

Guida per l'utente

AWS Strumenti per PowerShell (versione 4)



AWS Strumenti per PowerShell (versione 4): Guida per l'utente

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

I marchi e il trade dress di Amazon non possono essere utilizzati in relazione a prodotti o servizi che non siano di Amazon, in qualsiasi modo che possa causare confusione tra i clienti o in qualsiasi modo che denigri o discrediti Amazon. Tutti gli altri marchi non di proprietà di Amazon sono di proprietà dei rispettivi proprietari, che possono o meno essere affiliati, collegati o sponsorizzati da Amazon.

Table of Contents

.....	x
Cosa sono gli AWS Strumenti per PowerShell?	1
Manutenzione e supporto per le versioni principali dell'SDK	2
AWS.Tools	2
AWSPower- Conchiglia. NetCore	3
AWSPowerShell	3
Come utilizzare questa guida	4
Argomenti aggiuntivi in questa sezione	4
Cronologia delle revisioni	4
Cosa c'è di nuovo	5
Installazione	7
Installazione su Windows	7
Prerequisiti	8
Installazione di AWS.Tools	8
Installa AWSPower Shell. NetCore	11
Installa AWSPower Shell	12
Abilitazione dell'esecuzione di uno script	13
Controllo delle versioni	15
Aggiornamento AWS Strumenti per PowerShell	17
Installazione su Linux o macOS	18
Panoramica della configurazione	19
Prerequisiti	8
Installazione di AWS.Tools	20
Installa AWSPower Shell. NetCore	22
Esecuzione di uno script	13
Configurazione della console PowerShell	25
Inizializza la tua sessione PowerShell	25
Controllo delle versioni	15
Aggiornamento di AWS Strumenti per PowerShell su Linux o macOS	26
Informazioni correlate	27
Migrazione dalla AWS Strumenti per PowerShell versione 3.3 alla versione 4	28
Nuova versione AWS.Tools completamente modularizzata	28
Nuovo cmdlet Get-AWSService	29
Nuovo parametro -Select per controllare l'oggetto restituito da un cmdlet	29

Limitazione più coerente del numero di elementi nell'output	31
Parametri di flusso più facili da usare	31
Estendere la pipe in base al nome della proprietà	32
Parametri comuni statici	32
AWS.Tools Dichiara e applica i parametri obbligatori	33
Tutti i parametri sono "nullable"	33
Rimozione di funzionalità precedentemente deprecate	33
Inizia a usare	35
Configurazione dell'autenticazione degli strumenti	35
Abilitazione e configurazione di IAM Identity Center	36
Configura gli strumenti per PowerShell utilizzare IAM Identity Center.	36
Avviare una sessione del portale di AWS accesso	38
Esempio	39
Informazioni aggiuntive	40
Usa il AWS CLI	40
Specificare AWS le regioni	44
Specificare un endpoint personalizzato o non standard	46
Informazioni aggiuntive	46
Configurazione dell'identità federata	46
Prerequisiti	47
In che modo un utente con identità federata ottiene l'accesso federato al servizio AWS	
APIs	47
Come funziona il supporto SAML nel AWS Strumenti per PowerShell	49
Come utilizzare i cmdlet di configurazione SAML PowerShell	50
Ulteriori letture	55
Individuazione di cmdlet e alias	55
Individuare i cmdlet	55
Nomi dei cmdlet e degli alias	61
Pipeline, output e iterazione	66
Pipeline	66
Output del cmdlet	66
Iterazione tramite dati paginati	68
Risoluzione di credenziali e profili	70
Ordine di ricerca credenziali	70
Utenti e ruoli	71
Utenti e set di autorizzazioni	71

Ruoli di servizio	71
Uso delle credenziali legacy	72
Avvertenze e linee guida importanti	73
AWS Credenziali	74
Credenziali condivise	83
Funzionalità	89
Osservabilità	89
Lavora con AWS i servizi	93
PowerShell Codifica della concatenazione di file	93
Oggetti PowerShell restituiti per gli strumenti	94
Amazon EC2	94
Amazon S3	94
AWS Lambda e AWS Strumenti per PowerShell	95
Amazon SNS e Amazon SQS	95
CloudWatch	95
Vedi anche	95
Argomenti	95
Amazon S3 e strumenti per Windows PowerShell	96
Creare un bucket Amazon S3, verificare la sua Regione e rimuoverlo se si desidera	97
Configurare un bucket Amazon S3 come website e abilitare la registrazione	98
Caricare oggetti in un bucket Amazon S3	98
Eliminare bucket e oggetti Amazon S3	100
Caricamento dei contenuti di testo in linea su Amazon S3	102
Amazon EC2 e strumenti per Windows PowerShell	102
Crea una coppia di chiavi	103
Crea un gruppo di sicurezza	105
Trovare una AMI	108
Avviare un'istanza	112
AWS Lambda e AWS Strumenti per PowerShell	114
Prerequisiti	8
Installa il modulo AWSLambda PSCore	115
Vedi anche	95
Amazon SQS, Amazon SNS e strumenti per Windows PowerShell	116
Creare una coda Amazon SQS e ottenere l'ARN della coda	116
Creare un argomento Amazon SNS.	117
Concedere le autorizzazioni all'argomento SNS	117

Sottoscrivere la coda all'argomento SNS	118
Concedere le autorizzazioni	118
Verificare i risultati	118
CloudWatch dal AWS Tools for Windows PowerShell	120
Pubblicare un parametro personalizzato nel pannello di controllo CloudWatch	120
Vedi anche	95
Utilizzando ClientConfig	121
Uso del parametro ClientConfig	121
Uso di una proprietà non definita	122
Specificando il Regione AWS	122
Esempi di codice	123
ACM	125
Azioni	125
Application Auto Scaling	130
Azioni	125
WorkSpaces Applicazioni	137
Azioni	125
Aurora	164
Azioni	125
Auto Scaling	165
Azioni	125
Budget AWS	201
Azioni	125
AWS Cloud9	203
Azioni	125
CloudFormation	209
Azioni	125
CloudFront	222
Azioni	125
CloudTrail	230
Azioni	125
CloudWatch	235
Azioni	125
CodeCommit	240
Azioni	125
CodeDeploy	245

Azioni	125
CodePipeline	265
Azioni	125
Amazon Cognito Identity	283
Azioni	125
AWS Config	287
Azioni	125
Device Farm	307
Azioni	125
Directory Service	307
Azioni	125
AWS DMS	334
Azioni	125
DynamoDB	335
Azioni	125
Amazon EC2	350
Azioni	125
Amazon ECR	483
Azioni	125
Amazon ECS	485
Azioni	125
Amazon EFS	491
Azioni	125
Amazon EKS	498
Azioni	125
ELB - Versione 1	511
Azioni	125
ELB - Versione 2	531
Azioni	125
Amazon FSx	555
Azioni	125
Amazon Glacier	563
Azioni	125
AWS Glue	567
Azioni	125
AWS Health	569

Azioni	125
IAM	570
Azioni	125
Kinesis	645
Azioni	125
Lambda	649
Azioni	125
Amazon ML	662
Azioni	125
Macie	668
Azioni	125
Listino prezzi AWS	669
Azioni	125
Gruppi di risorse	672
Azioni	125
Resource Groups Tagging API	680
Azioni	125
Route 53	685
Azioni	125
Simple Storage Service (Amazon S3)	700
Azioni	125
Security Hub CSPM	736
Azioni	125
Amazon SES	738
Azioni	125
API Amazon SES v2	740
Azioni	125
Amazon SNS	740
Azioni	125
Amazon SQS	742
Azioni	125
AWS STS	754
Azioni	125
Supporto	758
Azioni	125
Systems Manager	765

Azioni	125
Amazon Translate	838
Azioni	125
AWS WAFV2	839
Azioni	125
WorkSpaces	840
Azioni	125
Sicurezza	857
Protezione dei dati	857
Crittografia dei dati	858
Identity and Access Management	859
Destinatari	859
Autenticazione con identità	860
Gestione dell'accesso con l'utilizzo delle policy	861
Come Servizi AWS lavorare con IAM	863
Risoluzione dei problemi di AWS identità e accesso	864
Convalida della conformità	865
Applicazione di una versione minima di TLS	866
Ulteriori considerazioni sulla sicurezza	866
Registrazione di informazioni sensibili	866
Riferimenti cmdlet	868
Cronologia dei documenti	869
.....	ccccclxxvii

È stata rilasciata la versione 5 (V5) di! AWS Strumenti per PowerShell

Per iniziare a utilizzare la nuova versione degli strumenti, consulta la [Guida per l'AWS Strumenti per PowerShell utente \(V5\)](#), in particolare l'argomento per la [migrazione](#) alla V5.

Cosa sono gli AWS Strumenti per PowerShell?

AWS Strumenti per PowerShell Sono un insieme di PowerShell moduli che si basano sulle funzionalità esposte da AWS SDK per .NET. Ti AWS Strumenti per PowerShell consentono di eseguire operazioni di script sulle tue AWS risorse dalla PowerShell riga di comando.

I cmdlet offrono un' PowerShell esperienza idiomatica per la specificazione dei parametri e la gestione dei risultati, anche se vengono implementati utilizzando le query HTTP dei vari servizi. AWS APIs Ad esempio, i cmdlet per la PowerShell pipeline di AWS Strumenti per PowerShell supporto, ovvero è possibile reindirizzare oggetti all'interno e all'esterno dei cmdlet. PowerShell

AWS Strumenti per PowerShell Sono flessibili nel modo in cui consentono di gestire le credenziali, incluso il supporto per l'infrastruttura (IAM). AWS Identity and Access Management Puoi utilizzare gli strumenti con credenziali utente IAM, token di sicurezza temporanei e ruoli IAM.

AWS Strumenti per PowerShell Supportano lo stesso set di servizi e AWS regioni supportati dall'SDK. Puoi installarlo AWS Strumenti per PowerShell su computer che eseguono sistemi operativi Windows, Linux o macOS.

Note

AWS Strumenti per PowerShell la versione 4 (V4) è un aggiornamento retrocompatibile alla versione 3.3. AWS Strumenti per PowerShell Aggiunge miglioramenti significativi pur mantenendo il comportamento dei cmdlet esistenti. Gli script esistenti dovrebbero continuare a funzionare dopo l'aggiornamento alla V4, ma consigliamo di testarli accuratamente prima dell'aggiornamento. Per ulteriori informazioni sulle modifiche apportate alla V4, vedere.

[Migrazione dalla AWS Strumenti per PowerShell versione 3.3 alla versione 4](#)

AWS Strumenti per PowerShell Sono disponibili nei seguenti tre pacchetti distinti:

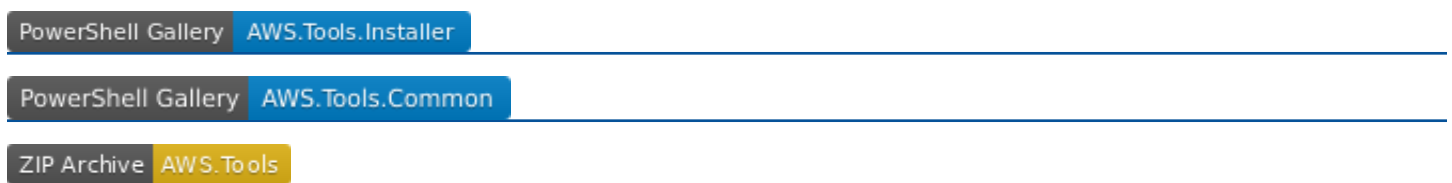
- [AWS.Tools](#)
- [AWSPowerConchiglia. NetCore](#)
- [AWSPowerConchiglia](#)

Manutenzione e supporto per le versioni principali dell'SDK

Per informazioni sulla manutenzione e il supporto per le versioni principali dell'SDK e le relative dipendenze sottostanti, consulta quanto segue nella Guida di [riferimento agli strumenti AWS SDKs e agli strumenti](#):

- [AWS SDKs e politica di manutenzione degli strumenti](#)
- [AWS SDKs e matrice di supporto delle versioni degli strumenti](#)

AWS.Tools- Una versione modulare del AWS Strumenti per PowerShell



Questa versione di AWS Strumenti per PowerShell è la versione consigliata per qualsiasi computer PowerShell in esecuzione in un ambiente di produzione. Poiché è suddiviso in moduli, devi scaricare e caricare solo i moduli per i servizi che desideri utilizzare. Ciò riduce i tempi di download, l'utilizzo della memoria e, nella maggior parte dei casi, consente l'importazione automatica dei cmdlet AWS.Tools con la necessità di chiamare manualmente prima `Import-Module`.

Questa è la versione più recente di AWS Strumenti per PowerShell e funziona su tutti i sistemi operativi supportati, inclusi Windows, Linux e macOS. Questo pacchetto fornisce un modulo di installazione `AWS.Tools.Installer`, un modulo comune e un modulo per ogni AWS servizio, ad esempio, `AWS.Tools.EC2`, `AWS.Tools.IdentityManagement`, `AWS.Tools.S3`, e così via. `AWS.Tools.Common`

Il `AWS.Tools.Installer` modulo fornisce cmdlet che consentono di installare, aggiornare e rimuovere i moduli per ogni servizio. AWS I cmdlet in questo modulo garantiscono automaticamente di disporre di tutti i moduli dipendenti necessari per supportare i moduli che desideri utilizzare.

Il modulo `AWS.Tools.Common` fornisce cmdlet per la configurazione e l'autenticazione che non sono specifici del servizio. Per utilizzare i cmdlet per un AWS servizio, è sufficiente eseguire il comando. PowerShell importa automaticamente il `AWS.Tools.Common` modulo e il modulo per il AWS servizio

di cui si desidera eseguire il cmdlet. Questo modulo viene installato automaticamente se utilizzi il modulo `AWS.Tools.Installer` per installare i moduli di servizio.

È possibile installare questa versione di AWS Strumenti per PowerShell su computer che eseguono:

- PowerShell Core 6.0 o versione successiva su Windows, Linux o macOS.
- Windows PowerShell 5.1 o versione successiva su Windows con .NET Framework 4.7.2 o versione successiva.

In questa guida, quando abbiamo bisogno di specificare solo questa versione, vi facciamo riferimento con il nome del modulo: `AWS.Tools`.

AWSPowerShell. NetCore - Una versione a modulo singolo di AWS Strumenti per PowerShell

PowerShell Gallery **AWSPowerShell.NetCore**

ZIP Archive **AWSPowerShell.NetCore**

Questa versione è costituita da un unico modulo di grandi dimensioni che contiene il supporto per tutti i AWS servizi. Prima di poter utilizzare questo modulo, devi importarlo manualmente.

È possibile installare questa versione di AWS Strumenti per PowerShell su computer che eseguono:

- PowerShell Core 6.0 o versione successiva su Windows, Linux o macOS.
- Windows PowerShell 3.0 o versione successiva su Windows con .NET Framework 4.7.2 o versione successiva.

In questa guida, quando è necessario specificare solo questa versione, ci riferiamo ad essa con il nome del modulo: `AWSPowerShell. NetCore`.

AWSPowerShell: una versione a modulo singolo per Windows PowerShell

PowerShell Gallery **AWSPowerShell**

ZIP Archive **AWSPowerShell**

Questa versione di AWS Strumenti per PowerShell è compatibile e installabile solo su computer Windows che eseguono PowerShell le versioni di Windows da 2.0 a 5.1. Non è compatibile con PowerShell Core 6.0 o versioni successive o con qualsiasi altro sistema operativo (Linux o macOS). Questa versione è costituita da un unico modulo di grandi dimensioni che contiene il supporto per tutti i AWS servizi.

In questa guida, quando è necessario specificare solo questa versione, ci riferiamo ad essa con il nome del modulo: `AWSPowerShell`.

Come utilizzare questa guida

La guida è suddivisa nelle seguenti sezioni principali.

[Installazione del AWS Strumenti per PowerShell](#)

Questa sezione spiega come installare AWS Strumenti per PowerShell. Include come registrarsi AWS se non si dispone già di un account e come creare un utente IAM da utilizzare per eseguire i cmdlet.

[Inizia con AWS Tools for Windows PowerShell](#)

Questa sezione descrive i fondamenti dell'utilizzo di AWS Strumenti per PowerShell, ad esempio la specificazione di credenziali e AWS regioni, la ricerca di cmdlet per un particolare servizio e l'utilizzo di alias per i cmdlet.

[Lavora con AWS i servizi in AWS Strumenti per PowerShell](#)

Questa sezione include informazioni sull'utilizzo di per eseguire alcune delle attività più comuni AWS Strumenti per PowerShell . AWS

Argomenti aggiuntivi in questa sezione

- [Cronologia delle revisioni](#)
- [Cosa c'è di nuovo nel AWS Strumenti per PowerShell](#)

Cronologia delle revisioni

Per scoprire cosa è cambiato nelle varie versioni, consulta quanto segue:

- [I registri delle modifiche](#)
- [Cosa c'è di nuovo nel AWS Strumenti per PowerShell](#)
- [Cronologia dei documenti](#)

Cosa c'è di nuovo nel AWS Strumenti per PowerShell

Per informazioni di alto livello sui nuovi sviluppi relativi a AWS Strumenti per PowerShell, consulta la pagina del prodotto all'indirizzo <https://aws.amazon.com/powershell/> e i [registri delle modifiche](#).

Di seguito sono riportate le novità degli Strumenti per PowerShell

17 settembre 2025: in arrivo end-of-support la versione 4 di AWS Strumenti per PowerShell

È stata annunciata la versione end-of-support per questa versione (V4) di AWS Strumenti per PowerShell. Consulta la [guida alla migrazione V5](#) per migrare gli script V4 ed evitare interruzioni. [Per ulteriori informazioni, consulta il post sul blog Announcing the for v4. end-of-support AWS Strumenti per PowerShell](#)

23 giugno 2025: versione 5 del AWS Strumenti per PowerShell

La versione 5 (V5) di AWS Strumenti per PowerShell è generalmente disponibile! Per ulteriori informazioni, consulta la [Guida per l'AWS Strumenti per PowerShell utente \(V5\)](#), in particolare l'argomento per la [migrazione](#) alla V5.

10 febbraio 2025: versione GA per l'osservabilità

L'osservabilità è la misura in cui lo stato attuale di un sistema può essere dedotto dai dati che emette. L'osservabilità è stata aggiunta agli strumenti per PowerShell, inclusa l'implementazione di un provider di telemetria. Per ulteriori informazioni, consulta [Osservabilità](#) questa guida e il post di blog che [annuncia la disponibilità generale delle](#) librerie.NET. AWS OpenTelemetry

15 gennaio 2025: nuovo comportamento predefinito per la protezione dell'integrità

A partire dalla versione 4.1.737 di AWS Strumenti per PowerShell, gli strumenti forniscono protezioni di integrità predefinite calcolando automaticamente un checksum per i caricamenti. CRC32 Per ulteriori informazioni, si veda l'annuncio pubblicato all'indirizzo. GitHub <https://github.com/aws/aws-tools-for-powershell/issues/370> Gli strumenti forniscono anche impostazioni globali per la protezione dell'integrità dei dati che è possibile impostare esternamente, per maggiori informazioni si rimanda

alla sezione [Protezione dell'integrità dei dati nella Guida](#) di riferimento degli strumenti [AWS SDKs e degli strumenti](#).

18 novembre 2024: versione di anteprima 1 per la versione 5

L'anteprima 1 della AWS Strumenti per PowerShell versione 5 è stata rilasciata il 18 novembre 2024. Per ulteriori informazioni su questa anteprima, consulta il post sul blog [Preview 1 of AWS Strumenti per PowerShell V5](#).

Installazione del AWS Strumenti per PowerShell

Per installare e utilizzare correttamente i AWS Strumenti per PowerShell cmdlet, vedere i passaggi nei seguenti argomenti.

Argomenti

- [Installazione di AWS Strumenti per PowerShell su Windows](#)
- [Installazione AWS Strumenti per PowerShell su Linux o macOS](#)
- [Migrazione dalla AWS Strumenti per PowerShell versione 3.3 alla versione 4](#)

Installazione di AWS Strumenti per PowerShell su Windows

Un computer basato su Windows può eseguire qualsiasi opzione del pacchetto: AWS Strumenti per PowerShell

- [AWS.Tools](#) - La versione modulare di AWS Strumenti per PowerShell. Ogni AWS servizio è supportato da un proprio piccolo modulo individuale, con moduli `AWS.Tools.Common` di supporto condivisi e `AWS.Tools.Installer`.
- [AWSPowerConchiglia.NetCore](#) - La versione a modulo singolo di grandi dimensioni di AWS Strumenti per PowerShell. Tutti i AWS servizi sono supportati da questo unico modulo di grandi dimensioni.

Note

Tieni presente che il singolo modulo potrebbe essere troppo grande per essere utilizzato con le funzioni [AWS Lambda](#). Utilizza invece la versione modulare mostrata in precedenza.

- [AWSPowerShell](#): la versione precedente, specifica per Windows, a modulo singolo e di grandi dimensioni di AWS Strumenti per PowerShell. Tutti i AWS servizi sono supportati da questo unico modulo di grandi dimensioni.

Il pacchetto scelto dipende dalla versione e dall'edizione di Windows in esecuzione.

Note

AWS Strumenti per PowerShell Sono installati per impostazione predefinita su tutte le Amazon Machine Images () AMIs basate su Windows. L'opzione installata dipende dall'AMI. Molti AMIs hanno il modulo AWSPower Shell, ma alcuni potrebbero avere un'opzione diversa. Ad esempio, Amazon EC2 AMIs for Windows Server 2025 utilizza l'`AWS.Tools` opzione modulare.

La configurazione di AWS Strumenti per PowerShell prevede le seguenti attività di alto livello, descritte in dettaglio in questo argomento.

1. Installa l'opzione di AWS Strumenti per PowerShell pacchetto appropriata per il tuo ambiente.
2. Verificare che l'esecuzione dello script sia abilitata eseguendo il cmdlet `Get-ExecutionPolicy`.
3. Importa il AWS Strumenti per PowerShell modulo nella tua PowerShell sessione.

Prerequisiti

Le versioni più recenti di PowerShell, tra cui PowerShell Core, sono disponibili come download da Microsoft all'indirizzo [Installazione di varie versioni del PowerShell](#) sito Web di Microsoft.

Installazione di **AWS.Tools** su Windows

È possibile installare la versione modulare di AWS Strumenti per PowerShell su computer che eseguono Windows con Windows PowerShell 5.1 o PowerShell Core 6.0 o versione successiva. Per informazioni su come installare PowerShell Core, vedere [Installazione di varie versioni di PowerShell sul sito Web di Microsoft](#).

È possibile installare `AWS.Tools` in uno dei tre modi:

- Utilizzo dei cmdlet nel modulo `AWS.Tools.Installer`. Questo modulo semplifica l'installazione e l'aggiornamento di altri `AWS.Tools` moduli. `AWS.Tools.Installer` richiede `PowerShellGet` e ne scarica e installa automaticamente una versione aggiornata. `AWS.Tools.Installer` mantiene automaticamente sincronizzate le versioni del modulo. Quando si installa o si esegue l'aggiornamento a una versione più recente di un modulo, i cmdlet inclusi aggiornano `AWS.Tools.Installer` automaticamente tutti gli altri `AWS.Tools` moduli alla stessa versione.

Questo metodo è descritto nella procedura che segue.

- Scaricando i moduli da [AWS.Tools.zip](#) ed estraendoli in una delle cartelle dei moduli. È possibile scoprire le cartelle dei moduli visualizzando il valore della variabile di ambiente `PSModulePath`.

Warning

Dopo aver scaricato il file ZIP e prima di estrarne il contenuto, potrebbe essere necessario sbloccarlo. Questa operazione viene in genere eseguita aprendo le proprietà del file, visualizzando la scheda Generale e selezionando la casella di controllo Sblocca, se presente.

Se il file ZIP deve essere sbloccato ma non lo fai, potresti ricevere errori simili ai seguenti: «Import-Module: Could not load file or assembly».

- Installazione di ogni modulo di servizio dalla PowerShell Galleria utilizzando il cmdlet. `Install-Module`

Per l'installazione **AWS.Tools** su Windows utilizzando il modulo **AWS.Tools.Installer**

1. Avvia una PowerShell sessione.

Note

Ti consigliamo di non lavorare PowerShell come amministratore con autorizzazioni elevate, tranne quando richiesto dall'attività in questione. Ciò è dovuto al potenziale rischio per la sicurezza ed è in contrasto con il principio del privilegio minimo.

2. Per installare il pacchetto **AWS.Tools** suddiviso in moduli, esegui il comando seguente.

```
PS > Install-Module -Name AWS.Tools.Installer
```

```
Untrusted repository
```

```
You are installing the modules from an untrusted repository. If you trust this repository, change its InstallationPolicy value by running the Set-PSRepository cmdlet. Are you sure
```

```
you want to install the modules from 'PSGallery'?
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): y
```

Se ricevi una notifica in cui è indicato che il repository non è attendibile, ti viene chiesto se desideri installarlo comunque. Inserisci **y** per consentire l' PowerShell installazione del modulo.

Per evitare il prompt e installare il modulo senza dover considerare attendibile il repository, puoi eseguire il comando con il parametro `-Force`.

```
PS > Install-Module -Name AWS.Tools.Installer -Force
```

3. È ora possibile installare il modulo per ogni AWS servizio che si desidera utilizzare utilizzando il `Install-AWSToolsModule` cmdlet. Ad esempio, il comando seguente installa i moduli Amazon EC2 e Amazon S3. Questo comando installa anche tutti i moduli dipendenti necessari per il funzionamento del modulo specificato. Ad esempio, quando installi il primo modulo di servizio `AWS.Tools`, viene installato anche `AWS.Tools.Common`. Si tratta di un modulo condiviso richiesto da tutti i moduli di AWS servizio. Rimuove anche le versioni precedenti dei moduli e aggiorna altri moduli alla stessa versione più recente.

```
PS > Install-AWSToolsModule AWS.Tools.EC2,AWS.Tools.S3 -Cleanup
Confirm
Are you sure you want to perform this action?
Performing the operation "Install-AWSToolsModule" on target "AWS Tools version 4.0.0.0".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):

Installing module AWS.Tools.Common version 4.0.0.0
Installing module AWS.Tools.EC2 version 4.0.0.0
Installing module AWS.Tools.Glacier version 4.0.0.0
Installing module AWS.Tools.S3 version 4.0.0.0

Uninstalling AWS.Tools version 3.3.618.0
Uninstalling module AWS.Tools.Glacier
Uninstalling module AWS.Tools.S3
Uninstalling module AWS.Tools.SimpleNotificationService
Uninstalling module AWS.Tools.SQS
Uninstalling module AWS.Tools.Common
```

Note

Il `Install-AWSToolsModule` cmdlet scarica tutti i moduli richiesti dal file PSRepository denominato PSGallery (<https://www.powershellgallery.com/>) e lo considera una fonte attendibile. Usa il comando `Get-PSRepository -Name PSGallery` per ulteriori informazioni su PSRepository.

Per impostazione predefinita, il comando precedente installa i moduli nella cartella %USERPROFILE%\Documents\WindowsPowerShell\Modules. Per installarlo AWS Strumenti per PowerShell per tutti gli utenti di un computer, è necessario eseguire il comando seguente in una PowerShell sessione avviata come amministratore. Ad esempio, il seguente comando installa il modulo IAM nella cartella %ProgramFiles%\WindowsPowerShell\Modules che è accessibile a tutti gli utenti.

```
PS > Install-AWSToolsModule AWS.Tools.IdentityManagement -Scope AllUsers
```

Per installare altri moduli, esegui comandi simili con i nomi dei moduli appropriati, come si trova nella [PowerShell Galleria](#).

Installa AWSPower Shell. NetCore su Windows

Puoi installare la AWSPower Shell. NetCore su computer che eseguono Windows con PowerShell versioni da 3 a 5.1 oppure PowerShell Core 6.0 o versioni successive. Per informazioni su come installare PowerShell Core, vedi [Installazione di varie versioni PowerShell](#) sul PowerShell sito Web di Microsoft.

È possibile installare AWSPower Shell. NetCore in due modi

- Scaricamento del modulo da [AWSPowerShell. NetCore.zip](#) ed estraendolo in una delle directory del modulo. Puoi individuare le directory dei moduli visualizzando il valore della variabile di ambiente PSModulePath.

Warning

Dopo aver scaricato il file ZIP e prima di estrarne il contenuto, potrebbe essere necessario sbloccarlo. Questa operazione viene in genere eseguita aprendo le proprietà del file, visualizzando la scheda Generale e selezionando la casella di controllo Sblocca, se presente.

Se il file ZIP deve essere sbloccato ma non lo fai, potresti ricevere errori simili ai seguenti: «Import-Module: Could not load file or assembly».

- Installazione dalla PowerShell Galleria utilizzando il Install-Module cmdlet, come descritto nella procedura seguente.

Per installare Shell AWSPower. NetCore dalla PowerShell Galleria utilizzando il cmdlet Install-Module

Per installare Shell. AWSPower NetCore dalla PowerShell Galleria, sul computer deve essere installata la PowerShell versione 5.0 o versione successiva oppure la versione PowerShell 3 o versione successiva. [PowerShellGet](#) Esegui il comando seguente.

```
PS > Install-Module -name AWSPowerShell.NetCore
```

Se esegui PowerShell come amministratore, il comando precedente viene installato AWS Strumenti per PowerShell per tutti gli utenti del computer. Se si esegue PowerShell come utente standard senza autorizzazioni di amministratore, lo stesso comando viene installato solo AWS Strumenti per PowerShell per l'utente corrente.

Per effettuare l'installazione solo per l'utente corrente quando tale utente dispone delle autorizzazioni di amministratore, esegui il comando con il set di parametri -Scope CurrentUser, come indicato di seguito.

```
PS > Install-Module -name AWSPowerShell.NetCore -Scope CurrentUser
```

Sebbene la PowerShell versione 3.0 e le versioni successive in genere carichino i moduli nella PowerShell sessione la prima volta che si esegue un cmdlet nel modulo Shell. AWSPower NetCore il modulo è troppo grande per supportare questa funzionalità. È invece necessario caricare esplicitamente la AWSPower Shell. NetCore Modulo principale nella PowerShell sessione eseguendo il comando seguente.

```
PS > Import-Module AWSPowerShell.NetCore
```

Per caricare la AWSPower Shell. NetCore inserisci automaticamente il modulo in una PowerShell sessione, aggiungi quel comando al tuo PowerShell profilo. Per ulteriori informazioni sulla modifica del PowerShell profilo, consulta [About Profiles](#) nella PowerShell documentazione.

Installa AWSPower Shell su Windows PowerShell

Puoi installarlo AWS Tools for Windows PowerShell in due modi:

- Scaricando il modulo da [AWSPowerShell.zip](#) ed estraendolo in una delle directory del modulo. Puoi individuare le directory dei moduli visualizzando il valore della variabile di ambiente PSModulePath.

 Warning

Dopo aver scaricato il file ZIP e prima di estrarne il contenuto, potrebbe essere necessario sbloccarlo. Questa operazione viene in genere eseguita aprendo le proprietà del file, visualizzando la scheda Generale e selezionando la casella di controllo Sblocca, se presente.

Se il file ZIP deve essere sbloccato ma non lo fai, potresti ricevere errori simili ai seguenti: «Import-Module: Could not load file or assembly».

- Installazione dalla PowerShell Galleria utilizzando il `Install-Module` cmdlet come descritto nella procedura seguente.

Per installare AWSPower Shell dalla PowerShell Gallery utilizzando il cmdlet `Install-Module`

È possibile installare AWSPower Shell dalla PowerShell Galleria se si esegue la PowerShell versione 5.0 o successiva oppure se è installata [PowerShellGet](#) la versione 3 o successiva. PowerShell Puoi installare e aggiornare AWSPower Shell dalla [PowerShellGalleria](#) di Microsoft eseguendo il comando seguente.

```
PS > Install-Module -Name AWSPowerShell
```

Per caricare automaticamente il modulo AWSPower Shell in una PowerShell sessione, aggiungi il `import-module` cmdlet precedente al tuo PowerShell profilo. Per ulteriori informazioni sulla modifica del PowerShell profilo, vedere [Informazioni sui profili](#) nella PowerShell documentazione.

 Note

Gli strumenti per Windows PowerShell sono installati per impostazione predefinita su tutte le Amazon Machine Images () AMIs basate su Windows.

Abilitazione dell'esecuzione di uno script

Per caricare i AWS Strumenti per PowerShell moduli, è necessario abilitare l'esecuzione degli PowerShell script. Per abilitare l'esecuzione dello script, eseguire il cmdlet `Set-ExecutionPolicy` per impostare una policy `RemoteSigned`. Per ulteriori informazioni, consulta [Informazioni sulle policy di esecuzione](#) sul sito Web Microsoft Technet.

 Note

Questo è un requisito solo per i computer che eseguono Windows. La restrizione di sicurezza `ExecutionPolicy` non è presente su altri sistemi operativi.

Per abilitare l'esecuzione dello script

1. Per impostare la policy di esecuzione sono necessari i diritti di amministratore. Se non hai effettuato l'accesso come utente con diritti di amministratore, apri una PowerShell sessione come amministratore. Scegli Start, quindi seleziona Tutti i programmi. Scegli Accessori, quindi scegli Windows PowerShell. Fate clic con PowerShell il pulsante destro del mouse su Windows e nel menu contestuale scegliete Esegui come amministratore.
2. Nel prompt dei comandi inserisci quanto segue.

```
PS > Set-ExecutionPolicy RemoteSigned
```

 Note

Su un sistema a 64 bit, è necessario eseguire questa operazione separatamente per la versione a 32 bit di PowerShell Windows PowerShell (x86).

Se la politica di esecuzione non è impostata correttamente, PowerShell mostra il seguente errore ogni volta che si tenta di eseguire uno script, ad esempio il profilo.

```
File C:\Users\username\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1
cannot be loaded because the execution
of scripts is disabled on this system. Please see "get-help about_signing" for more
details.
At line:1 char:2
+ . <<<< 'C:\Users\username\Documents\WindowsPowerShell
\Microsoft.PowerShell_profile.ps1'
+ CategoryInfo          : NotSpecified: (:) [], PSSecurityException
+ FullyQualifiedErrorId : RuntimeException
```

Il programma di PowerShell installazione di Tools for Windows aggiorna automaticamente il [PSModulePath](#) per includere la posizione della directory che contiene il `AWSPowerShell` modulo.

Poiché PSModulePath include la posizione della directory del AWS modulo, il `Get-Module -ListAvailable` cmdlet mostra il modulo.

```
PS > Get-Module -ListAvailable
```

ModuleType	Name	ExportedCommands
Manifest	AppLocker	{}
Manifest	BitsTransfer	{}
Manifest	PSDiagnostics	{}
Manifest	TroubleshootingPack	{}
Manifest	AWSPowerShell	{Update-EBApplicationVersion, Set-DPStatus, Remove-IAMGroupPol...

Controllo delle versioni

AWS rilascia AWS Strumenti per PowerShell periodicamente nuove versioni di per supportare nuovi AWS servizi e funzionalità. Per determinare la versione degli strumenti installata, eseguire il `AWSPowerShellVersion` cmdlet [Get-](#).

Per esempio:

```
PS > Get-AWSPowerShellVersion
```

```
AWS Tools for PowerShell
```

```
Version 4.1.849
```

```
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
Amazon Web Services SDK for .NET
```

```
Core Runtime Version 3.7.402.75
```

```
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
Release notes: https://github.com/aws/aws-tools-for-powershell/blob/v4.1/changelogs/CHANGELOG.ALL.md
```

```
This software includes third party software subject to the following copyrights:
```

```
- Logging from log4net, Apache License
```

```
[http://logging.apache.org/log4net/license.html]
```

È inoltre possibile aggiungere il `-ListServiceVersionInfo` parametro a un `AWSPowerShellVersion` comando [Get-](#) per visualizzare un elenco dei AWS servizi supportati nella versione

corrente degli strumenti. Se utilizzi l'opzione `AWS.Tools.*` suddivisa in moduli, vengono visualizzati solo i moduli attualmente importati.

Per esempio:

```
PS > Get-AWSPowerShellVersion -ListServiceVersionInfo
...

Service                                Noun Prefix Module Name                                SDK
-----
Assembly
Version
-----
-----
AWS IAM Access Analyzer                IAMAA      AWS.Tools.AccessAnalyzer
3.7.400.33
AWS Account                           ACCT       AWS.Tools.Account
3.7.400.33
AWS Certificate Manager Private... PCA        AWS.Tools.ACMPCA
3.7.400.34
AWS Amplify                           AMP        AWS.Tools.Amplify
3.7.401.28
Amplify Backend                       AMPB       AWS.Tools.AmplifyBackend
3.7.400.33
...
```

Per determinare la versione PowerShell che stai utilizzando, inserisci `$PSVersionTable` per visualizzare il contenuto della [variabile automatica](#) `$PSVersionTable`.

Per esempio:

```
PS > $PSVersionTable

Name                                Value
----                                -
PSVersion                          6.2.2
PSEdition                          Core
GitCommitId                        6.2.2
OS                                  Darwin 18.7.0 Darwin Kernel Version 18.7.0: Tue Aug 20
16:57:14 PDT 2019; root:xnu-4903.271.2~2/RELEASE_X86_64
Platform                          Unix
PSCompatibleVersions               {1.0, 2.0, 3.0, 4.0...}
```

PSRemotingProtocolVersion	2.3
SerializationVersion	1.1.0.1
WSManStackVersion	3.0

Aggiornamento di AWS Strumenti per PowerShell su Windows

Periodicamente, man mano che AWS Strumenti per PowerShell vengono rilasciate versioni aggiornate di, è necessario aggiornare la versione in esecuzione localmente.

Aggiornate i moduli modulari **AWS.Tools**

Per aggiornare i AWS.Tools moduli alla versione più recente, esegui il seguente comando:

```
PS > Update-AWSToolsModule -CleanUp
```

Questo comando aggiorna tutti i moduli AWS.Tools attualmente installati e, dopo un aggiornamento riuscito, rimuove altre versioni installate.

Note

Il Update-AWSToolsModule cmdlet scarica tutti i moduli dal file PSRepository denominato PSGallery (<https://www.powershellgallery.com/>) e lo considera una fonte attendibile. Usa il comando Get-PSRepository -Name PSGallery per ulteriori informazioni su questo PSRepository.

Aggiorna gli strumenti per Core PowerShell

Esegui il Get-AWSPowerShellVersion cmdlet per determinare la versione in esecuzione e confrontala con la versione di Tools for Windows PowerShell disponibile sul sito Web [PowerShell Gallery](https://www.powershellgallery.com/). Consigliamo di controllare ogni due o tre settimane. Il supporto per nuovi comandi e AWS servizi è disponibile solo dopo l'aggiornamento a una versione con tale supporto.

Prima di installare una versione più recente di AWSPowerShell.NetCore, disinstalla il modulo esistente. Chiudi tutte PowerShell le sessioni aperte prima di disinstallare il pacchetto esistente. Per disinstallare il pacchetto, eseguire il comando seguente.

```
PS > Uninstall-Module -Name AWSPowerShell.NetCore -AllVersions
```

Dopo aver disinstallato il pacchetto, installa il modulo aggiornato eseguendo il comando riportato qui di seguito.

```
PS > Install-Module -Name AWSPowerShell.NetCore
```

Dopo l'installazione, esegui il comando `Import-Module AWSPowerShell.NetCore` per caricare i cmdlet aggiornati nella sessione PowerShell.

Aggiorna gli strumenti per Windows PowerShell

Esegui il `Get-AWSPowerShellVersion` cmdlet per determinare la versione in esecuzione e confrontala con la versione di Tools for Windows PowerShell disponibile sul sito Web [PowerShell Gallery](#). Consigliamo di controllare ogni due o tre settimane. Il supporto per nuovi comandi e AWS servizi è disponibile solo dopo l'aggiornamento a una versione con tale supporto.

- Se hai effettuato l'installazione utilizzando il cmdlet `Install-Module`, esegui il comando riportato qui di seguito.

```
PS > Uninstall-Module -Name AWSPowerShell -AllVersions
PS > Install-Module -Name AWSPowerShell
```

- Se hai effettuato l'installazione utilizzando un file ZIP scaricato:
 1. Scarica la versione più recente dal sito PowerShell Web [Tools for](#). Confronta il numero di versione del pacchetto nel nome del file scaricato con il numero di versione visualizzato quando esegui il cmdlet `Get-AWSPowerShellVersion`.
 2. Se il numero della versione scaricata è superiore a quello della versione installata, chiudi tutti gli strumenti per le PowerShell console Windows.
 3. Installa la versione più recente degli strumenti per Windows. PowerShell

Dopo l'installazione, `Import-Module AWSPowerShell` esegui per caricare i cmdlet aggiornati nella sessione. PowerShell Oppure esegui la AWS Strumenti per PowerShell console personalizzata dal menu Start.

Installazione AWS Strumenti per PowerShell su Linux o macOS

Questo argomento fornisce istruzioni su come installarlo AWS Strumenti per PowerShell su Linux o macOS.

Panoramica della configurazione

Per l'installazione AWS Strumenti per PowerShell su un computer Linux o macOS, puoi scegliere tra due opzioni di pacchetto:

- [AWS.Tools](#) — La versione modulare di. AWS Strumenti per PowerShell Ogni AWS servizio è supportato da un proprio piccolo modulo individuale, con moduli di supporto condivisi. `AWS.Tools.Common`
- [AWSPowerConchiglia. NetCore](#) — La versione a modulo singolo di grandi dimensioni di AWS Strumenti per PowerShell. Tutti i AWS servizi sono supportati da questo unico modulo di grandi dimensioni.

Note

Tieni presente che il singolo modulo potrebbe essere troppo grande per essere utilizzato con le funzioni [AWS Lambda](#). Utilizza invece la versione modulare mostrata in precedenza.

L'impostazione di una di queste versioni su un computer su cui è in esecuzione Linux o macOS comporta le seguenti attività, descritte in dettaglio più avanti in questo argomento:

1. Installa PowerShell Core 6.0 o versione successiva su un sistema supportato.
2. Dopo aver installato PowerShell Core, inizia PowerShell eseguendolo `pwsh` nella shell del sistema.
3. Installa una delle due `AWS.Tools` o `AWSPower Shell. NetCore`.
4. Esegui il `Import-Module` cmdlet appropriato per importare il modulo nella sessione PowerShell.
5. Esegui il cmdlet [Initialize- AWSDefault Configuration per fornire le](#) tue credenziali. AWS

Prerequisiti

Per eseguire AWS Tools for PowerShell Core, sul computer deve essere in esecuzione PowerShell Core 6.0 o versione successiva.

- Per un elenco delle versioni supportate della piattaforma Linux e per informazioni su come installare la versione più recente di PowerShell su un computer basato su Linux, vedi [Installazione PowerShell su Linux sul sito Web](#) di Microsoft. Alcuni sistemi operativi basati su Linux, ad esempio Arch, Kali e Raspbian non sono supportati ufficialmente, ma hanno vari livelli di supporto dalla comunità.

- Per informazioni sulle versioni di macOS supportate e su come installare la versione più recente di PowerShell macOS, consulta Installazione [su PowerShell macOS sul sito Web](#) di Microsoft.

Installazione di **AWS.Tools** su Linux o macOS

È possibile installare la versione modulare di AWS Strumenti per PowerShell su computer che PowerShell eseguono Core 6.0 o versione successiva. Per informazioni su come installare PowerShell Core, vedi [Installazione di varie versioni PowerShell](#) sul PowerShell sito Web di Microsoft.

È possibile installare **AWS.Tools** in uno dei tre modi:

- Utilizzo dei cmdlet nel modulo **AWS.Tools.Installer**. Questo modulo semplifica l'installazione e l'aggiornamento di altri **AWS.Tools** moduli. **AWS.Tools.Installer** richiede **PowerShellGet** e ne scarica e installa automaticamente una versione aggiornata. **AWS.Tools.Installer** mantiene automaticamente sincronizzate le versioni del modulo. Quando si installa o si esegue l'aggiornamento a una versione più recente di un modulo, i cmdlet inclusi aggiornano **AWS.Tools.Installer** automaticamente tutti gli altri **AWS.Tools** moduli alla stessa versione.

Questo metodo è descritto nella procedura che segue.

- Scaricando i moduli da [AWS.Tools.zip](#) ed estraendoli in una delle directory dei moduli. È possibile scoprire le directory dei moduli stampando il valore della variabile `$Env:PSModulePath`.
- Installazione di ogni modulo di servizio dalla PowerShell Galleria utilizzando il `Install-Module` cmdlet.

Per installare **AWS.Tools** su Linux o macOS utilizzando il modulo **AWS.Tools.Installer**

1. Avvia una sessione PowerShell Core eseguendo il comando seguente.

```
$ pwsh
```

Note

Ti consigliamo di non lavorare PowerShell come amministratore con autorizzazioni elevate, tranne quando richiesto dall'attività in questione. Ciò è dovuto al potenziale rischio per la sicurezza ed è in contrasto con il principio del privilegio minimo.

2. Per installare il pacchetto `AWS.Tools` modularizzato utilizzando il modulo `AWS.Tools.Installer`, eseguire il seguente comando.

```
PS > Install-Module -Name AWS.Tools.Installer
```

Untrusted repository

You are installing the modules from an untrusted repository. If you trust this repository, change its `InstallationPolicy` value by running the `Set-PSRepository` cmdlet. Are you sure you want to install the modules from 'PSGallery'?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): **y**

Se ricevi una notifica in cui è indicato che il repository non è attendibile, ti viene chiesto se vuoi installarlo comunque. Inserisci **y** per consentire l' PowerShell installazione del modulo. Per evitare il prompt e installare il modulo senza dover considerare attendibile il repository, puoi eseguire il seguente comando.

```
PS > Install-Module -Name AWS.Tools.Installer -Force
```

3. Ora puoi installare il modulo per ogni servizio che desideri utilizzare. Ad esempio, il comando seguente installa i moduli Amazon EC2 e Amazon S3. Questo comando installa anche tutti i moduli dipendenti necessari per il funzionamento del modulo specificato. Ad esempio, quando installi il primo modulo di servizio `AWS.Tools`, viene installato anche `AWS.Tools.Common`. Si tratta di un modulo condiviso richiesto da tutti i moduli di AWS servizio. Rimuove anche le versioni precedenti dei moduli e aggiorna altri moduli alla stessa versione più recente.

```
PS > Install-AWSToolsModule AWS.Tools.EC2,AWS.Tools.S3 -CleanUp
```

Confirm

Are you sure you want to perform this action?

Performing the operation "Install-AWSToolsModule" on target "AWS Tools version 4.0.0.0".

[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):

Installing module `AWS.Tools.Common` version 4.0.0.0

Installing module `AWS.Tools.EC2` version 4.0.0.0

Installing module `AWS.Tools.Glacier` version 4.0.0.0

Installing module `AWS.Tools.S3` version 4.0.0.0

Uninstalling `AWS.Tools` version 3.3.618.0

```
Uninstalling module AWS.Tools.Glacier
Uninstalling module AWS.Tools.S3
Uninstalling module AWS.Tools.SimpleNotificationService
Uninstalling module AWS.Tools.SQS
Uninstalling module AWS.Tools.Common
```

Note

Il `Install-AWSToolsModule` cmdlet scarica tutti i moduli richiesti dal file PSRepository denominato PSGallery (<https://www.powershellgallery.com/>) e considera il repository come una fonte attendibile. Usa il comando `Get-PSRepository -Name PSGallery` per ulteriori informazioni su PSRepository.

Il comando precedente installa i moduli nelle directory predefinite del sistema. Le directory effettive dipendono dalla distribuzione e dalla versione del sistema operativo e dalla versione installata. PowerShell Ad esempio, se avete installato PowerShell 7 su un sistema simile a RHEL, molto probabilmente i moduli predefiniti si trovano in `/opt/microsoft/powershell/7/Modules` (o `$PSHOME/Modules`) e i moduli utente si trovano molto probabilmente in `~/.local/share/powershell/Modules`. Per ulteriori informazioni, vedere [Installazione PowerShell su Linux](#) sul PowerShell sito Web Microsoft. Per vedere dove sono installati i moduli, esegui il comando seguente:

```
PS > Get-Module -ListAvailable
```

Per installare altri moduli, esegui comandi simili con i nomi dei moduli appropriati, come si trova nella [PowerShell Galleria](#).

Installa AWSPower Shell. NetCore su Linux o macOS

Per eseguire l'aggiornamento a una versione più recente di AWSPower Shell. NetCore, segui le istruzioni riportate in [Aggiornamento di AWS Strumenti per PowerShell su Linux o macOS](#). Disinstalla le versioni precedenti di AWSPower Shell. NetCore prima.

Puoi installare AWSPower Shell. NetCore in due modi:

- Scaricando il modulo da [AWSPowerShell.NetCore.zip](#) ed estraendolo in una delle directory del modulo. È possibile scoprire le directory dei moduli stampando il valore della variabile `$Env:PSModulePath`.
- Installazione dalla PowerShell Galleria utilizzando il `Install-Module` cmdlet come descritto nella procedura seguente.

Per installare Shell AWSPower. NetCore su Linux o macOS utilizzando il cmdlet `Install-Module`

Avviare una sessione PowerShell Core eseguendo il comando seguente.

```
$ pwsh
```

Note

Ti consigliamo di non iniziare PowerShell eseguendo `sudo pwsh` l'esecuzione PowerShell con diritti di amministratore elevati. Ciò è dovuto al potenziale rischio per la sicurezza ed è in contrasto con il principio del privilegio minimo.

Per installare la AWSPower Shell. NetCore pacchetto a modulo singolo dalla PowerShell Galleria, esegui il seguente comando.

```
PS > Install-Module -Name AWSPowerShell.NetCore
```

```
Untrusted repository
```

```
You are installing the modules from an untrusted repository. If you trust this repository, change its InstallationPolicy value by running the Set-PSRepository cmdlet. Are you sure
```

```
you want to install the modules from 'PSGallery'?
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): y
```

Se ricevi una notifica in cui è indicato che il repository non è attendibile, ti viene chiesto se vuoi installarlo comunque. Inserisci **y** per consentire PowerShell l'installazione del modulo. Per evitare il prompt senza dover considerare attendibile il repository, puoi eseguire il comando riportato qui di seguito.

```
PS > Install-Module -Name AWSPowerShell.NetCore -Force
```

Non è necessario eseguire questo comando come root, a meno che non si desideri installarlo AWS Strumenti per PowerShell per tutti gli utenti di un computer. Per fare ciò, esegui il comando seguente in una PowerShell sessione con cui hai iniziato `sudo pwsh`.

```
PS > Install-Module -Scope AllUsers -Name AWSPowerShell.NetCore -Force
```

Esecuzione di uno script

Il comando `Set-ExecutionPolicy` non è disponibile sui sistemi non Windows. Puoi eseguire `Get-ExecutionPolicy`, il che dimostra che l'impostazione predefinita dei criteri di esecuzione in PowerShell Core in esecuzione su sistemi non Windows è `Unrestricted`. Per ulteriori informazioni, consulta [Informazioni sulle policy di esecuzione](#) sul sito Web Microsoft Technet.

Poiché `PSModulePath` include la posizione della directory del AWS modulo, il `Get-Module -ListAvailable` cmdlet mostra il modulo installato.

AWS.Tools

```
PS > Get-Module -ListAvailable
```

Directory: /Users/*username*/.local/share/powershell/Modules

ModuleType	Version	Name	PSEdition	ExportedCommands
Binary	3.3.563.1	AWS.Tools.Common	Desk	{Clear-AWSHistory, Set-AWSHistoryConfiguration, Initialize-AWSDefaultConfiguration, Clear-AWSDefaultConfigurat...

AWSPowerShell.NetCore

```
PS > Get-Module -ListAvailable
```

Directory: /Users/*username*/.local/share/powershell/Modules

ModuleType	Version	Name	ExportedCommands
Binary	3.3.563.1	AWSPowerShell.NetCore	

Configurare una PowerShell console per utilizzare AWS Tools for PowerShell Core (AWSPowerShell. NetCore Solo)

PowerShell Core in genere carica automaticamente i moduli ogni volta che si esegue un cmdlet nel modulo. Ma questo non funziona per AWSPowerShell. NetCore a causa delle sue grandi dimensioni. Per iniziare a eseguire AWSPowerShell. NetCore cmdlet, è necessario innanzitutto eseguire il `Import-Module AWSPowerShell. NetCore` comando. Questo non è necessario per i cmdlet nei moduli `AWS.Tools`.

Inizializza la tua sessione PowerShell

Quando si avvia PowerShell su un sistema basato su Linux o macOS dopo aver installato il AWS Strumenti per PowerShell, è necessario eseguire [Initialize- AWSDefault Configuration](#) per specificare la chiave di accesso da utilizzare. AWS Per ulteriori informazioni su `Initialize- AWSDefaultConfiguration`, consulta [Utilizzo delle AWS credenziali](#).

Note

Nelle versioni precedenti (precedenti alla 3.3.96.0) di, questo cmdlet era denominato. `AWS Strumenti per PowerShellInitialize-AWSDefaults`

Controllo delle versioni

AWS rilascia AWS Strumenti per PowerShell periodicamente nuove versioni per supportare nuovi servizi e funzionalità. AWS Per determinare la versione di AWS Strumenti per PowerShell che è stata installata, eseguire il `AWSPowerShellVersion` cmdlet [Get-](#).

Per esempio:

```
PS > Get-AWSPowerShellVersion
```

```
AWS Tools for PowerShell
```

```
Version 4.1.849
```

```
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
Amazon Web Services SDK for .NET
```

```
Core Runtime Version 3.7.402.75
```

```
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

Release notes: <https://github.com/aws/aws-tools-for-powershell/blob/v4.1/changelogs/CHANGELOG.ALL.md>

This software includes third party software subject to the following copyrights:

- Logging from log4net, Apache License
[<http://logging.apache.org/log4net/license.html>]

Per visualizzare un elenco dei AWS servizi supportati nella versione corrente degli strumenti, aggiungere il `-ListServiceVersionInfo` parametro a un cmdlet [Get- AWSPower ShellVersion](#).

Per determinare la versione in esecuzione, immettere `$PSVersionTable` per visualizzare il contenuto della variabile `$PSVersionTable` [automatica](#). PowerShell

Per esempio:

```
PS > $PSVersionTable
Name                           Value
----                           -
PSVersion                      6.2.2
PSEdition                      Core
GitCommitId                    6.2.2
OS                              Darwin 18.7.0 Darwin Kernel Version 18.7.0: Tue Aug 20
 16:57:14 PDT 2019; root:xnu-4903.271.2~2/RELEASE_X86_64
Platform                       Unix
PSCompatibleVersions           {1.0, 2.0, 3.0, 4.0...}
PSRemotingProtocolVersion      2.3
SerializationVersion           1.1.0.1
WSManStackVersion              3.0
```

Aggiornamento di AWS Strumenti per PowerShell su Linux o macOS

Periodicamente, man mano che AWS Strumenti per PowerShell vengono rilasciate versioni aggiornate di, è necessario aggiornare la versione in esecuzione localmente.

Aggiorna i moduli modularizzati **AWS.Tools**

Per aggiornare i **AWS.Tools** moduli alla versione più recente, esegui il seguente comando:

```
PS > Update-AWSToolsModule -CleanUp
```

Questo comando aggiorna tutti i moduli `AWS.Tools` attualmente installati e, per i moduli che sono stati aggiornati correttamente, rimuove le versioni precedenti.

Note

Il `Update-AWSToolsModule` cmdlet scarica tutti i moduli dal file `PSRepository` denominato `PSGallery` (<https://www.powershellgallery.com/>) e lo considera una fonte attendibile. Usa il comando `Get-PSRepository -Name PSGallery` per ulteriori informazioni su `PSRepository`.

Aggiorna gli strumenti per Core PowerShell

Esegui il `Get-AWSPowerShellVersion` cmdlet per determinare la versione in esecuzione e confrontala con la versione di Tools for Windows PowerShell disponibile sul sito Web [PowerShell Gallery](https://www.powershellgallery.com/). Consigliamo di controllare ogni due o tre settimane. Il supporto per nuovi comandi e AWS servizi è disponibile solo dopo l'aggiornamento a una versione con tale supporto.

Prima di installare una versione più recente di `AWSPowerShell.NetCore`, disinstalla il modulo esistente. Chiudi tutte PowerShell le sessioni aperte prima di disinstallare il pacchetto esistente. Per disinstallare il pacchetto, eseguire il comando seguente.

```
PS > Uninstall-Module -Name AWSPowerShell.NetCore -AllVersions
```

Dopo aver disinstallato il pacchetto, installa il modulo aggiornato eseguendo il comando riportato qui di seguito.

```
PS > Install-Module -Name AWSPowerShell.NetCore
```

Dopo l'installazione, esegui il comando `Import-Module AWSPowerShell.NetCore` per caricare i cmdlet aggiornati nella sessione PowerShell.

Informazioni correlate

- [Inizia con AWS Tools for Windows PowerShell](#)
- [Lavora con AWS i servizi in AWS Strumenti per PowerShell](#)

Migrazione dalla AWS Strumenti per PowerShell versione 3.3 alla versione 4

AWS Strumenti per PowerShell la versione 4 è un aggiornamento retrocompatibile alla versione 3.3. AWS Strumenti per PowerShell Aggiunge miglioramenti significativi pur mantenendo il comportamento dei cmdlet esistenti.

Gli script esistenti dovrebbero continuare a funzionare dopo l'aggiornamento alla nuova versione, ma si consiglia di verificarli accuratamente prima di eseguire l'aggiornamento agli ambienti di produzione.

Questa sezione descrive le modifiche e spiega come potrebbero influire sugli script.

Nuova versione **AWS.Tools** completamente modularizzata

La Shell. AWSPower NetCore e i pacchetti AWSPower Shell erano «monolitici». Ciò significava che tutti i AWS servizi erano supportati nello stesso modulo, il che lo rendeva molto grande e cresceva man mano che venivano aggiunti nuovi AWS servizi e funzionalità. Il nuovo **AWS.Tools** pacchetto è suddiviso in moduli più piccoli che offrono la flessibilità di scaricare e installare solo quelli necessari per i AWS servizi utilizzati. Il pacchetto include un modulo **AWS.Tools.Common** condiviso richiesto da tutti gli altri moduli e un modulo **AWS.Tools.Installer** che semplifica l'installazione, l'aggiornamento e la rimozione dei moduli secondo necessità.

Ciò consente anche l'importazione automatica dei cmdlet alla prima chiamata, senza dover effettuare la prima chiamata **Import-Module**. Tuttavia, per interagire con gli oggetti.NET associati prima di chiamare un cmdlet, è comunque necessario chiamare **Import-Module** per comunicare i PowerShell tipi.NET pertinenti.

Ad esempio, il comando seguente ha un riferimento a **Amazon.EC2.Model.Filter**. Questo tipo di riferimento non può attivare l'importazione automatica, quindi è necessario chiamare prima **Import-Module** o il comando fallisce.

```
PS > $filter = [Amazon.EC2.Model.Filter]@{Name="vpc-id";Values="vpc-1234abcd"}
InvalidOperation: Unable to find type [Amazon.EC2.Model.Filter].
```

```
PS > Import-Module AWS.Tools.EC2
PS > $filter = [Amazon.EC2.Model.Filter]@{Name="vpc-id";Values="vpc-1234abcd"}
PS > Get-EC2Instance -Filter $filter -Select Reservations.Instances.InstanceId
i-0123456789abcdefg
i-0123456789hijklmn
```

Nuovo cmdlet **Get-AWSService**

Per facilitare l'individuazione dei nomi dei moduli per ogni AWS servizio nell'AWS.Tools insieme di moduli, è possibile utilizzare il Get-AWSService cmdlet.

```
PS > Get-AWSService
Service : ACMPCA
CmdletNounPrefix : PCA
ModuleName : AWS.Tools.ACMPCA
SDKAssemblyVersion : 3.3.101.56
ServiceName : Certificate Manager Private Certificate Authority

Service : AlexaForBusiness
CmdletNounPrefix : ALXB
ModuleName : AWS.Tools.AlexaForBusiness
SDKAssemblyVersion : 3.3.106.26
ServiceName : Alexa For Business
...
```

Nuovo parametro **-Select** per controllare l'oggetto restituito da un cmdlet

La maggior parte dei cmdlet nella versione 4 supporta un nuovo parametro -Select. Ogni cmdlet chiama il AWS servizio automaticamente utilizzando APIs . AWS SDK per .NET Il AWS Strumenti per PowerShell client converte quindi la risposta in un oggetto che è possibile utilizzare negli PowerShell script e reindirizzare ad altri comandi. A volte l' PowerShell oggetto finale ha più campi o proprietà nella risposta originale del necessario e altre volte potresti volere che l'oggetto includa campi o proprietà della risposta che non sono presenti per impostazione predefinita. Il parametro -Select consente di specificare gli elementi inclusi nell'oggetto.NET restituito dal cmdlet.

Ad esempio, il [Get-S3Object](#) cmdlet richiama l'operazione SDK Amazon S3. [ListObjects](#) Tale operazione restituisce un oggetto. [ListObjectsResponse](#) Tuttavia, per impostazione predefinita, il Get-S3Object cmdlet restituisce all'utente solo l'S3Object elemento della risposta SDK. PowerShell Nell'esempio seguente, tale oggetto è un array con due elementi.

```
PS > Get-S3Object -BucketName amzn-s3-demo-bucket

ETag : "01234567890123456789012345678901111"
BucketName : amzn-s3-demo-bucket
Key : file1.txt
LastModified : 9/30/2019 1:31:40 PM
```

```

Owner : Amazon.S3.Model.Owner
Size : 568
StorageClass : STANDARD

ETag : "01234567890123456789012345678902222"
BucketName : amzn-s3-demo-bucket
Key : file2.txt
LastModified : 7/15/2019 9:36:54 AM
Owner : Amazon.S3.Model.Owner
Size : 392
StorageClass : STANDARD

```

Nella AWS Strumenti per PowerShell versione 4, è possibile specificare di `-Select *` restituire l'oggetto di risposta.NET completo restituito dalla chiamata API SDK.

```

PS > Get-S3Object -BucketName amzn-s3-demo-bucket -Select *
IsTruncated      : False
NextMarker       :
S3Objects        : {file1.txt, file2.txt}
Name             : amzn-s3-demo-bucket
Prefix           :
MaxKeys          : 1000
CommonPrefixes   : {}
Delimiter        :

```

È inoltre possibile specificare il percorso della proprietà nidificata specifica desiderata. L'esempio seguente restituisce solo la proprietà Key di ciascun elemento della matrice S3Objects.

```

PS > Get-S3Object -BucketName amzn-s3-demo-bucket -Select S3Objects.Key
file1.txt
file2.txt

```

In alcune situazioni può essere utile restituire un parametro cmdlet. Tale operazione può essere eseguita con `-Select ^ParameterName`. Questa funzione sostituisce il parametro `-PassThru`, che è ancora disponibile ma deprecato.

```

PS > Get-S3Object -BucketName amzn-s3-demo-bucket -Select S3Objects.Key |
>> Write-S3ObjectTagSet -Select ^Key -BucketName amzn-s3-demo-bucket -Tagging_TagSet
@{ Key='key'; Value='value'}
file1.txt
file2.txt

```


[L'argomento di riferimento](#) per ogni cmdlet consente di identificare se supporta il parametro `-Select`.

Limitazione più coerente del numero di elementi nell'output

Le versioni precedenti di AWS Strumenti per PowerShell consentivano di utilizzare il `-MaxItems` parametro per specificare il numero massimo di oggetti restituiti nell'output finale.

Questo comportamento è stato rimosso da `AWS.Tools`.

Questo comportamento è obsoleto in Shell. `AWSPower NetCore` e `AWSPower Shell`, e verranno rimossi da tali versioni in una versione futura.

Se l'API del servizio sottostante supporta un parametro `MaxItems`, è ancora disponibile e funziona come specificato dall'API. Ma non ha più il comportamento ulteriore di limitare il numero di elementi restituiti nell'output del cmdlet.

Per limitare il numero di elementi restituiti nell'output finale, reindirizzate l'output al `Select-Object` cmdlet e specificate il `-First n` parametro, dove *n* è il numero massimo di elementi da includere nell'output finale.

```
PS > Get-S3ObjectV2 -BucketName amzn-s3-demo-bucket -Select S3Objects.Key | select -first 2
file1.txt
file2.txt
```

Non tutti i AWS servizi sono supportati `-MaxItems` allo stesso modo, quindi questo elimina l'incoerenza e i risultati imprevisti che a volte si verificavano. Inoltre, `-MaxItems` in combinazione con il nuovo parametro [-Select](#) a volte potrebbe portare a risultati confusi.

Parametri di flusso più facili da usare

Parametri di tipo `Stream` o `byte[]` possono ora accettare valori `string`, `string[]` o `FileInfo`.

Ad esempio, è possibile utilizzare uno dei seguenti esempi.

```
PS > Invoke-LMFunction -FunctionName MyTestFunction -PayloadStream '{
>> "some": "json"
>> }'
```

```
PS > Invoke-LMFunction -FunctionName MyTestFunction -PayloadStream (ls .\some.json)
```

```
PS > Invoke-LMFunction -FunctionName MyTestFunction -PayloadStream @('{', '"some":  
"json"', '}'')
```

AWS Strumenti per PowerShell converte tutte le stringhe `byte[]` utilizzando la codifica UTF-8.

Estendere la pipe in base al nome della proprietà

Per rendere l'esperienza utente più coerente, è ora possibile passare l'input della pipeline specificando il nome della proprietà per qualsiasi parametro.

Nell'esempio seguente viene creato un oggetto personalizzato con proprietà con nomi corrispondenti ai nomi dei parametri del cmdlet di destinazione. Quando viene eseguito il cmdlet, vengono automaticamente consumate tali proprietà come parametri.

```
PS > [pscustomobject] @{ BucketName='amzn-s3-demo-bucket'; Key='file1.txt';  
PartNumber=1 } | Get-S3ObjectMetadata
```

Note

Alcune proprietà lo supportavano nelle versioni precedenti di AWS Strumenti per PowerShell. La versione 4 rende questo più coerente abilitandolo per tutti i parametri.

Parametri comuni statici

Per migliorare la coerenza nella versione 4.0 di AWS Strumenti per PowerShell, tutti i parametri sono statici.

Nelle versioni precedenti di AWS Strumenti per PowerShell, alcuni parametri comuni come `AccessKey`, o `SecretKeyProfileName`, erano [dinamici Region](#), mentre tutti gli altri parametri erano statici. Ciò potrebbe creare problemi perché PowerShell associa i parametri statici prima di quelli dinamici. Ad esempio, supponiamo che tu abbia eseguito il seguente comando.

```
PS > Get-EC2Region -Region us-west-2
```

Le versioni precedenti di PowerShell associavano il valore `us-west-2` al parametro `-RegionName` statico anziché al parametro `-Region` dinamico. Probabilmente, questo potrebbe confondere gli utenti.

AWS.Tools Dichiarare e applica i parametri obbligatori

I moduli `AWS.Tools.*` ora dichiarano e applicano parametri cmdlet obbligatori. Quando un AWS servizio dichiara che è necessario un parametro di un'API, PowerShell richiede il parametro del cmdlet corrispondente se non lo si è specificato. Questo vale solo per `AWS.Tools`. Per garantire la compatibilità con le versioni precedenti, ciò non si applica a `Shell`, `AWSPowerNetCore` o `AWSPowerShell`.

Tutti i parametri sono “nullable”

È ora possibile assegnare `$null` ai parametri del tipo di valore (numeri e date). Questa modifica non dovrebbe influire sugli script esistenti. Ciò consente di ignorare la richiesta di un parametro obbligatorio. I parametri obbligatori sono applicati solo in `AWS.Tools`.

Se viene eseguito l'esempio seguente utilizzando la versione 4, il medesimo ignora effettivamente la convalida lato client perché si fornisce un «valore» per ogni parametro obbligatorio. Tuttavia, la chiamata al servizio Amazon EC2 API non riesce perché il AWS servizio richiede ancora tali informazioni.

```
PS > Get-EC2InstanceAttribute -InstanceId $null -Attribute $null
WARNING: You are passing $null as a value for parameter Attribute which is marked as
required.
In case you believe this parameter was incorrectly marked as required, report this by
opening
an issue at https://github.com/aws/aws-tools-for-powershell/issues.
WARNING: You are passing $null as a value for parameter InstanceId which is marked as
required.
In case you believe this parameter was incorrectly marked as required, report this by
opening
an issue at https://github.com/aws/aws-tools-for-powershell/issues.

Get-EC2InstanceAttribute : The request must contain the parameter instanceId
```

Rimozione di funzionalità precedentemente deprecate

Le seguenti funzionalità erano obsolete nelle versioni precedenti di AWS Strumenti per PowerShell e sono state rimosse nella versione 4:

- Rimosso il parametro `-Terminate` dal cmdlet `Stop-EC2Instance`. Usare invece `Remove-EC2Instance`.

- È stato rimosso il `-ProfileName` parametro dal cmdlet `Clear-AWSCredential`. Usare invece `Remove-AWSCredentialProfile`.
- Rimossi i cmdlet `Import-EC2Instance` e `Import-EC2Volume`.

Inizia con AWS Tools for Windows PowerShell

In alcuni argomenti di questa sezione vengono descritti i fondamenti dell'utilizzo degli strumenti per Windows PowerShell dopo l'[installazione degli](#) strumenti. Ad esempio, spiegano come specificare le [credenziali](#) e l'[AWS area](#) con cui gli Strumenti per Windows PowerShell devono utilizzare per interagire. AWS

Gli altri argomenti di questa sezione forniscono informazioni sui modi avanzati di configurazione degli strumenti, dell'ambiente e dei progetti.

Argomenti

- [Configura l'autenticazione dello strumento con AWS](#)
- [Specificare AWS le regioni](#)
- [Configurare l'identità federata con AWS Strumenti per PowerShell](#)
- [Individuazione di cmdlet e alias](#)
- [Pipeline, output e iterazione in AWS Strumenti per PowerShell](#)
- [Risoluzione di credenziali e profili](#)
- [Ulteriori informazioni su utenti e ruoli](#)
- [Uso delle credenziali legacy](#)

Configura l'autenticazione dello strumento con AWS

È necessario stabilire in che modo il codice si autentica AWS durante lo sviluppo con. Servizi AWS Esistono diversi modi in cui è possibile configurare l'accesso programmatico alle AWS risorse, a seconda dell'ambiente e dell' AWS accesso a disposizione.

Per visualizzare i vari metodi di autenticazione per Tools for PowerShell, consulta [Autenticazione e accesso](#) nella Guida di riferimento agli strumenti AWS SDKs e agli strumenti.

In questo argomento si presuppone che un nuovo utente stia sviluppando localmente, non abbia ricevuto un metodo di autenticazione dal datore di lavoro e che lo utilizzerà AWS IAM Identity Center per ottenere credenziali temporanee. Se l'ambiente in uso non rientra in questi presupposti, alcune delle informazioni contenute in questo argomento potrebbero non riguardarti l'utente o potrebbero esserti già state fornite.

La configurazione di questo ambiente richiede diversi passaggi, riassunti di seguito:

1. [Abilitazione e configurazione di IAM Identity Center](#)
2. [Configura gli strumenti per PowerShell utilizzare IAM Identity Center.](#)
3. [Avviare una sessione del portale di AWS accesso](#)

Abilitazione e configurazione di IAM Identity Center

Per utilizzarlo AWS IAM Identity Center, deve prima essere abilitato e configurato. Per maggiori dettagli su come eseguire questa operazione PowerShell, consulta la Fase 1 dell'argomento relativo all'[autenticazione di IAM Identity Center](#) nella AWS SDKs and Tools Reference Guide. In particolare, segui le istruzioni necessarie riportate nella sezione I do not have established access through IAM Identity Center (Non ho stabilito l'accesso tramite IAM Identity Center).

Configura gli strumenti per PowerShell utilizzare IAM Identity Center.

Note

A partire dalla versione 4.1.538 di Tools for PowerShell, il metodo consigliato per configurare le credenziali SSO e avviare una sessione del portale di AWS accesso consiste nell'utilizzare i [Invoke-AWSSSOLogin](#) cmdlet [Initialize-AWSSSOConfiguration](#) and, come descritto in questo argomento. Se non si ha accesso a quella versione degli strumenti per PowerShell (o successiva) o non è possibile utilizzare tali cmdlet, è comunque possibile eseguire queste attività utilizzando il. AWS CLI Per scoprire come, vedi. [Usa il AWS CLI per accedere al portale](#)

La procedura seguente aggiorna il AWS config file condiviso con le informazioni SSO che Tools for PowerShell utilizza per ottenere credenziali temporanee. Come conseguenza di questa procedura, viene avviata anche una sessione del portale di AWS accesso. Se il config file condiviso contiene già informazioni SSO e desideri semplicemente sapere come avviare una sessione del portale di accesso utilizzando gli Strumenti per PowerShell, consulta la sezione successiva di questo argomento, [Avviare una sessione del portale di AWS accesso](#).

1. Se non l'hai ancora fatto, apri PowerShell e installa il file appropriato per AWS Strumenti per PowerShell il tuo sistema operativo e ambiente, inclusi i cmdlet comuni. Per informazioni su come eseguire questa attività, consultare [Installazione del AWS Strumenti per PowerShell](#).

Ad esempio, se installi la versione modulare di Tools for PowerShell on Windows, molto probabilmente eseguirai comandi simili ai seguenti:

```
Install-Module -Name AWS.Tools.Installer
Install-AWSToolsModule AWS.Tools.Common
```

2. Esegui il comando seguente. Sostituisci i valori delle proprietà di esempio con i valori della configurazione di IAM Identity Center. Per informazioni su queste proprietà e su come trovarle, consulta le [impostazioni del provider di credenziali IAM Identity Center](#) nella AWS SDKs and Tools Reference Guide.

```
$params = @{
  ProfileName = 'my-sso-profile'
  AccountId = '111122223333'
  RoleName = 'SamplePermissionSet'
  SessionName = 'my-sso-session'
  StartUrl = 'https://provided-domain.awsapps.com/start'
  SSORegion = 'us-west-2'
  RegistrationScopes = 'sso:account:access'
};
Initialize-AWSSSOConfiguration @params
```

In alternativa, è possibile utilizzare semplicemente il cmdlet da solo e la Initialize-AWSSSOConfiguration finestra Tools for PowerShell richiede i valori delle proprietà.

Considerazioni per determinati valori di proprietà:

- Se hai semplicemente seguito le istruzioni per [abilitare e configurare IAM Identity Center](#), il valore per -RoleName potrebbe essere PowerUserAccess. Ma se hai creato un set di autorizzazioni IAM Identity Center appositamente per PowerShell il lavoro, utilizzalo invece.
 - Assicurati di utilizzare il Regione AWS punto in cui hai configurato IAM Identity Center.
3. A questo punto, il AWS config file condiviso contiene un profilo chiamato my-sso-profile con un set di valori di configurazione a cui è possibile fare riferimento negli Strumenti per PowerShell. Per trovare la posizione di questo file, consulta [Posizione dei file condivisi nella Guida di riferimento agli strumenti AWS SDKs e strumenti](#).

The Tools for PowerShell utilizza il provider di token SSO del profilo per acquisire le credenziali prima di inviare richieste a. AWS Il sso_role_name valore, che è un ruolo IAM connesso a

un set di autorizzazioni IAM Identity Center, dovrebbe consentire l'accesso ai dati Servizi AWS utilizzati nell'applicazione.

L'esempio seguente mostra il profilo che è stato creato utilizzando il comando mostrato sopra. Alcuni valori delle proprietà e il relativo ordine potrebbero essere diversi nel profilo attuale. La `sso-session` proprietà del profilo si riferisce alla sezione denominata `my-sso-session`, che contiene le impostazioni per avviare una sessione del portale di AWS accesso.

```
[profile my-sso-profile]
sso_account_id=111122223333
sso_role_name=SamplePermissionSet
sso_session=my-sso-session

[sso-session my-sso-session]
sso_region=us-west-2
sso_registration_scopes=sso:account:access
sso_start_url=https://provided-domain.awsapps.com/start/
```

4. Se hai già una sessione attiva del portale di AWS accesso, la sezione Strumenti ti PowerShell informa che hai già effettuato l'accesso.

In caso contrario, la sezione Strumenti PowerShell tenta di aprire automaticamente la pagina di autorizzazione SSO nel browser Web predefinito. Segui le istruzioni del tuo browser, che potrebbero includere un codice di autorizzazione SSO, nome utente e password e l'autorizzazione ad accedere agli AWS IAM Identity Center account e ai set di autorizzazioni.

Gli strumenti per PowerShell informarti che l'accesso SSO è avvenuto con successo.

Avviare una sessione del portale di AWS accesso

Prima di eseguire i comandi di accesso Servizi AWS, è necessaria una sessione attiva del portale di AWS accesso in modo che Tools for PowerShell possa utilizzare l'autenticazione IAM Identity Center per risolvere le credenziali. Per accedere al portale di AWS accesso, esegui il seguente comando in PowerShell, `-ProfileName my-sso-profile` dov'è il nome del profilo che è stato creato nel config file condiviso quando hai seguito la procedura nella sezione precedente di questo argomento.

```
Invoke-AWSSSOLogin -ProfileName my-sso-profile
```


Se hai già una sessione attiva del portale di AWS accesso, il menu Strumenti ti PowerShell informa che hai già effettuato l'accesso.

In caso contrario, la sezione Strumenti PowerShell tenta di aprire automaticamente la pagina di autorizzazione SSO nel browser Web predefinito. Segui le istruzioni del tuo browser, che potrebbero includere un codice di autorizzazione SSO, nome utente e password e l'autorizzazione ad accedere agli AWS IAM Identity Center account e ai set di autorizzazioni.

Gli strumenti per PowerShell informarti che l'accesso SSO è avvenuto con successo.

Per verificare se hai già una sessione attiva, esegui il comando seguente dopo aver installato o importato il `AWS.Tools.SecurityToken` modulo, se necessario.

```
Get-STSCallerIdentity -ProfileName my-sso-profile
```

La risposta al `Get-STSCallerIdentity` cmdlet riporta l'account IAM Identity Center e il set di autorizzazioni configurati nel file condiviso. `config`

Esempio

Di seguito è riportato un esempio di come utilizzare IAM Identity Center con gli strumenti per PowerShell. Si presume quanto segue:

- Hai abilitato IAM Identity Center e lo hai configurato come descritto in questo argomento. Le proprietà SSO si trovano nel `my-sso-profile` profilo, che è stato configurato in precedenza in questo argomento.
- Quando accedi tramite i `Invoke-AWSSSOLogin` cmdlet `Initialize-AWSSSOConfiguration` or, l'utente dispone almeno delle autorizzazioni di sola lettura per Amazon S3.
- Alcuni bucket S3 sono disponibili per la visualizzazione da parte dell'utente.

Installa o importa il `AWS.Tools.S3` modulo secondo necessità, quindi utilizza il PowerShell comando seguente per visualizzare un elenco dei bucket S3.

```
Get-S3Bucket -ProfileName my-sso-profile
```

Informazioni aggiuntive

- Per ulteriori opzioni sull'autenticazione per gli strumenti per PowerShell, come l'uso di profili e variabili di ambiente, consulta il capitolo sulla [configurazione](#) nella Guida di riferimento agli strumenti AWS SDKs e agli strumenti.
- Alcuni comandi richiedono la specificazione di una AWS regione. Esistono diversi modi per farlo, tra cui l'opzione del `-Region` cmdlet, il `[default]` profilo e la variabile di `AWS_REGION` ambiente. Per ulteriori informazioni, vedere [Specificare AWS le regioni](#) in questa guida e [AWS Region nella AWS SDKs and Tools Reference Guide](#).
- Per ulteriori informazioni sulle best practice, consulta [Best practice per la sicurezza in IAM](#) nella Guida per l'utente di IAM.
- Per creare AWS credenziali a breve termine, consulta [Temporary Security Credentials](#) nella IAM User Guide.
- Per ulteriori informazioni su altri fornitori di credenziali, consulta Provider di [credenziali standardizzati nella and Tools Reference](#) Guide AWS SDKs .

Argomenti

- [Usa il AWS CLI per accedere al portale](#)

Usa il AWS CLI per accedere al portale

A partire dalla versione 4.1.538 di Tools for PowerShell, il metodo consigliato per configurare le credenziali SSO e avviare una sessione del portale di AWS accesso consiste nell'utilizzare i [Invoke-AWSSSOLogin](#) cmdlet [Initialize-AWSSSOConfiguration](#) and, come descritto in. [Configura l'autenticazione dello strumento con AWS](#) Se non si ha accesso a quella versione di Tools for PowerShell (o successiva) o non è possibile utilizzare tali cmdlet, è comunque possibile eseguire queste attività utilizzando il. AWS CLI

Configura gli strumenti per PowerShell utilizzare IAM Identity Center tramite. AWS CLI

Se non l'hai già fatto, assicurati di [abilitare e configurare IAM Identity Center](#) prima di procedere.

Le informazioni su come configurare gli strumenti PowerShell per utilizzare IAM Identity Center tramite IAM sono riportate nella Fase 2 dell'argomento relativo all'[autenticazione di IAM Identity Center](#) nella Guida di riferimento agli strumenti AWS SDKs e agli strumenti. AWS CLI Dopo aver completato questa configurazione, il sistema deve contenere i seguenti elementi:

- Il AWS CLI, che viene utilizzato per avviare una sessione del portale di AWS accesso prima di eseguire l'applicazione.
- Il AWS config file condiviso che contiene un [\[default\]profilo](#) con un set di valori di configurazione a cui è possibile fare riferimento negli Strumenti per PowerShell. Per trovare la posizione di questo file, consulta [Posizione dei file condivisi nella](#) Guida di riferimento agli strumenti AWS SDKs e strumenti. The Tools for PowerShell utilizza il provider di token SSO del profilo per acquisire le credenziali prima di inviare richieste a. AWS Il sso_role_name valore, che è un ruolo IAM connesso a un set di autorizzazioni IAM Identity Center, dovrebbe consentire l'accesso ai dati Servizi AWS utilizzati nell'applicazione.

Il seguente config file di esempio mostra un [default] profilo configurato con un provider di token SSO. L'impostazione sso_session del profilo si riferisce alla sezione sso-session denominata. La sso-session sezione contiene le impostazioni per avviare una sessione del portale di AWS accesso.

```
[default]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole
region = us-east-1
output = json

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://provided-domain.awsapps.com/start
sso_registration_scopes = sso:account:access
```

Important

La PowerShell sessione deve avere i seguenti moduli installati e importati in modo che la risoluzione SSO possa funzionare:

- AWS.Tools.SSO
- AWS.Tools.SSOIDC

Se utilizzi una versione precedente di Tools for PowerShell e non disponi di questi moduli, riceverai un errore simile al seguente: «Assembly AWSSDK.SSOIDC could not be found...».

Avvia una sessione del portale di accesso AWS

Prima di eseguire i comandi di accesso Servizi AWS, è necessaria una sessione attiva del portale di AWS accesso in modo che Tools for Windows PowerShell possa utilizzare l'autenticazione IAM Identity Center per risolvere le credenziali. A seconda della durata della sessione configurata, l'accesso alla fine scadrà e gli Strumenti per Windows PowerShell riscontreranno un errore di autenticazione. Per accedere al portale di AWS accesso, esegui il seguente comando in AWS CLI

```
aws sso login
```

Poiché si utilizza il [default] profilo, non è necessario richiamare il comando con l'--profileopzione. Se la configurazione del provider di token SSO utilizza un profilo denominato, il comando è `aws sso login --profile named-profile` invece. Per ulteriori informazioni sui profili denominati, consulta la sezione [Profili](#) nella Guida di riferimento agli strumenti AWS SDKs e agli strumenti.

Per verificare se hai già una sessione attiva, esegui il AWS CLI comando seguente (con la stessa considerazione per named profile):

```
aws sts get-caller-identity
```

La risposta a questo comando dovrebbe restituire l'account IAM Identity Center e il set di autorizzazioni configurati nel file config condiviso.

Note

Se hai già una sessione attiva del portale di AWS accesso ed esegui `aws sso login`, non ti verrà richiesto di fornire credenziali.

La procedura di accesso potrebbe richiedere all'utente di consentire l'AWS CLI accesso ai dati. Poiché AWS CLI è basato sull'SDK per Python, i messaggi di autorizzazione possono contenere variazioni del botocore nome.

Esempio

Di seguito è riportato un esempio di come utilizzare IAM Identity Center con gli strumenti per PowerShell. Si presume quanto segue:

- Hai abilitato IAM Identity Center e lo hai configurato come descritto in questo argomento. Le proprietà SSO si trovano nel profilo [default].
- Quando accedi tramite AWS CLI by using `aws sso login`, quell'utente dispone almeno delle autorizzazioni di sola lettura per Amazon S3.
- Alcuni bucket S3 sono disponibili per la visualizzazione da parte dell'utente.

Usa i seguenti PowerShell comandi per visualizzare un elenco dei bucket S3:

```
Install-Module AWS.Tools.Installer
Install-AWSToolsModule S3
# And if using an older version of the AWS Tools for PowerShell:
Install-AWSToolsModule SSO, SSO0IDC

# In older versions of the AWS Tools for PowerShell, we're not invoking a cmdlet from
these modules directly,
# so we must import them explicitly:
Import-Module AWS.Tools.SSO
Import-Module AWS.Tools.SSO0IDC

# Older versions of the AWS Tools for PowerShell don't support the SSO login flow, so
login with the CLI
aws sso login

# Now we can invoke cmdlets using the SSO profile
Get-S3Bucket
```

Come accennato in precedenza, poiché si utilizza il [default] profilo, non è necessario chiamare il `Get-S3Bucket` cmdlet con l'opzione `-ProfileName`. Se la configurazione del provider di token SSO utilizza un profilo denominato, il comando è `Get-S3Bucket -ProfileName named-profile`. Per ulteriori informazioni sui profili denominati, vedere la sezione [Profili](#) nella Guida di riferimento agli strumenti AWS SDKs e agli strumenti.

Informazioni aggiuntive

- Per ulteriori opzioni sull'autenticazione per gli Strumenti per PowerShell, come l'uso di profili e variabili di ambiente, consultate il capitolo sulla [configurazione](#) nella Guida di riferimento agli strumenti AWS SDKs e agli strumenti.
- Alcuni comandi richiedono la specificazione di una AWS regione. Esistono diversi modi per farlo, tra cui l'opzione del `-Region` cmdlet, il `[default]` profilo e la variabile di `AWS_REGION` ambiente. Per ulteriori informazioni, vedere [Specificare AWS le regioni](#) in questa guida e [AWS Region nella AWS SDKs and Tools Reference Guide](#).
- Per ulteriori informazioni sulle best practice, consulta [Best practice per la sicurezza in IAM](#) nella Guida per l'utente di IAM.
- Per creare AWS credenziali a breve termine, consulta [Temporary Security Credentials](#) nella IAM User Guide.
- Per ulteriori informazioni su altri fornitori di credenziali, consulta Provider di [credenziali standardizzati nella and Tools Reference Guide AWS SDKs](#).

Specificare AWS le regioni

Esistono due modi per specificare la AWS regione da utilizzare durante l'esecuzione dei AWS Strumenti per PowerShell comandi:

- Utilizza il parametro `-Region` comune sui singoli comandi.
- Utilizza il comando `Set-DefaultAWSRegion` per impostare una regione predefinita per tutti i comandi.

Molti AWS cmdlet hanno esito negativo se gli Strumenti per Windows non PowerShell riescono a capire quale regione utilizzare. Le eccezioni includono i cmdlet per Amazon S3, [Amazon](#) SES AWS Identity and Access Management e, che utilizzano automaticamente per impostazione predefinita un endpoint globale.

Per specificare la regione per un singolo comando AWS

Aggiungi il parametro `-Region` al comando, come il seguente.

```
PS > Get-EC2Image -Region us-west-2
```

Per impostare una regione predefinita per tutti i comandi AWS CLI nella sessione corrente

Dal PowerShell prompt dei comandi, digitare il comando seguente.

```
PS > Set-DefaultAWSRegion -Region us-west-2
```

Note

Questa impostazione persiste solo nella sessione corrente. Per applicare l'impostazione a tutte le PowerShell sessioni, aggiungi questo comando al tuo PowerShell profilo come hai fatto per il `Import-Module` comando.

Per visualizzare l'area predefinita corrente per tutti i comandi AWS CLI

Dal PowerShell prompt dei comandi, digitare il seguente comando.

```
PS > Get-DefaultAWSRegion
```

Region	Name	IsShellDefault
-----	----	-----
us-west-2	US West (Oregon)	True

Per cancellare la regione predefinita corrente per tutti i comandi AWS CLI

Dal PowerShell prompt dei comandi, digitare il seguente comando.

```
PS > Clear-DefaultAWSRegion
```

Per visualizzare un elenco di tutte le regioni disponibili AWS

Dal PowerShell prompt dei comandi, digitare il comando seguente. La terza colonna nell'output di esempio identifica la Regione predefinita per la sessione corrente.

```
PS > Get-AWSRegion
```

Region	Name	IsShellDefault
-----	----	-----
ap-east-1	Asia Pacific (Hong Kong)	False
ap-northeast-1	Asia Pacific (Tokyo)	False
...		

us-east-2	US East (Ohio)	False
us-west-1	US West (N. California)	False
us-west-2	US West (Oregon)	True
...		

Note

Alcune Regioni potrebbero essere supportate ma non incluse negli output del cmdlet `Get-AWSRegion`. Ad esempio, questo è talvolta vero per le Regioni che non sono ancora globali. Se non sei in grado di specificare una Regione aggiungendo il parametro `-Region` a un comando, prova invece a specificare la Regione in un endpoint personalizzato, come illustrato nella sezione successiva.

Specificare un endpoint personalizzato o non standard

Specificate un endpoint personalizzato come URL aggiungendo il parametro `-EndpointUrl` common al PowerShell comando Tools for Windows, nel seguente formato di esempio.

```
PS > Some-AWS-PowerShellCmdlet -EndpointUrl "custom endpoint URL" -Other -Parameters
```

L'esempio seguente utilizza il cmdlet `Get-EC2Instance`. L'endpoint personalizzato si trova in `us-west-2` o nella Regione Stati Uniti occidentali (Oregon) in questo esempio, ma puoi utilizzare qualsiasi altra Regione AWS supportata, incluse le Regioni che non sono enumerate da `Get-AWSRegion`.

```
PS > Get-EC2Instance -EndpointUrl "https://service-custom-url.us-west-2.amazonaws.com"
-InstanceId "i-0555a30a2000000e1"
```

Informazioni aggiuntive

Per ulteriori informazioni sulle AWS regioni, vedere [AWS Region nella AWS SDKs and Tools Reference Guide](#).

Configurare l'identità federata con AWS Strumenti per PowerShell

Per consentire agli utenti dell'organizzazione di accedere alle AWS risorse, è necessario configurare un metodo di autenticazione standard e ripetibile per scopi di sicurezza, verificabilità, conformità e

capacità di supportare la separazione di ruoli e account. Sebbene sia comune fornire agli utenti la possibilità di accedere AWS APIs, senza l'accesso alle API federate, è necessario creare anche utenti AWS Identity and Access Management (IAM), il che vanifica lo scopo dell'utilizzo della federazione. Questo argomento descrive il supporto SAML (Security Assertion Markup Language) AWS Strumenti per PowerShell che semplifica la soluzione di accesso federato.

Il supporto SAML AWS Strumenti per PowerShell consente di fornire agli utenti un accesso federato ai servizi. AWS SAML è un formato standard aperto basato su XML per la trasmissione dei dati di autenticazione e autorizzazione degli utenti tra i servizi, in particolare tra un provider di identità (come [Active Directory Federation Services](#)) e un fornitore di servizi (come). AWS Per ulteriori informazioni su SAML e su come funziona, consultare la pagina [SAML](#) su Wikipedia o [SAML Technical Specifications](#) sul sito del consorzio della Organization for the Advancement of Structured Information Standards (OASIS). Il supporto SAML in è compatibile con SAML 2.0 AWS Strumenti per PowerShell .

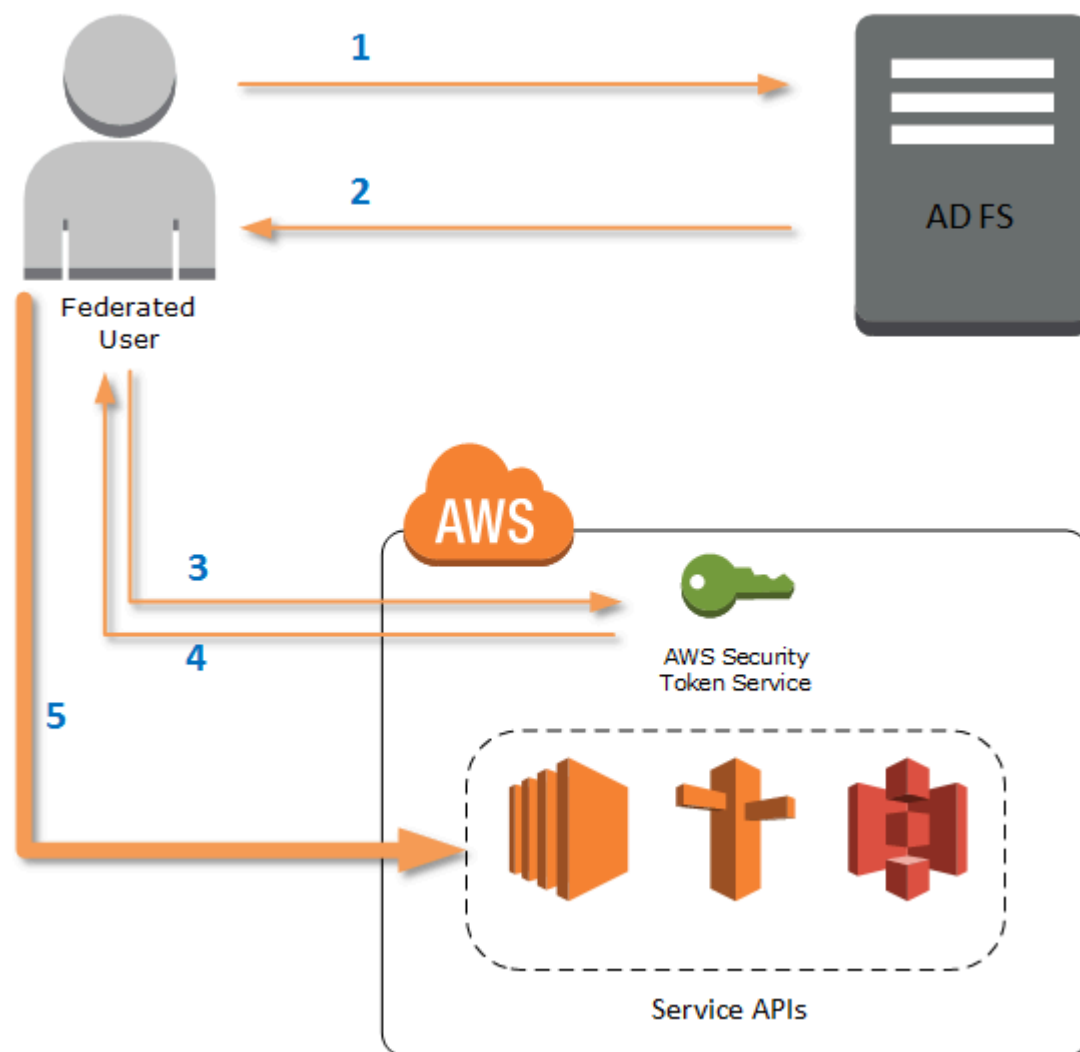
Prerequisiti

È necessario soddisfare i seguenti prerequisiti prima di provare a utilizzare il supporto a SAML per la prima volta.

- Una soluzione di identità federata correttamente integrata con l'account AWS per l'accesso alla console utilizzando solo le credenziali della propria organizzazione. Per ulteriori informazioni su come eseguire questa operazione, in particolare per Active Directory Federation Services, consulta [About SAML 2.0 Federation](#) nella IAM User Guide e il post del blog [Enabling Federation to AWS Using Windows Active Directory, AD FS e SAML 2.0](#). Anche se il blog post copre AD FS 2.0, le fasi sono simili se si esegue AD FS 3.0.
- Versione 3.1.31.0 o successiva di quella AWS Strumenti per PowerShell installata sulla workstation locale.

In che modo un utente con identità federata ottiene l'accesso federato al servizio AWS APIs

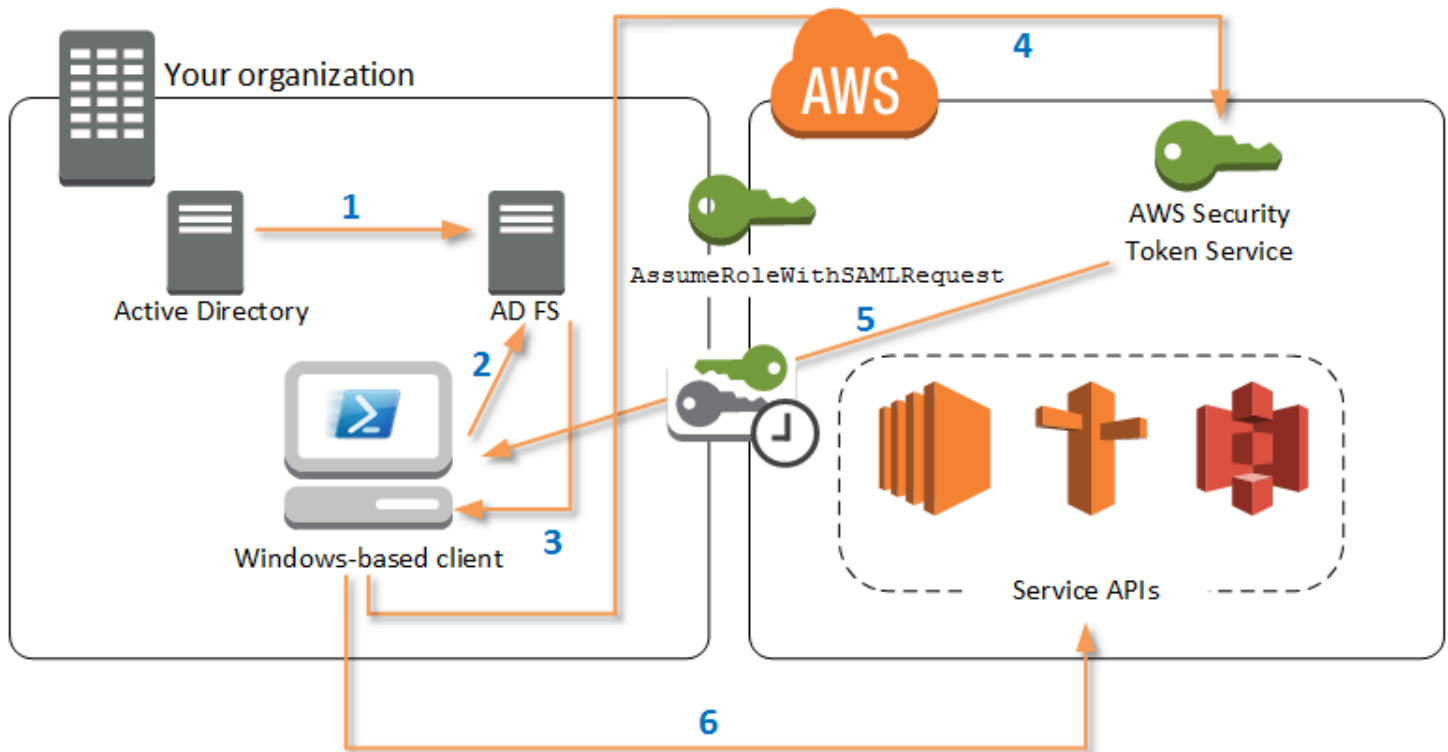
Il processo seguente descrive, a livello generale, come un utente di Active Directory (AD) viene federato da AD FS per accedere alle risorse. AWS



1. Il client nel computer dell'utente federato esegue l'autenticazione tramite AD FS.
2. Se l'autenticazione ha esito positivo, AD FS invia all'utente un'asserzione SAML.
3. Il client dell'utente invia l'asserzione SAML a AWS Security Token Service (STS) come parte di una richiesta di federazione SAML.
4. STS restituisce una risposta SAML che contiene credenziali AWS temporanee per un ruolo che l'utente può assumere.
5. L'utente accede al AWS servizio APIs includendo tali credenziali temporanee nella richiesta effettuata da. AWS Strumenti per PowerShell

Come funziona il supporto SAML nel AWS Strumenti per PowerShell

Questa sezione descrive come i AWS Strumenti per PowerShell cmdlet abilitano la configurazione della federazione delle identità basata su SAML per gli utenti.



1. AWS Strumenti per PowerShell esegue l'autenticazione con AD FS utilizzando le credenziali correnti dell'utente Windows o in modo interattivo, quando l'utente tenta di eseguire un cmdlet che richiede credenziali per la chiamata AWS.
2. AD FS autentica l'utente.
3. AD FS genera una risposta di autenticazione SAML 2.0 che include un'asserzione; lo scopo dell'asserzione è identificare e fornire informazioni sull'utente. AWS Strumenti per PowerShell estrae l'elenco dei ruoli autorizzati dell'utente dall'asserzione SAML.
4. AWS Strumenti per PowerShell inoltra la richiesta SAML, inclusi gli Amazon Resource Names (ARN) del ruolo richiesto, a STS effettuando la chiamata API `AssumeRoleWithSAMLRequest`.
5. Se la richiesta SAML è valida, STS restituirà una risposta che contiene i parametri `AccessKeyId`, `SecretAccessKey` e `SessionToken` di AWS. Queste credenziali dureranno per 3.600 secondi (1 ora).
6. L'utente ora dispone di credenziali valide per lavorare con qualsiasi AWS servizio a APIs cui il ruolo dell'utente è autorizzato ad accedere. AWS Strumenti per PowerShell applica

automaticamente queste credenziali per tutte le chiamate AWS API successive e le rinnova automaticamente quando scadono.

Note

Quando le credenziali scadono e sono necessarie nuove credenziali, gli AWS Strumenti per PowerShell eseguono automaticamente una nuova autenticazione con AD FS e ottengono nuove credenziali per l'ora successiva. Per gli account degli utenti aggiunti al dominio, questo processo viene eseguito automaticamente. Per gli account che non fanno parte di un dominio, AWS Strumenti per PowerShell richiede agli utenti di inserire le proprie credenziali prima di poter effettuare nuovamente l'autenticazione.

Come utilizzare i cmdlet di configurazione SAML PowerShell

AWS Strumenti per PowerShell include due nuovi cmdlet che forniscono il supporto SAML.

- `Set-AWSSamlEndpoint` configura gli endpoint AD FS, assegna un nome descrittivo per l'endpoint e, facoltativamente, descrive il tipo di autenticazione dell'endpoint.
- `Set-AWSSamlRoleProfile` crea o modifica un profilo di account utente che si desidera associare a un endpoint AD FS, identificato specificando il nome descrittivo fornito al cmdlet `Set-AWSSamlEndpoint`. Ogni ruolo del profilo viene mappato a un singolo ruolo che un utente è autorizzato a eseguire.

Proprio come con i profili di AWS credenziali, si assegna un nome descrittivo al profilo del ruolo. È possibile utilizzare lo stesso nome descrittivo con il `Set-AWSCredential` cmdlet o come valore del `-ProfileName` parametro per qualsiasi cmdlet che richiama il servizio AWS APIs

Aprire una nuova sessione. AWS Strumenti per PowerShell Se si esegue la PowerShell versione 3.0 o una versione successiva, il AWS Strumenti per PowerShell modulo viene importato automaticamente quando si esegue uno dei relativi cmdlet. Se si esegue la PowerShell versione 2.0, è necessario importare il modulo manualmente eseguendo il cmdlet `Import-Module`, come illustrato nell'esempio seguente.

```
PS > Import-Module "C:\Program Files (x86)\AWS Tools\PowerShell\AWSPowerShell\AWSPowerShell.psd1"
```

Come eseguire i cmdlet **Set-AWSSamlEndpoint** e **Set-AWSSamlRoleProfile**

1. In primo luogo, configurare le impostazioni degli endpoint per il sistema AD FS. Il modo più semplice per eseguire questa operazione è quello di archiviare l'endpoint in una variabile, come illustrato in questa fase. Assicurati di sostituire l'account segnaposto e il nome host AD FS con il tuo account IDs e il tuo nome host AD FS. IDs Specificare il nome dell'host dell'AD FS nel parametro Endpoint.

```
PS > $endpoint = "https://adfs.example.com/adfs/ls/IdpInitiatedSignOn.aspx?loginToRp=urn:amazon:webservices"
```

2. Per creare le impostazioni dell'endpoint, eseguire il cmdlet **Set-AWSSamlEndpoint** specificando il valore corretto per il parametro **AuthenticationType**. I valori validi includono Basic, Digest, Kerberos, Negotiate e NTLM. Se non viene specificato questo parametro, il valore predefinito è Kerberos.

```
PS > $sepName = Set-AWSSamlEndpoint -Endpoint $endpoint -StoreAs ADFS-Demo -AuthenticationType NTLM
```

Il cmdlet restituisce il nome descrittivo assegnato utilizzando il parametro **-StoreAs**, in modo che sia possibile utilizzarlo quando si esegue **Set-AWSSamlRoleProfile** nella riga successiva.

3. Ora, esegui il cmdlet **Set-AWSSamlRoleProfile** per effettuare l'autenticazione con il fornitore di identità AD FS e ottenere il set dei ruoli (nell'asserzione SAML) che l'utente è autorizzato a eseguire.

Il cmdlet **Set-AWSSamlRoleProfile** utilizza il set di ruoli restituito per richiedere all'utente di selezionare un ruolo per associarlo con il profilo specificato o convalidare i dati forniti del ruolo in parametri se presenti (in caso contrario all'utente viene richiesto di scegliere). Se l'utente è autorizzato per un solo ruolo, il cmdlet associa il ruolo con il profilo automaticamente, senza richiedere l'intervento dell'utente. Non è necessario fornire credenziali per configurare un profilo per l'utilizzo aggiunto al dominio.

```
PS > Set-AWSSamlRoleProfile -StoreAs SAMLDemoProfile -EndpointName $sepName
```

In alternativa, per non-domain-joined gli account, puoi fornire le credenziali di Active Directory e quindi selezionare un AWS ruolo a cui l'utente ha accesso, come illustrato nella riga seguente. Questa funzione è utile se si dispone di diversi account utente Active Directory per distinguere i ruoli all'interno della propria organizzazione (ad esempio per le funzioni di amministrazione).

```
PS > $credential = Get-Credential -Message "Enter the domain credentials for the endpoint"
PS > Set-AWSSamlRoleProfile -EndpointName $epName -NetworkCredential $credential -StoreAs SAMLDemoProfile
```

4. In entrambi i casi, il cmdlet `Set-AWSSamlRoleProfile` richiede di scegliere quale ruolo deve essere memorizzato nel profilo. Il seguente esempio mostra due ruoli disponibili: `ADFS-Dev` e `ADFS-Production`. I ruoli IAM sono associati alle tue credenziali di accesso AD dall'amministratore di AD FS.

```
Select Role
Select the role to be assumed when this profile is active
[1] 1 - ADFS-Dev [2] 2 - ADFS-Production [?] Help (default is "1"):
```

In alternativa, puoi specificare un ruolo senza il prompt, inserendo i parametri `RoleARN`, `PrincipalARN` e `NetworkCredential` opzionale. Se il ruolo specificato non è elencato nell'asserzione restituita dall'autenticazione, all'utente viene richiesto di scegliere tra i ruoli disponibili.

```
PS > $params = @{ "NetworkCredential"=$credential,
  "PrincipalARN"="{arn:aws:iam::012345678912:saml-provider/ADFS}",
  "RoleARN"="{arn:aws:iam::012345678912:role/ADFS-Dev}"
}
PS > $epName | Set-AWSSamlRoleProfile @params -StoreAs SAMLDemoProfile1 -Verbose
```

5. È possibile creare profili per tutti i ruoli in un unico comando aggiungendo il parametro `StoreAllRoles`, come mostrato nel codice seguente. Si noti che il nome del ruolo viene usato come nome del profilo.

```
PS > Set-AWSSamlRoleProfile -EndpointName $epName -StoreAllRoles
ADFS-Dev
ADFS-Production
```

Come utilizzare i profili di ruolo per eseguire cmdlet che richiedono credenziali AWS

Per eseguire cmdlet che richiedono AWS credenziali, è possibile utilizzare i profili di ruolo definiti nel file di credenziali condiviso. AWS Fornite il nome di un profilo di ruolo a `Set-AWSCredential` (o

come valore per qualsiasi `ProfileName` parametro in AWS Strumenti per PowerShell) per ottenere automaticamente AWS le credenziali temporanee per il ruolo descritto nel profilo.

Anche se si utilizza un solo profilo del ruolo alla volta, è possibile passare tra i profili all'interno di una sessione shell. Il cmdlet `Set-AWSCredential` non autentica e non ottiene le credenziali quando viene eseguito di per sé; il cmdlet registra il desiderio di utilizzare un specifico profilo di ruolo. Finché esegui un cmdlet che richiede le credenziali AWS, non occorre l'autenticazione o la richiesta di credenziali.

È ora possibile utilizzare le AWS credenziali temporanee ottenute con il `SAMLDemoProfile` profilo per lavorare con AWS il servizio. APIs Le seguenti sezioni mostrano degli esempi su come utilizzare i profili di ruolo.

Esempio 1: Impostare un ruolo predefinito con **Set-AWSCredential**

Questo esempio imposta un ruolo predefinito per una AWS Strumenti per PowerShell sessione utilizzando `Set-AWSCredential`. Quindi, è possibile eseguire i cmdlet che richiedono le credenziali e sono autorizzati dal ruolo specificato. In questo esempio vengono elencate tutte le istanze Amazon Elastic Compute Cloud nella Regione Stati Uniti occidentali (Oregon) associate al profilo specificato con il cmdlet `Set-AWSCredential`.

```
PS > Set-AWSCredential -ProfileName SAMLDemoProfile
PS > Get-EC2Instance -Region us-west-2 | Format-Table -Property Instances,GroupNames
```

Instances	GroupNames
-----	-----
{TestInstance1}	{default}
{TestInstance2}	{}
{TestInstance3}	{launch-wizard-6}
{TestInstance4}	{default}
{TestInstance5}	{}
{TestInstance6}	{AWS-OpsWorks-Default-
Server}	

Esempio 2: modifica dei profili dei ruoli durante una PowerShell sessione

Questo esempio elenca tutti i bucket Amazon S3 disponibili nell' AWS account del ruolo associato al profilo. `SAMLDemoProfile` L'esempio mostra che, sebbene sia stato possibile utilizzare un altro profilo in precedenza nella AWS Strumenti per PowerShell sessione, è possibile modificare i profili specificando un valore diverso per il `-ProfileName` parametro con i cmdlet che lo supportano. Si

tratta di un'attività comune per gli amministratori che gestiscono Amazon S3 PowerShell dalla riga di comando.

```
PS > Get-S3Bucket -ProfileName SAMLDemoProfile
```

CreationDate	BucketName
-----	-----
7/25/2013 3:16:56 AM	<i>amzn-s3-demo-bucket</i>
4/15/2015 12:46:50 AM	<i>amzn-s3-demo-bucket1</i>
4/15/2015 6:15:53 AM	<i>amzn-s3-demo-bucket2</i>
1/12/2015 11:20:16 PM	<i>amzn-s3-demo-bucket3</i>

Si noti che il cmdlet `Get-S3Bucket` specifica il nome del profilo creato eseguendo il cmdlet `Set-AWSSamlRoleProfile`. Questo comando può essere utile se è stato impostato un profilo all'inizio della sessione (per esempio, eseguendo il cmdlet `Set-AWSCredential`) e si voleva utilizzare un profilo di ruolo differente per il cmdlet `Get-S3Bucket`. Il gestore del profilo rende disponibili le credenziali provvisorie al cmdlet `Get-S3Bucket`.

Anche se le credenziali scadono dopo 1 ora (un limite imposto da STS), gli AWS Strumenti per PowerShell aggiornano automaticamente le credenziali richiedendo una nuova asserzione SAML quando gli strumenti rilevano che le attuali credenziali sono scadute.

Per gli utenti aggiunti al dominio, questo processo viene eseguito senza interruzioni, perché l'attuale identità dell'utente Windows viene utilizzata durante l'autenticazione. Per gli account non-domain-joined utente, AWS Strumenti per PowerShell mostra una richiesta di PowerShell credenziali che richiede la password dell'utente. L'utente fornisce le credenziali che sono state utilizzate per la sua nuova autenticazione e per ottenere una nuova asserzione.

Esempio 3: Ottenere istanze in una regione

L'esempio seguente elenca tutte EC2 le istanze Amazon nella regione Asia Pacifico (Sydney) associate all'account utilizzato dal `ADFS-Production` profilo. Si tratta di un comando utile per restituire tutte le EC2 istanze Amazon in una regione.

```
PS > (Get-Ec2Instance -ProfileName ADFS-Production -Region ap-southeast-2).Instances |
Select InstanceType, @{Name="Servername";Expression={$_.tags | where key -eq "Name" |
Select Value -Expand Value}}
```

InstanceType	Servername
-----	-----
t2.small	DC2

t1.micro	NAT1
t1.micro	RDGW1
t1.micro	RDGW2
t1.micro	NAT2
t2.small	DC1
t2.micro	BUILD

Ulteriori letture

Per informazioni generali su come implementare l'accesso federato alle API, consultare la pagina [Come implementare una soluzione generale per l'accesso federato API/CLI tramite SAML 2.0](#).

Per domande o commenti di supporto, visita i forum per AWS sviluppatori di [PowerShell scripting o.NET Development](#).

Individuazione di cmdlet e alias

Questa sezione illustra come elencare i servizi supportati da AWS Strumenti per PowerShell, come mostrare il set di cmdlet fornito da AWS Strumenti per PowerShell a supporto di tali servizi e come trovare nomi di cmdlet alternativi (chiamati anche alias) per accedere a tali servizi.

Individuare i cmdlet

Tutte le operazioni (o APIs) del AWS servizio sono documentate nella Guida di riferimento delle API per ogni servizio. Ad esempio, consulta la [Guida di riferimento dell'API IAM](#). Nella maggior parte dei casi esiste una one-to-one corrispondenza tra un'API di AWS servizio e un AWS PowerShell cmdlet. Per ottenere il nome del cmdlet che corrisponde al nome di un'API di AWS servizio, esegui il AWS `Get-AWSCmdletName` cmdlet con il parametro e il nome dell'API del `-ApiOperation` servizio. AWS Ad esempio, per ottenere tutti i possibili nomi di cmdlet basati su qualsiasi API di `DescribeInstances` AWS servizio disponibile, esegui il comando seguente:

```
PS > Get-AWSCmdletName -ApiOperation DescribeInstances
```

CmdletName	ServiceOperation	ServiceName	CmdletNounPrefix
-----	-----	-----	-----
Get-EC2Instance	DescribeInstances	Amazon Elastic Compute Cloud	EC2
Get-GMLInstance	DescribeInstances	Amazon GameLift Service	GML

Il parametro `-ApiOperation` è il parametro predefinito, quindi puoi omettere il nome del parametro. L'esempio seguente è uguale al precedente:

```
PS > Get-AWSCmdletName DescribeInstances
```

Se si conoscono i nomi sia dell'API che del servizio, è possibile includere il `-Service` parametro insieme al prefisso sostantivo del cmdlet o a parte del nome del servizio. AWS Ad esempio, il prefisso sostantivo del cmdlet per Amazon è. EC2 EC2 Per ottenere il nome del cmdlet che corrisponde all'`DescribeInstancesAPI` nel EC2 servizio Amazon, esegui uno dei seguenti comandi. Sono tutti risultati nello stesso output:

```
PS > Get-AWSCmdletName -ApiOperation DescribeInstances -Service EC2
PS > Get-AWSCmdletName -ApiOperation DescribeInstances -Service Compute
PS > Get-AWSCmdletName -ApiOperation DescribeInstances -Service "Compute Cloud"
```

CmdletName	ServiceOperation	ServiceName	CmdletNounPrefix
-----	-----	-----	-----
Get-EC2Instance	DescribeInstances	Amazon Elastic Compute Cloud	EC2

I valori del parametro in questi comandi fanno distinzione tra maiuscola e minuscola.

Se non conosci il nome dell'API di AWS servizio desiderata o del AWS servizio, puoi utilizzare il `-ApiOperation` parametro, insieme al modello da abbinare e al `-MatchWithRegex` parametro. Ad esempio, per ottenere tutti i nomi dei cmdlet disponibili che contengono `SecurityGroup`, esegui il comando seguente:

```
PS > Get-AWSCmdletName -ApiOperation SecurityGroup -MatchWithRegex
```

CmdletName	ServiceOperation	CmdletNounPrefix
-----	-----	-----
Approve-ECCacheSecurityGroupIngress	AuthorizeCacheSecurityGroupIngress	
Amazon ElastiCache	EC	
Get-ECCacheSecurityGroup	DescribeCacheSecurityGroups	
Amazon ElastiCache	EC	
New-ECCacheSecurityGroup	CreateCacheSecurityGroup	
Amazon ElastiCache	EC	
Remove-ECCacheSecurityGroup	DeleteCacheSecurityGroup	
Amazon ElastiCache	EC	
Revoke-ECCacheSecurityGroupIngress	RevokeCacheSecurityGroupIngress	
Amazon ElastiCache	EC	
Add-EC2SecurityGroupToClientVpnTargetNetwrk		
ApplySecurityGroupsToClientVpnTargetNetwork	Amazon Elastic Compute Cloud	EC2

Get-EC2SecurityGroup		DescribeSecurityGroups
Amazon Elastic Compute Cloud	EC2	
Get-EC2SecurityGroupReference		DescribeSecurityGroupReferences
Amazon Elastic Compute Cloud	EC2	
Get-EC2StaleSecurityGroup		DescribeStaleSecurityGroups
Amazon Elastic Compute Cloud	EC2	
Grant-EC2SecurityGroupEgress		AuthorizeSecurityGroupEgress
Amazon Elastic Compute Cloud	EC2	
Grant-EC2SecurityGroupIngress		AuthorizeSecurityGroupIngress
Amazon Elastic Compute Cloud	EC2	
New-EC2SecurityGroup		CreateSecurityGroup
Amazon Elastic Compute Cloud	EC2	
Remove-EC2SecurityGroup		DeleteSecurityGroup
Amazon Elastic Compute Cloud	EC2	
Revoke-EC2SecurityGroupEgress		RevokeSecurityGroupEgress
Amazon Elastic Compute Cloud	EC2	
Revoke-EC2SecurityGroupIngress		RevokeSecurityGroupIngress
Amazon Elastic Compute Cloud	EC2	
Update-EC2SecurityGroupRuleEgressDescription		UpdateSecurityGroupRuleDescriptionsEgress
Amazon Elastic Compute Cloud	EC2	
Update-EC2SecurityGroupRuleIngressDescription		
UpdateSecurityGroupRuleDescriptionsIngress	Amazon Elastic Compute Cloud	EC2
Edit-EFSMountTargetSecurityGroup		ModifyMountTargetSecurityGroups
Amazon Elastic File System	EFS	
Get-EFSMountTargetSecurityGroup		DescribeMountTargetSecurityGroups
Amazon Elastic File System	EFS	
Join-ELBSecurityGroupToLoadBalancer		ApplySecurityGroupsToLoadBalancer
Elastic Load Balancing	ELB	
Set-ELB2SecurityGroup		SetSecurityGroups
Elastic Load Balancing V2	ELB2	
Enable-RDSDBSecurityGroupIngress		AuthorizeDBSecurityGroupIngress
Amazon Relational Database Service	RDS	
Get-RDSDBSecurityGroup		DescribeDBSecurityGroups
Amazon Relational Database Service	RDS	
New-RDSDBSecurityGroup		CreateDBSecurityGroup
Amazon Relational Database Service	RDS	
Remove-RDSDBSecurityGroup		DeleteDBSecurityGroup
Amazon Relational Database Service	RDS	
Revoke-RDSDBSecurityGroupIngress		RevokeDBSecurityGroupIngress
Amazon Relational Database Service	RDS	
Approve-RSClusterSecurityGroupIngress		AuthorizeClusterSecurityGroupIngress
Amazon Redshift	RS	
Get-RSClusterSecurityGroup		DescribeClusterSecurityGroups
Amazon Redshift	RS	

New-RSClusterSecurityGroup		CreateClusterSecurityGroup
Amazon Redshift	RS	
Remove-RSClusterSecurityGroup		DeleteClusterSecurityGroup
Amazon Redshift	RS	
Revoke-RSClusterSecurityGroupIngress		RevokeClusterSecurityGroupIngress
Amazon Redshift	RS	

Se conosci il nome del AWS servizio ma non l'API del AWS servizio, includi sia il `-MatchWithRegex` parametro che il `-Service` parametro per circoscrivere la ricerca a un singolo servizio. Ad esempio, per ottenere tutti i nomi di cmdlet che contengono SecurityGroup solo il EC2 servizio Amazon, esegui il comando seguente

```
PS > Get-AWSCmdletName -ApiOperation SecurityGroup -MatchWithRegex -Service EC2
```

CmdletName	ServiceName	ServiceOperation	CmdletNounPrefix
-----	-----	-----	-----
Add-EC2SecurityGroupToClientVpnTargetNetwrk			
ApplySecurityGroupsToClientVpnTargetNetwork	Amazon Elastic Compute Cloud	EC2	
Get-EC2SecurityGroup		DescribeSecurityGroups	
Amazon Elastic Compute Cloud	EC2		
Get-EC2SecurityGroupReference		DescribeSecurityGroupReferences	
Amazon Elastic Compute Cloud	EC2		
Get-EC2StaleSecurityGroup		DescribeStaleSecurityGroups	
Amazon Elastic Compute Cloud	EC2		
Grant-EC2SecurityGroupEgress		AuthorizeSecurityGroupEgress	
Amazon Elastic Compute Cloud	EC2		
Grant-EC2SecurityGroupIngress		AuthorizeSecurityGroupIngress	
Amazon Elastic Compute Cloud	EC2		
New-EC2SecurityGroup		CreateSecurityGroup	
Amazon Elastic Compute Cloud	EC2		
Remove-EC2SecurityGroup		DeleteSecurityGroup	
Amazon Elastic Compute Cloud	EC2		
Revoke-EC2SecurityGroupEgress		RevokeSecurityGroupEgress	
Amazon Elastic Compute Cloud	EC2		
Revoke-EC2SecurityGroupIngress		RevokeSecurityGroupIngress	
Amazon Elastic Compute Cloud	EC2		
Update-EC2SecurityGroupRuleEgressDescription		UpdateSecurityGroupRuleDescriptionsEgress	
Amazon Elastic Compute Cloud	EC2		
Update-EC2SecurityGroupRuleIngressDescription			
UpdateSecurityGroupRuleDescriptionsIngress	Amazon Elastic Compute Cloud	EC2	

Se conosci il nome del comando AWS Command Line Interface (AWS CLI), puoi utilizzare il `-AwsCliCommand` parametro e il nome del AWS CLI comando desiderato per ottenere il nome del cmdlet basato sulla stessa API. Ad esempio, per ottenere il nome del cmdlet che corrisponde alla chiamata di `authorize-security-group-ingress` AWS CLI comando nel EC2 servizio Amazon, esegui il comando seguente:

```
PS > Get-AWSCmdletName -AwsCliCommand "aws ec2 authorize-security-group-ingress"
```

CmdletName	ServiceOperation	ServiceName
CmdletNounPrefix		
-----	-----	-----

Grant-EC2SecurityGroupIngress	AuthorizeSecurityGroupIngress	Amazon Elastic Compute Cloud EC2

Il `Get-AWSCmdletName` cmdlet necessita solo del nome del AWS CLI comando sufficiente per identificare il servizio e l'API. AWS

Per ottenere un elenco di tutti i cmdlet presenti in Tools for PowerShell Core, eseguire il PowerShell `Get-Command` cmdlet, come illustrato nell'esempio seguente.

```
PS > Get-Command -Module AWSPowerShell.NetCore
```

Puoi eseguire lo stesso comando con `-Module AWSPowerShell` per visualizzare i cmdlet negli AWS Tools for Windows PowerShell.

Il cmdlet `Get-Command` genera l'elenco di cmdlet in ordine alfabetico. Si noti che per impostazione predefinita l'elenco è ordinato per PowerShell verbo, anziché per sostantivo. PowerShell

Per ordinare i risultati in base al servizio, esegui il comando seguente.

```
PS > Get-Command -Module AWSPowerShell.NetCore | Sort-Object Noun,Verb
```

Per filtrare i cmdlet restituiti dal cmdlet, reindirizzate l'`Get-Command` output al cmdlet. PowerShell `Select-String` Ad esempio, per visualizzare il set di cmdlet che funzionano con le regioni, esegui il comando seguente: AWS

```
PS > Get-Command -Module AWSPowerShell.NetCore | Select-String region
```

```
Clear-DefaultAWSRegion
```

```
Copy-HSM2BackupToRegion
Get-AWSRegion
Get-DefaultAWSRegion
Get-EC2Region
Get-LSRegionList
Get-RDSSourceRegion
Set-DefaultAWSRegion
```

È inoltre possibile trovare i cmdlet per un servizio specifico filtrando per il prefisso del servizio dei sostantivi del cmdlet. Per visualizzare l'elenco dei prefissi del servizio disponibili, esegui `Get-AWSPowerShellVersion -ListServiceVersionInfo`. L'esempio seguente restituisce cmdlet che supportano il servizio Amazon CloudWatch Events.

```
PS > Get-Command -Module AWSPowerShell -Noun CWE*
```

CommandType	Name	Version	Source
-----	----	-----	-----
Cmdlet	Add-CWEResourceTag	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Disable-CWEEventSource	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Disable-CWERule	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Enable-CWEEventSource	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Enable-CWERule	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEEventBus	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEEventBusList	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEEventSource	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEEventSourceList	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEPartnerEventSource	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEPartnerEventSourceAccountList	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEPartnerEventSourceList	3.3.563.1	
	AWSPowerShell.NetCore		
Cmdlet	Get-CWEResourceTag	3.3.563.1	
	AWSPowerShell.NetCore		

Cmdlet	Get-CWRule	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWRuleDetail	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWRuleNamesByTarget	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Get-CWETargetsByRule	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	New-CWEventBus	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	New-CWPartnerEventSource	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWEventBus	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWPartnerEventSource	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWEPermission	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWEResourceTag	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWRule	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Remove-CWETarget	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Test-CWEventPattern	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Write-CWEvent	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Write-CWPartnerEvent	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Write-CWEPermission	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Write-CWRule	3.3.563.1
	AWSPowerShell.NetCore	
Cmdlet	Write-CWETarget	3.3.563.1
	AWSPowerShell.NetCore	

Nomi dei cmdlet e degli alias

I cmdlet inclusi in AWS Strumenti per PowerShell for each service si basano sui metodi forniti dall' AWS SDK per il servizio. Tuttavia, a causa delle convenzioni PowerShell di denominazione obbligatorie, il nome di un cmdlet potrebbe essere diverso dal nome della chiamata API o del

metodo su cui è basato. Ad esempio, il `Get-EC2Instance` cmdlet si basa sul metodo `Amazon EC2DescribeInstances`.

In alcuni casi, il nome del cmdlet può essere simile a un nome del metodo, ma potrebbe effettivamente eseguire una funzione differente. Ad esempio, il metodo `Amazon S3GetObject` recupera un oggetto Amazon S3. Tuttavia, il cmdlet `Get-S3Object` restituisce le informazioni su un oggetto Amazon S3 anziché l'oggetto stesso.

```
PS > Get-S3Object -BucketName text-content -Key aws-tech-docs
```

```
ETag          : "df000002a0fe0000f3c000004EXAMPLE"
BucketName    : aws-tech-docs
Key           : javascript/frameset.js
LastModified  : 6/13/2011 1:24:18 PM
Owner         : Amazon.S3.Model.Owner
Size          : 512
StorageClass  : STANDARD
```

Per ottenere un oggetto S3 con AWS Strumenti per PowerShell, esegui il cmdlet: `Read-S3Object`

```
PS > Read-S3Object -BucketName text-content -Key text-object.txt -file c:\tmp\text-object-download.txt
```

Mode	LastWriteTime	Length	Name
----	-----	-----	----
-a---	11/5/2012 7:29 PM	20622	text-object-download.txt

Note

Il cmdlet help per un AWS cmdlet fornisce il nome dell'API AWS SDK su cui è basato il cmdlet.

[Per ulteriori informazioni sui verbi standard e sui relativi significati, PowerShell vedere Verbi approvati per i comandi. PowerShell](#)

Tutti i AWS cmdlet che utilizzano il `Remove` verbo, e il `Stop-EC2Instance` cmdlet quando si aggiunge il `-Terminate` parametro, richiedono una conferma prima di procedere. Per ignorare la conferma, aggiungere il parametro `-Force` per il comando.

⚠ Important

AWS i cmdlet non supportano lo switch. `-WhatIf`

Alias

L'installazione di AWS Strumenti per PowerShell installa un file di alias che contiene alias per molti cmdlet. AWS Questi alias potrebbero risultare più intuitivi dei nomi dei cmdlet. Ad esempio, i nomi dei servizi e i nomi dei metodi AWS SDK sostituiscono PowerShell verbi e sostantivi in alcuni alias. Un esempio è l'alias `EC2-DescribeInstances`.

Altri alias utilizzano verbi che, sebbene non seguano le PowerShell convenzioni standard, possono essere più descrittivi dell'operazione effettiva. Ad esempio, il file alias associa l'alias `Get-S3Content` al cmdlet `Read-S3Object`.

```
PS > Set-Alias -Name Get-S3Content -Value Read-S3Object
```

Il file `aliases` si trova nella directory di installazione. AWS Strumenti per PowerShell Per caricare gli alias nell'ambiente, eseguire il file dot-source. Il seguente è un esempio basato su Windows.

```
PS > . "C:\Program Files (x86)\AWS Tools\PowerShell\AWSPowerShell\AWSAliases.ps1"
```

Per una shell Linux o macOS, potrebbe essere simile a questo:

```
. ~/.local/share/powershell/Modules/AWSPowerShell.NetCore/3.3.563.1/AWSAliases.ps1
```

Per mostrare tutti gli AWS Strumenti per PowerShell alias, esegui il comando seguente. Questo comando utilizza l'`?alias` per il PowerShell `Where-Object` cmdlet e la `Source` proprietà per filtrare solo gli alias che provengono dal modulo `AWSPowerShell.NetCore`

```
PS > Get-Alias | ? Source -like "AWSPowerShell.NetCore"
```

CommandType	Name	Version	Source
-----	----	-----	-----
Alias	Add-ASInstances	3.3.343.0	
AWSPowerShell			
Alias	Add-CTTag	3.3.343.0	
AWSPowerShell			

Alias AWSPowerShell	Add-DPTags	3.3.343.0
Alias AWSPowerShell	Add-DSIpRoutes	3.3.343.0
Alias AWSPowerShell	Add-ELBTags	3.3.343.0
Alias AWSPowerShell	Add-EMRTag	3.3.343.0
Alias AWSPowerShell	Add-ESTag	3.3.343.0
Alias AWSPowerShell	Add-MLTag	3.3.343.0
Alias AWSPowerShell	Clear-AWSCredentials	3.3.343.0
Alias AWSPowerShell	Clear-AWSDefaults	3.3.343.0
Alias AWSPowerShell	Dismount-ASInstances	3.3.343.0
Alias AWSPowerShell	Edit-EC2Hosts	3.3.343.0
Alias AWSPowerShell	Edit-RSClusterIamRoles	3.3.343.0
Alias AWSPowerShell	Enable-ORGAllFeatures	3.3.343.0
Alias AWSPowerShell	Find-CTEvents	3.3.343.0
Alias AWSPowerShell	Get-ASACases	3.3.343.0
Alias AWSPowerShell	Get-ASAccountLimits	3.3.343.0
Alias AWSPowerShell	Get-ASACommunications	3.3.343.0
Alias AWSPowerShell	Get-ASAServices	3.3.343.0
Alias AWSPowerShell	Get-ASASeverityLevels	3.3.343.0
Alias AWSPowerShell	Get-ASATrustedAdvisorCheckRefreshStatuses	3.3.343.0
Alias AWSPowerShell	Get-ASATrustedAdvisorChecks	3.3.343.0
Alias AWSPowerShell	Get-ASATrustedAdvisorCheckSummaries	3.3.343.0
Alias AWSPowerShell	Get-ASLifecycleHooks	3.3.343.0

Alias	Get-ASLifecycleHookTypes	3.3.343.0
AWSPowerShell		
Alias	Get-AWSCredentials	3.3.343.0
AWSPowerShell		
Alias	Get-CDApplications	3.3.343.0
AWSPowerShell		
Alias	Get-CDDeployments	3.3.343.0
AWSPowerShell		
Alias	Get-CFCloudFrontOriginAccessIdentities	3.3.343.0
AWSPowerShell		
Alias	Get-CFDistributions	3.3.343.0
AWSPowerShell		
Alias	Get-CFGConfigRules	3.3.343.0
AWSPowerShell		
Alias	Get-CFGConfigurationRecorders	3.3.343.0
AWSPowerShell		
Alias	Get-CFGDeliveryChannels	3.3.343.0
AWSPowerShell		
Alias	Get-CFInvalidations	3.3.343.0
AWSPowerShell		
Alias	Get-CFNAccountLimits	3.3.343.0
AWSPowerShell		
Alias	Get-CFNStackEvents	3.3.343.0
AWSPowerShell		
...		

Per aggiungere alias personalizzati a questo file, potrebbe essere necessario aumentare il valore della [variabile di PowerShell \\$MaximumAliasCount preferenza a un valore superiore a 5500](#). Il valore predefinito è 4096; è possibile aumentarlo a un massimo di 32768. Per farlo, eseguire il comando seguente.

```
PS > $MaximumAliasCount = 32768
```

Per verificare che la modifica abbia avuto successo, inserire il nome della variabile per visualizzarne il valore corrente.

```
PS > $MaximumAliasCount
32768
```

Pipeline, output e iterazione in AWS Strumenti per PowerShell

Pipeline

PowerShell incoraggia gli utenti a connettere i cmdlet in [pipeline](#) che indirizzano l'output di un cmdlet all'input del successivo. L'esempio seguente mostra questo comportamento quando si utilizza `AWS Strumenti per PowerShell`. Il comando ottiene e quindi arresta tutte le EC2 istanze Amazon nella regione predefinita corrente.

```
PS > Get-EC2Instance | Stop-EC2Instance
```

Output del cmdlet

Per supportare meglio il pipelining, alcuni dati delle risposte di AWS SDK per .NET potrebbero essere eliminati per impostazione predefinita. L'output dei `AWS Strumenti per PowerShell` cmdlet non viene rimodellato per includere la risposta del servizio e le istanze dei risultati come `Note` proprietà dell'oggetto di raccolta emesso. Invece, per le chiamate che emettono una singola raccolta come output, la raccolta viene ora enumerata nella pipeline. PowerShell Ciò significa che i dati della risposta e dei risultati dell'SDK non possono esistere nella pipeline perché non esiste un oggetto di raccolta contenitore a cui possano essere allegati.

Sebbene la maggior parte degli utenti probabilmente non abbia bisogno di questi dati, possono essere utili per scopi diagnostici perché è possibile vedere esattamente cosa è stato inviato e ricevuto dalle chiamate di AWS servizio sottostanti effettuate dal cmdlet. A partire dalla `AWS Strumenti per PowerShell V4`, i cmdlet possono utilizzare il `-Select *` parametro e l'argomento per restituire l'intera risposta del servizio.

Note

Nelle versioni `AWS Strumenti per PowerShell` precedenti alla V4, `$AWSHistory` è stata introdotta una variabile di sessione denominata che mantiene un record delle chiamate dei `AWS` cmdlet e delle risposte di servizio ricevute per ogni chiamata. Nella versione 4 di `Tools for PowerShell`, questa variabile di sessione è stata sostituita dal `-Select *` parametro e dall'argomento, che possono essere utilizzati per restituire l'intera risposta del servizio. Questo parametro è descritto in questo argomento.

Note

Si tratta di una documentazione di pre-rilascio di una caratteristica nella versione di anteprima. ed è soggetta a modifiche.

La `$AWSHistory` variabile verrà rimossa nella V5 di AWS Strumenti per PowerShell. Per maggiori informazioni, consulta il post sul blog [Avviso della prossima versione principale 5 di AWS Tools for PowerShell](#).

Per illustrare come è possibile restituire tutti i dati di una risposta, considera i seguenti esempi.

Il primo esempio restituisce semplicemente un elenco di bucket Amazon S3. Questo è il comportamento che segue di default.

PS > **Get-S3Bucket**

CreationDate	BucketName
-----	-----
9/22/2023 10:54:35 PM	amzn-s3-demo-bucket1
9/22/2023 11:04:37 AM	amzn-s3-demo-bucket2
9/22/2023 12:54:34 PM	amzn-s3-demo-bucket3

Il secondo esempio restituisce un oggetto di AWS SDK per .NET risposta. Poiché `-Select *` è stato specificato, l'output include l'intera risposta API, che contiene la raccolta di bucket nella `Buckets` proprietà. In questo esempio, il `Format-List` cmdlet non è strettamente necessario, ma è presente per garantire la visualizzazione di tutte le proprietà.

PS > **Get-S3Bucket -Select * | Format-List**

```

LoggedAt      : 10/1/2023 9:45:52 AM
Buckets       : {amzn-s3-demo-bucket1, amzn-s3-demo-bucket2,
                  amzn-s3-demo-bucket3}
Owner         : Amazon.S3.Model.Owner
ContinuationToken :
ResponseMetadata : Amazon.Runtime.ResponseMetadata
ContentLength  : 0
HttpStatusCode : OK
  
```

Iterazione tramite dati paginati

Le sezioni seguenti descrivono i vari tipi di iterazione possibili.

Iterazione automatica

Per i servizi APIs che impongono un numero massimo predefinito di oggetti restituiti per una determinata chiamata o che supportano set di risultati paginabili, la maggior parte dei cmdlet implementa l'iterazione automatica, che abilita il comportamento predefinito di "». page-to-completion. In questo scenario, un cmdlet effettua tutte le chiamate necessarie per conto dell'utente per restituire il set di dati completo alla pipeline.

Nell'esempio seguente, che utilizza il `Get-S3Object` cmdlet, la `$result` variabile contiene `S3Object` istanze per ogni chiave in un bucket chiamato `amzn-s3-demo-bucket1`, che è potenzialmente un set di dati molto grande.

```
PS > $result = Get-S3Object -BucketName amzn-s3-demo-bucket1
```

L'esempio seguente riduce il numero di risultati per ogni pagina durante l'iterazione automatica dal valore predefinito di 1000 a 500. L'esempio esegue il doppio delle chiamate di iterazione automatica perché per ogni chiamata viene restituito solo la metà dei risultati.

```
PS > $result = Get-S3Object -BucketName amzn-s3-demo-bucket1 -MaxKey 500
```

Note

Nella AWS Strumenti per PowerShell V4, alcuni cmdlet per le operazioni su pagina non implementano l'iterazione automatica. Se un cmdlet non dispone del `-NoAutoIteration` parametro, illustrato nella sezione successiva, non implementa l'iterazione automatica.

Disabilita l'iterazione automatica

Se desideri che gli strumenti PowerShell restituiscano solo la prima pagina di dati, puoi aggiungere il `-NoAutoIteration` parametro per evitare che vengano restituite pagine di dati aggiuntive.

L'esempio seguente utilizza i `-MaxKey` parametri `-NoAutoIteration` and per limitare il numero di `S3Object` istanze restituite a non più delle prime 500 trovate nel bucket.

```
PS > $result = Get-S3Object -BucketName amzn-s3-demo-bucket1 -MaxKey 500 -  
NoAutoIteration
```

Per determinare se sono disponibili più dati ma non sono stati restituiti, utilizzate il `-Select *` parametro e l'argomento e verificate se è presente un valore nella proprietà del token successivo.

L'esempio seguente restituisce `$true` se ci sono più di 500 oggetti nel bucket e `$false` se altro.

```
PS > $result = Get-S3Object -BucketName amzn-s3-demo-bucket1 -MaxKey 500 -  
NoAutoIteration -Select *  
PS > $null -eq $result.NextMarker
```

Note

I nomi della proprietà di risposta del token successivo e del parametro del cmdlet variano tra i cmdlet. Per i dettagli, fare riferimento alla documentazione di aiuto per ogni cmdlet.

Iterazione manuale

L'esempio seguente restituisce tutti gli oggetti S3 da un bucket utilizzando un ciclo [do](#), che valuta la condizione dopo ogni iterazione. Il `do` ciclo esegue iterazioni fino a quando non `Get-S3Object` viene impostato `$result.NextMarker` su `$null`, a indicare che non rimangono più dati paginati. L'output del loop viene assegnato alla `$s3objects` variabile.

```
$s3objects = do  
{  
    $splatParams = @{  
        BucketName = 'amzn-s3-demo-bucket1'  
        MaxKey = 500  
        Marker = $result.NextMarker  
        NoAutoIteration = $true  
        Select = '*'  
    }  
    $result = Get-S3Object @splatParams  
  
    $result.S3Objects  
}  
while ($null -ne $result.NextMarker)
```

Questo esempio utilizza lo PowerShell [splatting](#) per evitare una lunga riga di codice causata dalla dichiarazione di parametri e argomenti in linea.

Risoluzione di credenziali e profili

Ordine di ricerca credenziali

Quando si esegue un comando, AWS Strumenti per PowerShell cerca le credenziali nell'ordine seguente. Si interrompe quando trova le credenziali utilizzabili.

1. Credenziali letterali incorporate come parametri nella riga di comando.

Consigliamo di utilizzare i profili anziché inserire credenziali letterali nelle righe di comando.

2. Un nome specifico del profilo o la posizione del profilo.
 - Se specificate solo un nome di profilo, il comando cerca il profilo specificato nell'archivio AWS SDK e, se non esiste, il profilo specificato dal file delle credenziali AWS condivise nella posizione predefinita.
 - Se specifichi solo una posizione del profilo, il comando cerca il profilo default dal file delle credenziali.
 - Se specifichi sia un nome sia un percorso, il comando cerca il profilo specificato nel file delle credenziali.

Se il profilo specificato o il percorso non è stato trovato, il comando genera un'eccezione. La ricerca procede nelle seguenti fasi solo se non hai specificato un profilo o una posizione.

3. Credenziali specificate dal parametro `-Credential`.
4. Il profilo di sessione, se esiste.
5. Il profilo predefinito nell'ordine seguente:
 - a. Il default profilo nell'archivio AWS SDK.
 - b. Il default profilo nel file delle credenziali AWS condivise.
 - c. Il AWS PS Default profilo nell'archivio AWS SDK.
6. Se il comando è in esecuzione su un' EC2 istanza Amazon configurata per utilizzare un ruolo IAM, le credenziali temporanee dell' EC2istanza sono accessibili dal profilo dell'istanza.

Per ulteriori informazioni sull'utilizzo dei ruoli IAM per EC2 le istanze Amazon, consulta il [AWS SDK per .NET](#).

Se la ricerca ha esito negativo per individuare le credenziali specificate, il comando genera un'eccezione.

Ulteriori informazioni su utenti e ruoli

Per eseguire Strumenti per PowerShell i comandi su AWS, è necessario disporre di una combinazione di utenti, set di autorizzazioni e ruoli di servizio appropriata per le proprie attività.

Gli utenti, i set di autorizzazioni e i ruoli di servizio specifici creati e il modo in cui li utilizzi dipenderanno dai tuoi requisiti. Di seguito sono riportate alcune informazioni aggiuntive sul motivo per cui potrebbero essere utilizzati e su come crearli.

Utenti e set di autorizzazioni

Sebbene sia possibile utilizzare un account utente IAM con credenziali a lungo termine per accedere ai servizi AWS, questa non è più una best practice e dovrebbe essere evitata. Anche durante lo sviluppo, è consigliabile creare utenti e set di autorizzazioni AWS IAM Identity Center e utilizzare credenziali temporanee fornite da una fonte di identità.

Per lo sviluppo, puoi utilizzare l'utente che hai creato o assegnato in [Configurazione dell'autenticazione degli strumenti](#). Se disponi Console di gestione AWS delle autorizzazioni appropriate, puoi anche creare diversi set di autorizzazioni con il privilegio minimo per quell'utente o creare nuovi utenti specificamente per progetti di sviluppo, fornendo set di autorizzazioni con il privilegio minimo. L'eventuale operazione scelta dipenderà dalle circostanze.

Per ulteriori informazioni su questi utenti e set di autorizzazioni e su come crearli, consulta [Autenticazione e accesso](#) nella Guida di riferimento agli strumenti AWS SDKs e Guida [introduttiva](#) nella Guida per l'utente.AWS IAM Identity Center

Ruoli di servizio

È possibile impostare un ruolo AWS di servizio per accedere ai AWS servizi per conto degli utenti. Questo tipo di accesso è appropriato se più persone eseguiranno l'applicazione in remoto, ad esempio su un' EC2 istanza Amazon che hai creato per questo scopo.

Il processo di creazione di un ruolo di servizio varia a seconda della situazione, ma è essenzialmente simile a quanto indicato di seguito.

1. Accedi Console di gestione AWS e apri la console IAM all'indirizzo <https://console.aws.amazon.com/iam/>.

2. Selezionare Roles (Ruoli), quindi selezionare Create role (Crea ruolo).
3. Scegli AWS il servizio, trova e seleziona EC2(ad esempio), quindi scegli il caso EC2d'uso (ad esempio).
4. Scegli Avanti e seleziona le [politiche appropriate](#) per i AWS servizi che l'applicazione utilizzerà.

 Warning

NON scegliete la AdministratorAccess politica perché tale politica consente le autorizzazioni di lettura e scrittura per quasi tutto il contenuto del vostro account.

5. Scegli Next (Successivo). Immetti un valore in Nome ruolo e in Descrizione e quindi immetti qualsiasi altro tag desiderato.

Puoi trovare informazioni sui tag in [Controlling access using AWS resource tags](#) nella [IAM User Guide](#).

6. Scegliere Crea ruolo.

Per ulteriori informazioni di alto livello sui ruoli IAM, consulta [Identità IAM \(utenti, gruppi di utenti e ruoli\)](#) nella [Guida per l'utente di IAM](#). Per informazioni dettagliate sui ruoli, consulta l'argomento [Ruoli IAM](#).

Uso delle credenziali legacy

Negli argomenti di questa sezione vengono fornite informazioni sull'utilizzo di credenziali a lungo termine o a breve termine senza utilizzare AWS IAM Identity Center.

 Warning

Per evitare rischi per la sicurezza, non utilizzare gli utenti IAM per l'autenticazione quando sviluppi software creato ad hoc o lavori con dati reali. Utilizza invece la federazione con un provider di identità come [AWS IAM Identity Center](#).

 Note

Le informazioni contenute in questi argomenti si riferiscono a scenari in cui è necessario ottenere e gestire manualmente le credenziali a breve o lungo termine. Per ulteriori

informazioni sulle credenziali a breve e lungo termine, consulta [Altri modi di autenticazione nella Guida](#) di riferimento AWS SDKs e agli strumenti.

Per le migliori pratiche di sicurezza, utilizzare AWS IAM Identity Center, come descritto in. [Configurazione dell'autenticazione degli strumenti](#)

Avvertenze e linee guida importanti per le credenziali

Avvertenze per le credenziali

- NON utilizzate le credenziali root del vostro account per accedere alle AWS risorse. Queste credenziali forniscono un accesso illimitato all'account e sono difficili da revocare.
- NON inserire chiavi di accesso letterali o informazioni sulle credenziali nei comandi o negli script. In caso contrario, si corre il rischio di esporre accidentalmente le proprie credenziali.
- Tieni presente che tutte le credenziali memorizzate nel AWS `credentials` file condiviso vengono archiviate in testo non crittografato.

Linee guida aggiuntive per la gestione sicura delle credenziali

[Per una discussione generale su come gestire in modo sicuro le credenziali, consulta AWS le credenziali di sicurezza nella sezione Credenziali AWS di sicurezza e le best practice Riferimenti generali di AWS e i casi d'uso di sicurezza nella IAM User Guide.](#) Considera inoltre quanto segue:

- Crea utenti aggiuntivi, come gli utenti in IAM Identity Center, e utilizza le loro credenziali invece di utilizzare le credenziali dell'utente root AWS . Le credenziali per altri utenti possono essere revocate, se necessario, o sono temporanee per natura. Inoltre, puoi applicare una policy a ciascun utente per consentire l'accesso solo a determinate risorse e azioni e quindi adottare autorizzazioni basate sul con privilegio minimo.
- Usa [ruoli IAM](#) per le attività di Amazon Elastic Container Service (Amazon ECS).
- Usa [i ruoli IAM](#) per le applicazioni in esecuzione su EC2 istanze Amazon.

Argomenti

- [Utilizzo delle AWS credenziali](#)
- [Credenziali condivise in AWS Strumenti per PowerShell](#)

Utilizzo delle AWS credenziali

Ogni AWS Strumenti per PowerShell comando deve includere un set di AWS credenziali, che vengono utilizzate per firmare crittograficamente la richiesta di servizio Web corrispondente. Puoi specificare le credenziali per comando, per sessione o per tutte le sessioni.

Warning

Per evitare rischi per la sicurezza, non utilizzare gli utenti IAM per l'autenticazione quando sviluppi software creato ad hoc o lavori con dati reali. Utilizza invece la federazione con un provider di identità come [AWS IAM Identity Center](#).

Note

Le informazioni contenute in questo argomento riguardano le circostanze in cui è necessario ottenere e gestire manualmente le credenziali a breve o lungo termine. Per ulteriori informazioni sulle credenziali a breve e lungo termine, consulta [Altri modi di autenticazione nella and Tools Reference Guide](#).AWS SDKs

Per le migliori pratiche di sicurezza, utilizzare AWS IAM Identity Center, come descritto in. [Configurazione dell'autenticazione degli strumenti](#)

Come buona pratica, evitare di esporre le proprie credenziali, non utilizzare le credenziali letterali in un comando. Invece, creare un profilo per ogni set di credenziali che si desidera utilizzare e memorizzare il profilo in uno dei due archivi delle credenziali. Specificando il profilo corretto per nome nel comando, gli AWS Strumenti per PowerShell recuperano le credenziali associate. Per una discussione generale su come gestire in modo sicuro AWS le credenziali, vedere [Best Practices for Managing AWS Access Keys](#) nel Riferimenti generali di Amazon Web Services.

Note

È necessario un AWS account per ottenere le credenziali e utilizzare il. AWS Strumenti per PowerShell Per creare un AWS account, vedi [Guida introduttiva: Sei un utente principiante AWS?](#) nella Guida di Gestione dell'account AWS riferimento.

Argomenti

- [Posizioni degli archivi per credenziali](#)
- [Gestione dei profili](#)
- [Specifica delle credenziali](#)
- [Ordine di ricerca credenziali](#)
- [Gestione delle credenziali negli AWS Tools for PowerShell Core](#)

Posizioni degli archivi per credenziali

AWS Strumenti per PowerShell Possono utilizzare uno dei due archivi di credenziali:

- L'archivio AWS SDK, che crittografa le credenziali e le archivia nella cartella home. In Windows, questo archivio si trova in: `C:\Users\username\AppData\Local\AWSToolkit\RegisteredAccounts.json`.

L'[AWS SDK per .NET](#) e il [Toolkit for Visual Studio](#) possono anche utilizzare l'archivio SDK AWS .

- Il file delle credenziali condiviso, che si trova anche nella tua cartella home, ma archivia le credenziali come testo semplice.

Per impostazione predefinita, i file delle credenziali vengono archiviati qui:

- Su Windows: `C:\Users\username\.aws\credentials`
- Su Mac/Linux: `~/.aws/credentials`

Inoltre, AWS SDKs AWS Command Line Interface possono utilizzare il file delle credenziali. Se state eseguendo uno script al di fuori del contesto AWS utente, assicuratevi che il file contenente le credenziali venga copiato in una posizione in cui tutti gli account utente (sistema locale e utente) possano accedere alle vostre credenziali.

Gestione dei profili

I profili consentono di fare riferimento a diversi set di credenziali con. AWS Strumenti per PowerShell. È possibile utilizzare i AWS Strumenti per PowerShell cmdlet per gestire i profili nell'archivio SDK. AWS. Inoltre, puoi gestire i profili nell'archivio SDK AWS utilizzando il [Toolkit for Visual Studio](#) o a livello di programmazione utilizzando l'[AWS SDK per .NET](#). Per istruzioni su come gestire i profili nel file delle credenziali, consulta [Best practice for Managing Access Keys](#). AWS

Aggiunta di un nuovo profilo

Per aggiungere un nuovo profilo all'archivio AWS SDK, esegui il comando `Set-AWSCredential`. In questo modo la chiave di accesso e la chiave segreta vengono archiviate nel file delle credenziali predefinito sotto il nome del profilo specificato.

```
PS > Set-AWSCredential `
    -AccessKey AKIA0123456787EXAMPLE `
    -SecretKey wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY `
    -StoreAs MyNewProfile
```

- `-AccessKey` – L'ID chiave di accesso.
- `-SecretKey` – La chiave segreta.
- `-StoreAs`– Il nome del profilo, deve essere univoco. Per specificare il profilo predefinito, utilizza il nome `default`.

Aggiornamento di un profilo

L'archivio AWS SDK deve essere gestito manualmente. Se successivamente si modificano le credenziali nel servizio, ad esempio utilizzando la [console IAM](#), eseguendo un comando con le credenziali archiviate in locale e questo ha esito negativo con il seguente messaggio di errore:

```
The Access Key Id you provided does not exist in our records.
```

È possibile aggiornare un profilo ripetendo il comando `Set-AWSCredential` per il profilo e passando a esso le nuove chiavi segrete e di accesso.

Elenco dei profili

Puoi controllare l'elenco corrente dei nomi con il seguente comando. In questo esempio, un utente denominato Shirley ha accesso a tre profili che sono tutti archiviati nel file delle credenziali condiviso (`~/.aws/credentials`).

```
PS > Get-AWSCredential -ListProfileDetail
```

ProfileName	StoreTypeName	ProfileLocation
-----	-----	-----
default	SharedCredentialsFile	/Users/shirley/.aws/credentials
production	SharedCredentialsFile	/Users/shirley/.aws/credentials

```
test          SharedCredentialsFile /Users/shirley/.aws/credentials
```

Rimozione di un profilo

Per rimuovere un profilo non più necessario, utilizza il comando seguente.

```
PS > Remove-AWSCredentialProfile -ProfileName an-old-profile-I-do-not-need
```

Il parametro `-ProfileName` specifica il profilo da eliminare.

Il comando obsoleto [Clear-AWSCredential](#) è ancora disponibile per la compatibilità con le versioni precedenti, ma è preferito. `Remove-AWSCredentialProfile`

Specifiche delle credenziali

Esistono vari modi per specificare le credenziali. Il metodo preferito consiste nell'identificare un profilo anziché incorporare credenziali letterali nella riga di comando. AWS Strumenti per PowerShell [individua il profilo utilizzando un ordine di ricerca descritto in Ordine di ricerca delle credenziali](#).

In Windows, AWS le credenziali archiviate nell'archivio AWS SDK sono crittografate con l'identità utente Windows che ha effettuato l'accesso. Non è possibile decrittografarle utilizzando un altro account o utilizzarle su un dispositivo diverso da quello in cui sono state create originariamente. Per eseguire attività che richiedono le credenziali di un altro utente, ad esempio un account utente utilizzato per eseguire un'attività pianificata, configura un profilo delle credenziali, come descritto nella sezione precedente, che è possibile utilizzare quando accedi al computer come tale utente. Accedi come utente che esegue attività per completare le fasi di configurazione delle credenziali e creare un profilo che funzioni per tale utente. Quindi disconnettiti e accedi di nuovo con le tue credenziali per configurare l'attività pianificata.

Note

Utilizzare il parametro comune `-ProfileName` per specificare un profilo. Questo parametro è equivalente al `-StoredCredentials` parametro delle versioni precedenti. AWS Strumenti per PowerShell Per garantire la compatibilità con le versioni precedenti, `-StoredCredentials` è ancora supportato.

Profilo predefinito (consigliato)

Tutti AWS SDKs gli strumenti di gestione sono in grado di trovare automaticamente le credenziali sul computer locale se le credenziali sono archiviate in un profilo denominato. `default` Ad esempio, se disponi di un profilo denominato `default` sul computer locale, non devi eseguire il cmdlet `Initialize-AWSDefaultConfiguration` o il cmdlet `Set-AWSCredential`. Gli strumenti utilizzano automaticamente i dati della chiave segreta e di accesso archiviati in quel profilo. Per usare una regione AWS diversa dalla regione predefinita (i risultati di `Get-DefaultAWSRegion`), puoi eseguire `Set-DefaultAWSRegion` e specificare una regione.

Se il profilo non è denominato `default`, ma si desidera utilizzarlo come profilo predefinito per la sessione corrente, eseguire `Set-AWSCredential` per impostare il profilo come predefinito.

Sebbene l'esecuzione `Initialize-AWSDefaultConfiguration` consenta di specificare un profilo predefinito per ogni PowerShell sessione, il cmdlet carica le credenziali dal profilo con nome personalizzato, ma sovrascrive il profilo con il profilo denominato. `default`

Ti consigliamo di non eseguire l'operazione `Initialize-AWSDefaultConfiguration` a meno che tu non stia eseguendo una PowerShell sessione su un' EC2 istanza Amazon che non è stata avviata con un profilo di istanza e desideri configurare manualmente il profilo delle credenziali. Tieni presente che il profilo delle credenziali in questo caso non contiene le credenziali. Il profilo di credenziali che risulta dall'esecuzione `Initialize-AWSDefaultConfiguration` su un' EC2 istanza non memorizza direttamente le credenziali, ma rimanda invece ai metadati dell'istanza (che forniscono credenziali temporanee che ruotano automaticamente). Tuttavia, archivia la regione dell'istanza. Un altro scenario che potrebbe richiedere l'esecuzione di `Initialize-AWSDefaultConfiguration` si verifica se desideri eseguire una chiamata su una regione differente rispetto a quella in cui è in esecuzione l'istanza. L'esecuzione di questo comando sostituisce definitivamente la regione archiviata nei metadati delle istanze.

```
PS > Initialize-AWSDefaultConfiguration -ProfileName MyProfileName -Region us-west-2
```

Note

Le credenziali predefinite sono incluse nell'archivio AWS SDK sotto il nome del profilo. `default` Il comando sovrascrive qualsiasi profilo esistente con lo stesso nome.

Se l' EC2 istanza è stata avviata con un profilo di istanza, ottiene PowerShell automaticamente AWS le credenziali e le informazioni sulla regione dal profilo dell'istanza. Non devi eseguire `Initialize-`

`AWSDefaultConfiguration`. L'esecuzione del `Initialize-AWSDefaultConfiguration` cmdlet su un' EC2 istanza avviata con un profilo di istanza non è necessaria, poiché utilizza gli stessi dati del profilo di istanza PowerShell già utilizzati per impostazione predefinita.

Profilo di sessione

Utilizzare `Set-AWSCredential` per specificare un profilo predefinito per una sessione particolare. Questo profilo sostituisce ogni profilo predefinito per la durata della sessione. Questo è consigliato se desideri utilizzare un profilo con nome personalizzato nella sessione anziché il profilo `default` corrente.

```
PS > Set-AWSCredential -ProfileName MyProfileName
```

Note

Nelle versioni degli strumenti per Windows PowerShell precedenti alla 1.1, il `Set-AWSCredential` cmdlet non funzionava correttamente e sovrascriveva il profilo specificato da `">".` `MyProfileName` Si consiglia di utilizzare una versione più recente degli strumenti per Windows. PowerShell

Profilo di comando

Su singoli comandi, puoi aggiungere il parametro `-ProfileName` per specificare un profilo che si applica a un solo comando. Questo profilo sostituisce qualsiasi profilo predefinito o di sessione, come illustrato nell'esempio seguente.

```
PS > Get-EC2Instance -ProfileName MyProfileName
```

Note

Quando specifichi un profilo predefinito o di sessione, puoi anche aggiungere un parametro `-Region` per sostituire una sessione predefinita o la regione della sessione. Per ulteriori informazioni, consulta [Specificare AWS le regioni](#). L'esempio seguente specifica un profilo predefinito e la regione.

```
PS > Initialize-AWSDefaultConfiguration -ProfileName MyProfileName -Region us-west-2
```

Per impostazione predefinita, si presume che il file delle credenziali AWS condivise si trovi nella cartella home dell'utente (C:\Users\username\.awssu Windows o ~/.aws su Linux). Per specificare un file delle credenziali in un percorso diverso, includi il parametro `-ProfileLocation` e specifica il percorso del file delle credenziali. L'esempio seguente specifica un file delle credenziali non predefinito per un determinato comando.

```
PS > Get-EC2Instance -ProfileName MyProfileName -ProfileLocation C:\aws_service_credentials\credentials
```

Note

Se state eseguendo PowerShell uno script durante un periodo a cui normalmente non siete connessi, ad AWS esempio, state eseguendo uno PowerShell script come attività pianificata al di fuori del normale orario di lavoro, aggiungete il `-ProfileLocation` parametro quando specificate il profilo che desiderate utilizzare e impostate il valore sul percorso del file in cui sono memorizzate le credenziali. Per essere certi che AWS Strumenti per PowerShell lo script venga eseguito con le credenziali dell'account corrette, è necessario aggiungere il `-ProfileLocation` parametro ogni volta che lo script viene eseguito in un contesto o in un processo che non utilizza un AWS account. È anche possibile copiare il file delle credenziali in un percorso accessibile al sistema locale o ad altri account che lo script utilizza per eseguire le attività.

Ordine di ricerca credenziali

Quando esegui un comando, AWS Strumenti per PowerShell cerca le credenziali nell'ordine seguente. Si interrompe quando trova le credenziali utilizzabili.

1. Credenziali letterali incorporate come parametri nella riga di comando.

Consigliamo di utilizzare i profili anziché inserire credenziali letterali nelle righe di comando.

2. Un nome specifico del profilo o la posizione del profilo.
 - Se specificate solo un nome di profilo, il comando cerca il profilo specificato nell'archivio AWS SDK e, se non esiste, il profilo specificato dal file delle credenziali AWS condivise nella posizione predefinita.
 - Se specifichi solo una posizione del profilo, il comando cerca il profilo default dal file delle credenziali.

- Se specifichi sia un nome sia un percorso, il comando cerca il profilo specificato nel file delle credenziali.

Se il profilo specificato o il percorso non è stato trovato, il comando genera un'eccezione. La ricerca procede nelle seguenti fasi solo se non hai specificato un profilo o una posizione.

3. Credenziali specificate dal parametro `-Credential`.
4. Il profilo di sessione, se esiste.
5. Il profilo predefinito nell'ordine seguente:
 - a. Il default profilo nell'archivio AWS SDK.
 - b. Il default profilo nel file delle credenziali AWS condivise.
 - c. Il AWS PS Default profilo nell'archivio AWS SDK.
6. Se il comando è in esecuzione su un' EC2 istanza Amazon configurata per utilizzare un ruolo IAM, le credenziali temporanee dell' EC2 istanza sono accessibili dal profilo dell'istanza.

Per ulteriori informazioni sull'utilizzo dei ruoli IAM per EC2 le istanze Amazon, consulta il [AWS SDK per .NET](#).

Se la ricerca ha esito negativo per individuare le credenziali specificate, il comando genera un'eccezione.

Gestione delle credenziali negli AWS Tools for PowerShell Core

I cmdlet AWS Tools for PowerShell Core accettano le chiavi di AWS accesso e segrete o i nomi dei profili di credenziali quando vengono eseguiti, in modo analogo a AWS Tools for Windows PowerShell. Quando vengono eseguiti su Windows, entrambi i moduli hanno accesso ai file di archivio delle credenziali AWS SDK per .NET (archiviati per ogni utente in `AppData\Local\AWSToolkit\RegisteredAccounts.json`).

Questo file memorizza le proprie chiavi in formato crittografato e non può essere utilizzato su un altro computer. È il primo file che AWS Strumenti per PowerShell cerca un profilo di credenziali, ed è anche il file in cui vengono archiviati i profili delle credenziali. AWS Strumenti per PowerShell [Per ulteriori informazioni sul file di archiviazione delle AWS SDK per .NET credenziali, vedere Configurazione delle credenziali. AWS](#) Il PowerShell modulo Tools for Windows attualmente non supporta la scrittura di credenziali su altri file o posizioni.

Entrambi i moduli possono leggere i profili dal file di credenziali AWS condivise utilizzato da altri AWS SDKs e da AWS CLI. In Windows, la posizione predefinita per questo file è `C:\Users\<userid>`

\.aws\credentials. Su piattaforme non Windows, questo file viene memorizzato in ~/.aws/credentials. Il parametro -ProfileLocation può essere utilizzato per puntare a un nome di file non predefinito o a una posizione del file.

L'archivio di credenziali SDK contiene le credenziali in forma crittografata utilizzando la crittografia di Windows. APIs Queste non APIs sono disponibili su altre piattaforme, quindi il AWS Tools for PowerShell Core modulo utilizza esclusivamente il file di credenziali AWS condivise e supporta la scrittura di nuovi profili di credenziali nel file di credenziali condiviso.

Gli script di esempio seguenti che utilizzano il **Set-AWSCredential** cmdlet mostrano le opzioni per la gestione dei profili di credenziali in Windows con Shell o Shell. AWSPower AWSPower NetCoremoduli.

```
# Writes a new (or updates existing) profile with name "myProfileName"
# in the encrypted SDK store file

Set-AWSCredential -AccessKey akey -SecretKey skey -StoreAs myProfileName

# Checks the encrypted SDK credential store for the profile and then
# falls back to the shared credentials file in the default location

Set-AWSCredential -ProfileName myProfileName

# Bypasses the encrypted SDK credential store and attempts to load the
# profile from the ini-format credentials file "mycredentials" in the
# folder C:\MyCustomPath

Set-AWSCredential -ProfileName myProfileName -ProfileLocation C:\MyCustomPath
\mycredentials
```

I seguenti esempi mostrano il comportamento della AWSPowerShell. NetCoremodulo sui sistemi operativi Linux o macOS.

```
# Writes a new (or updates existing) profile with name "myProfileName"
# in the default shared credentials file ~/.aws/credentials

Set-AWSCredential -AccessKey akey -SecretKey skey -StoreAs myProfileName

# Writes a new (or updates existing) profile with name "myProfileName"
# into an ini-format credentials file "~/mycustompath/mycredentials"
```

```
Set-AWSCredential -AccessKey akey -SecretKey skey -StoreAs myProfileName -
ProfileLocation ~/mycustompath/mycredentials

# Reads the default shared credential file looking for the profile "myProfileName"

Set-AWSCredential -ProfileName myProfileName

# Reads the specified credential file looking for the profile "myProfileName"

Set-AWSCredential -ProfileName myProfileName -ProfileLocation ~/mycustompath/
mycredentials
```

Credenziali condivise in AWS Strumenti per PowerShell

Gli strumenti per Windows PowerShell supportano l'uso del file di credenziali AWS condivise, in modo analogo a AWS CLI e altri. AWS SDKs Gli Strumenti per Windows PowerShell ora supportano la lettura e la scrittura e i profili di basic assume role credenziali sia nel file di credenziali.NET che nel file di credenziali AWS condiviso. session Questa funzionalità è abilitata da un nuovo namespace `Amazon.Runtime.CredentialManagement`.

Warning

Per evitare rischi per la sicurezza, non utilizzare gli utenti IAM per l'autenticazione quando sviluppi software creato ad hoc o lavori con dati reali. Utilizza invece la federazione con un provider di identità come [AWS IAM Identity Center](#).

Note

Le informazioni contenute in questo argomento riguardano le circostanze in cui è necessario ottenere e gestire manualmente le credenziali a breve o lungo termine. Per ulteriori informazioni sulle credenziali a breve e lungo termine, consulta [Altri modi di autenticazione nella and Tools](#) Reference Guide.AWS SDKs

Per le migliori pratiche di sicurezza, utilizzare AWS IAM Identity Center, come descritto in. [Configurazione dell'autenticazione degli strumenti](#)

[I nuovi tipi di profilo e l'accesso al file di credenziali AWS condiviso sono supportati dai seguenti parametri che sono stati aggiunti ai cmdlet relativi alle credenziali, Initialize- AWSDefault](#)

[Configuration, New- e Set- AWSCredential AWSCredential](#) Nei cmdlet del servizio, puoi fare riferimento ai nuovi profili aggiungendo il parametro comune `-ProfileName`.

Utilizzo di un ruolo IAM con gli AWS Strumenti per PowerShell

Il file di credenziali AWS condiviso consente tipi di accesso aggiuntivi. Ad esempio, puoi accedere alle tue AWS risorse utilizzando un ruolo IAM anziché le credenziali a lungo termine di un utente IAM. A tale scopo, devi disporre di un profilo standard con autorizzazioni per assumere il ruolo. Quando gli dici AWS Strumenti per PowerShell di utilizzare un profilo che ha specificato un ruolo, AWS Strumenti per PowerShell cerca il profilo identificato dal `SourceProfile` parametro. Tali credenziali vengono utilizzate per richiedere credenziali temporanee per il ruolo specificato dal parametro `RoleArn`. Puoi richiedere l'utilizzo di un dispositivo di autenticazione a più fattori (Multi-Factor Authentication (MFA)) o di un codice `ExternalId` quando terze parti assumono il ruolo.

Nome parametro	Descrizione
<code>ExternalId</code>	L'ID esterno definito dall'utente da utilizzare quando assume un ruolo, se richiesto dal ruolo. In genere, è necessario solo quando deleghi l'accesso al tuo account a terze parti. La terza parte deve includere il parametro <code>ExternalId</code> come parametro quando assume il ruolo assegnato. Per ulteriori informazioni, consulta Come utilizzare un ID esterno per concedere l'accesso alle AWS risorse a terzi nella Guida per l'utente IAM.
<code>MfaSerial</code>	Il numero seriale MFA da utilizzare quando assume un ruolo, se richiesto dal ruolo. Per ulteriori informazioni, consulta la sezione Utilizzo dell'autenticazione a più fattori (MFA) in AWS nella Guida per l'utente IAM.
<code>RoleArn</code>	L'ARN del ruolo da assumere per assumere le credenziali di ruolo. Per ulteriori informazioni sulla creazione e sull'utilizzo dei ruoli IAM, consulta la sezione relativa ai ruoli IAM nella Guida per l'utente IAM.

Nome parametro	Descrizione
SourceProfile	Il nome del profilo di origine da utilizzare assumendo credenziali di ruolo. Le credenziali trovate in questo profilo vengono utilizzate per assumere il ruolo specificato dal parametro RoleArn.

Configurazione dei profili per assumere un ruolo

Di seguito è riportato un esempio che mostra come configurare un profilo fonte che consente di assumere direttamente un ruolo IAM.

Il primo comando crea un profilo di origine a cui fa riferimento il profilo del ruolo. Il secondo comando crea il profilo che quel ruolo assumerà. Il terzo comando mostra le credenziali per il profilo del ruolo.

```
PS > Set-AWSCredential -StoreAs my_source_profile -AccessKey access_key_id -
SecretKey secret_key
PS > Set-AWSCredential -StoreAs my_role_profile -SourceProfile my_source_profile -
RoleArn arn:aws:iam::123456789012:role/role-i-want-to-assume
PS > Get-AWSCredential -ProfileName my_role_profile
```

```
SourceCredentials          RoleArn
-----
RoleSessionName           Options
-----
-----
Amazon.Runtime.BasicAWSCredentials arn:aws:iam::123456789012:role/
role-i-want-to-assume aws-dotnet-sdk-session-636238288466144357
Amazon.Runtime.AssumeRoleAWSCredentialsOptions
```

Per utilizzare questo profilo di ruolo con i cmdlet di PowerShell servizio Tools for Windows, aggiungi il parametro `-ProfileName` common al comando per fare riferimento al profilo del ruolo. L'esempio seguente utilizza il profilo del ruolo definito nell'esempio precedente per accedere al [Get-S3Bucket](#) cmdlet. AWS Strumenti per PowerShell cerca le credenziali `my_source_profile`, utilizza tali credenziali per chiamare `AssumeRole` conto dell'utente e quindi utilizza quelle credenziali di ruolo temporanee per chiamare `Get-S3Bucket`.

```
PS > Get-S3Bucket -ProfileName my_role_profile
```

CreationDate	BucketName
-----	-----
2/27/2017 8:57:53 AM	4ba3578c-f88f-4d8b-b95f-92a8858dac58-bucket1
2/27/2017 10:44:37 AM	2091a504-66a9-4d69-8981-aaef812a02c3-bucket2

Utilizzo dei tipi di credenziali del profilo

Per impostare il tipo di credenziali del profilo, comprendere quali parametri forniscono le informazioni necessarie per il tipo di profilo.

Tipo di credenziali	Parametri da utilizzare
Base	-AccessKey
Queste sono le credenziali a lungo termine per un utente IAM	-SecretKey
Sessione: Queste sono le credenziali a breve termine per un ruolo IAM recuperabili manualmente, ad esempio chiamando direttamente il cmdlet Use- STSRole	-AccessKey -SecretKey -SessionToken
Ruolo:	-SourceProfile
Queste sono credenziali a breve termine per un ruolo IAM che gli AWS Strumenti per PowerShell recuperano per tuo conto.	-RoleArn facoltativo: -ExternalId facoltativo: -MfaSerial

Il parametro comune **ProfilesLocation**

È possibile utilizzare -ProfileLocation per scrivere nel file delle credenziali condivise e istruire un cmdlet per leggere il file delle credenziali. L'aggiunta del -ProfileLocation parametro controlla se Tools for Windows PowerShell utilizza il file di credenziali condiviso o il file di credenziali.NET. La tabella seguente descrive come funziona il parametro in Tools for Windows. PowerShell

Valore posizione profilo	Comportamento della risoluzione del profilo
null (non impostato) o vuoto	In primo luogo, cerca il file delle credenziali .NET per un profilo con il nome specificato. Se il profilo non viene trovato, cerca nel file delle credenziali AWS condivise in <i>(user's home directory)</i> \.aws\credentials .
Il percorso di un file nel formato di file con credenziali AWS condivise	Cerca solo un profilo con uno specifico nome nel file specificato.

Salvare le credenziali in un file delle credenziali

Per scrivere e salvare le credenziali in uno dei due file delle credenziali, eseguire il cmdlet `Set-AWSCredential`. Gli esempi seguenti mostrano come fare: Il primo comando utilizza `Set-AWSCredential` con `-ProfileLocation` per aggiungere chiavi segrete e di accesso a un profilo specificato dal parametro `-ProfileName`. Nella seconda riga, eseguire il cmdlet [Get-Content](#) per visualizzare il contenuto del file delle credenziali.

```
PS > Set-AWSCredential -ProfileLocation C:\Users\auser\.aws\credentials -ProfileName
    basic_profile -AccessKey access_key2 -SecretKey secret_key2
PS > Get-Content C:\Users\auser\.aws\credentials

aws_access_key_id=access_key2
aws_secret_access_key=secret_key2
```

Visualizzazione dei profili delle credenziali

Eseguire il `AWSCredential` cmdlet [Get-](#) e aggiungere il `-ListProfileDetail` parametro per restituire i tipi e le posizioni dei file delle credenziali e un elenco di nomi di profilo.

```
PS > Get-AWSCredential -ListProfileDetail

ProfileName                StoreTypeName                ProfileLocation
-----
source_profile             NetSDKCredentialsFile
assume_role_profile        NetSDKCredentialsFile
basic_profile              SharedCredentialsFile C:\Users\auser\.aws\credentials
```

Rimozione dei profili delle credenziali

Per rimuovere i profili di credenziali, eseguire il nuovo cmdlet [Remove-Profile](#). [AWSCredential Clear-AWSCredential](#) è obsoleto, ma è ancora disponibile per la compatibilità con le versioni precedenti.

Note importanti

Solo [Initialize-AWSDefault Configuration](#), [New- e Set-AWSCredential](#) [AWSCredential](#) supportano i parametri per i profili di ruolo. Non puoi specificare i parametri del ruolo direttamente in un comando come `Get-S3Bucket -SourceProfile source_profile_name -RoleArn arn:aws:iam::999999999999:role/role_name`. Ciò non funziona perché i cmdlet di servizio non supportano direttamente i parametri `SourceProfile` o `RoleArn`. Devi invece archiviare tali parametri in un profilo, quindi chiamare il comando con il parametro `-ProfileName`.

Caratteristiche del AWS Tools for Windows PowerShell

In alcuni argomenti di questa sezione vengono fornite informazioni sulle funzionalità degli Strumenti per Windows PowerShell che potrebbe essere necessario prendere in considerazione durante la creazione di progetti e script. Gli altri argomenti di questa sezione forniscono informazioni sui modi avanzati di configurazione degli strumenti, dell'ambiente e dei progetti. Assicuratevi innanzitutto di aver [installato](#) e [configurato](#) gli strumenti.

Per informazioni sullo sviluppo di software per AWS servizi specifici e esempi di codice, consulta [Lavora con AWS i servizi](#). Per ulteriori esempi di codice, vedere [Strumenti per esempi di codice PowerShell V4](#).

Argomenti

- [Osservabilità](#)

Osservabilità

L'osservabilità è la misura in cui lo stato attuale di un sistema può essere dedotto dai dati che emette. I dati emessi vengono comunemente definiti telemetria. [Per ulteriori informazioni sulla telemetria durante l'utilizzo dei AWS servizi, consulta Observability nella Developer Guide.AWS SDK per .NET](#)

Il codice seguente mostra un esempio di come l'osservabilità può essere abilitata in AWS Strumenti per PowerShell

```
<#
    This is an example of generating telemetry for AWS Tools for PowerShell.
    Each cmdlet that interacts with an Amazon Web Service creates a trace containing
    spans
    for underlying processes and AWS SDK for .NET operations.
    This example is written using PowerShell 7 and .NET 8.
    It requires the installation of the .NET CLI tool.
    Note that implementation varies by the exporter/endpoint, which is not specified in
    this example.
    For more information, see https://opentelemetry.io/docs/languages/net/exporters/.
#>

# Set this value to a common folder path on your computer for local development of code
# repositories.
$devProjectsPath = [System.IO.Path]::Join('C:', 'Dev', 'Repos')
```

```
# If these values are changed, update the hardcoded method invocation toward the end of
this script.
# Values must follow constraints for namespaces and classes.
$telemetryProjectName = 'ExampleAWSPowerShellTelemetryImplementation'
$serviceName = 'ExamplePowerShellService'

# This example supposes that the OTLP exporter requires these two properties,
# but some exporters require different properties or no properties.
$telemetryEndPoint = 'https://example-endpoint-provider.io'
$telemetryHeaders = 'x-example-header=abc123'

$dllsPath = [System.IO.Path]::Join($devProjectsPath, $telemetryProjectName, 'bin',
'Release', 'net8.0', 'publish')

$telemetryProjectPath = [System.IO.Path]::Join($devProjectsPath, $telemetryProjectName)

# This script is designed to recreate the example telemetry project each time it's
executed.
Remove-Item -Path $telemetryProjectPath -Recurse -Force -ErrorAction 'SilentlyContinue'
$null = New-Item -Path $devProjectsPath -Name $telemetryProjectName -ItemType
'Directory'

<#
    Create and build a C#-based .NET 8 project that implements
    OpenTelemetry Instrumentation for the AWS Tools for PowerShell.
#>

Set-Location -Path $telemetryProjectPath

dotnet new classlib

# Other exporters are available.
# For more information, see https://opentelemetry.io/docs/languages/net/exporters/.
dotnet add package OpenTelemetry.Exporter.OpenTelemetryProtocol
dotnet add package OpenTelemetry.Instrumentation.AWS

$classContent = @"
using OpenTelemetry;
using OpenTelemetry.Resources;
using OpenTelemetry.Trace;

namespace Example.Telemetry;
```

```

public class AWSToolsForPowerShellTelemetry
{
    public static void InitializeAWSInstrumentation()
    {
        Sdk.CreateTracerProviderBuilder()
            .ConfigureResource(e => e.AddService("$ServiceName"))
            .AddAWSInstrumentation()
            // Exporters vary so options might need to be changed or omitted.
            .AddOtlpExporter(options =>
            {
                options.Endpoint = new Uri("$telemetryEndPoint");
                options.Headers = "$telemetryHeaders";
            })
            .Build();
    }
}
"@

$csFilePath = [System.IO.Path]::Join($telemetryProjectPath, ($serviceName + '.cs'))
Set-Content -Path $csFilePath -Value $classContent

dotnet build
dotnet publish -c Release

<#
    Add additional modules here for any other cmdlets that you require.
    Beyond this point, additional AWS Tools for PowerShell modules will fail to import
    due to conflicts with the AWS SDK for .NET assemblies that are added next.
#>

Import-Module -Name 'AWS.Tools.Common'
Import-Module -Name 'AWS.Tools.DynamoDBv2'

# Load assemblies for the telemetry project, excluding the AWS SDK for .NET assemblies
# that were already loaded by importing AWS Tools for PowerShell modules.
$dlls = (Get-ChildItem $dllsPath -Filter *.dll -Recurse ).FullName
$AWSSDKAssembliesAlreadyLoaded =
    [Threading.Thread]::GetDomain().GetAssemblies().Location | Where-Object {$_. -like
        '*AWSSDK*' } | Split-Path -Leaf
$dlls.Where{$AWSSDKAssembliesAlreadyLoaded -notcontains ($_. | Split-Path -
    Leaf)}.ForEach{Add-Type -Path $_}

# Invoke the method defined earlier in this script.

```

```
[Example.Telemetry.AWSToolsForPowerShellTelemetry]::InitializeAWSInstrumentation()
```

```
<#
```

```
    Now telemetry will be exported for AWS Tools for PowerShell cmdlets  
    that are invoked directly or indirectly.
```

```
    Execute this cmdlet or execute your own PowerShell script.
```

```
#>
```

```
Get-DDBTable -TableName 'DotNetTests-HashTable' -Region 'us-east-1'
```

Lavora con AWS i servizi in AWS Strumenti per PowerShell

Questa sezione fornisce esempi di utilizzo di AWS Strumenti per PowerShell per accedere ai AWS servizi. Questi esempi aiutano a dimostrare come utilizzare i cmdlet per eseguire attività effettive AWS . Questi esempi si basano sui cmdlet forniti da Tools for. PowerShell Per vedere quali cmdlet sono disponibili, consulta [AWS Strumenti per PowerShell Cmdlet Reference](#).

PowerShell Codifica della concatenazione di file

Alcuni cmdlet consentono di AWS Strumenti per PowerShell modificare i file o i record esistenti in uso. AWS Un esempio è `Edit-R53ResourceRecordSet` che chiama l'[ChangeResourceRecordSets](#) API per Amazon Route 53.

Quando modifichi o concateni file in versioni PowerShell 5.1 o precedenti, PowerShell codifica l'output in UTF-16, non in UTF-8. In questo modo si possono aggiungere caratteri indesiderati e creare risultati non validi. Un editor esadecimale è in grado di visualizzare i caratteri indesiderati.

Per evitare di convertire l'output del file in UTF-16, è possibile reindirizzare il comando nel `Out-File` cmdlet e specificare la codifica UTF-8, PowerShell come illustrato nell'esempio seguente:

```
PS > *some file concatenation command* | Out-File filename.txt -Encoding utf8
```

Se si eseguono AWS CLI comandi dall'interno della console, si applica lo stesso comportamento. PowerShell È possibile reindirizzare l'output di un AWS CLI comando `Out-File` nella PowerShell console. Altri cmdlet, ad esempio `Export-Csv` o `Export-Clixml`, inoltre, dispongono di un parametro `Encoding`. Per un elenco completo dei cmdlet che dispongono di un parametro `Encoding` e che consentono di correggere la codifica di un file concatenato di output, esegui il comando seguente:

```
PS > Get-Command -ParameterName "Encoding"
```

Note

PowerShell 6.0 e versioni successive, incluso PowerShell Core, mantengono automaticamente la codifica UTF-8 per l'output di file concatenati.

Oggetti PowerShell restituiti per gli strumenti

Per renderlo AWS Strumenti per PowerShell più utile in un PowerShell ambiente nativo, l'oggetto restituito da un AWS Strumenti per PowerShell cmdlet è un oggetto.NET, non l'oggetto di testo JSON che in genere viene restituito dall'API corrispondente nell'SDK. AWS Ad esempio, `Get-S3Bucket` restituisce una raccolta `Buckets`, non un oggetto di risposta JSON Amazon S3. La `Buckets` raccolta può essere inserita nella PowerShell pipeline e con essa è possibile interagire nei modi appropriati. Allo stesso modo, `Get-EC2Instance` emette una raccolta di oggetti .NET `Reservation`, non un oggetto risultato JSON `DescribeEC2Instances`. Questo comportamento è dovuto alla progettazione e consente all' AWS Strumenti per PowerShell esperienza di essere più coerente con l'idiomatico. PowerShell

Le risposte effettive del servizio sono disponibili se ne hai bisogno. Sono archiviate come proprietà note sugli oggetti restituiti. Per le azioni API che supportano i campi `NextToken`, queste sono anche collegate come proprietà note.

Amazon EC2

Questa sezione illustra i passaggi necessari per avviare un' EC2 istanza Amazon, incluso come:

- Recupera un elenco di Amazon Machine Images (AMIs).
- Creare una coppia di chiavi per l'autenticazione SSH.
- Crea e configura un gruppo EC2 di sicurezza Amazon.
- Avviare l'istanza e recuperare le relative informazioni.

Amazon S3

La sezione esamina le fasi necessarie per creare un website statico ospitato in Amazon S3. Dimostra come fare a:

- Creare ed eliminare bucket Amazon S3.
- Caricare file in un bucket Amazon S3 come oggetti.
- Eliminare gli oggetti da un bucket Amazon S3.
- Designare un bucket Amazon S3 come website.

[AWS Lambda e AWS Strumenti per PowerShell](#)

Questa sezione fornisce una breve panoramica del PowerShell modulo AWS Lambda Tools for e descrive i passaggi necessari per configurare il modulo.

[Amazon SNS e Amazon SQS](#)

Questa sezione esamina le fasi necessarie per sottoscrivere una coda Amazon SQS a un argomento Amazon SNS. Dimostra come fare a:

- Creazione di un argomento Amazon SNS.
- Creare una coda Amazon SQS.
- Sottoscrivere la coda all'argomento .
- Inviare un messaggio all'argomento.
- Ricevere il messaggio dalla coda.

[CloudWatch](#)

Questa sezione mostra un esempio di come pubblicare dati personalizzati su CloudWatch.

- Pubblica una metrica personalizzata nella tua CloudWatch dashboard.

Vedi anche

- [Inizia con AWS Tools for Windows PowerShell](#)

Argomenti

- [Amazon S3 e strumenti per Windows PowerShell](#)
- [Amazon EC2 e strumenti per Windows PowerShell](#)
- [AWS Lambda e AWS Strumenti per PowerShell](#)
- [Amazon SQS, Amazon SNS e strumenti per Windows PowerShell](#)
- [CloudWatch dal AWS Tools for Windows PowerShell](#)

- [Utilizzo del ClientConfig parametro nei cmdlet](#)

Amazon S3 e strumenti per Windows PowerShell

In questa sezione, creiamo un sito Web statico AWS Tools for Windows PowerShell utilizzando Amazon S3 e CloudFront. Durante il processo, dimostriamo una serie di operazioni comuni con questi servizi. Questa spiegazione passo per passo è modellata sulla base della Guida introduttiva per l'[Hosting di un website statico](#), in cui viene descritto un processo analogo utilizzando la [Console di gestione AWS](#).

I comandi mostrati qui presuppongono che tu abbia impostato credenziali predefinite e una regione predefinite per la tua PowerShell sessione. Pertanto, le credenziali e le regioni non sono incluse nelle invocazioni dei cmdlet.

Note

Al momento non esiste un'API Amazon S3 per rinominare un bucket o un oggetto e, pertanto, non esiste un singolo PowerShell cmdlet Tools for Windows per eseguire questa attività. Per rinominare un oggetto in S3, si consiglia di copiare l'oggetto in uno con un nuovo nome, eseguendo il cmdlet, e quindi di eliminare l'oggetto originale eseguendo il [Copy-S3Object](#) cmdlet. [Remove-S3Object](#)

Consulta anche

- [Lavora con AWS i servizi in AWS Strumenti per PowerShell](#)
- [Hosting di un sito Web statico su Amazon S3](#)
- [Console Amazon S3](#)

Argomenti

- [Creare un bucket Amazon S3, verificare la sua Regione e rimuoverlo se si desidera](#)
- [Configurare un bucket Amazon S3 come website e abilitare la registrazione](#)
- [Caricare oggetti in un bucket Amazon S3](#)
- [Eliminare bucket e oggetti Amazon S3](#)
- [Caricamento dei contenuti di testo in linea su Amazon S3](#)

Creare un bucket Amazon S3, verificare la sua Regione e rimuoverlo se si desidera

Utilizza il cmdlet `New-S3Bucket` per creare un nuovo bucket Amazon S3. Negli esempi seguenti viene creato un bucket denominato `website-example`. Il nome del bucket deve essere univoco in tutte le regioni. Nell'esempio viene creato il bucket nella regione `us-west-1`.

```
PS > New-S3Bucket -BucketName website-example -Region us-west-2
```

CreationDate	BucketName
-----	-----
8/16/19 8:45:38 PM	website-example

È possibile verificare la regione in cui si trova il bucket utilizzando il cmdlet `Get-S3BucketLocation`.

```
PS > Get-S3BucketLocation -BucketName website-example
```

Value

us-west-2

Al termine del tutorial, puoi usare la riga seguente per rimuovere questo bucket. Suggeriamo di lasciare questo bucket al suo posto visto che lo utilizziamo negli esempi successivi.

```
PS > Remove-S3Bucket -BucketName website-example
```

Il completamento del processo di rimozione del bucket potrebbe richiedere tempo. Se provi a ricreare immediatamente un bucket con lo stesso nome, il cmdlet `New-S3Bucket` può non avere esito positivo fino a quando il vecchio non è completamente eliminato.

Vedi anche

- [Lavora con AWS i servizi in AWS Strumenti per PowerShell](#)
- [Inserimento del bucket \(Amazon S3 Service Reference\)](#)
- [AWS PowerShell Regioni per Amazon S3](#)

Configurare un bucket Amazon S3 come website e abilitare la registrazione

Utilizza il cmdlet `Write-S3BucketWebsite` per configurare un bucket Amazon S3 come website statico. L'esempio seguente specifica un nome `index.html` per la pagina Web di contenuti e un nome `error.html` per la pagina Web di errore predefinita. Questo cmdlet non consente di creare tali pagine. Devono essere [caricate come oggetti di Amazon S3](#).

```
PS > Write-S3BucketWebsite -BucketName website-example -  
WebsiteConfiguration_IndexDocumentSuffix index.html -WebsiteConfiguration_ErrorDocument  
error.html  
RequestId      : A1813E27995FFDDD  
AmazonId2      : T7h1D0eLqA5Q2XfTe8j2q3SLop3/5XwhUU3RyJBGHU/LnC+CIWLeGgP0MY24xA1I  
ResponseStream :  
Headers        : {x-amz-id-2, x-amz-request-id, Content-Length, Date...}  
Metadata       : {}  
ResponseXml    :
```

Vedi anche

- [Lavora con AWS i servizi in AWS Strumenti per PowerShell](#)
- [Inserire sito Web bucket \(Amazon S3 API Reference\)](#)
- [Inserire Bucket ACL \(Amazon S3 API Reference\)](#)

Caricare oggetti in un bucket Amazon S3

Utilizza il cmdlet `Write-S3Object` per caricare file dal file system locale in un bucket Amazon S3 come oggetti. L'esempio seguente crea e carica due semplici file HTML in un bucket Amazon S3 e verifica che gli oggetti siano stati caricati. Il parametro `-File` in `Write-S3Object` specifica il nome del file nel file system locale. Il parametro `-Key` specifica il nome che avrà l'oggetto corrispondente in Amazon S3.

Amazon deduce automaticamente il tipo di contenuto degli oggetti dalle estensioni dei file, in questo caso, `".html"`.

```
PS > # Create the two files using here-strings and the Set-Content cmdlet  
PS > $index_html = @"  
>> <html>  
>>   <body>  
>>     <p>
```

```

>>      Hello, World!
>>      </p>
>>      </body>
>> </html>
>> "@"
>>
PS > $index_html | Set-Content index.html
PS > $error_html = @"
>> <html>
>>   <body>
>>     <p>
>>       This is an error page.
>>     </p>
>>   </body>
>> </html>
>> @"
>>
>>$error_html | Set-Content error.html
>># Upload the files to Amazon S3 using a foreach loop
>>foreach ($f in "index.html", "error.html") {
>> Write-S3Object -BucketName website-example -File $f -Key $f -CannedACLName public-
read
>> }
>>
PS > # Verify that the files were uploaded
PS > Get-S3BucketWebsite -BucketName website-example

IndexDocumentSuffix                                ErrorDocument
-----
index.html                                            error.html

```

Opzioni di ACL predefinite

I valori per specificare canned ACLs con Tools for Windows PowerShell sono gli stessi utilizzati da AWS SDK per .NET. Tieni presente, tuttavia, che questi sono diversi dai valori utilizzati dall'operazione `Amazon S3Put Object`. Gli strumenti per Windows PowerShell supportano quanto segue preimpostato: ACLs

- NoACL
- private
- public-read
- public-read-write

- `aws-exec-read`
- `authenticated-read`
- `bucket-owner-read`
- `bucket-owner-full-control`
- `log-delivery-write`

Per ulteriori informazioni su queste impostazioni di ACL predefinite, consultare la pagina [Panoramica lista di controllo accessi \(ACL\)](#).

Nota riguardo il caricamento in più parti

Se si utilizza l'API di Amazon S3 per caricare un file di dimensioni superiori a 5 GB, è necessario utilizzare il caricamento in più parti. Tuttavia, il `Write-S3Object` cmdlet fornito da Tools for Windows PowerShell può gestire in modo trasparente i caricamenti di file superiori a 5 GB.

Test del sito Web

A questo punto, è possibile testare il sito Web accedendovi tramite un browser. URLs per i siti Web statici ospitati in Amazon S3 seguono un formato standard.

```
http://<bucket-name>.s3-website-<region>.amazonaws.com
```

Per esempio:

```
http://website-example.s3-website-us-west-1.amazonaws.com
```

Vedi anche

- [Lavora con AWS i servizi in AWS Strumenti per PowerShell](#)
- [Inserisci oggetto \(Amazon S3 API Reference\)](#)
- [In scatola ACLs \(riferimento all'API Amazon S3\)](#)

Eliminare bucket e oggetti Amazon S3

Questa sezione descrive come eliminare il sito Web creato nelle sezioni precedenti. È possibile semplicemente eliminare gli oggetti per i file HTML e quindi eliminare il bucket Amazon S3 per il sito.

Innanzitutto, esegui il cmdlet `Remove-S3Object` per eliminare gli oggetti per i file HTML dal bucket Amazon S3.

```
PS > foreach ( $obj in "index.html", "error.html" ) {  
>> Remove-S3Object -BucketName website-example -Key $obj  
>> }  
>>  
IsDeleteMarker  
-----  
False
```

La risposta `False` è un artefatto previsto del modo in cui Amazon S3 elabora la richiesta. In questo contesto, non indica un problema.

Ora puoi eseguire il cmdlet `Remove-S3Bucket` per eliminare il bucket Amazon S3 ormai vuoto per il sito.

```
PS > Remove-S3Bucket -BucketName website-example  
  
RequestId      : E480ED92A2EC703D  
AmazonId2      : k6tqaqC1nMkoeYwbuJXUx1/UDa49BJd6dfLN0Ls1mWYNPHjbc8/Nyvm6AGbWcc2P  
ResponseStream :  
Headers        : {x-amz-id-2, x-amz-request-id, Date, Server}  
Metadata       : {}  
ResponseXml    :
```

Nella versione 1.1 e successive di AWS Strumenti per PowerShell, è possibile aggiungere il `-DeleteBucketContent` parametro a `Remove-S3Bucket`, che elimina prima tutti gli oggetti e le versioni degli oggetti nel bucket specificato prima di provare a rimuovere il bucket stesso. A seconda del numero degli oggetti o delle versioni degli oggetti nel bucket, questa operazione potrebbe richiedere una notevole quantità di tempo. Nelle versioni degli Strumenti per Windows PowerShell precedenti alla 1.1, il bucket doveva essere vuoto prima di poterlo eliminare. `Remove-S3Bucket`

Note

A meno che non si aggiunga il `-Force` parametro, AWS Strumenti per PowerShell richiede una conferma prima dell'esecuzione del cmdlet.

Vedi anche

- [Lavora con AWS i servizi in AWS Strumenti per PowerShell](#)
- [Eliminazione di un oggetto \(Amazon S3 API Reference\)](#)
- [DeleteBucket \(Riferimento all'API Amazon S3\)](#)

Caricamento dei contenuti di testo in linea su Amazon S3

Il cmdlet `Write-S3Object` supporta la possibilità di caricare contenuti di testo in linea su Amazon S3. Utilizzando il parametro `-Content` (alias `-Text`), è possibile specificare contenuti basati su testo che devono essere caricati su Amazon S3 senza la necessità di trasferirli prima in un file. Il parametro accetta stringhe semplici di una sola riga e anche here strings che contengono più righe.

```
PS > # Specifying content in-line, single line text:
PS > write-s3object amzn-s3-demo-bucket -key myobject.txt -content "file content"

PS > # Specifying content in-line, multi-line text: (note final newline needed to end
in-line here-string)
PS > write-s3object amzn-s3-demo-bucket -key myobject.txt -content @"
>> line 1
>> line 2
>> line 3
>> "@
>>

PS > # Specifying content from a variable: (note final newline needed to end in-line
here-string)
PS > $x = @"
>> line 1
>> line 2
>> line 3
>> "@
>>
PS > write-s3object amzn-s3-demo-bucket -key myobject.txt -content $x
```

Amazon EC2 e strumenti per Windows PowerShell

Puoi eseguire attività comuni relative ad Amazon EC2 utilizzando AWS Strumenti per PowerShell.

I comandi di esempio mostrati qui presuppongono che tu abbia impostato credenziali predefinite e una regione predefinite per la tua PowerShell sessione. Pertanto, non includiamo le credenziali o la

regione quando invochiamo il cmdlet. Per ulteriori informazioni, consulta [Inizia con AWS Tools for Windows PowerShell](#).

Argomenti

- [Creazione di una coppia di chiavi](#)
- [Creare un gruppo di sicurezza utilizzando Windows PowerShell](#)
- [Trova l'immagine di una macchina Amazon utilizzando Windows PowerShell](#)
- [Avvio di un' EC2 istanza Amazon tramite Windows PowerShell](#)

Creazione di una coppia di chiavi

L'New-EC2KeyPair esempio seguente crea una coppia di key pair e la memorizza nella PowerShell variabile \$myPSKeyPair

```
PS > $myPSKeyPair = New-EC2KeyPair -KeyName myPSKeyPair
```

Indirizza l'oggetto della coppia di chiavi nel cmdlet Get-Member per visualizzare la struttura dell'oggetto.

```
PS > $myPSKeyPair | Get-Member
```

```
TypeName: Amazon.EC2.Model.KeyPair
```

Name	MemberType	Definition
----	-----	-----
Equals	Method	bool Equals(System.Object obj)
GetHashCode	Method	int GetHashCode()
GetType	Method	type GetType()
ToString	Method	string ToString()
KeyFingerprint	Property	System.String KeyFingerprint {get;set;}
KeyMaterial	Property	System.String KeyMaterial {get;set;}
KeyName	Property	System.String KeyName {get;set;}

Indirizzare l'oggetto della coppia di chiavi nel cmdlet Format-List per visualizzare i valori di KeyName, KeyFingerprint e i membri di KeyMaterial. (L'output è stato accorciato per ragioni di leggibilità).

```
PS > $myPSKeyPair | Format-List KeyName, KeyFingerprint, KeyMaterial
```

```

KeyName       : myPSKeyPair
KeyFingerprint : 09:06:70:8e:26:b6:e7:ef:8f:fe:4a:1d:bc:9c:6a:63:11:ac:ad:3c
KeyMaterial    : ----BEGIN RSA PRIVATE KEY----
                 MIIIEogIBAAKCAQEAKK+ANYUS9c7niNjYfaCn6KYj/D0I6djnFoQE...
                 Mz6btoxPcE7EMeH1wySUP8nouAS9xbl9l7+VkD74bN9KmNcPa/Mu...
                 Zyn4vVe0Q5il/MpkrRogHq0B0rigeTeV5Yc3lv00RFFPu0Kz4kcm...
                 w3Jg8dKsWn0pl0pX7V3sRC02KgJIbejQUvBFGi50QK9bm4tXBIeC...
                 daxKIAQMtDUdmBDrhR1/YMv8itFe5DiLLbq7Ga+FDcS85NstBa3h...
                 iuskGkcvgWkcFQkLmRHRoDpPb+0dFsZtjHZDpMVfMA9tT8EdbkEF...
                 3SrNeqZPsxJJIX0odb3CxLJpg75JU5kyWnb0+sDNVHoJiZCULCr0...
                 GGLLfEgB95KjGIk7zEv2Q7K6s+DHclrDeMZWa7KFNRZuCuX7jssC...
                 x098abxMr3o3TNU6p1ZYRJEQ0oJr0W+kc+/8SWb8NIwFLtwhmJEy...
                 1BX9X8WFX/A8VLHrT1elrKmlkNECgYEAwltkV1p0JAFhz9p7ZFEv...
                 vvVsPaF0Ev9bk9pqhx269PB50x2KokwCagDMMaYvasWobuLmNu/1...
                 lmwRx7KTeQ7W1J30LgxHA1QNMkip9c4Tb3q9vVc3t/fPf8vwfJ8C...
                 63g6N6rk2FkHZX1E62BgbewUd3eZ0S05Ip4VUdvtGcuc8/qa+e5C...
                 KXgyt9n164pMv+VaXfXkZhdLAdY0Khc9TGB9++VMSG5TrD15YJId...
                 gYALEI7m1jJKpHWAes0hiemw5VmKyIZpzGstSJsFStER1AjiETDH...
                 YAtnI4J8dRyP9I7B0V0n3wNfIjk85gi1/00c+j8S65giLAfndWGR...
                 9R9wIkm5BMUcSRRcDy0yuwKBgEbkOnGGSD0ah4HkvrUkepIbUDTD...
                 AnEBM1cXI5UT7BfKInpUihZi59QhgdK/hk0SmWhlZGWikJ5VizBf...
                 drkBr/vTKVRMTi3lVFB7KkIV1xJxC5E/BZ+YdZEpWoCZAoGAC/Cd...
                 TTld5N6opg0XAcQJwzqoGa9ZMwc5Q9f4bfRc67emkw0ZAAwSsvWR...
                 x302duuy7/smTwWwskEWRK5IrUxoMv/VVYaqdzc0ajwieNrblr7c...
                 -----END RSA PRIVATE KEY-----

```

Il membro `KeyMaterial` archivia la chiave privata per la coppia di chiavi. La chiave pubblica è memorizzata in AWS. Non è possibile recuperare la chiave pubblica da AWS, ma è possibile verificare la chiave pubblica confrontando `KeyFingerprint` la chiave privata con quella restituita dalla AWS chiave pubblica.

Visualizzazione dell'impronta della coppia di chiave

È possibile utilizzare il cmdlet `Get-EC2KeyPair` per visualizzare l'impronta per la coppia di chiavi.

```

PS > Get-EC2KeyPair -KeyName myPSKeyPair | format-list KeyName, KeyFingerprint

KeyName       : myPSKeyPair
KeyFingerprint : 09:06:70:8e:26:b6:e7:ef:8f:fe:4a:1d:bc:9c:6a:63:11:ac:ad:3c

```

Archiviazione della chiave privata

Per archiviare la chiave privata in un file, indirizzare il membro `KeyFingerMaterial` al cmdlet `Out-File`.

```
PS > $myPSKeyPair.KeyMaterial | Out-File -Encoding ascii myPSKeyPair.pem
```

È necessario specificare `-Encoding ascii` quando si scrive la chiave privata per un file. In caso contrario, strumenti come `openssl` potrebbero non essere in grado di leggere il file correttamente. È possibile verificare la correttezza del formato del file risultante utilizzando un comando, ad esempio il seguente:

```
PS > openssl rsa -check < myPSKeyPair.pem
```

(Lo `openssl` strumento non è incluso nella versione AWS Strumenti per PowerShell o nella AWS SDK per .NET.)

Rimozione di una coppia di chiavi

Per avviare un'istanza e connetterti a essa è necessaria una coppia di chiavi. Al termine dell'utilizzo di una coppia di chiavi, puoi rimuoverla. Per rimuovere la chiave pubblica da AWS, utilizzare il `Remove-EC2KeyPair` cmdlet. Quando richiesto, premi `Enter` per rimuovere la coppia di chiavi.

```
PS > Remove-EC2KeyPair -KeyName myPSKeyPair
```

Confirm

Performing the operation "Remove-EC2KeyPair (DeleteKeyPair)" on target "myPSKeyPair".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):

La variabile `$myPSKeyPair`, esiste ancora nella PowerShell sessione corrente e contiene ancora le informazioni sulla coppia di chiavi. Anche il file `myPSKeyPair.pem` esiste. Tuttavia, la chiave privata non è più valida perché la chiave pubblica per la coppia di chiavi non è più archiviata in AWS.

Creare un gruppo di sicurezza utilizzando Windows PowerShell

È possibile utilizzare il AWS Strumenti per PowerShell per creare e configurare un gruppo di sicurezza. La risposta è l'ID del gruppo di sicurezza.

Se hai bisogno di connetterti a un'istanza, devi configurare il gruppo di sicurezza per consentire il traffico SSH (Linux) o il traffico RDP (Windows).

Argomenti

- [Prerequisiti](#)
- [Creazione di un gruppo di sicurezza per EC2 -VPC](#)

Prerequisiti

È necessario l'indirizzo IP pubblico del computer, nella notazione CIDR. Puoi ottenere l'indirizzo IP pubblico del computer locale utilizzando un servizio. Ad esempio, Amazon fornisce il seguente servizio: <http://checkip.amazonaws.com/> o <https://checkip.amazonaws.com/>. Per individuare un altro servizio che fornisca l'indirizzo IP, usa la frase di ricerca "qual è il mio indirizzo IP". Se ti stai connettendo tramite un ISP o con la protezione di un firewall senza un indirizzo IP statico, devi individuare l'intervallo degli indirizzi IP che i computer client possono utilizzare.

Warning

Se specifichi `0.0.0.0/0`, abiliti il traffico da qualsiasi indirizzo IP nel mondo. Per i protocolli SSH e RDP, potresti considerarlo accettabile per un breve periodo di tempo in un ambiente di test, ma non è sicuro per gli ambienti di produzione. In produzione, assicurati di autorizzare l'accesso solo dall'indirizzo IP individuale o dall'intervallo di indirizzi appropriato.

Creazione di un gruppo di sicurezza per EC2 -VPC

Warning

EC2-Classic è stato ritirato il 15 agosto 2022. Ti consigliamo di migrare da EC2 -Classic a un VPC. Per ulteriori informazioni, consulta il post sul blog [EC2-Classic Networking is Retiring — Ecco](#) come prepararsi.

L'esempio `New-EC2SecurityGroup` seguente aggiunge il parametro `-VpcId` per creare un gruppo di sicurezza per il VPC specificato.

```
PS > $groupid = New-EC2SecurityGroup `
    -VpcId "vpc-da0013b3" `
    -GroupName "myPSSecurityGroup" `
    -GroupDescription "EC2-VPC from PowerShell"
```

Per visualizzare la configurazione iniziale del gruppo di sicurezza, utilizza il cmdlet `Get-EC2SecurityGroup`. Per impostazione predefinita, il gruppo di sicurezza per un VPC contiene una regola che abilita tutto il traffico in uscita. Nota che non puoi fare riferimento a un gruppo di sicurezza per EC2 -VPC per nome.

```
PS > Get-EC2SecurityGroup -GroupId sg-5d293231

OwnerId           : 123456789012
GroupName         : myPSSecurityGroup
GroupId           : sg-5d293231
Description       : EC2-VPC from PowerShell
IpPermissions     : {}
IpPermissionsEgress : {Amazon.EC2.Model.IpPermission}
VpcId             : vpc-da0013b3
Tags              : {}
```

Per definire le autorizzazioni per il traffico in entrata sulla porta TCP 22 (SSH) e sulla porta TCP 3389, utilizza il cmdlet `New-Object`. Lo script di esempio seguente definisce le autorizzazioni per le porte TCP 22 e 3389 da un singolo indirizzo IP, 203.0.113.25/32.

```
$ip1 = new-object Amazon.EC2.Model.IpPermission
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")
$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")
Grant-EC2SecurityGroupIngress -GroupId $groupid -IpPermissions @( $ip1, $ip2 )
```

Per verificare che il gruppo di sicurezza sia stato aggiornato, utilizza di nuovo il cmdlet `Get-EC2SecurityGroup`.

```
PS > Get-EC2SecurityGroup -GroupIds sg-5d293231

OwnerId           : 123456789012
GroupName         : myPSSecurityGroup
GroupId           : sg-5d293231
Description       : EC2-VPC from PowerShell
```

```
IpPermissions      : {Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {Amazon.EC2.Model.IpPermission}
VpcId              : vpc-da0013b3
Tags               : {}
```

Per visualizzare le regole in entrata, puoi recuperare la proprietà `IpPermissions` dall'oggetto raccolto restituito dal comando precedente.

```
PS > (Get-EC2SecurityGroup -GroupIds sg-5d293231).IpPermissions
```

```
IpProtocol      : tcp
FromPort        : 22
ToPort          : 22
UserIdGroupPairs : {}
IpRanges        : {203.0.113.25/32}

IpProtocol      : tcp
FromPort        : 3389
ToPort          : 3389
UserIdGroupPairs : {}
IpRanges        : {203.0.113.25/32}
```

Trova l'immagine di una macchina Amazon utilizzando Windows PowerShell

Quando avvii un' EC2 istanza Amazon, specifichi un'Amazon Machine Image (AMI) che funga da modello per l'istanza. Tuttavia, la versione IDs per AWS Windows AMIs cambia frequentemente perché AWS fornisce ai nuovi AMIs utenti gli ultimi aggiornamenti e miglioramenti della sicurezza. È possibile utilizzare i [Get-EC2ImageByNames](#) cmdlet [Get-EC2Image](#) and per trovare le finestre AMIs correnti e ottenerle. IDs

Argomenti

- [Get-EC2Image](#)
- [Get-EC2ImageByNames](#)

Get-EC2Image

Il `Get-EC2Image` cmdlet recupera un elenco di quelli che è possibile utilizzare. AMIs

Utilizza il `-Owner` parametro con il valore dell'array `amazon, self` in modo `Get-EC2Image` da recuperare solo AMIs ciò che appartiene ad Amazon o a te. In questo contesto, devi fare riferimento all'utente di cui hai usato le credenziali per richiamare il cmdlet.

```
PS > Get-EC2Image -Owner amazon, self
```

È possibile definire i risultati utilizzando il parametro `-Filter`. Per specificare il filtro, creare un oggetto di tipo `Amazon.EC2.Model.Filter`. Ad esempio, utilizza il seguente filtro per visualizzare solo Windows AMIs.

```
$platform_values = New-Object 'collections.generic.list[string]'
$platform_values.add("windows")
$filter_platform = New-Object Amazon.EC2.Model.Filter -Property @{Name = "platform";
    Values = $platform_values}
Get-EC2Image -Owner amazon, self -Filter $filter_platform
```

Di seguito è riportato un esempio di uno dei valori AMIs restituiti dal cmdlet; l'output effettivo del comando precedente fornisce informazioni per molti AMIs

```
Architecture      : x86_64
BlockDeviceMappings : {/dev/sda1, xvdca, xvdcb, xvdcc...}
CreationDate      : 2019-06-12T10:41:31.000Z
Description       : Microsoft Windows Server 2019 Full Locale English with SQL Web
                   2017 AMI provided by Amazon
EnaSupport        : True
Hypervisor        : xen
ImageId           : ami-000226b77608d973b
ImageLocation     : amazon/Windows_Server-2019-English-Full-SQL_2017_Web-2019.06.12
ImageOwnerAlias   : amazon
ImageType         : machine
KernelId          :
Name              : Windows_Server-2019-English-Full-SQL_2017_Web-2019.06.12
OwnerId           : 801119661308
Platform          : Windows
ProductCodes      : {}
Public            : True
RamdiskId         :
RootDeviceName    : /dev/sda1
RootDeviceType    : ebs
SriovNetSupport   : simple
State             : available
```

```
StateReason      :  
Tags             : {}  
VirtualizationType : hvm
```

Get-EC2ImageByName

Il `Get-EC2ImageByName` cmdlet consente di filtrare l'elenco di AWS Windows in AMIs base al tipo di configurazione del server a cui si è interessati.

Quando viene eseguito senza parametri, come di seguito, il cmdlet emette il set completo di nomi di filtro corrente:

```
PS > Get-EC2ImageByName
```

```
WINDOWS_2016_BASE  
WINDOWS_2016_NANO  
WINDOWS_2016_CORE  
WINDOWS_2016_CONTAINER  
WINDOWS_2016_SQL_SERVER_ENTERPRISE_2016  
WINDOWS_2016_SQL_SERVER_STANDARD_2016  
WINDOWS_2016_SQL_SERVER_WEB_2016  
WINDOWS_2016_SQL_SERVER_EXPRESS_2016  
WINDOWS_2012R2_BASE  
WINDOWS_2012R2_CORE  
WINDOWS_2012R2_SQL_SERVER_EXPRESS_2016  
WINDOWS_2012R2_SQL_SERVER_STANDARD_2016  
WINDOWS_2012R2_SQL_SERVER_WEB_2016  
WINDOWS_2012R2_SQL_SERVER_EXPRESS_2014  
WINDOWS_2012R2_SQL_SERVER_STANDARD_2014  
WINDOWS_2012R2_SQL_SERVER_WEB_2014  
WINDOWS_2012_BASE  
WINDOWS_2012_SQL_SERVER_EXPRESS_2014  
WINDOWS_2012_SQL_SERVER_STANDARD_2014  
WINDOWS_2012_SQL_SERVER_WEB_2014  
WINDOWS_2012_SQL_SERVER_EXPRESS_2012  
WINDOWS_2012_SQL_SERVER_STANDARD_2012  
WINDOWS_2012_SQL_SERVER_WEB_2012  
WINDOWS_2012_SQL_SERVER_EXPRESS_2008  
WINDOWS_2012_SQL_SERVER_STANDARD_2008  
WINDOWS_2012_SQL_SERVER_WEB_2008  
WINDOWS_2008R2_BASE  
WINDOWS_2008R2_SQL_SERVER_EXPRESS_2012  
WINDOWS_2008R2_SQL_SERVER_STANDARD_2012
```



```

WINDOWS_2008R2_SQL_SERVER_WEB_2012
WINDOWS_2008R2_SQL_SERVER_EXPRESS_2008
WINDOWS_2008R2_SQL_SERVER_STANDARD_2008
WINDOWS_2008R2_SQL_SERVER_WEB_2008
WINDOWS_2008RTM_BASE
WINDOWS_2008RTM_SQL_SERVER_EXPRESS_2008
WINDOWS_2008RTM_SQL_SERVER_STANDARD_2008
WINDOWS_2008_BEANSTALK_IIS75
WINDOWS_2012_BEANSTALK_IIS8
VPC_NAT

```

Per restringere il set di immagini restituito, specificare uno o più nomi di filtri utilizzando il parametro **Names**.

```
PS > Get-EC2ImageByName -Names WINDOWS_2016_CORE
```

```

Architecture      : x86_64
BlockDeviceMappings : {/dev/sda1, xvdca, xvdcb, xvdcc...}
CreationDate      : 2019-08-16T09:36:09.000Z
Description       : Microsoft Windows Server 2016 Core Locale English AMI provided by
                    Amazon
EnaSupport        : True
Hypervisor        : xen
ImageId           : ami-06f2a2afca06f15fc
ImageLocation     : amazon/Windows_Server-2016-English-Core-Base-2019.08.16
ImageOwnerAlias   : amazon
ImageType        : machine
KernelId         :
Name              : Windows_Server-2016-English-Core-Base-2019.08.16
OwnerId          : 801119661308
Platform         : Windows
ProductCodes     : {}
Public           : True
RamdiskId        :
RootDeviceName    : /dev/sda1
RootDeviceType   : ebs
SriovNetSupport  : simple
State            : available
StateReason      :
Tags             : {}
VirtualizationType : hvm

```

Avvio di un' EC2 istanza Amazon tramite Windows PowerShell

Per avviare un' EC2 istanza Amazon, sono necessari la coppia di chiavi e il gruppo di sicurezza creati nelle sezioni precedenti. È inoltre necessario l'ID di una Amazon Machine Image (AMI). Per ulteriori informazioni, consulta la seguente documentazione :

- [Creazione di una coppia di chiavi](#)
- [Crea un gruppo di sicurezza utilizzando Windows PowerShell](#)
- [Trova l'immagine di una macchina Amazon utilizzando Windows PowerShell](#)

Important

Se avvii un'istanza che non rientra nel piano gratuito, i costi ti verranno addebitati dopo l'avvio dell'istanza e per il periodo di tempo in cui l'istanza viene eseguita, anche se rimane inattiva.

Argomenti

- [Avviare un'istanza in una VPC](#)
- [Avviare un'istanza Spot in una VPC](#)

Avviare un'istanza in una VPC

Warning

EC2-Classic è stato ritirato il 15 agosto 2022. Ti consigliamo di migrare da EC2 -Classic a un VPC. Per ulteriori informazioni, consulta il post sul blog [EC2-Classic Networking is Retiring — Ecco come prepararsi](#).

Il comando seguente crea una singola istanza `m1.small` nella sottorete privata specificata. Il gruppo di sicurezza deve essere valido per la sottorete specificata.

```
PS > New-EC2Instance `
    -ImageId ami-c49c0dac `
    -MinCount 1 -MaxCount 1 `
    -KeyName myPSKeyPair `
```

```
-SecurityGroupId sg-5d293231 `
-InstanceType m1.small `
-SubnetId subnet-d60013bf
```

```
ReservationId    : r-b70a0ef1
OwnerId          : 123456789012
RequesterId      :
Groups           : {}
GroupName        : {}
Instances        : {}
```

Inizialmente l'istanza è nello stato pending, ma in pochi minuti passa allo stato running. Per visualizzare le informazioni sull'istanza, utilizza il cmdlet `Get-EC2Instance`. Se hai più di un'istanza, puoi filtrare i risultati dell'ID della prenotazione con il parametro `Filter`. Innanzitutto, devi creare un oggetto di tipo `Amazon.EC2.Model.Filter`. Successivamente, devi chiamare `Get-EC2Instance` che utilizza il filtro e quindi visualizza la proprietà `Instances`.

```
PS > $reservation = New-Object 'collections.generic.list[string]'
PS > $reservation.add("r-b70a0ef1")
PS > $filter_reservation = New-Object Amazon.EC2.Model.Filter -Property @{Name =
    "reservation-id"; Values = $reservation}
PS > (Get-EC2Instance -Filter $filter_reservation).Instances
```

```
AmiLaunchIndex    : 0
Architecture      : x86_64
BlockDeviceMappings : {/dev/sda1}
ClientToken       :
EbsOptimized      : False
Hypervisor        : xen
IamInstanceProfile :
ImageId           : ami-c49c0dac
InstanceId        : i-5203422c
InstanceLifecycle :
InstanceType      : m1.small
KernelId          :
KeyName           : myPSKeyPair
LaunchTime        : 12/2/2018 3:38:52 PM
Monitoring        : Amazon.EC2.Model.Monitoring
NetworkInterfaces : {}
Placement         : Amazon.EC2.Model.Placement
Platform          : Windows
PrivateDnsName    :
PrivateIpAddress  : 10.25.1.11
```

```

ProductCodes      : {}
PublicDnsName     :
PublicIpAddress   : 198.51.100.245
RamdiskId         :
RootDeviceName    : /dev/sda1
RootDeviceType    : ebs
SecurityGroups    : {myPSSecurityGroup}
SourceDestCheck   : True
SpotInstanceRequestId :
SriovNetSupport   :
State             : Amazon.EC2.Model.InstanceState
StateReason       :
StateTransitionReason :
SubnetId          : subnet-d60013bf
Tags              : {}
VirtualizationType : hvm
VpcId             : vpc-a01106c2

```

Avviare un'istanza Spot in una VPC

Il seguente script di esempio richiede un'istanza Spot nella sottorete specificata. Il gruppo di sicurezza deve essere uno creato per la VPC che contiene la sottorete specificata.

```

$interface1 = New-Object Amazon.EC2.Model.InstanceNetworkInterfaceSpecification
$interface1.DeviceIndex = 0
$interface1.SubnetId = "subnet-b61f49f0"
$interface1.PrivateIpAddress = "10.0.1.5"
$interface1.Groups.Add("sg-5d293231")
Request-EC2SpotInstance `
    -SpotPrice 0.007 `
    -InstanceCount 1 `
    -Type one-time `
    -LaunchSpecification_ImageId ami-7527031c `
    -LaunchSpecification_InstanceType m1.small `
    -Region us-west-2 `
    -LaunchSpecification_NetworkInterfaces $interface1

```

AWS Lambda e AWS Strumenti per PowerShell

Utilizzando il [AWSLambdaPSCore](#) modulo, è possibile sviluppare AWS Lambda funzioni in PowerShell Core 6.0 utilizzando il runtime .NET Core 2.1. PowerShell gli sviluppatori possono gestire

AWS le risorse e scrivere script di automazione nell' PowerShell ambiente utilizzando Lambda. PowerShell il supporto in Lambda consente di eseguire PowerShell script o funzioni in risposta a qualsiasi evento Lambda, ad esempio un evento Amazon S3 o un evento pianificato da Amazon. CloudWatch Il AWSLambda PSCore modulo è un AWS modulo separato per PowerShell; non fa parte di AWS Strumenti per PowerShell, né l'installazione del modulo installa il AWSLambdaPSCore . AWS Strumenti per PowerShell

Dopo aver installato il AWSLambda PSCore modulo, è possibile utilizzare qualsiasi PowerShell cmdlet disponibile, o svilupparne uno personalizzato, per creare funzioni serverless. Il PowerShell modulo AWS Lambda Tools for include modelli di progetto per applicazioni serverless PowerShell basate su server e strumenti su cui pubblicare progetti. AWS

AWSLambdaPSCore il supporto per i moduli è disponibile in tutte le regioni che supportano Lambda. Per ulteriori informazioni sulle Regioni supportate, consulta la [tabella delle Regioni AWS](#).

Prerequisiti

I seguenti passaggi sono necessari prima di poter installare e utilizzare il AWSLambda PSCore modulo. Per maggiori dettagli su questi passaggi, consulta [Configurazione di un ambiente di PowerShell sviluppo](#) nella Guida per gli AWS Lambda sviluppatori.

- Installa la versione corretta di PowerShell: il supporto di Lambda per PowerShell basa sulla versione PowerShell Core 6.0 multiplatforma. Puoi sviluppare funzioni PowerShell Lambda su Windows, Linux o Mac. Se non disponi almeno di questa versione di PowerShell installazione, le istruzioni sono disponibili nel [sito Web della PowerShell documentazione Microsoft](#).
- Installazione di .NET Core 2.1 SDK: poiché PowerShell Core è basato su .NET Core, il supporto Lambda utilizza lo stesso runtime Lambda PowerShell per entrambe le funzioni .NET Core e Lambda. PowerShell I cmdlet di PowerShell pubblicazione Lambda utilizzano l'SDK .NET Core 2.1 per creare il pacchetto di distribuzione Lambda. L'SDK .NET Core 2.1 SDK è disponibile nell'[Area download Microsoft](#). Assicurati di installare l'SDK, non il runtime.

Installa il modulo AWSLambda PSCore

Dopo aver completato i prerequisiti, sei pronto per installare il AWSLambda PSCore modulo. Eseguite il comando seguente in una sessione PowerShell Core.

```
PS> Install-Module AWSLambdaPSCore -Scope CurrentUser
```

Sei pronto per iniziare a sviluppare funzioni Lambda in PowerShell? Per ulteriori informazioni su come iniziare, consulta [Programming Model for Authoring Lambda Functions PowerShell](#) nella AWS Lambda Developer Guide.

Vedi anche

- [Annuncio del PowerShell supporto Lambda per Core AWS sul blog degli sviluppatori](#)
- [AWSLambdaPSCore modulo sul sito web della Galleria PowerShell](#)
- [Configurazione di un ambiente di PowerShell sviluppo](#)
- [AWS Strumenti Lambda per Powershell su GitHub](#)
- [AWS Console Lambda](#)

Amazon SQS, Amazon SNS e strumenti per Windows PowerShell

In questa sezione vengono forniti esempi in cui viene illustrato come:

- Creare una coda Amazon SQS e ottenere l'ARN della coda (Amazon Resource Name).
- Creazione di un argomento Amazon SNS.
- Concedere l'autorizzazione all'argomento SNS, in modo che possa inviare messaggi alla coda.
- Sottoscrivere la coda all'argomento SNS
- Concedi agli utenti o agli AWS account IAM le autorizzazioni per pubblicare sull'argomento SNS e leggere i messaggi dalla coda SQS.
- Verificare i risultati pubblicando un messaggio nell'argomento e leggendo il messaggio dalla coda.

Creare una coda Amazon SQS e ottenere l'ARN della coda

Il comando seguente crea una coda SQS nella regione predefinita. L'output mostra l'URL della nuova coda.

```
PS > New-SQSQueue -QueueName myQueue
https://sqs.us-west-2.amazonaws.com/123456789012/myQueue
```

Il comando seguente recupera l'ARN della coda.

```
PS > Get-SQSQueueAttribute -QueueUrl https://sqs.us-west-2.amazonaws.com/123456789012/myQueue -AttributeName QueueArn
```

```
...
QueueARN           : arn:aws:sqs:us-west-2:123456789012:myQueue
...
```

Creare un argomento Amazon SNS.

Il comando seguente crea un argomento SNS nella regione predefinita e restituisce l'ARN del nuovo argomento.

```
PS > New-SNSTopic -Name myTopic
arn:aws:sns:us-west-2:123456789012:myTopic
```

Concedere le autorizzazioni all'argomento SNS

Il seguente script di esempio crea sia una coda SQS sia un argomento SNS e concede le autorizzazioni all'argomento SNS in modo che possa inviare messaggi alla coda SQS:

```
# create the queue and topic to be associated
$qurl = New-SQSQueue -QueueName "myQueue"
$topicarn = New-SNSTopic -Name "myTopic"

# get the queue ARN to inject into the policy; it will be returned
# in the output's QueueARN member but we need to put it into a variable
# so text expansion in the policy string takes effect
$qarn = (Get-SQSQueueAttribute -QueueUrl $qurl -AttributeNames "QueueArn").QueueARN

# construct the policy and inject arns
$policy = @"
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": "*",
    "Action": "SQS:SendMessage",
    "Resource": "$qarn",
    "Condition": { "ArnEquals": { "aws:SourceArn": "$topicarn" } }
  }
}
"@

# set the policy
```

```
Set-SQSQueueAttribute -QueueUrl $qurl -Attribute @{ Policy=$policy }
```

Sottoscrivere la coda all'argomento SNS

Il comando seguente sottoscrive la coda myQueue all'argomento SNS myTopic e restituisce l'ID sottoscrizione:

```
PS > Connect-SNSNotification `  
    -TopicARN arn:aws:sns:us-west-2:123456789012:myTopic `  
    -Protocol SQS `  
    -Endpoint arn:aws:sqs:us-west-2:123456789012:myQueue  
arn:aws:sns:us-west-2:123456789012:myTopic:f8ff77c6-e719-4d70-8e5c-a54d41feb754
```

Concedere le autorizzazioni

Il comando seguente concede l'autorizzazione per eseguire l'azione sns:Publish sull'argomento myTopic.

```
PS > Add-SNSPermission `  
    -TopicArn arn:aws:sns:us-west-2:123456789012:myTopic `  
    -Label ps-cmdlet-topic `  
    -AWSAccountIds 123456789012 `  
    -ActionNames publish
```

Il comando seguente concede l'autorizzazione per eseguire le azioni sqs:ReceiveMessage e sqs:DeleteMessage sulla coda myQueue.

```
PS > Add-SQSPermission `  
    -QueueUrl https://sqs.us-west-2.amazonaws.com/123456789012/myQueue `  
    -AWSAccountId "123456789012" `  
    -Label queue-permission `  
    -ActionName SendMessage, ReceiveMessage
```

Verificare i risultati

Il comando seguente verifica la nuova coda e l'argomento pubblicando un messaggio nell'argomento SNS myTopic e restituisce MessageId.

```
PS > Publish-SNSMessage `
```



```
-TopicArn arn:aws:sns:us-west-2:123456789012:myTopic `
```

```
-Message "Have A Nice Day!"
```

```
728180b6-f62b-49d5-b4d3-3824bb2e77f4
```

Il comando seguente recupera il messaggio dalla coda SQS myQueue e lo mostra.

```
PS > Receive-SQSMessage -QueueUrl https://sqs.us-west-2.amazonaws.com/123456789012/myQueue
```

```
Attributes : {}
```

```
Body : {
```

```
    "Type" : "Notification",
```

```
    "MessageId" : "491c687d-b78d-5c48-b7a0-3d8d769ee91b",
```

```
    "TopicArn" : "arn:aws:sns:us-west-2:123456789012:myTopic",
```

```
    "Message" : "Have A Nice Day!",
```

```
    "Timestamp" : "2019-09-09T21:06:27.201Z",
```

```
    "SignatureVersion" : "1",
```

```
    "Signature" :
```

```
    "11E17A2+X0uJZnw3TlgcXz4C4KPLXZxbxoEMIirelh13u/oxkWmz5+9tJKFMns1Z0qQvKxk
```

```
+ExfEZcD5yWt6biVuBb8pyRmZ1b03hUEN13ayv2WQiQT1vpLpM7VEQN5m+hLIiPFcs
```

```
vyuGkJReV710JWPHnCN
```

```
+qTE2lId2RPkF0eGtLGawTsSPTWEvJdDbLlf7E0zZ0q1niXTUtpsZ8Swx01X3Q06u9i9qBFt0ekJFZNJp6Avu05hIk1b4yo
```

```
y0a8Yl9lWp7a7EoWaBn0zhCESe7o
```

```
kZC6ncBJWphX7KCGVYD0qhVf/5VDgBuv9w8T+higJyvr3WbaSvg==",
```

```
    "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/
```

```
SimpleNotificationService-6aad65c2f9911b05cd53efda11f913f9.pem",
```

```
    "UnsubscribeURL" :
```

```
    "https://sns.us-west-2.amazonaws.com/?
```

```
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-west-2:123456789012:myTopic:22b77de7-a216-4000-9a23-bf465744ca84"
```

```
    }
```

```
MD5ofBody : 5b5ee4f073e9c618eda3718b594fa257
```

```
MD5ofMessageAttributes :
```

```
MessageAttributes : {}
```

```
MessageId : 728180b6-f62b-49d5-b4d3-3824bb2e77f4
```

```
ReceiptHandle :
```

```
AQEB2vvk1e5c0KFjeIWJticabkc664yuDEjhucnIOqdVUmie7bX7GiJb17F0enABUgaI2XjEcNPxixhVc/wfsAJZLNHn18S1bQa0R/kD+Saq40Ivfj8x3M40h1yM1cVKpYmhAzsYrAwAD5g5FvxNBD6zs
```

```
+HmXdkax2Wd+9AxrH1QZV5ur1MoByKWwbDbsqoYJTJquCc10gWIak/sBx/
```

```
daBRMTiVQ4GHsrQWMVHtNC14q7Jy/0L2dkmb4dzJfJq0VbFSX1G+u/lrSLpgae+Dfux646y8yFiPFzY4ua4mCF/SVUn63Spy
```

```
sHN12776axknhg3j9K/Xwj54DixdsegnrKoLx+ctI
```

```
+0jzAetBR66Q1VhIoJAq7s0a2Msey0eM/Jjucg6Sr9VUnTWVhV8ErXmotoiEg==
```

CloudWatch dal AWS Tools for Windows PowerShell

Questa sezione mostra un esempio di come utilizzare gli Strumenti per Windows per PowerShell pubblicare dati metrici personalizzati su CloudWatch

Questo esempio presuppone che tu abbia impostato credenziali predefinite e un'area predefinite per la sessione. PowerShell

Pubblicare un parametro personalizzato nel pannello di controllo CloudWatch

Il PowerShell codice seguente inizializza un CloudWatch `MetricDatum` oggetto e lo invia al servizio.

[È possibile visualizzare il risultato di questa operazione accedendo alla console. CloudWatch](#)

```
$dat = New-Object Amazon.CloudWatch.Model.MetricDatum
$dat.Timestamp = (Get-Date).ToUniversalTime()
$dat.MetricName = "New Posts"
$dat.Unit = "Count"
$dat.Value = ".50"
Write-CWMetricData -Namespace "Usage Metrics" -MetricData $dat
```

Tieni presente quanto segue:

- Le informazioni su data e ora utilizzate per l'inizializzazione `$dat.Timestamp` devono essere nel formato Universal Time (UTC).
- Il valore che si utilizza per l'inizializzazione `$dat.Value` può essere un valore di stringa tra virgolette o un valore numerico (senza virgolette). L'esempio mostra un valore di stringa.

Vedi anche

- [Lavora con AWS i servizi in AWS Strumenti per PowerShell](#)
- [AmazonCloudWatchClient.PutMetricData](#)(Riferimento.NET SDK)
- [MetricDatum](#)(Riferimento all'API del servizio)
- [CloudWatch Console Amazon](#)

Utilizzo del ClientConfig parametro nei cmdlet

Il parametro `ClientConfig` può essere usato per specificare determinate impostazioni di configurazione durante la connessione a un servizio. La maggior parte delle possibili proprietà di questo parametro sono definite nella [Amazon.Runtime.ClientConfig](#) classe, che viene ereditata dai AWS servizi APIs for. Per un esempio di ereditarietà semplice, fai riferimento alla classe [Amazon.Keyspaces.AmazonKeyspacesConfig](#). Inoltre, alcuni servizi definiscono proprietà aggiuntive che sono appropriate solo per il servizio specifico. Per un esempio di definizione di proprietà aggiuntive, fai riferimento alla classe [Amazon.S3.AmazonS3Config](#) e in particolare alla proprietà `ForcePathStyle`.

Uso del parametro **ClientConfig**

Per utilizzare il `ClientConfig` parametro, è possibile specificarlo nella riga di comando come `ClientConfig` oggetto o utilizzare PowerShell splatting per passare una raccolta di valori di parametro a un comando come unità. Questi metodi vengono mostrati negli esempi seguenti. Gli esempi presuppongono che sia stato installato e importato il modulo `AWS.Tools.S3` e che tu abbia un profilo di credenziali [default] con le autorizzazioni appropriate.

Definizione di un oggetto **ClientConfig**

```
$s3Config = New-Object -TypeName Amazon.S3.AmazonS3Config
$s3Config.ForcePathStyle = $true
$s3Config.Timeout = [TimeSpan]::FromMilliseconds(150000)
Get-S3Object -BucketName <BUCKET_NAME> -ClientConfig $s3Config
```

Aggiungere **ClientConfig** proprietà utilizzando lo splatting PowerShell

```
$params=@{
    ClientConfig=@{
        ForcePathStyle=$true
        Timeout=[TimeSpan]::FromMilliseconds(150000)
    }
    BucketName="<BUCKET_NAME>"
}

Get-S3Object @params
```

Uso di una proprietà non definita

Quando si utilizza lo PowerShell splatting, se si specifica una `ClientConfig` proprietà che non esiste, l'errore AWS Strumenti per PowerShell viene rilevato solo in fase di esecuzione, momento in cui restituisce un'eccezione. Modificando l'esempio precedente:

```
$params=@{
    ClientConfig=@{
        ForcePathStyle=$true
        UndefinedProperty="Value"
        Timeout=[TimeSpan]::FromMilliseconds(150000)
    }
    BucketName="<BUCKET_NAME>"
}

Get-S3Object @params
```

Questo esempio genera un'eccezione simile alla seguente:

```
Cannot bind parameter 'ClientConfig'. Cannot create object of type
"Amazon.S3.AmazonS3Config". The UndefinedProperty property was not found for the
Amazon.S3.AmazonS3Config object.
```

Specificando il Regione AWS

È possibile utilizzare il `ClientConfig` parametro Regione AWS per impostare il comando. La regione viene impostata tramite la proprietà `RegionEndpoint`. AWS Strumenti per PowerShell calcola la regione da utilizzare in base alla seguente precedenza:

1. Parametro `-Region`
2. Regione passata nel parametro `ClientConfig`
3. Lo stato della sessione PowerShell
4. Il AWS config file condiviso
5. Variabili di ambiente
6. I metadati dell' EC2 istanza Amazon, se abilitati.

Strumenti per esempi di codice PowerShell V4

Gli esempi di codice in questo argomento mostrano come utilizzare la AWS Strumenti per PowerShell V4 con AWS.

Nozioni di base: esempi di codice che mostrano come eseguire le operazioni essenziali all'interno di un servizio.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Scenari: esempi di codice che mostrano come eseguire un'attività specifica chiamando più funzioni all'interno dello stesso servizio o combinate con altri Servizi AWS.

Alcuni servizi contengono ulteriori categorie di esempi che mostrano come sfruttare le librerie o le funzioni specifiche del servizio.

Servizi

- [Esempi ACM che utilizzano Tools for V4 PowerShell](#)
- [Esempi di Application Auto Scaling con Tools for V4 PowerShell](#)
- [WorkSpaces Esempi di applicazioni che utilizzano Tools for V4 PowerShell](#)
- [Esempi di Aurora con Tools for V4 PowerShell](#)
- [Esempi di Auto Scaling con Tools for V4 PowerShell](#)
- [Budget AWS esempi di utilizzo di Tools for V4 PowerShell](#)
- [AWS Cloud9 esempi di utilizzo di Tools for V4 PowerShell](#)
- [CloudFormation esempi di utilizzo di Tools for V4 PowerShell](#)
- [CloudFront esempi di utilizzo di Tools for V4 PowerShell](#)
- [CloudTrail esempi di utilizzo di Tools for V4 PowerShell](#)
- [CloudWatch esempi di utilizzo di Tools for V4 PowerShell](#)
- [CodeCommit esempi di utilizzo di Tools for V4 PowerShell](#)
- [CodeDeploy esempi di utilizzo di Tools for V4 PowerShell](#)
- [CodePipeline esempi di utilizzo di Tools for V4 PowerShell](#)

- [Esempi di identità di Amazon Cognito con Tools for V4 PowerShell](#)
- [AWS Config esempi di utilizzo di Tools for V4 PowerShell](#)
- [Esempi di Device Farm con Tools for PowerShell V4](#)
- [Directory Service esempi di utilizzo di Tools for V4 PowerShell](#)
- [AWS DMS esempi di utilizzo di Tools for V4 PowerShell](#)
- [Esempi di DynamoDB con Tools for V4 PowerShell](#)
- [EC2 Esempi di Amazon con Tools for PowerShell V4](#)
- [Esempi di Amazon ECR con Tools for PowerShell V4](#)
- [Esempi di Amazon ECS con Tools for PowerShell V4](#)
- [Esempi di Amazon EFS con Tools for PowerShell V4](#)
- [Esempi di Amazon EKS con Tools for PowerShell V4](#)
- [ELB - Esempi della versione 1 che utilizzano Tools for V4 PowerShell](#)
- [ELB - Esempi della versione 2 che utilizzano Tools for V4 PowerShell](#)
- [FSx Esempi di Amazon con Tools for PowerShell V4](#)
- [Esempi di Amazon Glacier con Tools for V4 PowerShell](#)
- [AWS Glue esempi di utilizzo di Tools for V4 PowerShell](#)
- [AWS Health esempi di utilizzo di Tools for V4 PowerShell](#)
- [Esempi IAM che utilizzano Tools for V4 PowerShell](#)
- [Esempi di Kinesis con Tools for V4 PowerShell](#)
- [Esempi di Lambda con Tools for V4 PowerShell](#)
- [Esempi di Amazon ML con Tools for PowerShell V4](#)
- [Esempi di Macie che utilizzano Tools for V4 PowerShell](#)
- [Listino prezzi AWS esempi di utilizzo di Tools for V4 PowerShell](#)
- [Esempi di Resource Groups con Tools for PowerShell V4](#)
- [Esempi di API Resource Groups Tagging con Tools for PowerShell V4](#)
- [Esempi di Route 53 utilizzando Tools for V4 PowerShell](#)
- [Esempi di Amazon S3 con Tools for V4 PowerShell](#)
- [Esempi CSPM di Security Hub con Tools for V4 PowerShell](#)
- [Esempi di Amazon SES con Tools for PowerShell V4](#)

- [Esempi di API Amazon SES v2 con Tools for PowerShell V4](#)
- [Esempi di Amazon SNS con Tools for V4 PowerShell](#)
- [Esempi di Amazon SQS con Tools for V4 PowerShell](#)
- [AWS STS esempi di utilizzo di Tools for V4 PowerShell](#)
- [Supporto esempi di utilizzo di Tools for V4 PowerShell](#)
- [Esempi di Systems Manager con Tools for PowerShell V4](#)
- [Esempi di Amazon Translate con Tools for PowerShell V4](#)
- [AWS WAFV2 esempi di utilizzo di Tools for V4 PowerShell](#)
- [WorkSpaces esempi di utilizzo di Tools for V4 PowerShell](#)

Esempi ACM che utilizzano Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con ACM.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-ACMCertificate

Il seguente esempio di codice mostra come utilizzare. Get-ACMCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra come restituire un certificato e la relativa catena utilizzando l'ARN del certificato.

```
Get-ACMCertificate -CertificateArn "arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012"
```

- Per i dettagli sull'API, vedere [GetCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ACMCertificateDetail

Il seguente esempio di codice mostra come utilizzare `Get-ACMCertificateDetail`

Strumenti per PowerShell V4

Esempio 1: restituisce i dettagli del certificato specificato.

```
Get-ACMCertificateDetail -CertificateArn "arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012"
```

Output:

```
CertificateArn      : arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012
CreatedAt          : 1/21/2016 5:55:59 PM
DomainName         : www.example.com
DomainValidationOptions : {www.example.com}
InUseBy            : {}
IssuedAt           : 1/1/0001 12:00:00 AM
Issuer             :
KeyAlgorithm        : RSA-2048
NotAfter           : 1/1/0001 12:00:00 AM
NotBefore          : 1/1/0001 12:00:00 AM
RevocationReason    :
RevokedAt          : 1/1/0001 12:00:00 AM
Serial             :
SignatureAlgorithm  : SHA256WITHRSA
Status             : PENDING_VALIDATION
Subject            : CN=www.example.com
SubjectAlternativeNames : {www.example.net}
```

- Per i dettagli sull'API, vedere [DescribeCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ACMCertificateList

Il seguente esempio di codice mostra come utilizzare. Get-ACMCertificateList

Strumenti per PowerShell V4

Esempio 1: recupera un elenco di tutti i certificati ARNs e il nome di dominio per ciascuno.

Il cmdlet verrà impaginato automaticamente per recuperare tutti i. ARNs Per controllare manualmente l'impaginazione, utilizzare il MaxItem parametro - per controllare il numero di certificati ARNs restituiti per ogni chiamata di servizio e il NextToken parametro - per indicare il punto di partenza per ogni chiamata.

```
Get-ACMCertificateList
```

Output:

```
CertificateArn
DomainName
-----
-----
arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012
www.example.com
```

Esempio 2: recupera un elenco di tutti i certificati ARNs in cui lo stato del certificato corrisponde a quello degli stati forniti.

```
Get-ACMCertificateList -CertificateStatus "VALIDATION_TIMED_OUT","FAILED"
```

Esempio 3: questo esempio restituisce un elenco di tutti i certificati nella Regione us-east-1 con un tipo di chiave RSA_2048 e l'utilizzo o scopo esteso CODE_SIGNING della chiave. È possibile trovare i valori per questi parametri di filtro nell'argomento di riferimento dell'API ListCertificates Filters: https://docs.aws.amazon.com/acm/latest/APIReference/API_Filters.html.

```
Get-ACMCertificateList -Region us-east-1 -Includes_KeyType RSA_2048 -
Includes_ExtendedKeyUsage CODE_SIGNING
```

Output:

```
CertificateArn
DomainName
```

```

-----
-----
arn:aws:acm:us-east-1:8xxxxxxxxxx:certificate/xxxxxxxx-d7c0-48c1-af8d-2133d8f30zzz
*.route53docs.com
arn:aws:acm:us-east-1:8xxxxxxxxxx:certificate/xxxxxxxx-98a5-443d-a734-800430c80zzz
nerdzizm.net
arn:aws:acm:us-east-1:8xxxxxxxxxx:certificate/xxxxxxxx-2be6-4376-8fa7-bad559525zzz

arn:aws:acm:us-east-1:8xxxxxxxxxx:certificate/xxxxxxxx-e7ca-44c5-803e-24d9f2f36zzz

arn:aws:acm:us-east-1:8xxxxxxxxxx:certificate/xxxxxxxx-1241-4b71-80b1-090305a62zzz

arn:aws:acm:us-east-1:8xxxxxxxxxx:certificate/xxxxxxxx-8709-4568-8c64-f94617c99zzz

arn:aws:acm:us-east-1:8xxxxxxxxxx:certificate/xxxxxxxx-a8fa-4a61-98cf-e08ccc0eezzz

arn:aws:acm:us-east-1:8xxxxxxxxxx:certificate/xxxxxxxx-fa47-40fe-a714-2d277d3eezzz
*.route53docs.com

```

- Per i dettagli sull'API, vedere [ListCertificates](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ACMCertificate

Il seguente esempio di codice mostra come utilizzare New-ACMCertificate

Strumenti per PowerShell V4

Esempio 1: crea un nuovo certificato. Il servizio restituisce l'ARN del nuovo certificato.

```
New-ACMCertificate -DomainName "www.example.com"
```

Output:

```
arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012
```

Esempio 2: crea un nuovo certificato. Il servizio restituisce l'ARN del nuovo certificato.

```
New-ACMCertificate -DomainName "www.example.com" -SubjectAlternativeName
"example.com", "www.example.net"
```

Output:

```
arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012
```

- Per i dettagli sull'API, vedere [RequestCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ACMCertificate

Il seguente esempio di codice mostra come utilizzare. Remove-ACMCertificate

Strumenti per PowerShell V4

Esempio 1: elimina il certificato identificato dall'ARN fornito e dalla chiave privata associata. Il cmdlet richiederà una conferma prima di procedere; aggiungi l'opzione -Force per disabilitare la conferma.

```
Remove-ACMCertificate -CertificateArn "arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012"
```

- Per i dettagli sull'API, vedere [DeleteCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Send-ACMValidationEmail

Il seguente esempio di codice mostra come utilizzare. Send-ACMValidationEmail

Strumenti per PowerShell V4

Esempio 1: richiede l'invio dell'e-mail per convalidare la proprietà del dominio per "www.example.com". Se il valore \$ della shell ConfirmPreference è impostato su 'Medium' o inferiore, il cmdlet richiederà una conferma prima di procedere. Aggiungi l'opzione -Force per disattivare le richieste di conferma.

```
$params = @{  
    CertificateArn="arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012"  
    Domain="www.example.com"  
    ValidationDomain="example.com"
```

```
}  
Send-ACMValidationEmail @params
```

- Per i dettagli sull'API, vedere [ResendValidationEmail](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Application Auto Scaling con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Application Auto Scaling.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-AASScalableTarget

Il seguente esempio di codice mostra come utilizzare. Add-AASScalableTarget

Strumenti per PowerShell V4

Esempio 1: questo cmdlet registra o aggiorna una destinazione dimensionabile. Una destinazione dimensionabile è una risorsa la cui dimensione può essere aumentata e ridotta orizzontalmente da Application Auto Scaling.

```
Add-AASScalableTarget -ServiceNamespace AppStream -ResourceId fleet/MyFleet -  
ScalableDimension appstream:fleet:DesiredCapacity -MinCapacity 2 -MaxCapacity 10
```

- Per i dettagli sull'API, vedere [RegisterScalableTarget](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-AASScalableTarget

Il seguente esempio di codice mostra come utilizzare. Get-AASScalableTarget

Strumenti per PowerShell V4

Esempio 1: questo esempio fornirà informazioni sulle destinazioni dimensionabili di Application Autoscaling nel namespace specificato.

```
Get-AASScalableTarget -ServiceNamespace "AppStream"
```

Output:

```
CreationTime      : 11/7/2019 2:30:03 AM
MaxCapacity       : 5
MinCapacity       : 1
ResourceId        : fleet/Test
RoleARN           : arn:aws:iam::012345678912:role/aws-
service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
SuspendedState    : Amazon.ApplicationAutoScaling.Model.SuspendedState
```

- Per i dettagli sull'API, vedere [DescribeScalableTargets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-AASScalingActivity

Il seguente esempio di codice mostra come utilizzare. Get-AASScalingActivity

Strumenti per PowerShell V4

Esempio 1: fornisce informazioni descrittive sulle attività di dimensionamento nel namespace specificato per le sei settimane precedenti.

```
Get-AASScalingActivity -ServiceNamespace AppStream
```

Output:

```
ActivityId        : 2827409f-b639-4cdb-a957-8055d5d07434
```

```

Cause           : monitor alarm Appstream2-MyFleet-default-scale-in Alarm in state
                  ALARM triggered policy default-scale-in
Description     : Setting desired capacity to 2.
Details         :
EndTime         : 12/14/2019 11:32:49 AM
ResourceId      : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace : appstream
StartTime       : 12/14/2019 11:32:14 AM
StatusCode      : Successful
StatusMessage   : Successfully set desired capacity to 2. Change successfully
                  fulfilled by appstream.

```

- Per i dettagli sull'API, vedere [DescribeScalingActivities](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-AASScalingPolicy

Il seguente esempio di codice mostra come utilizzare. Get-AASScalingPolicy

Strumenti per PowerShell V4

Esempio 1: questo cmdlet descrive le policy di dimensionamento di Application Auto Scaling per il namespace del servizio specificato.

```
Get-AASScalingPolicy -ServiceNamespace AppStream
```

Output:

```

Alarms           : {Appstream2-LabFleet-default-scale-out-
Alarm}
CreationTime     : 9/3/2019 2:48:15 AM
PolicyARN        : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                  policyName/default-scale-out
PolicyName       : default-scale-out
PolicyType       : StepScaling
ResourceId       : fleet/LabFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace : appstream

```

```

StepScalingPolicyConfiguration      :
  Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

Alarms                             : {Appstream2-LabFleet-default-scale-in-
Alarm}
CreationTime                       : 9/3/2019 2:48:15 AM
PolicyARN                          : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                   policyName/default-scale-in
PolicyName                         : default-scale-in
PolicyType                         : StepScaling
ResourceId                         : fleet/LabFleet
ScalableDimension                  : appstream:fleet:DesiredCapacity
ServiceNamespace                   : appstream
StepScalingPolicyConfiguration      :
  Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

```

- Per i dettagli sull'API, vedere [DescribeScalingPolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-AASScheduledAction

Il seguente esempio di codice mostra come utilizzare. Get-AASScheduledAction

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elenca le azioni pianificate del gruppo Auto Scaling che non sono state eseguite o che non hanno raggiunto l'ora di fine.

```
Get-AASScheduledAction -ServiceNamespace AppStream
```

Output:

```

CreationTime      : 12/22/2019 9:25:52 AM
EndTime           : 1/1/0001 12:00:00 AM
ResourceId         : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ScalableTargetAction : Amazon.ApplicationAutoScaling.Model.ScalableTargetAction

```

```

Schedule           : cron(0 0 8 ? * MON-FRI *)
ScheduledActionARN  : arn:aws:autoscaling:us-
west-2:012345678912:scheduledAction:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:scheduledActionName
                    /WeekDaysFleetScaling
ScheduledActionName : WeekDaysFleetScaling
ServiceNamespace    : appstream
StartTime            : 1/1/0001 12:00:00 AM

```

- Per i dettagli sull'API, vedere [DescribeScheduledActions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-AASScalableTarget

Il seguente esempio di codice mostra come utilizzare. Remove-AASScalableTarget

Strumenti per PowerShell V4

Esempio 1: questo cmdlet annulla la registrazione di una destinazione dimensionabile Application Auto Scaling. L'annullamento della registrazione di una destinazione dimensionabile elimina le policy di dimensionamento ad essa associate.

```

Remove-AASScalableTarget -ResourceId fleet/MyFleet -ScalableDimension
appstream:fleet:DesiredCapacity -ServiceNamespace AppStream

```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScalableTarget (DeregisterScalableTarget)" on
target "fleet/MyFleet".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

```

- Per i dettagli sull'API, vedere [DeregisterScalableTarget](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-AASScalingPolicy

Il seguente esempio di codice mostra come utilizzare. Remove-AASScalingPolicy

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elimina la policy di dimensionamento specificata per una destinazione dimensionabile di Application Auto Scaling.

```
Remove-AASScalingPolicy -ServiceNamespace AppStream -PolicyName "default-scale-out"  
-ResourceId fleet/Test -ScalableDimension appstream:fleet:DesiredCapacity
```

- Per i dettagli sull'API, vedere [DeleteScalingPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-AASScheduledAction

Il seguente esempio di codice mostra come utilizzare. Remove-AASScheduledAction

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elimina l'azione pianificata specificata per una destinazione dimensionabile di Application Auto Scaling.

```
Remove-AASScheduledAction -ServiceNamespace AppStream -ScheduledActionName  
WeekDaysFleetScaling -ResourceId fleet/MyFleet -ScalableDimension  
appstream:fleet:DesiredCapacity
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing the operation "Remove-AASScheduledAction (DeleteScheduledAction)" on  
target "WeekDaysFleetScaling".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): Y
```

- Per i dettagli sull'API, vedere [DeleteScheduledAction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-AASScalingPolicy

Il seguente esempio di codice mostra come utilizzare. Set-AASScalingPolicy

Strumenti per PowerShell V4

Esempio 1: questo cmdlet crea o aggiorna una policy di una destinazione dimensionabile di Application Auto Scaling. Ogni destinazione dimensionabile è identificata da un namespace del servizio, un ID risorsa e una dimensione dimensionabile.

```
Set-AASScalingPolicy -ServiceNamespace AppStream -PolicyName ASFleetScaleInPolicy
-PolicyType StepScaling -ResourceId fleet/MyFleet -ScalableDimension
appstream:fleet:DesiredCapacity -StepScalingPolicyConfiguration_AdjustmentType
ChangeInCapacity -StepScalingPolicyConfiguration_Cooldown 360
-StepScalingPolicyConfiguration_MetricAggregationType Average -
StepScalingPolicyConfiguration_StepAdjustments @{ScalingAdjustment = -1;
MetricIntervalUpperBound = 0}
```

Output:

```
Alarms      PolicyARN
-----
{}          arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:policyName/ASFleetScaleInPolicy
```

- Per i dettagli sull'API, vedere [PutScalingPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-AASScheduledAction

Il seguente esempio di codice mostra come utilizzare. Set-AASScheduledAction

Strumenti per PowerShell V4

Esempio 1: questo cmdlet crea o aggiorna un'azione pianificata per una destinazione dimensionabile di Application Auto Scaling. Ogni destinazione dimensionabile è identificata da un namespace del servizio, un ID risorsa e una dimensione dimensionabile.

```
Set-AASScheduledAction -ServiceNamespace AppStream -ResourceId fleet/
MyFleet -Schedule "cron(0 0 8 ? * MON-FRI *)" -ScalableDimension
appstream:fleet:DesiredCapacity -ScheduledActionName WeekDaysFleetScaling -
ScalableTargetAction_MinCapacity 5 -ScalableTargetAction_MaxCapacity 10
```

- Per i dettagli sull'API, vedere [PutScheduledAction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

WorkSpaces Esempi di applicazioni che utilizzano Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 with WorkSpaces Applications.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-APSResourceTag

Il seguente esempio di codice mostra come utilizzare Add-APSResourceTag.

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un tag di risorsa alla AppStream risorsa

```
Add-APSResourceTag -ResourceArn arn:aws:appstream:us-east-1:123456789012:stack/SessionScriptTest -Tag @{StackState='Test'} -Select ^Tag
```

Output:

Name	Value
----	----
StackState	Test

- Per i dettagli sull'API, vedere [TagResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Copy-APSIImage

Il seguente esempio di codice mostra come utilizzare. Copy-APSIImage

Strumenti per PowerShell V4

Esempio 1: questo esempio copia un'immagine in un'altra Regione

```
Copy-APSIImage -DestinationImageName TestImageCopy -DestinationRegion us-west-2 -  
SourceImageName Powershell
```

Output:

```
TestImageCopy
```

- Per i dettagli sull'API, vedere [CopyImage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Disable-APSUser

Il seguente esempio di codice mostra come utilizzare. Disable-APSUser

Strumenti per PowerShell V4

Esempio 1: questo esempio disabilita un utente in USERPOOL

```
Disable-APSUser -AuthenticationType USERPOOL -UserName TestUser@lab.com
```

- Per i dettagli sull'API, vedere [DisableUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-APSUser

Il seguente esempio di codice mostra come utilizzare. Enable-APSUser

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita un utente disabilitato in USERPOOL

```
Enable-APUser -AuthenticationType USERPOOL -UserName TestUser@lab.com
```

- Per i dettagli sull'API, vedere [EnableUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSAssociatedFleetList

Il seguente esempio di codice mostra come utilizzare. Get-APSAssociatedFleetList

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza il parco associato a uno stack

```
Get-APSAssociatedFleetList -StackName PowershellStack
```

Output:

```
PowershellFleet
```

- Per i dettagli sull'API, vedere [ListAssociatedFleets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSAssociatedStackList

Il seguente esempio di codice mostra come utilizzare. Get-APSAssociatedStackList

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza lo stack associato a un parco

```
Get-APSAssociatedStackList -FleetName PowershellFleet
```

Output:

```
PowershellStack
```

- Per i dettagli sull'API, vedere [ListAssociatedStacks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSDirectoryConfigList

Il seguente esempio di codice mostra come utilizzare. Get-APSDirectoryConfigList

Strumenti per PowerShell V4

Esempio 1: Questo esempio visualizza le configurazioni di directory create in AppStream

```
Get-APSDirectoryConfigList | Select DirectoryName,
OrganizationalUnitDistinguishedNames, CreatedTime
```

Output:

```
DirectoryName OrganizationalUnitDistinguishedNames CreatedTime
-----
Test.com      {OU=AppStream,DC=Test,DC=com}    9/6/2019 10:56:40 AM
contoso.com   {OU=AppStream,OU=contoso,DC=contoso,DC=com} 8/9/2019 9:08:50 AM
```

- Per i dettagli sull'API, vedere [DescribeDirectoryConfigs](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSFleetList

Il seguente esempio di codice mostra come utilizzare. Get-APSFleetList

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza i dettagli di un parco

```
Get-APSFleetList -Name Test
```

Output:

```
Arn                : arn:aws:appstream:us-east-1:1234567890:fleet/Test
ComputeCapacityStatus : Amazon.AppStream.Model.ComputeCapacityStatus
CreatedTime        : 9/12/2019 5:00:45 PM
```

```

Description           : Test
DisconnectTimeoutInSeconds : 900
DisplayName           : Test
DomainJoinInfo        :
EnableDefaultInternetAccess : False
FleetErrors           : {}
FleetType             : ON_DEMAND
IamRoleArn            :
IdleDisconnectTimeoutInSeconds : 900
ImageArn              : arn:aws:appstream:us-east-1:1234567890:image/Test
ImageName             : Test
InstanceType          : stream.standard.medium
MaxUserDurationInSeconds : 57600
Name                  : Test
State                 : STOPPED
VpcConfig             : Amazon.AppStream.Model.VpcConfig

```

- Per i dettagli sull'API, vedere [DescribeFleets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSIImageBuilderList

Il seguente esempio di codice mostra come utilizzare. Get-APSIImageBuilderList

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra i dettagli di un ImageBuilder

```
Get-APSIImageBuilderList -Name TestImage
```

Output:

```

AccessEndpoints       : {}
AppstreamAgentVersion : 06-19-2019
Arn                   : arn:aws:appstream:us-east-1:1234567890:image-builder/
TestImage
CreatedTime           : 1/14/2019 4:33:05 AM
Description           :
DisplayName           : TestImage
DomainJoinInfo        :
EnableDefaultInternetAccess : False
IamRoleArn            :

```

```

ImageArn           : arn:aws:appstream:us-east-1::image/Base-Image-
Builder-05-02-2018
ImageBuilderErrors : {}
InstanceType       : stream.standard.large
Name               : TestImage
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
Platform          : WINDOWS
State              : STOPPED
StateChangeReason  :
VpcConfig          : Amazon.AppStream.Model.VpcConfig

```

- Per i dettagli sull'API, vedere [DescribeImageBuilders](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSIImageList

Il seguente esempio di codice mostra come utilizzare. Get-APSIImageList

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra immagini private AppStream

```
Get-APSIImageList -Type PRIVATE | select DisplayName, ImageBuilderName, Visibility,
arn
```

Output:

DisplayName	ImageBuilderName	Visibility	Arn
-----	-----	-----	---
OfficeApps	OfficeApps	PRIVATE	arn:aws:appstream:us-east-1:123456789012:image/OfficeApps
SessionScriptV2	SessionScriptTest	PRIVATE	arn:aws:appstream:us-east-1:123456789012:image/SessionScriptV2

- Per i dettagli sull'API, vedere [DescribeImages](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSIImagePermission

Il seguente esempio di codice mostra come utilizzare. Get-APSIImagePermission

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza le autorizzazioni relative alle immagini su un'immagine condivisa AppStream

```
Get-APSIImagePermission -Name Powershell | select SharedAccountId,
@{n="AllowFleet";e={$_.ImagePermissions.AllowFleet}},
@{n="AllowImageBuilder";e={$_.ImagePermissions.AllowImageBuilder}}
```

Output:

```
SharedAccountId AllowFleet AllowImageBuilder
-----
123456789012      True              True
```

- Per i dettagli sull'API, vedere [DescribeImagePermissions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSSessionList

Il seguente esempio di codice mostra come utilizzare. Get-APSSessionList

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza l'elenco delle sessioni per un parco

```
Get-APSSessionList -FleetName PowershellFleet -StackName PowershellStack
```

Output:

```
AuthenticationType      : API
ConnectionState         : CONNECTED
FleetName               : PowershellFleet
Id                     : d8987c70-4394-4324-a396-2d485c26f2a2
MaxExpirationTime       : 12/27/2019 4:54:07 AM
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
StackName              : PowershellStack
StartTime              : 12/26/2019 12:54:12 PM
State                  : ACTIVE
UserId                 : Test
```

- Per i dettagli sull'API, vedere [DescribeSessions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSSStackList

Il seguente esempio di codice mostra come utilizzare. Get-APSSStackList

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra un elenco di Stack AppStream

```
Get-APSSStackList | Select DisplayName, Arn, CreatedTime
```

Output:

DisplayName	Arn
-----	---

PowershellStack	arn:aws:appstream:us-east-1:123456789012:stack/
PowershellStack	4/24/2019 8:49:29 AM
SessionScriptTest	arn:aws:appstream:us-east-1:123456789012:stack/
SessionScriptTest	9/12/2019 3:23:12 PM

- Per i dettagli sull'API, vedere [DescribeStacks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSTagsForResourceList

Il seguente esempio di codice mostra come utilizzare. Get-APSTagsForResourceList

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza i tag su una risorsa AppStream

```
Get-APSTagsForResourceList -ResourceArn arn:aws:appstream:us-east-1:123456789012:stack/SessionScriptTest
```

Output:

Key	Value
---	-----
StackState	Test

- Per i dettagli sull'API, vedere [ListTagsForResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSUsageReportSubscription

Il seguente esempio di codice mostra come utilizzare. Get-APSUsageReportSubscription

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza i dettagli AppStreamUsageReport di configurazione

```
Get-APSUsageReportSubscription
```

Output:

LastGeneratedReportDate	S3BucketName	SubscriptionErrors	Schedule
-----	-----		-----
1/1/0001 12:00:00 AM	appstream-logs-us-east-1-123456789012-sik1hnxe		DAILY {}

- Per i dettagli sull'API, vedere [DescribeUsageReportSubscriptions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSUser

Il seguente esempio di codice mostra come utilizzare. Get-APSUser

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza un elenco di utenti con stato abilitato

```
Get-APSUser -AuthenticationType USERPOOL | Select-Object UserName,
AuthenticationType, Enabled
```

Output:

UserName	AuthenticationType	Enabled
-----	-----	-----
foo1@contoso.com	USERPOOL	True
foo2@contoso.com	USERPOOL	True
foo3@contoso.com	USERPOOL	True
foo4@contoso.com	USERPOOL	True
foo5@contoso.com	USERPOOL	True

- Per i dettagli sull'API, vedere [DescribeUsers](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-APSUserStackAssociation

Il seguente esempio di codice mostra come utilizzare. Get-APSUserStackAssociation
Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza l'elenco degli utenti assegnati a uno stack

```
Get-APSUserStackAssociation -StackName PowershellStack
```

Output:

AuthenticationType	SendEmailNotification	StackName	UserName
-----	-----	-----	-----
USERPOOL	False	PowershellStack	TestUser1@lab.com
USERPOOL	False	PowershellStack	TestUser2@lab.com

- Per i dettagli sull'API, vedere [DescribeUserStackAssociations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-APSDirectoryConfig

Il seguente esempio di codice mostra come utilizzare. New-APSDirectoryConfig
Strumenti per PowerShell V4

Esempio 1: Questo esempio crea una configurazione di directory in AppStream

```
New-APSDirectoryConfig -ServiceAccountCredentials_AccountName contoso\ServiceAccount
-ServiceAccountCredentials_AccountPassword MyPass -DirectoryName contoso.com -
OrganizationalUnitDistinguishedName "OU=AppStream,OU=Contoso,DC=Contoso,DC=com"
```

Output:

```
CreatedTime          DirectoryName OrganizationalUnitDistinguishedNames
ServiceAccountCredentials
-----
-----
12/27/2019 11:00:30 AM contoso.com {OU=AppStream,OU=Contoso,DC=Contoso,DC=com}
Amazon.AppStream.Model.ServiceAccountCredentials
```

- Per i dettagli sull'API, vedere [CreateDirectoryConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-APSFleet

Il seguente esempio di codice mostra come utilizzare. New-APSFleet

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo parco veicoli AppStream

```
New-APSFleet -ComputeCapacity_DesiredInstance 1 -InstanceType stream.standard.medium
-Name TestFleet -DisplayName TestFleet -FleetType ON_DEMAND -
EnableDefaultInternetAccess $True -VpcConfig_SubnetIds "subnet-123ce32","subnet-
a1234cfd" -VpcConfig_SecurityGroupIds sg-4d012a34 -ImageName SessionScriptTest -
Region us-west-2
```

Output:

```
Arn                  : arn:aws:appstream:us-west-2:123456789012:fleet/
TestFleet
ComputeCapacityStatus : Amazon.AppStream.Model.ComputeCapacityStatus
CreatedTime          : 12/27/2019 11:24:42 AM
Description           :
DisconnectTimeoutInSeconds : 900
DisplayName           : TestFleet
DomainJoinInfo        :
```

```

EnableDefaultInternetAccess    : True
FleetErrors                    : {}
FleetType                      : ON_DEMAND
IamRoleArn                     :
IdleDisconnectTimeoutInSeconds : 0
ImageArn                      : arn:aws:appstream:us-west-2:123456789012:image/
SessionScriptTest              :
ImageName                      : SessionScriptTest
InstanceType                   : stream.standard.medium
MaxUserDurationInSeconds       : 57600
Name                           : TestFleet
State                          : STOPPED
VpcConfig                      : Amazon.AppStream.Model.VpcConfig

```

- Per i dettagli sull'API, vedere [CreateFleet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-APSImageBuilder

Il seguente esempio di codice mostra come utilizzare `New-APSImageBuilder`

Strumenti per PowerShell V4

Esempio 1: Questo esempio crea un Image Builder in AppStream

```

New-APSImageBuilder -InstanceType stream.standard.medium -Name TestIB -DisplayName
TestIB -ImageName AppStream-WinServer2012R2-12-12-2019 -EnableDefaultInternetAccess
$True -VpcConfig_SubnetId subnet-a1234cfd -VpcConfig_SecurityGroupIds sg-2d012a34 -
Region us-west-2

```

Output:

```

AccessEndpoints                : {}
AppstreamAgentVersion          : 12-16-2019
Arn                            : arn:aws:appstream:us-west-2:123456789012:image-
builder/TestIB
CreatedTime                    : 12/27/2019 11:39:24 AM
Description                    :
DisplayName                    : TestIB
DomainJoinInfo                 :
EnableDefaultInternetAccess    : True
IamRoleArn                     :

```

```

ImageArn          : arn:aws:appstream:us-west-2::image/AppStream-
WinServer2012R2-12-12-2019
ImageBuilderErrors : {}
InstanceType      : stream.standard.medium
Name              : TestIB
NetworkAccessConfiguration :
Platform          : WINDOWS
State              : PENDING
StateChangeReason :
VpcConfig         : Amazon.AppStream.Model.VpcConfig

```

- Per i dettagli sull'API, vedere [CreateImageBuilder](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-APSIImageBuilderStreamingURL

Il seguente esempio di codice mostra come utilizzare. New-APSIImageBuilderStreamingURL
Strumenti per PowerShell V4

Esempio 1: questo esempio crea un URL di ImageBuilder streaming con una validità di 2 ore

```
New-APSIImageBuilderStreamingURL -Name TestIB -Validity 7200 -Region us-west-2
```

Output:

```

Expires          StreamingURL
-----
12/27/2019 1:49:13 PM https://appstream2.us-west-2.aws.amazon.com/authenticate?
parameters=eyJ0eXB1IjoiQURNSU4iLCJleHBpcmVzIjoimTU3NzQ1NDU1MyIsImF3c0FjY291bnRJZCI6IjM5MzQwM

```

- Per i dettagli sull'API, vedere [CreateImageBuilderStreamingURL](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-APSSStack

Il seguente esempio di codice mostra come utilizzare. New-APSSStack
Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo stack AppStream

```
New-APSSStack -Name TestStack -DisplayName TestStack -ApplicationSettings_Enabled
$True -ApplicationSettings_SettingsGroup TestStack -Region us-west-2
```

Output:

```
AccessEndpoints      : {}
ApplicationSettings  : Amazon.AppStream.Model.ApplicationSettingsResponse
Arn                  : arn:aws:appstream:us-west-2:123456789012:stack/TestStack
CreatedTime          : 12/27/2019 12:34:19 PM
Description           :
DisplayName           : TestStack
EmbedHostDomains     : {}
FeedbackURL          :
Name                  : TestStack
RedirectURL           :
StackErrors           : {}
StorageConnectors    : {}
UserSettings         : {Amazon.AppStream.Model.UserSetting,
Amazon.AppStream.Model.UserSetting, Amazon.AppStream.Model.UserSetting,
Amazon.AppStream.Model.UserSetting}
```

- Per i dettagli sull'API, vedere [CreateStack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-APSSStreamingURL

Il seguente esempio di codice mostra come utilizzare New-APSSStreamingURL

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un URL di streaming per lo stack

```
New-APSSStreamingURL -StackName SessionScriptTest -FleetName SessionScriptNew -UserId
TestUser
```

Output:

```
Expires              StreamingURL
-----
12/27/2019 12:43:37 PM https://appstream2.us-east-1.aws.amazon.com/authenticate?
parameters=eyJ0eXB1IjoiriRU5EX1VTRVIlLCJleHBpcmVzIjoimTU3NzQ1MDYxNyIsImF3c0FjY291bnRjZCI6IjM5M...
```


- Per i dettagli sull'API, vedere [CreateStreamingURL](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-APSUsageReportSubscription

Il seguente esempio di codice mostra come utilizzare. New-APSUsageReportSubscription
Strumenti per PowerShell V4

Esempio 1: questo esempio abilita i report di AppStream utilizzo

```
New-APSUsageReportSubscription
```

Output:

S3BucketName	Schedule
-----	-----
appstream-logs-us-east-1-123456789012-sik2hnxe	DAILY

- Per i dettagli sull'API, vedere [CreateUsageReportSubscription](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-APSUser

Il seguente esempio di codice mostra come utilizzare. New-APSUser
Strumenti per PowerShell V4

Esempio 1: questo esempio crea un utente in USERPOOL

```
New-APSUser -UserName Test@lab.com -AuthenticationType USERPOOL -FirstName 'kt' -  
LastName 'aws' -Select ^UserName
```

Output:

```
Test@lab.com
```

- Per i dettagli sull'API, vedere [CreateUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-APSFleet

Il seguente esempio di codice mostra come utilizzare. Register-APSFleet

Strumenti per PowerShell V4

Esempio 1: questo esempio registra il parco con uno stack

```
Register-APSFleet -StackName TestStack -FleetName TestFleet -Region us-west-2
```

- Per i dettagli sull'API, vedere [AssociateFleet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-APSUserStackBatch

Il seguente esempio di codice mostra come utilizzare. Register-APSUserStackBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio assegna lo stack a un utente in USERPOOL

```
Register-APSUserStackBatch -UserStackAssociation  
@{AuthenticationType="USERPOOL";SendEmailNotification=  
$False;StackName="PowershellStack";UserName="TestUser1@lab.com"}
```

- Per i dettagli sull'API, vedere [BatchAssociateUserStack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-APSDirectoryConfig

Il seguente esempio di codice mostra come utilizzare. Remove-APSDirectoryConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la configurazione della AppStream directory

```
Remove-APSDirectoryConfig -DirectoryName contoso.com
```

Output:

```
Confirm
```

```
Are you sure you want to perform this action?  
Performing the operation "Remove-APSDirectoryConfig (DeleteDirectoryConfig)" on  
target "contoso.com".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): A
```

- Per i dettagli sull'API, vedere [DeleteDirectoryConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-APSFleet

Il seguente esempio di codice mostra come utilizzare. Remove-APSFleet

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove le eliminazioni di un gruppo AppStream

```
Remove-APSFleet -Name TestFleet -Region us-west-2
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing the operation "Remove-APSFleet (DeleteFleet)" on target "TestFleet".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): A
```

- Per i dettagli sull'API, vedere [DeleteFleet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-APSIImage

Il seguente esempio di codice mostra come utilizzare. Remove-APSIImage

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina un'immagine

```
Remove-APSIImage -Name TestImage -Region us-west-2
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSIImage (DeleteImage)" on target "TestImage".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): A

Applications           : {}
AppstreamAgentVersion  : LATEST
Arn                    : arn:aws:appstream:us-west-2:123456789012:image/
TestImage
BaseImageArn           :
CreatedTime            : 12/27/2019 1:34:10 PM
Description            :
DisplayName            : TestImage
ImageBuilderName       :
ImageBuilderSupported  : True
ImagePermissions       :
Name                   : TestImage
Platform               : WINDOWS
PublicBaseImageReleasedDate : 6/12/2018 12:00:00 AM
State                  : AVAILABLE
StateChangeReason      :
Visibility              : PRIVATE

```

- Per i dettagli sull'API, vedere [DeleteImage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-APSIImageBuilder

Il seguente esempio di codice mostra come utilizzare `Remove-APSIImageBuilder`

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina un ImageBuilder

```
Remove-APSIImageBuilder -Name TestIB -Region us-west-2
```

Output:

```

Confirm
Are you sure you want to perform this action?

```

```

Performing the operation "Remove-APSIImageBuilder (DeleteImageBuilder)" on target
"TestIB".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): A

AccessEndpoints           : {}
AppstreamAgentVersion     : 12-16-2019
Arn                       : arn:aws:appstream:us-west-2:123456789012:image-
builder/TestIB
CreatedTime               : 12/27/2019 11:39:24 AM
Description               :
DisplayName               : TestIB
DomainJoinInfo            :
EnableDefaultInternetAccess : True
IamRoleArn                :
ImageArn                  : arn:aws:appstream:us-west-2::image/AppStream-
WinServer2012R2-12-12-2019
ImageBuilderErrors        : {}
InstanceType              : stream.standard.medium
Name                      : TestIB
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
Platform                  : WINDOWS
State                     : DELETING
StateChangeReason         :
VpcConfig                  : Amazon.AppStream.Model.VpcConfig

```

- Per i dettagli sull'API, vedere [DeleteImageBuilder](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-APSIImagePermission

Il seguente esempio di codice mostra come utilizzare. Remove-APSIImagePermission

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove le autorizzazioni di un'immagine

```
Remove-APSIImagePermission -Name Powershell -SharedAccountId 123456789012
```

Output:

```
Confirm
```

```
Are you sure you want to perform this action?  
Performing the operation "Remove-APSIImagePermission (DeleteImagePermissions)" on  
target "Powershell".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): A
```

- Per i dettagli sull'API, vedere [DeleteImagePermissions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-APSResourceTag

Il seguente esempio di codice mostra come utilizzare. Remove-APSResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove un tag di risorsa dalla AppStream risorsa

```
Remove-APSResourceTag -ResourceArn arn:aws:appstream:us-east-1:123456789012:stack/  
SessionScriptTest -TagKey StackState
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing the operation "Remove-APSResourceTag (UntagResource)" on target  
"arn:aws:appstream:us-east-1:123456789012:stack/SessionScriptTest".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): A
```

- Per i dettagli sull'API, vedere [UntagResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-APSStack

Il seguente esempio di codice mostra come utilizzare. Remove-APSStack

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina uno stack

```
Remove-APSStack -Name TestStack -Region us-west-2
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSSStack (DeleteStack)" on target "TestStack".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): A
```

- Per i dettagli sull'API, vedere [DeleteStack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-APSUsageReportSubscription

Il seguente esempio di codice mostra come utilizzare. Remove-APSUsageReportSubscription
Strumenti per PowerShell V4

Esempio 1: questo esempio disattiva l'abbonamento a AppStream Usage Report

```
Remove-APSUsageReportSubscription
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APSUsageReportSubscription
(DeleteUsageReportSubscription)" on target "".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): A
```

- Per i dettagli sull'API, vedere [DeleteUsageReportSubscription](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-APSUser

Il seguente esempio di codice mostra come utilizzare. Remove-APSUser
Strumenti per PowerShell V4

Esempio 1: questo esempio elimina un utente da USERPOOL

```
Remove-APUser -UserName TestUser@lab.com -AuthenticationType USERPOOL
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-APUser (DeleteUser)" on target "TestUser@lab.com".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): A
```

- Per i dettagli sull'API, vedere [DeleteUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Revoke-APSSession

Il seguente esempio di codice mostra come utilizzare. `Revoke-APSSession`

Strumenti per PowerShell V4

Esempio 1: questo esempio revoca una sessione a fleet AppStream

```
Revoke-APSSession -SessionId 6cd2f9a3-f948-4aa1-8014-8a7dcde14877
```

- Per i dettagli sull'API, vedere [ExpireSession](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-APSFleet

Il seguente esempio di codice mostra come utilizzare. `Start-APSFleet`

Strumenti per PowerShell V4

Esempio 1: questo esempio avvia un parco

```
Start-APSFleet -Name PowershellFleet
```

- Per i dettagli sull'API, vedere [StartFleet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-APSIImageBuilder

Il seguente esempio di codice mostra come utilizzare. Start-APSIImageBuilder

Strumenti per PowerShell V4

Esempio 1: Questo esempio avvia un ImageBuilder

```
Start-APSIImageBuilder -Name TestImage
```

Output:

```
AccessEndpoints      : {}
AppstreamAgentVersion : 06-19-2019
Arn                  : arn:aws:appstream:us-east-1:123456789012:image-
builder/TestImage
CreatedTime           : 1/14/2019 4:33:05 AM
Description           :
DisplayName           : TestImage
DomainJoinInfo        :
EnableDefaultInternetAccess : False
IamRoleArn            :
ImageArn              : arn:aws:appstream:us-east-1::image/Base-Image-
Builder-05-02-2018
ImageBuilderErrors    : {}
InstanceType          : stream.standard.large
Name                  : TestImage
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
Platform              : WINDOWS
State                  : PENDING
StateChangeReason      :
VpcConfig             : Amazon.AppStream.Model.VpcConfig
```

- Per i dettagli sull'API, vedere [StartImageBuilder](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-APSFleet

Il seguente esempio di codice mostra come utilizzare. Stop-APSFleet

Strumenti per PowerShell V4

Esempio 1: questo esempio arresta un parco

```
Stop-APSFleet -Name PowershellFleet
```

- Per i dettagli sull'API, vedere [StopFleet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-APSIImageBuilder

Il seguente esempio di codice mostra come utilizzare. Stop-APSIImageBuilder

Strumenti per PowerShell V4

Esempio 1: questo esempio interrompe un ImageBuilder

```
Stop-APSIImageBuilder -Name TestImage
```

Output:

```
AccessEndpoints           : {}
AppstreamAgentVersion     : 06-19-2019
Arn                       : arn:aws:appstream:us-east-1:123456789012:image-
builder/TestImage
CreatedTime               : 1/14/2019 4:33:05 AM
Description               :
DisplayName               : TestImage
DomainJoinInfo            :
EnableDefaultInternetAccess : False
IamRoleArn                :
ImageArn                  : arn:aws:appstream:us-east-1::image/Base-Image-
Builder-05-02-2018
ImageBuilderErrors        : {}
InstanceType              : stream.standard.large
Name                      : TestImage
NetworkAccessConfiguration : Amazon.AppStream.Model.NetworkAccessConfiguration
Platform                  : WINDOWS
State                     : STOPPING
StateChangeReason          :
VpcConfig                  : Amazon.AppStream.Model.VpcConfig
```

- Per i dettagli sull'API, vedere [StopImageBuilder](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-APSFleet

Il seguente esempio di codice mostra come utilizzare. Unregister-APSFleet

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la registrazione di un parco in uno stack

```
Unregister-APSFleet -StackName TestStack -FleetName TestFleet -Region us-west-2
```

- Per i dettagli sull'API, vedere [DisassociateFleet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-APSUserStackBatch

Il seguente esempio di codice mostra come utilizzare. Unregister-APSUserStackBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove un utente da uno stack assegnato

```
Unregister-APSUserStackBatch -UserStackAssociation  
@{AuthenticationType="USERPOOL";SendEmailNotification=  
$False;StackName="PowershellStack";UserName="TestUser1@lab.com"}
```

- Per i dettagli sull'API, vedere [BatchDisassociateUserStack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-APSDirectoryConfig

Il seguente esempio di codice mostra come utilizzare. Update-APSDirectoryConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la configurazione della directory creata in AppStream

```
Update-APSDirectoryConfig -ServiceAccountCredentials_AccountName contoso  
\ServiceAccount -ServiceAccountCredentials_AccountPassword MyPass@1$@#
```

```
-DirectoryName contoso.com -OrganizationalUnitDistinguishedName
"OU=AppStreamNew,OU=Contoso,DC=Contoso,DC=com"
```

Output:

```
CreatedTime      DirectoryName  OrganizationalUnitDistinguishedNames
ServiceAccountCredentials
-----
-----
12/27/2019 3:50:02 PM contoso.com    {OU=AppStreamNew,OU=Contoso,DC=Contoso,DC=com}
Amazon.AppStream.Model.ServiceAccountCredentials
```

- Per i dettagli sull'API, vedere [UpdateDirectoryConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-APSFleet

Il seguente esempio di codice mostra come utilizzare. Update-APSFleet

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna le proprietà di un parco

```
Update-APSFleet -Name PowershellFleet -EnableDefaultInternetAccess $True -
DisconnectTimeoutInSeconds 950
```

Output:

```
Arn              : arn:aws:appstream:us-east-1:123456789012:fleet/
PowershellFleet
ComputeCapacityStatus : Amazon.AppStream.Model.ComputeCapacityStatus
CreatedTime        : 4/24/2019 8:39:41 AM
Description        : PowershellFleet
DisconnectTimeoutInSeconds : 950
DisplayName        : PowershellFleet
DomainJoinInfo     :
EnableDefaultInternetAccess : True
FleetErrors        : {}
FleetType          : ON_DEMAND
IamRoleArn         :
IdleDisconnectTimeoutInSeconds : 900
```

```

ImageArn          : arn:aws:appstream:us-east-1:123456789012:image/
Powershell
ImageName         : Powershell
InstanceType      : stream.standard.medium
MaxUserDurationInSeconds : 57600
Name              : PowershellFleet
State             : STOPPED
VpcConfig         : Amazon.AppStream.Model.VpcConfig

```

- Per i dettagli sull'API, vedere [UpdateFleet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-APSIImagePermission

Il seguente esempio di codice mostra come utilizzare. Update-APSIImagePermission

Strumenti per PowerShell V4

Esempio 1: questo esempio condivide un' AppStream immagine con un altro account

```

Update-APSIImagePermission -Name Powershell -SharedAccountId 123456789012 -
ImagePermissions_AllowFleet $True -ImagePermissions_AllowImageBuilder $True

```

- Per i dettagli sull'API, vedere [UpdateImagePermissions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-APSSStack

Il seguente esempio di codice mostra come utilizzare. Update-APSSStack

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna (abilita) la persistenza delle impostazioni dell'applicazione e le cartelle Home in uno stack

```

Update-APSSStack -Name PowershellStack -ApplicationSettings_Enabled $True
-ApplicationSettings_SettingsGroup PowershellStack -StorageConnector
@{ConnectorType="HOMEFOLDERS"}

```

Output:

```

AccessEndpoints      : {}
ApplicationSettings  : Amazon.AppStream.Model.ApplicationSettingsResponse
Arn                  : arn:aws:appstream:us-east-1:123456789012:stack/PowershellStack
CreatedTime          : 4/24/2019 8:49:29 AM
Description           : PowershellStack
DisplayName           : PowershellStack
EmbedHostDomains     : {}
FeedbackURL           :
Name                  : PowershellStack
RedirectURL           :
StackErrors           : {}
StorageConnectors    : {Amazon.AppStream.Model.StorageConnector,
  Amazon.AppStream.Model.StorageConnector}
UserSettings         : {Amazon.AppStream.Model.UserSetting,
  Amazon.AppStream.Model.UserSetting, Amazon.AppStream.Model.UserSetting,
  Amazon.AppStream.Model.UserSetting}

```

- Per i dettagli sull'API, vedere [UpdateStack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Aurora con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Aurora.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-RDSOrderableDBInstanceOption

Il seguente esempio di codice mostra come utilizzare. Get-RDSOrderableDBInstanceOption

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le versioni del motore DB che supportano una classe di istanza database specifica in una Regione AWS.

```
$params = @{  
    Engine = 'aurora-postgresql'  
    DBInstanceClass = 'db.r5.large'  
    Region = 'us-east-1'  
}  
Get-RDSOrderableDBInstanceOption @params
```

Esempio 2: questo esempio elenca le classi di istanza database che supportano una versione specifica del motore DB in una Regione AWS.

```
$params = @{  
    Engine = 'aurora-postgresql'  
    EngineVersion = '13.6'  
    Region = 'us-east-1'  
}  
Get-RDSOrderableDBInstanceOption @params
```

- Per i dettagli sull'API, vedere [DescribeOrderableDBInstanceOpzioni](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Auto Scaling con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Auto Scaling.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-ASLoadBalancer

Il seguente esempio di codice mostra come utilizzare. Add-ASLoadBalancer

Strumenti per PowerShell V4

Esempio 1: questo esempio collega il bilanciatore del carico specificato al gruppo Auto Scaling specificato.

```
Add-ASLoadBalancer -LoadBalancerName my-lb -AutoScalingGroupName my-asg
```

- Per i dettagli sull'API, vedere [AttachLoadBalancers](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Complete-ASLifecycleAction

Il seguente esempio di codice mostra come utilizzare. Complete-ASLifecycleAction

Strumenti per PowerShell V4

Esempio 1: questo esempio completa l'azione del ciclo di vita specificata.

```
Complete-ASLifecycleAction -LifecycleHookName myLifecycleHook -  
AutoScalingGroupName my-asg -LifecycleActionResult CONTINUE -LifecycleActionToken  
bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

- Per i dettagli sull'API, vedere [CompleteLifecycleAction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Disable-ASMetricsCollection

Il seguente esempio di codice mostra come utilizzare. Disable-ASMetricsCollection

Strumenti per PowerShell V4

Esempio 1: questo esempio disabilita il monitoraggio dei parametri specificati per il gruppo Auto Scaling specificato.


```
Disable-ASMetricsCollection -AutoScalingGroupName my-asg -Metric @("GroupMinSize",
"GroupMaxSize")
```

Esempio 2: questo esempio disabilita il monitoraggio di tutte le metriche specificate per il gruppo Auto Scaling specificato.

```
Disable-ASMetricsCollection -AutoScalingGroupName my-asg
```

- Per i dettagli sull'API, vedere [DisableMetricsCollection](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Dismount-ASInstance

Il seguente esempio di codice mostra come utilizzare. Dismount-ASInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio scollega l'istanza specificata dal gruppo Auto Scaling specificato e diminuisce la capacità desiderata in modo che Auto Scaling non avvii un'istanza di sostituzione.

```
Dismount-ASInstance -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $true
```

Output:

```
ActivityId           : 06733445-ce94-4039-be1b-b9f1866e276e
AutoScalingGroupName : my-asg
Cause                : At 2015-11-20T22:34:59Z instance i-93633f9b was detached in
                      response to a user request, shrinking
                      the capacity from 2 to 1.
Description          : Detaching EC2 instance: i-93633f9b
Details              : {"Availability Zone":"us-west-2b","Subnet
                      ID":"subnet-5264e837"}
EndTime              :
Progress              : 50
StartTime             : 11/20/2015 2:34:59 PM
StatusCode            : InProgress
StatusMessage        :
```

Esempio 2: questo esempio scollega l'istanza specificata dal gruppo Auto Scaling specificato senza diminuire la capacità desiderata. Auto Scaling avvia un'istanza di sostituzione.

```
Dismount-ASInstance -InstanceId i-7bf746a2 -AutoScalingGroupName my-asg -  
ShouldDecrementDesiredCapacity $false
```

Output:

```
ActivityId           : f43a3cd4-d38c-4af7-9fe0-d76ec2307b6d  
AutoScalingGroupName : my-asg  
Cause                : At 2015-11-20T22:34:59Z instance i-7bf746a2 was detached in  
                      response to a user request.  
Description          : Detaching EC2 instance: i-7bf746a2  
Details              : {"Availability Zone":"us-west-2b","Subnet  
                      ID":"subnet-5264e837"}  
EndTime              :  
Progress             : 50  
StartTime            : 11/20/2015 2:34:59 PM  
StatusCode           : InProgress  
StatusMessage        :
```

- Per i dettagli sull'API, vedere [DetachInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Dismount-ASLoadBalancer

Il seguente esempio di codice mostra come utilizzare Dismount-ASLoadBalancer

Strumenti per PowerShell V4

Esempio 1: questo esempio scollega il bilanciatore del carico specificato dal gruppo Auto Scaling specificato.

```
Dismount-ASLoadBalancer -LoadBalancerName my-lb -AutoScalingGroupName my-asg
```

- Per i dettagli sull'API, vedere [DetachLoadBalancers](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-ASMetricsCollection

Il seguente esempio di codice mostra come utilizzare. Enable-ASMetricsCollection

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita il monitoraggio delle metriche specificate per il gruppo Auto Scaling specificato.

```
Enable-ASMetricsCollection -Metric @("GroupMinSize", "GroupMaxSize") -  
AutoScalingGroupName my-asg -Granularity 1Minute
```

Esempio 2: questo esempio abilita il monitoraggio di tutte le metriche del gruppo Auto Scaling specificato.

```
Enable-ASMetricsCollection -AutoScalingGroupName my-asg -Granularity 1Minute
```

- Per i dettagli sull'API, vedere [EnableMetricsCollection](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enter-ASStandby

Il seguente esempio di codice mostra come utilizzare. Enter-ASStandby

Strumenti per PowerShell V4

Esempio 1: questo esempio mette l'istanza specificata in modalità standby e diminuisce la capacità desiderata in modo che Auto Scaling non avvii un'istanza di sostituzione.

```
Enter-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg -  
ShouldDecrementDesiredCapacity $true
```

Output:

```
ActivityId           : e36a5a54-ced6-4df8-bd19-708e2a59a649  
AutoScalingGroupName : my-asg  
Cause                : At 2015-11-22T15:48:06Z instance i-95b8484f was moved to  
                      standby in response to a user request,  
                      shrinking the capacity from 2 to 1.  
Description          : Moving EC2 instance to Standby: i-95b8484f
```

```

Details           : {"Availability Zone":"us-west-2b","Subnet
ID":"subnet-5264e837"}
EndTime           :
Progress          : 50
StartTime         : 11/22/2015 7:48:06 AM
StatusCode        : InProgress
StatusMessage     :

```

Esempio 2: questo esempio mette l'istanza specificata in modalità standby senza diminuire la capacità desiderata. Auto Scaling avvia un'istanza di sostituzione.

```

Enter-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $false

```

Output:

```

ActivityId         : e36a5a54-ced6-4df8-bd19-708e2a59a649
AutoScalingGroupName : my-asg
Cause              : At 2015-11-22T15:48:06Z instance i-95b8484f was moved to
standby in response to a user request.
Description        : Moving EC2 instance to Standby: i-95b8484f
Details           : {"Availability Zone":"us-west-2b","Subnet
ID":"subnet-5264e837"}
EndTime           :
Progress          : 50
StartTime         : 11/22/2015 7:48:06 AM
StatusCode        : InProgress
StatusMessage     :

```

- Per i dettagli sull'API, vedere [EnterStandby](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Exit-ASStandby

Il seguente esempio di codice mostra come utilizzare. Exit-ASStandby

Strumenti per PowerShell V4

Esempio 1: questo esempio imposta l'uscita dell'istanza specificata dalla modalità standby.

```

Exit-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg

```

Output:

```
ActivityId           : 1833d3e8-e32f-454e-b731-0670ad4c6934
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:51:21Z instance i-95b8484f was moved out of
                      standby in response to a user
                      request, increasing the capacity from 1 to 2.
Description          : Moving EC2 instance out of Standby: i-95b8484f
Details              : {"Availability Zone":"us-west-2b","Subnet
                      ID":"subnet-5264e837"}
EndTime              :
Progress             : 30
StartTime            : 11/22/2015 7:51:21 AM
StatusCode           : PreInService
StatusMessage        :
```

- Per i dettagli sull'API, vedere [ExitStandby](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASAccountLimit

Il seguente esempio di codice mostra come utilizzare. `Get-ASAccountLimit`

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive i limiti delle risorse Auto Scaling per il tuo AWS account.

```
Get-ASAccountLimit
```

Output:

```
MaxNumberOfAutoScalingGroups    : 20
MaxNumberOfLaunchConfigurations : 100
```

- Per i dettagli sull'API, vedere [DescribeAccountLimits](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASAdjustmentType

Il seguente esempio di codice mostra come utilizzare. `Get-ASAdjustmentType`

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive i tipi di regolazione supportati da Auto Scaling.

```
Get-ASAdjustmentType
```

Output:

```
Type
----
ChangeInCapacity
ExactCapacity
PercentChangeInCapacity
```

- Per i dettagli sull'API, vedere [DescribeAdjustmentTypes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASAutoScalingGroup

Il seguente esempio di codice mostra come utilizzare. Get-ASAutoScalingGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i nomi dei gruppi Auto Scaling.

```
Get-ASAutoScalingGroup | format-table -property AutoScalingGroupName
```

Output:

```
AutoScalingGroupName
-----
my-asg-1
my-asg-2
my-asg-3
my-asg-4
my-asg-5
my-asg-6
```

Esempio 2: questo esempio descrive il gruppo Auto Scaling specificato.

```
Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1
```

Output:

```
AutoScalingGroupARN      : arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-a11a-7b0acd480
                           f03:autoScalingGroupName/my-asg-1
AutoScalingGroupName     : my-asg-1
AvailabilityZones        : {us-west-2b, us-west-2a}
CreatedTime              : 3/1/2015 9:05:31 AM
DefaultCooldown          : 300
DesiredCapacity          : 2
EnabledMetrics            : {}
HealthCheckGracePeriod   : 300
HealthCheckType          : EC2
Instances                : {my-1c}
LaunchConfigurationName  : my-1c
LoadBalancerNames        : {}
MaxSize                   : 0
MinSize                   : 0
PlacementGroup           :
Status                   :
SuspendedProcesses       : {}
Tags                     : {}
TerminationPolicies      : {Default}
VPCZoneIdentifier         : subnet-e4f33493,subnet-5264e837
```

Esempio 3: questo esempio descrive i due gruppi Auto Scaling specificati.

```
Get-ASAutoScalingGroup -AutoScalingGroupName @"("my-asg-1", "my-asg-2")
```

Esempio 4: questo esempio descrive le istanze Auto Scaling per il gruppo Auto Scaling specificato.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1).Instances
```

Esempio 5: questo esempio descrive tutti i gruppi Auto Scaling.

```
Get-ASAutoScalingGroup
```

Esempio 6: Questo LaunchTemplate esempio descrive il gruppo Auto Scaling specificato. Questo esempio presuppone che l'opzione "Opzioni di acquisto delle istanze" sia impostata su "Aderisci al modello di avvio". Nel caso in cui questa opzione sia impostata su «Combina opzioni di acquisto e tipi di istanze», è LaunchTemplate possibile accedervi utilizzando "MixedInstancesPolicy". LaunchTemplate«proprietà.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-ag-1).LaunchTemplate
```

Output:

LaunchTemplateId	LaunchTemplateName	Version
-----	-----	-----
lt-06095fd619cb40371	test-launch-template	\$Default

- Per i dettagli sull'API, vedere [DescribeAutoScalingGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASAutoScalingInstance

Il seguente esempio di codice mostra come utilizzare. Get-ASAutoScalingInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le IDs istanze di Auto Scaling.

```
Get-ASAutoScalingInstance | format-table -property InstanceId
```

Output:

InstanceId

i-12345678
i-87654321
i-abcd1234

Esempio 2: questo esempio descrive l'istanza Auto Scaling specificata.

```
Get-ASAutoScalingInstance -InstanceId i-12345678
```


Output:

```
AutoScalingGroupName    : my-asg
AvailabilityZone         : us-west-2b
HealthStatus            : HEALTHY
InstanceId               : i-12345678
LaunchConfigurationName : my-lc
LifecycleState           : InService
```

Esempio 3: questo esempio descrive le due istanze Auto Scaling specificate.

```
Get-ASAutoScalingInstance -InstanceId @"i-12345678", "i-87654321")
```

Esempio 4: questo esempio descrive le istanze Auto Scaling per il gruppo Auto Scaling specificato.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-asg).Instances | Get-
ASAutoScalingInstance
```

Esempio 5: questo esempio descrive tutte le istanze Auto Scaling.

```
Get-ASAutoScalingInstance
```

- Per i dettagli sull'API, vedere [DescribeAutoScalingInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASAutoScalingNotificationType

Il seguente esempio di codice mostra come utilizzare `Get-ASAutoScalingNotificationType`

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i tipi di notifica supportati da Auto Scaling.

```
Get-ASAutoScalingNotificationType
```

Output:

```
autoscaling:EC2_INSTANCE_LAUNCH
```

```
autoscaling:EC2_INSTANCE_LAUNCH_ERROR
autoscaling:EC2_INSTANCE_TERMINATE
autoscaling:EC2_INSTANCE_TERMINATE_ERROR
autoscaling:TEST_NOTIFICATION
```

- Per i dettagli sull'API, vedere [DescribeAutoScalingNotificationTypes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASLaunchConfiguration

Il seguente esempio di codice mostra come utilizzare. Get-ASLaunchConfiguration

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i nomi delle configurazioni di avvio.

```
Get-ASLaunchConfiguration | format-table -property LaunchConfigurationName
```

Output:

```
LaunchConfigurationName
-----
my-lc-1
my-lc-2
my-lc-3
my-lc-4
my-lc-5
```

Esempio 2: questo esempio descrive la configurazione di avvio specificata.

```
Get-ASLaunchConfiguration -LaunchConfigurationName my-lc-1
```

Output:

```
AssociatePublicIpAddress      : True
BlockDeviceMappings           : {/dev/xvda}
ClassicLinkVPCId              :
ClassicLinkVPCSecurityGroups  : {}
CreatedTime                   : 12/12/2014 3:22:08 PM
EbsOptimized                   : False
```

```

IamInstanceProfile      :
ImageId                 : ami-043a5034
InstanceMonitoring      : Amazon.AutoScaling.Model.InstanceMonitoring
InstanceType            : t2.micro
KernelId                :
KeyName                 :
LaunchConfigurationARN  : arn:aws:autoscaling:us-
                        west-2:123456789012:launchConfiguration:7e5f31e4-693b-4604-9322-
                        e6f68d7fafad:launchConfigurationName/my-lc-1
LaunchConfigurationName : my-lc-1
PlacementTenancy        :
RamdiskId               :
SecurityGroups           : {sg-67ef0308}
SpotPrice               :
UserData                :

```

Esempio 3: questo esempio descrive le due configurazioni di avvio specificate.

```
Get-ASLaunchConfiguration -LaunchConfigurationName @("my-lc-1", "my-lc-2")
```

Esempio 4: questo esempio descrive tutte le configurazioni di avvio.

```
Get-ASLaunchConfiguration
```

- Per i dettagli sull'API, vedere [DescribeLaunchConfigurations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASLifecycleHook

Il seguente esempio di codice mostra come utilizzare. Get-ASLifecycleHook

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'hook del ciclo di vita specificato.

```
Get-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName myLifecycleHook
```

Output:

```
AutoScalingGroupName : my-asg
```

```
DefaultResult      : ABANDON
GlobalTimeout      : 172800
HeartbeatTimeout   : 3600
LifecycleHookName  : myLifecycleHook
LifecycleTransition : auto-scaling:EC2_INSTANCE_LAUNCHING
NotificationMetadata :
NotificationTargetARN : arn:aws:sns:us-west-2:123456789012:my-topic
RoleARN            : arn:aws:iam::123456789012:role/my-iam-role
```

Esempio 2: questo esempio descrive tutti gli hook del ciclo di vita del gruppo Auto Scaling specificato.

```
Get-ASLifecycleHook -AutoScalingGroupName my-asg
```

Esempio 3: questo esempio descrive tutti gli hook del ciclo di vita di tutti i gruppi Auto Scaling.

```
Get-ASLifecycleHook
```

- Per i dettagli sull'API, vedere [DescribeLifecycleHooks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASLifecycleHookType

Il seguente esempio di codice mostra come utilizzare. Get-ASLifecycleHookType

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i tipi di hook del ciclo di vita supportati da Auto Scaling.

```
Get-ASLifecycleHookType
```

Output:

```
autoscaling:EC2_INSTANCE_LAUNCHING
auto-scaling:EC2_INSTANCE_TERMINATING
```

- Per i dettagli sull'API, vedere [DescribeLifecycleHookTypes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASLoadBalancer

Il seguente esempio di codice mostra come utilizzare. Get-ASLoadBalancer

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive i bilanciatori del carico del gruppo Auto Scaling specificato.

```
Get-ASLoadBalancer -AutoScalingGroupName my-asg
```

Output:

LoadBalancerName	State
-----	----
my-lb	Added

- Per i dettagli sull'API, vedere [DescribeLoadBalancers](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASMetricCollectionType

Il seguente esempio di codice mostra come utilizzare. Get-ASMetricCollectionType

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i tipi di raccolta delle metriche supportati da Auto Scaling.

```
(Get-ASMetricCollectionType).Metrics
```

Output:

```
Metric
-----
GroupMinSize
GroupMaxSize
GroupDesiredCapacity
GroupInServiceInstances
GroupPendingInstances
GroupTerminatingInstances
GroupStandbyInstances
```

```
GroupTotalInstances
```

Esempio 2: questo esempio elenca le granularità corrispondenti.

```
(Get-ASMetricCollectionType).Granularities
```

Output:

```
Granularity
-----
1Minute
```

- Per i dettagli sull'API, vedere [DescribeMetricCollectionTypes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASNotificationConfiguration

Il seguente esempio di codice mostra come utilizzare Get-ASNotificationConfiguration

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le azioni di notifica associate al gruppo Auto Scaling specificato.

```
Get-ASNotificationConfiguration -AutoScalingGroupName my-asg | format-list
```

Output:

```
AutoScalingGroupName : my-asg
NotificationType      : auto-scaling:EC2_INSTANCE_LAUNCH
TopicARN              : arn:aws:sns:us-west-2:123456789012:my-topic

AutoScalingGroupName : my-asg
NotificationType      : auto-scaling:EC2_INSTANCE_TERMINATE
TopicARN              : arn:aws:sns:us-west-2:123456789012:my-topic
```

Esempio 2: questo esempio descrive le azioni di notifica associate a tutti i gruppi Auto Scaling.

```
Get-ASNotificationConfiguration
```

- Per i dettagli sull'API, vedere [DescribeNotificationConfigurations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASPolicy

Il seguente esempio di codice mostra come utilizzare. Get-ASPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive tutte le policy del gruppo Auto Scaling specificato.

```
Get-ASPolicy -AutoScalingGroupName my-asg
```

Output:

```
AdjustmentType      : ChangeInCapacity
Alarms              : {}
AutoScalingGroupName : my-asg
Cooldown            : 0
EstimatedInstanceWarmup : 0
MetricAggregationType :
MinAdjustmentMagnitude : 0
MinAdjustmentStep    : 0
PolicyARN           : arn:aws:auto-scaling:us-
west-2:123456789012:scalingPolicy:aa3836ab-5462-42c7-adab-e1d769fc24ef
                    :autoScalingGroupName/my-asg:policyName/myScaleInPolicy
PolicyName          : myScaleInPolicy
PolicyType           : SimpleScaling
ScalingAdjustment    : -1
StepAdjustments      : {}
```

Esempio 2: questo esempio descrive tutte le policy specificate per il gruppo Auto Scaling specificato.

```
Get-ASPolicy -AutoScalingGroupName my-asg -PolicyName @("myScaleOutPolicy",
"myScaleInPolicy")
```

Esempio 3: questo esempio descrive tutte le policy per tutti i gruppi Auto Scaling.

```
Get-ASPolicy
```

- Per i dettagli sull'API, vedere [DescribePolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASScalingActivity

Il seguente esempio di codice mostra come utilizzare. Get-ASScalingActivity

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le attività di dimensionamento nelle ultime sei settimane per il gruppo Auto Scaling specificato.

```
Get-ASScalingActivity -AutoScalingGroupName my-asg
```

Output:

```
ActivityId           : 063308ae-aa22-4a9b-94f4-9fae4EXAMPLE
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:45:16Z a user request explicitly set group
                      desired capacity changing the desired
                      capacity from 1 to 2. At 2015-11-22T15:45:34Z an instance
                      was started in response to a difference
                      between desired and actual capacity, increasing the capacity
                      from 1 to 2.
Description          : Launching a new EC2 instance: i-26e715fc
Details              : {"Availability Zone":"us-west-2b","Subnet
                      ID":"subnet-5264e837"}
EndTime              : 11/22/2015 7:46:09 AM
Progress              : 100
StartTime             : 11/22/2015 7:45:35 AM
StatusCode            : Successful
StatusMessage        :

ActivityId           : ce719997-086d-4c73-a2f1-ab703EXAMPLE
AutoScalingGroupName : my-asg
Cause                : At 2015-11-20T22:57:53Z a user request created an
                      AutoScalingGroup changing the desired capacity
                      from 0 to 1. At 2015-11-20T22:57:58Z an instance was
                      started in response to a difference betwe
                      en desired and actual capacity, increasing the capacity from
                      0 to 1.
Description          : Launching a new EC2 instance: i-93633f9b
```



```

Details           : {"Availability Zone":"us-west-2b","Subnet
ID":"subnet-5264e837"}
EndTime           : 11/20/2015 2:58:32 PM
Progress          : 100
StartTime         : 11/20/2015 2:57:59 PM
StatusCode        : Successful
StatusMessage     :

```

Esempio 2: questo esempio descrive l'attività di dimensionamento specificata.

```
Get-ASScalingActivity -ActivityId "063308ae-aa22-4a9b-94f4-9fae4EXAMPLE"
```

Esempio 3: questo esempio descrive le attività di dimensionamento nelle ultime sei settimane per tutti i gruppi Auto Scaling.

```
Get-ASScalingActivity
```

- Per i dettagli sull'API, vedere [DescribeScalingActivities](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASScalingProcessType

Il seguente esempio di codice mostra come utilizzare `Get-ASScalingProcessType`

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i tipi di processo supportati da Auto Scaling.

```
Get-ASScalingProcessType
```

Output:

```

ProcessName
-----
AZRebalance
AddToLoadBalancer
AlarmNotification
HealthCheck
Launch
ReplaceUnhealthy

```

```
ScheduledActions
Terminate
```

- Per i dettagli sull'API, vedere [DescribeScalingProcessTypes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASScheduledAction

Il seguente esempio di codice mostra come utilizzare Get-ASScheduledAction

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le azioni di dimensionamento pianificate del gruppo Auto Scaling specificato.

```
Get-ASScheduledAction -AutoScalingGroupName my-asg
```

Output:

```
AutoScalingGroupName : my-asg
DesiredCapacity       : 10
EndTime               :
MaxSize               :
MinSize               :
Recurrence             :
ScheduledActionARN    : arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8a4c5f24-6ec6-4306-a2dd-f7
2c3af3a4d6:autoScalingGroupName/my-asg:scheduledActionName/
myScheduledAction
ScheduledActionName   : myScheduledAction
StartTime              : 11/30/2015 8:00:00 AM
Time                  : 11/30/2015 8:00:00 AM
```

Esempio 2: questo esempio descrive le azioni di dimensionamento pianificate specificate.

```
Get-ASScheduledAction -ScheduledActionName @"(myScheduledScaleOut",
"myScheduledScaleIn")
```

Esempio 3: questo esempio descrive le azioni di dimensionamento pianificate che iniziano entro l'ora specificata.

```
Get-ASScheduledAction -StartTime "2015-12-01T08:00:00Z"
```

Esempio 4: questo esempio descrive le azioni di dimensionamento pianificate che terminano entro l'ora specificata.

```
Get-ASScheduledAction -EndTime "2015-12-30T08:00:00Z"
```

Esempio 5: questo esempio descrive le azioni di dimensionamento pianificate di tutti i gruppi Auto Scaling specificati.

```
Get-ASScheduledAction
```

- Per i dettagli sull'API, vedere [DescribeScheduledActions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASTag

Il seguente esempio di codice mostra come utilizzare. Get-ASTag

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive i tag con un valore di chiave "myTag" o "myTag2". I valori possibili per il nome del filtro sono auto-scaling-group, 'key', 'value' e 'propagate-at-launch'. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
Get-ASTag -Filter @( @{ Name="key"; Values=@("myTag", "myTag2")} )
```

Output:

```
Key           : myTag2
PropagateAtLaunch : True
ResourceId     : my-asg
ResourceType   : auto-scaling-group
Value          : myTagValue2

Key           : myTag
PropagateAtLaunch : True
ResourceId     : my-asg
ResourceType   : auto-scaling-group
```

```
Value           : myTagValue
```

Esempio 2: con la PowerShell versione 2, è necessario utilizzare New-Object per creare il filtro per il parametro Filter.

```
$keys = New-Object string[] 2
$keys[0] = "myTag"
$keys[1] = "myTag2"
$filter = New-Object Amazon.AutoScaling.Model.Filter
$filter.Name = "key"
$filter.Values = $keys
Get-ASTag -Filter @( $filter )
```

Esempio 3: questo esempio descrive tutti i tag per tutti i gruppi Auto Scaling.

```
Get-ASTag
```

- Per i dettagli sull'API, vedere [DescribeTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASTerminationPolicyType

Il seguente esempio di codice mostra come utilizzare Get-ASTerminationPolicyType

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le policy di terminazione supportate da Auto Scaling.

```
Get-ASTerminationPolicyType
```

Output:

```
ClosestToNextInstanceHour
Default
NewestInstance
OldestInstance
OldestLaunchConfiguration
```

- Per i dettagli sull'API, vedere [DescribeTerminationPolicyTypes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Mount-ASInstance

Il seguente esempio di codice mostra come utilizzare. Mount-ASInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio collega l'istanza specificata al gruppo Auto Scaling specificato. Auto Scaling aumenta automaticamente la capacità desiderata del gruppo Auto Scaling.

```
Mount-ASInstance -InstanceId i-93633f9b -AutoScalingGroupName my-asg
```

- Per i dettagli sull'API, vedere [AttachInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ASAutoScalingGroup

Il seguente esempio di codice mostra come utilizzare. New-ASAutoScalingGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un gruppo Auto Scaling con il nome e gli attributi specificati. La capacità predefinita desiderata è la dimensione minima. Pertanto, questo gruppo Auto Scaling avvia due istanze, una in ciascuna delle due zone di disponibilità specificate.

```
New-ASAutoScalingGroup -AutoScalingGroupName my-asg -LaunchConfigurationName my-lc -  
MinSize 2 -MaxSize 6 -AvailabilityZone @("us-west-2a", "us-west-2b")
```

- Per i dettagli sull'API, vedere [CreateAutoScalingGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ASLaunchConfiguration

Il seguente esempio di codice mostra come utilizzare. New-ASLaunchConfiguration

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una configurazione di avvio denominata "my-lc". Le EC2 istanze lanciate dai gruppi Auto Scaling che utilizzano questa configurazione di avvio utilizzano il tipo di istanza, l'AMI, il gruppo di sicurezza e il ruolo IAM specificati.

```
New-ASLaunchConfiguration -LaunchConfigurationName my-lc -InstanceType "m3.medium" -ImageId "ami-12345678" -SecurityGroup "sg-12345678" -IamInstanceProfile "myIamRole"
```

- Per i dettagli sull'API, vedere [CreateLaunchConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ASAutoScalingGroup

Il seguente esempio di codice mostra come utilizzare. Remove-ASAutoScalingGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il gruppo Auto Scaling specificato se non include istanze in esecuzione. Prima di procedere con l'operazione, viene richiesta una conferma.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASAutoScalingGroup (DeleteAutoScalingGroup)" on Target
"my-asg".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Esempio 2: se si specifica il parametro Force, non viene richiesta la conferma per continuare l'operazione.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -Force
```

Esempio 3: questo esempio elimina il gruppo Auto Scaling specificato e termina qualsiasi istanza in esecuzione in esso contenuta.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -ForceDelete $true -Force
```

- Per i dettagli sull'API, vedere [DeleteAutoScalingGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ASLaunchConfiguration

Il seguente esempio di codice mostra come utilizzare. Remove-ASLaunchConfiguration

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la configurazione di avvio specificata se non è collegata a un gruppo Auto Scaling. Prima di procedere con l'operazione, viene richiesta una conferma.

```
Remove-ASLaunchConfiguration -LaunchConfigurationName my-lc
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASLaunchConfiguration (DeleteLaunchConfiguration)" on
Target "my-lc".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Esempio 2: se si specifica il parametro Force, non viene richiesta la conferma per continuare l'operazione.

```
Remove-ASLaunchConfiguration -LaunchConfigurationName my-lc -Force
```

- Per i dettagli sull'API, vedere [DeleteLaunchConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ASLifecycleHook

Il seguente esempio di codice mostra come utilizzare. Remove-ASLifecycleHook

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'hook del ciclo di vita specificato del gruppo Auto Scaling specificato. Prima di procedere con l'operazione, viene richiesta una conferma.

```
Remove-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
myLifecycleHook
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASLifecycleHook (DeleteLifecycleHook)" on Target
"myLifecycleHook".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Esempio 2: se si specifica il parametro Force, non viene richiesta la conferma per continuare l'operazione.

```
Remove-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
myLifecycleHook -Force
```

- Per i dettagli sull'API, vedere [DeleteLifecycleHook](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ASNotificationConfiguration

Il seguente esempio di codice mostra come utilizzare Remove-ASNotificationConfiguration

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'azione di notifica specificata. Prima di procedere con l'operazione, viene richiesta una conferma.

```
Remove-ASNotificationConfiguration -AutoScalingGroupName my-asg -TopicARN
"arn:aws:sns:us-west-2:123456789012:my-topic"
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASNotificationConfiguration
(DeleteNotificationConfiguration)" on Target
"arn:aws:sns:us-west-2:123456789012:my-topic".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```


Esempio 2: se si specifica il parametro `Force`, non viene richiesta la conferma per continuare l'operazione.

```
Remove-ASNotificationConfiguration -AutoScalingGroupName my-asg -TopicARN  
"arn:aws:sns:us-west-2:123456789012:my-topic" -Force
```

- Per i dettagli sull'API, vedere [DeleteNotificationConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ASPolicy

Il seguente esempio di codice mostra come utilizzare `Remove-ASPolicy`

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la policy specificata del gruppo Auto Scaling specificato. Prima di procedere con l'operazione, viene richiesta una conferma.

```
Remove-ASPolicy -AutoScalingGroupName my-asg -PolicyName myScaleInPolicy
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing operation "Remove-ASPolicy (DeletePolicy)" on Target "myScaleInPolicy".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"):
```

Esempio 2: se si specifica il parametro `Force`, non viene richiesta la conferma per continuare l'operazione.

```
Remove-ASPolicy -AutoScalingGroupName my-asg -PolicyName myScaleInPolicy -Force
```

- Per i dettagli sull'API, vedere [DeletePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ASScheduledAction

Il seguente esempio di codice mostra come utilizzare `Remove-ASScheduledAction`

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'azione pianificata specificata per il gruppo Auto Scaling specificato. Prima di procedere con l'operazione, viene richiesta una conferma.

```
Remove-ASScheduledAction -AutoScalingGroupName my-asg -ScheduledAction
"myScheduledAction"
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASScheduledAction (DeleteScheduledAction)" on Target
"myScheduledAction".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Esempio 2: se si specifica il parametro Force, non viene richiesta la conferma per continuare l'operazione.

```
Remove-ASScheduledAction -AutoScalingGroupName my-asg -ScheduledAction
"myScheduledAction" -Force
```

- Per i dettagli sull'API, vedere [DeleteScheduledAction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ASTag

Il seguente esempio di codice mostra come utilizzare. Remove-ASTag

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il tag specificato dal gruppo Auto Scaling specificato. Prima di procedere con l'operazione, viene richiesta una conferma. La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
Remove-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
Key="myTag" } )
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ASTag (DeleteTags)" on target
"Amazon.AutoScaling.Model.Tag".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Esempio 2: se si specifica il parametro `Force`, non viene richiesta la conferma per continuare l'operazione.

```
Remove-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
Key="myTag" } ) -Force
```

Esempio 3: con PowerShell versione 2, è necessario utilizzare `New-Object` per creare il tag per il parametro `Tag`.

```
$tag = New-Object Amazon.AutoScaling.Model.Tag
$tag.ResourceType = "auto-scaling-group"
$tag.ResourceId = "my-asg"
$tag.Key = "myTag"
Remove-ASTag -Tag $tag -Force
```

- Per i dettagli sull'API, vedere [DeleteTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Resume-ASProcess

Il seguente esempio di codice mostra come utilizzare `Resume-ASProcess`

Strumenti per PowerShell V4

Esempio 1: questo esempio riprende il processo di dimensionamento automatico specificato per il gruppo Auto Scaling specificato.

```
Resume-ASProcess -AutoScalingGroupName my-asg -ScalingProcess "AlarmNotification"
```

Esempio 2: questo esempio riprende i processi di dimensionamento automatico specificati per il gruppo Auto Scaling specificato.

```
Resume-ASProcess -AutoScalingGroupName my-asg
```

- Per i dettagli sull'API, vedere [ResumeProcesses](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ASDesiredCapacity

Il seguente esempio di codice mostra come utilizzare. Set-ASDesiredCapacity

Strumenti per PowerShell V4

Esempio 1: questo esempio imposta le dimensioni del gruppo Auto Scaling specificato.

```
Set-ASDesiredCapacity -AutoScalingGroupName my-asg -DesiredCapacity 2
```

Esempio 2: questo esempio imposta la dimensione del gruppo Auto Scaling specificato e attende il raggiungimento del tempo di raffreddamento prima di passare alla nuova dimensione.

```
Set-ASDesiredCapacity -AutoScalingGroupName my-asg -DesiredCapacity 2 -HonorCooldown $true
```

- Per i dettagli sull'API, vedere [SetDesiredCapacity](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ASInstanceHealth

Il seguente esempio di codice mostra come utilizzare. Set-ASInstanceHealth

Strumenti per PowerShell V4

Esempio 1: questo esempio imposta lo stato dell'istanza specificata su "Unhealthy" (Non integro), eliminandola dal servizio. Auto Scaling termina e sostituisce l'istanza.

```
Set-ASInstanceHealth -HealthStatus Unhealthy -InstanceId i-93633f9b
```

Esempio 2: questo esempio imposta lo stato dell'istanza specificata su "Unhealthy" (Non integro), mantenendola nel servizio. L'eventuale periodo di tolleranza definito per il controllo dell'integrità per il gruppo Auto Scaling non viene rispettato.

```
Set-ASInstanceHealth -HealthStatus Healthy -InstanceId i-93633f9b -  
ShouldRespectGracePeriod $false
```

- Per i dettagli sull'API, vedere [SetInstanceHealth](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ASInstanceProtection

Il seguente esempio di codice mostra come utilizzare. Set-ASInstanceProtection

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita la protezione dell'istanza specificata.

```
Set-ASInstanceProtection -AutoScalingGroupName my-asg -InstanceId i-12345678 -  
ProtectedFromScaleIn $true
```

Esempio 2: questo esempio disabilita la protezione dell'istanza specificata.

```
Set-ASInstanceProtection -AutoScalingGroupName my-asg -InstanceId i-12345678 -  
ProtectedFromScaleIn $false
```

- Per i dettagli sull'API, vedere [SetInstanceProtection](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ASTag

Il seguente esempio di codice mostra come utilizzare. Set-ASTag

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un unico tag al gruppo Auto Scaling specificato. La chiave del tag è 'myTag' e il valore del tag è ". myTagValue Auto Scaling propaga questo tag alle EC2 istanze successive avviate dal gruppo Auto Scaling. La sintassi utilizzata da questo esempio richiede PowerShell la versione 3 o successiva.

```
Set-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";  
Key="myTag"; Value="myTagValue"; PropagateAtLaunch=$true} )
```

Esempio 2: con PowerShell la versione 2, è necessario utilizzare New-Object per creare il tag per il parametro Tag.

```
$tag = New-Object Amazon.AutoScaling.Model.Tag
$tag.ResourceType = "auto-scaling-group"
$tag.ResourceId = "my-asg"
$tag.Key = "myTag"
$tag.Value = "myTagValue"
$tag.PropagateAtLaunch = $true
Set-ASTag -Tag $tag
```

- Per i dettagli sull'API, vedere [CreateOrUpdateTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-ASPolicy

Il seguente esempio di codice mostra come utilizzare. Start-ASPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio esegue la policy specificata per il gruppo Auto Scaling specificato.

```
Start-ASPolicy -AutoScalingGroupName my-asg -PolicyName "myScaleInPolicy"
```

Esempio 2: questo esempio esegue la policy specificata per il gruppo Auto Scaling specificato dopo aver atteso il raggiungimento del tempo di raffreddamento.

```
Start-ASPolicy -AutoScalingGroupName my-asg -PolicyName "myScaleInPolicy" -
HonorCooldown $true
```

- Per i dettagli sull'API, vedere [ExecutePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-ASInstanceInAutoScalingGroup

Il seguente esempio di codice mostra come utilizzare. Stop-ASInstanceInAutoScalingGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio termina l'istanza specificata e diminuisce la capacità desiderata del relativo gruppo Auto Scaling in modo che Auto Scaling non avvii un'istanza di sostituzione.

```
Stop-ASInstanceInAutoScalingGroup -InstanceId i-93633f9b -  
ShouldDecrementDesiredCapacity $true
```

Output:

```
ActivityId      : 2e40d9bd-1902-444c-abf3-6ea0002efdc5  
AutoScalingGroupName :  
Cause          : At 2015-11-22T16:09:03Z instance i-93633f9b was taken out of  
                service in response to a user  
                request, shrinking the capacity from 2 to 1.  
Description    : Terminating EC2 instance: i-93633f9b  
Details        : {"Availability Zone":"us-west-2b","Subnet  
                ID":"subnet-5264e837"}  
EndTime        :  
Progress       : 0  
StartTime      : 11/22/2015 8:09:03 AM  
StatusCode     : InProgress  
StatusMessage  :
```

Esempio 2: questo esempio termina l'istanza specificata senza diminuire la capacità desiderata del relativo gruppo Auto Scaling. Auto Scaling avvia un'istanza di sostituzione.

```
Stop-ASInstanceInAutoScalingGroup -InstanceId i-93633f9b -  
ShouldDecrementDesiredCapacity $false
```

Output:

```
ActivityId      : 2e40d9bd-1902-444c-abf3-6ea0002efdc5  
AutoScalingGroupName :  
Cause          : At 2015-11-22T16:09:03Z instance i-93633f9b was taken out of  
                service in response to a user  
                request.  
Description    : Terminating EC2 instance: i-93633f9b  
Details        : {"Availability Zone":"us-west-2b","Subnet  
                ID":"subnet-5264e837"}  
EndTime        :  
Progress       : 0
```

```
StartTime      : 11/22/2015 8:09:03 AM
StatusCode     : InProgress
StatusMessage  :
```

- Per i dettagli sull'API, vedere [TerminateInstanceInAutoScalingGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Suspend-ASProcess

Il seguente esempio di codice mostra come utilizzare. Suspend-ASProcess

Strumenti per PowerShell V4

Esempio 1: questo esempio sospende il processo di dimensionamento automatico specificato per il gruppo Auto Scaling specificato.

```
Suspend-ASProcess -AutoScalingGroupName my-asg -ScalingProcess "AlarmNotification"
```

Esempio 2: questo esempio riprende tutti i processi di dimensionamento automatico specificati per il gruppo Auto Scaling specificato.

```
Suspend-ASProcess -AutoScalingGroupName my-asg
```

- Per i dettagli sull'API, vedere [SuspendProcesses](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-ASAutoScalingGroup

Il seguente esempio di codice mostra come utilizzare. Update-ASAutoScalingGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio imposta le dimensioni minima e massima del gruppo Auto Scaling specificato.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -MaxSize 5 -MinSize 1
```

Esempio 2: questo esempio aggiorna il tempo di raffreddamento predefinito del gruppo Auto Scaling specificato.


```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -DefaultCooldown 10
```

Esempio 3: questo esempio aggiorna le zone di disponibilità del gruppo Auto Scaling specificato.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -AvailabilityZone @("us-west-2a", "us-west-2b")
```

Esempio 4: questo esempio aggiorna il gruppo Auto Scaling specificato in modo da utilizzare i controlli dell'integrità di Elastic Load Balancing.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -HealthCheckType ELB -  
HealthCheckGracePeriod 60
```

- Per i dettagli sull'API, vedere [UpdateAutoScalingGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-ASLifecycleActionHeartbeat

Il seguente esempio di codice mostra come utilizzare Write-ASLifecycleActionHeartbeat

Strumenti per PowerShell V4

Esempio 1: questo esempio registra un heartbeat per l'azione del ciclo di vita specificata. In questo modo l'istanza rimane in sospeso fino al completamento dell'azione personalizzata.

```
Write-ASLifecycleActionHeartbeat -AutoScalingGroupName my-asg -LifecycleHookName  
myLifecycleHook -LifecycleActionToken bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

- Per i dettagli sull'API, vedere [RecordLifecycleActionHeartbeat](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-ASLifecycleHook

Il seguente esempio di codice mostra come utilizzare Write-ASLifecycleHook

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge l'hook del ciclo di vita specificato al gruppo Auto Scaling specificato.

```
Write-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName  
"myLifecycleHook" -LifecycleTransition "autoscaling:EC2_INSTANCE_LAUNCHING" -  
NotificationTargetARN "arn:aws:sns:us-west-2:123456789012:my-sns-topic" -RoleARN  
"arn:aws:iam::123456789012:role/my-iam-role"
```

- Per i dettagli sull'API, vedere [PutLifecycleHook](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-ASNotificationConfiguration

Il seguente esempio di codice mostra come utilizzare Write-ASNotificationConfiguration Strumenti per PowerShell V4

Esempio 1: questo esempio configura il gruppo Auto Scaling specificato per inviare una notifica all'argomento SNS specificato quando avvia le istanze. EC2

```
Write-ASNotificationConfiguration -AutoScalingGroupName my-asg -NotificationType  
"autoscaling:EC2_INSTANCE_LAUNCH" -TopicARN "arn:aws:sns:us-west-2:123456789012:my-  
topic"
```

Esempio 2: questo esempio configura il gruppo Auto Scaling specificato per inviare una notifica all'argomento SNS specificato quando avvia o termina le istanze. EC2

```
Write-ASNotificationConfiguration -AutoScalingGroupName my-asg -NotificationType  
@"autoscaling:EC2_INSTANCE_LAUNCH", "autoscaling:EC2_INSTANCE_TERMINATE") -  
TopicARN "arn:aws:sns:us-west-2:123456789012:my-topic"
```

- Per i dettagli sull'API, vedere [PutNotificationConfiguration](#) in Cmdlet Reference (V4).AWS Strumenti per PowerShell

Write-ASScalingPolicy

Il seguente esempio di codice mostra come utilizzare Write-ASScalingPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge la policy specificata al gruppo Auto Scaling specificato. Il tipo di regolazione specificato determina come interpretare il ScalingAdjustment parametro. Con

'ChangeInCapacity', un valore positivo aumenta la capacità del numero specificato di istanze e un valore negativo diminuisce la capacità del numero di istanze specificato.

```
Write-ASScalingPolicy -AutoScalingGroupName my-asg -AdjustmentType  
"ChangeInCapacity" -PolicyName "myScaleInPolicy" -ScalingAdjustment -1
```

Output:

```
arn:aws:autoscaling:us-west-2:123456789012:scalingPolicy:aa3836ab-5462-42c7-adab-  
e1d769fc24ef:autoScalingGroupName/my-asg  
:policyName/myScaleInPolicy
```

- Per i dettagli sull'API, vedere [PutScalingPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-ASScheduledUpdateGroupAction

Il seguente esempio di codice mostra come utilizzare Write-ASScheduledUpdateGroupAction

Strumenti per PowerShell V4

Esempio 1: questo esempio crea o aggiorna un'azione pianificata una tantum per modificare la capacità desiderata all'ora di inizio specificata.

```
Write-ASScheduledUpdateGroupAction -AutoScalingGroupName my-asg -ScheduledActionName  
"myScheduledAction" -StartTime "2015-12-01T00:00:00Z" -DesiredCapacity 10
```

- Per i dettagli sull'API, vedere [PutScheduledUpdateGroupAction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Budget AWS esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Budget AWS

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

New-BGTBudget

Il seguente esempio di codice mostra come utilizzare `New-BGTBudget`.

Strumenti per PowerShell V4

Esempio 1: crea un nuovo budget con i vincoli di budget e di tempo specificati e con notifiche e-mail.

```
$notification = @{
    NotificationType = "ACTUAL"
    ComparisonOperator = "GREATER_THAN"
    Threshold = 80
}

$addressObject = @{
    Address = @"user@domain.com"
    SubscriptionType = "EMAIL"
}

$subscriber = New-Object Amazon.Budgets.Model.NotificationWithSubscribers
$subscriber.Notification = $notification
$subscriber.Subscribers.Add($addressObject)

$startDate = [datetime]::new(2017,09,25)
$endDate = [datetime]::new(2017,10,25)

New-BGTBudget -Budget_BudgetName "Tester" -Budget_BudgetType COST -
CostTypes_IncludeTax $true -Budget_TimeUnit MONTHLY -BudgetLimit_Unit USD -
TimePeriod_Start $startDate -TimePeriod_End $endDate -AccountId 123456789012 -
BudgetLimit_Amount 200 -NotificationsWithSubscriber $subscriber
```

- Per i dettagli sull'API, vedere [CreateBudget](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

AWS Cloud9 esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con AWS Cloud9

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-C9EnvironmentData

Il seguente esempio di codice mostra come utilizzare `Get-C9EnvironmentData`.

Strumenti per PowerShell V4

Esempio 1: Questo esempio ottiene informazioni sugli ambienti di sviluppo AWS Cloud9 specificati.

```
Get-C9EnvironmentData -EnvironmentId
685f892f431b45c2b28cb69eadcdb0EX,1980b80e5f584920801c09086667f0EX
```

Output:

```
Arn          : arn:aws:cloud9:us-
east-1:123456789012:environment:685f892f431b45c2b28cb69eadcdb0EX
Description  : Created from CodeStar.
Id           : 685f892f431b45c2b28cb69eadcdb0EX
Lifecycle    : Amazon.Cloud9.Model.EnvironmentLifecycle
Name         : my-demo-ec2-env
OwnerArn     : arn:aws:iam::123456789012:user/MyDemoUser
Type         : ec2
```

```

Arn          : arn:aws:cloud9:us-
east-1:123456789012:environment:1980b80e5f584920801c09086667f0EX
Description  :
Id           : 1980b80e5f584920801c09086667f0EX
Lifecycle    : Amazon.Cloud9.Model.EnvironmentLifecycle
Name         : my-demo-ssh-env
OwnerArn     : arn:aws:iam::123456789012:user/MyDemoUser
Type         : ssh

```

Esempio 2: Questo esempio ottiene informazioni sullo stato del ciclo di vita dell'ambiente di sviluppo Cloud9 specificato AWS .

```
(Get-C9EnvironmentData -EnvironmentId 685f892f431b45c2b28cb69eadcdb0EX).Lifecycle
```

Output:

```

FailureResource Reason Status
-----
                        -----
                                CREATED

```

- Per i dettagli sull'API, vedere [DescribeEnvironments](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-C9EnvironmentList

Il seguente esempio di codice mostra come utilizzare. Get-C9EnvironmentList

Strumenti per PowerShell V4

Esempio 1: Questo esempio ottiene un elenco di identificatori dell'ambiente di AWS sviluppo Cloud9 disponibili.

```
Get-C9EnvironmentList
```

Output:

```

685f892f431b45c2b28cb69eadcdb0EX
1980b80e5f584920801c09086667f0EX

```

- Per i dettagli sull'API, vedere [ListEnvironments](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-C9EnvironmentMembershipList

Il seguente esempio di codice mostra come utilizzare. Get-C9EnvironmentMembershipList
Strumenti per PowerShell V4

Esempio 1: Questo esempio ottiene informazioni sui membri dell'ambiente di sviluppo AWS Cloud9 specificato.

```
Get-C9EnvironmentMembershipList -EnvironmentId ffd88420d4824eeeeaeaa8a04bfde8cEX
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaeaa8a04bfde8cEX
LastAccess    : 1/1/0001 12:00:00 AM
Permissions   : read-write
UserArn       : arn:aws:iam::123456789012:user/AnotherDemoUser
UserId       : AIDAJ3BA602FMJWCXHEX

EnvironmentId : ffd88420d4824eeeeaeaa8a04bfde8cEX
LastAccess    : 1/1/0001 12:00:00 AM
Permissions   : owner
UserArn       : arn:aws:iam::123456789012:user/MyDemoUser
UserId       : AIDAJ3LOROM0UXTBSU6EX
```

Esempio 2: Questo esempio ottiene informazioni sul proprietario dell'ambiente di sviluppo AWS Cloud9 specificato.

```
Get-C9EnvironmentMembershipList -EnvironmentId ffd88420d4824eeeeaeaa8a04bfde8cEX -
Permission owner
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaeaa8a04bfde8cEX
LastAccess    : 1/1/0001 12:00:00 AM
Permissions   : owner
UserArn       : arn:aws:iam::123456789012:user/MyDemoUser
```

```
UserId      : AIDAJ3LOROM0UXTBSU6EX
```

Esempio 3: Questo esempio ottiene informazioni sul membro dell'ambiente specificato per più ambienti di sviluppo AWS Cloud9.

```
Get-C9EnvironmentMembershipList -UserArn arn:aws:iam::123456789012:user/MyDemoUser
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaea8a04bfde8cEX
LastAccess    : 1/17/2018 7:48:14 PM
Permissions    : owner
UserArn       : arn:aws:iam::123456789012:user/MyDemoUser
UserId        : AIDAJ3LOROM0UXTBSU6EX

EnvironmentId : 1980b80e5f584920801c09086667f0EX
LastAccess    : 1/16/2018 11:21:24 PM
Permissions    : owner
UserArn       : arn:aws:iam::123456789012:user/MyDemoUser
UserId        : AIDAJ3LOROM0UXTBSU6EX
```

- Per i dettagli sull'API, vedere [DescribeEnvironmentMemberships](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-C9EnvironmentStatus

Il seguente esempio di codice mostra come utilizzare. Get-C9EnvironmentStatus

Strumenti per PowerShell V4

Esempio 1: Questo esempio ottiene informazioni sullo stato dell'ambiente di sviluppo AWS Cloud9 specificato.

```
Get-C9EnvironmentStatus -EnvironmentId 349c86d4579e4e7298d500ff57a6b2EX
```

Output:

```
Message                Status
-----                -
Environment is ready to use ready
```


- Per i dettagli sull'API, vedere [DescribeEnvironmentStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-C9EnvironmentEC2

Il seguente esempio di codice mostra come utilizzare. New-C9EnvironmentEC2

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un AWS ambiente di sviluppo Cloud9 con le impostazioni specificate, avvia un'istanza Amazon Elastic Compute Cloud (EC2Amazon) e quindi si connette dall'istanza all'ambiente.

```
New-C9EnvironmentEC2 -Name my-demo-env -AutomaticStopTimeMinutes 60 -Description  
"My demonstration development environment." -InstanceType t2.micro -OwnerArn  
arn:aws:iam::123456789012:user/MyDemoUser -SubnetId subnet-d43a46EX
```

Output:

```
ffd88420d4824eeeeaeaa8a04bfde8cEX
```

- Per i dettagli sull'API, consulta [CreateEnvironmentEc2](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-C9EnvironmentMembership

Il seguente esempio di codice mostra come utilizzare. New-C9EnvironmentMembership

Strumenti per PowerShell V4

Esempio 1: Questo esempio aggiunge il membro dell'ambiente specificato all'ambiente di sviluppo AWS Cloud9 specificato.

```
New-C9EnvironmentMembership -UserArn arn:aws:iam::123456789012:user/AnotherDemoUser  
-EnvironmentId ffd88420d4824eeeeaeaa8a04bfde8cEX -Permission read-write
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaeaa8a04bfde8cEX  
LastAccess    : 1/1/0001 12:00:00 AM
```

```
Permissions    : read-write
UserArn        : arn:aws:iam::123456789012:user/AnotherDemoUser
UserId         : AIDAJ3BA602FMJWCWXHEX
```

- Per i dettagli sull'API, vedere [CreateEnvironmentMembership](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-C9Environment

Il seguente esempio di codice mostra come utilizzare. Remove-C9Environment

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'ambiente di sviluppo AWS Cloud9 specificato. Se un' EC2 istanza Amazon è connessa all'ambiente, interrompe anche l'istanza.

```
Remove-C9Environment -EnvironmentId ffd88420d4824eeeeaeaa8a04bfde8cEX
```

- Per i dettagli sull'API, vedere [DeleteEnvironment](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-C9EnvironmentMembership

Il seguente esempio di codice mostra come utilizzare. Remove-C9EnvironmentMembership

Strumenti per PowerShell V4

Esempio 1: Questo esempio elimina il membro dell'ambiente specificato dall'ambiente di sviluppo AWS Cloud9 specificato.

```
Remove-C9EnvironmentMembership -UserArn arn:aws:iam::123456789012:user/
AnotherDemoUser -EnvironmentId ffd88420d4824eeeeaeaa8a04bfde8cEX
```

- Per i dettagli sull'API, vedere [DeleteEnvironmentMembership](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-C9Environment

Il seguente esempio di codice mostra come utilizzare. Update-C9Environment

Strumenti per PowerShell V4

Esempio 1: Questo esempio modifica le impostazioni specificate dell'ambiente di sviluppo AWS Cloud9 esistente specificato.

```
Update-C9Environment -EnvironmentId ffd88420d4824eeeeaea8a04bfde8cEX -Description  
"My changed demonstration development environment." -Name my-changed-demo-env
```

- Per i dettagli sull'API, vedere [UpdateEnvironment](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-C9EnvironmentMembership

Il seguente esempio di codice mostra come utilizzare. Update-C9EnvironmentMembership

Strumenti per PowerShell V4

Esempio 1: Questo esempio modifica le impostazioni del membro di ambiente esistente specificato per l'ambiente di sviluppo AWS Cloud9 specificato.

```
Update-C9EnvironmentMembership -UserArn arn:aws:iam::123456789012:user/  
AnotherDemoUser -EnvironmentId ffd88420d4824eeeeaea8a04bfde8cEX -Permission read-  
only
```

Output:

```
EnvironmentId : ffd88420d4824eeeeaea8a04bfde8cEX  
LastAccess    : 1/1/0001 12:00:00 AM  
Permissions   : read-only  
UserArn       : arn:aws:iam::123456789012:user/AnotherDemoUser  
UserId       : AIDAJ3BA602FMJWCXHEX
```

- Per i dettagli sull'API, vedere [UpdateEnvironmentMembership](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

CloudFormation esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. CloudFormation

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-CFNStack

Il seguente esempio di codice mostra come utilizzare `Get-CFNStack`.

Strumenti per PowerShell V4

Esempio 1: restituisce una raccolta di istanze di stack che descrivono tutti gli stack dell'utente.

```
Get-CFNStack
```

Esempio 2: restituisce un'istanza di stack che descrive lo stack specificato

```
Get-CFNStack -StackName "myStack"
```

- Per i dettagli sull'API, vedere [DescribeStacks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFNStackEvent

Il seguente esempio di codice mostra come utilizzare `Get-CFNStackEvent`

Strumenti per PowerShell V4

Esempio 1: restituisce tutti gli eventi correlati allo stack per lo stack specificato.

```
Get-CFNStackEvent -StackName "myStack"
```

- Per i dettagli sull'API, vedere [DescribeStackEvents](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFNStackResource

Il seguente esempio di codice mostra come utilizzare. Get-CFNStackResource

Strumenti per PowerShell V4

Esempio 1: restituisce la descrizione di una risorsa identificata nel modello associato allo stack specificato dall'ID logico «MyDBInstance».

```
Get-CFNStackResource -StackName "myStack" -LogicalResourceId "MyDBInstance"
```

- Per i dettagli sull'API, vedere [DescribeStackResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFNStackResourceList

Il seguente esempio di codice mostra come utilizzare. Get-CFNStackResourceList

Strumenti per PowerShell V4

Esempio 1: restituisce le descrizioni AWS delle risorse per un massimo di 100 risorse associate allo stack specificato. Per ottenere i dettagli di tutte le risorse associate a uno stack, usa Get-CFNStackResourceSummary, che supporta anche la paginazione manuale dei risultati.

```
Get-CFNStackResourceList -StackName "myStack"
```

Esempio 2: restituisce la descrizione dell' EC2 istanza Amazon identificata nel modello associato allo stack specificato dall'ID logico «Ec2Instance».

```
Get-CFNStackResourceList -StackName "myStack" -LogicalResourceId "Ec2Instance"
```

Esempio 3: restituisce la descrizione di un massimo di 100 risorse associate allo stack contenente un' EC2 istanza Amazon identificata dall'ID di istanza «i-123456». Per ottenere i dettagli di tutte le risorse associate a uno stack, usa Get- CFNStackResourceSummary, che supporta anche la paginazione manuale dei risultati.

```
Get-CFNStackResourceList -PhysicalResourceId "i-123456"
```

Esempio 4: restituisce la descrizione dell' EC2 istanza Amazon identificata dall'ID logico «Ec2Instance» nel modello per uno stack. Lo stack viene identificato utilizzando l'ID della risorsa fisica di una risorsa che contiene, in questo caso anche un' EC2 istanza Amazon con ID di istanza «i-123456». È inoltre possibile utilizzare una risorsa fisica diversa per identificare lo stack in base al contenuto del modello, ad esempio un bucket Amazon S3.

```
Get-CFNStackResourceList -PhysicalResourceId "i-123456" -LogicalResourceId  
"Ec2Instance"
```

- Per i dettagli sull'API, vedere [DescribeStackResources](#) in Cmdlet Reference (V4).AWS Strumenti per PowerShell

Get-CFNStackResourceSummary

Il seguente esempio di codice mostra come utilizzare. Get-CFNStackResourceSummary
Strumenti per PowerShell V4

Esempio 1: restituisce le descrizioni di tutte le risorse associate allo stack specificato.

```
Get-CFNStackResourceSummary -StackName "myStack"
```

- Per i dettagli sull'API, vedere [ListStackResources](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFNStackSummary

Il seguente esempio di codice mostra come utilizzare. Get-CFNStackSummary
Strumenti per PowerShell V4

Esempio 1: restituisce informazioni di riepilogo per tutti gli stack.

```
Get-CFNStackSummary
```

Esempio 2: restituisce informazioni di riepilogo per tutti gli stack attualmente in fase di creazione.

```
Get-CFNStackSummary -StackStatusFilter "CREATE_IN_PROGRESS"
```

Esempio 3: restituisce informazioni di riepilogo per tutti gli stack attualmente in fase di creazione o aggiornamento.

```
Get-CFNStackSummary -StackStatusFilter @("CREATE_IN_PROGRESS", "UPDATE_IN_PROGRESS")
```

- Per i dettagli sull'API, vedere [ListStacks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFNTemplate

Il seguente esempio di codice mostra come utilizzare. Get-CFNTemplate

Strumenti per PowerShell V4

Esempio 1: restituisce il modello associato allo stack specificato.

```
Get-CFNTemplate -StackName "myStack"
```

- Per i dettagli sull'API, vedere [GetTemplate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Measure-CFNTemplateCost

Il seguente esempio di codice mostra come utilizzare. Measure-CFNTemplateCost

Strumenti per PowerShell V4

Esempio 1: restituisce l'URL di un calcolatore mensile AWS semplice con una stringa di query che descrive le risorse necessarie per eseguire il modello. Il modello è ottenuto dall'URL Amazon S3 specificato e dal parametro di personalizzazione applicato. Il parametro può anche essere specificato utilizzando «Chiave» e «Valore» anziché «ParameterKey» e «ParameterValue».

```
Measure-CFNTemplateCost -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
                        -Region us-west-1 `
                        -Parameter @{ ParameterKey="KeyName";
ParameterValue="myKeyPairName" }
```

Esempio 2: restituisce l'URL di un calcolatore mensile AWS semplice con una stringa di query che descrive le risorse necessarie per eseguire il modello. Il modello viene analizzato in base al contenuto fornito e ai parametri di personalizzazione applicati (l'esempio presuppone che il contenuto del modello abbia dichiarato due parametri, " e 'KeyName'). InstanceType I parametri di personalizzazione possono anche essere specificati utilizzando 'Key' e 'Value' anziché " e 'ParameterKey'. ParameterValue

```
Measure-CFNTemplateCost -TemplateBody "{TEMPLATE CONTENT HERE}" `
    -Parameter @( @{ ParameterKey="KeyName";
    ParameterValue="myKeyPairName" }, `
    @{ ParameterKey="InstanceType";
    ParameterValue="m1.large" })
```

Esempio 3: utilizza New-Object per creare il set di parametri del modello e restituisce un URL di AWS Simple Monthly Calculator con una stringa di query che descrive le risorse necessarie per eseguire il modello. Il modello viene analizzato a partire dal contenuto fornito, con parametri di personalizzazione (questo esempio presuppone che il contenuto del modello abbia dichiarato due parametri, " e "). KeyName InstanceType

```
$p1 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p1.ParameterKey = "KeyName"
$p1.ParameterValue = "myKeyPairName"

$p2 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p2.ParameterKey = "InstanceType"
$p2.ParameterValue = "m1.large"

Measure-CFNTemplateCost -TemplateBody "{TEMPLATE CONTENT HERE}" -Parameter @( $p1,
    $p2 )
```

- Per i dettagli sull'API, vedere [EstimateTemplateCost](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CFNStack

Il seguente esempio di codice mostra come utilizzare. New-CFNStack

Strumenti per PowerShell V4

Esempio 1: crea un nuovo stack con il nome specificato. Il modello viene analizzato a partire dal contenuto fornito con parametri di personalizzazione (" e PK1 'PK2' rappresentano i nomi dei parametri dichiarati nel contenuto del modello, " e PV1 'PV2' rappresentano i valori di tali parametri. I parametri di personalizzazione possono anche essere specificati utilizzando «Key» e «Value» anziché «» e «ParameterKey». ParameterValue Se la creazione dello stack ha esito negativo, non verrà eseguito il rollback dello stack.

```
New-CFNStack -StackName "myStack" `
    -TemplateBody "{TEMPLATE CONTENT HERE}" `
    -Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
    @{ ParameterKey="PK2"; ParameterValue="PV2" }) `
    -DisableRollback $true
```

Esempio 2: crea un nuovo stack con il nome specificato. Il modello viene analizzato a partire dal contenuto fornito con parametri di personalizzazione (" e PK1 'PK2' rappresentano i nomi dei parametri dichiarati nel contenuto del modello, " e PV1 'PV2' rappresentano i valori di tali parametri. I parametri di personalizzazione possono anche essere specificati utilizzando «Key» e «Value» anziché «» e «ParameterKey». ParameterValue Se la creazione dello stack ha esito negativo, verrà eseguito il rollback dello stack.

```
$p1 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p1.ParameterKey = "PK1"
$p1.ParameterValue = "PV1"

$p2 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p2.ParameterKey = "PK2"
$p2.ParameterValue = "PV2"

New-CFNStack -StackName "myStack" `
    -TemplateBody "{TEMPLATE CONTENT HERE}" `
    -Parameter @( $p1, $p2 ) `
    -OnFailure "ROLLBACK"
```

Esempio 3: crea un nuovo stack con il nome specificato. Il modello è ottenuto dall'URL di Amazon S3 con parametri di personalizzazione ('PK1' rappresenta il nome di un parametro dichiarato nel contenuto del modello, 'PV1' rappresenta il valore del parametro. I parametri di personalizzazione possono essere specificati anche utilizzando «Key» e «Value» anziché «» e «ParameterKey».

ParameterValue Se la creazione dello stack fallisce, verrà ripristinato (come specificare - **DisableRollback \$false**).

```
New-CFNStack -StackName "myStack" `
              -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
              -Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

Esempio 4: crea un nuovo stack con il nome specificato. Il modello è ottenuto dall'URL di Amazon S3 con parametri di personalizzazione ('PK1' rappresenta il nome di un parametro dichiarato nel contenuto del modello, 'PV1' rappresenta il valore del parametro. I parametri di personalizzazione possono essere specificati anche utilizzando «Key» e «Value» anziché «» e «ParameterKey». **ParameterValue** Se la creazione dello stack fallisce, verrà ripristinato (come specificare - **DisableRollback \$false**). La notifica specificata AENs riceverà gli eventi pubblicati relativi allo stack.

```
New-CFNStack -StackName "myStack" `
              -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
              -Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" } `
              -NotificationARN @( "arn1", "arn2" )
```

- Per i dettagli sull'API, vedere [CreateStack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CFNStack

Il seguente esempio di codice mostra come utilizzare. **Remove-CFNStack**

Strumenti per PowerShell V4

Esempio 1: elimina lo stack specificato.

```
Remove-CFNStack -StackName "myStack"
```

- Per i dettagli sull'API, vedere [DeleteStack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Resume-CFNUpdateRollback

Il seguente esempio di codice mostra come utilizzare. Resume-CFNUpdateRollback

Strumenti per PowerShell V4

Esempio 1: continua il rollback dello stack denominato, il cui stato deve essere "UPDATE_ROLLBACK_FAILED". Se il rollback continuato ha esito positivo, lo stack assumerà lo stato "UPDATE_ROLLBACK_COMPLETE".

```
Resume-CFNUpdateRollback -StackName "myStack"
```

- Per i dettagli sull'API, vedere [ContinueUpdateRollback](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-CFNUpdateStack

Il seguente esempio di codice mostra come utilizzare. Stop-CFNUpdateStack

Strumenti per PowerShell V4

Esempio 1: annulla un aggiornamento nello stack specificato.

```
Stop-CFNUpdateStack -StackName "myStack"
```

- Per i dettagli sull'API, vedere [CancelUpdateStack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Test-CFNStack

Il seguente esempio di codice mostra come utilizzare. Test-CFNStack

Strumenti per PowerShell V4

Esempio 1: verifica se lo stack ha raggiunto uno degli stati seguenti: UPDATE_ROLLBACK_COMPLETE, CREATE_COMPLETE, ROLLBACK_COMPLETE o UPDATE_COMPLETE.

```
Test-CFNStack -StackName MyStack
```

Output:

```
False
```

Esempio 2: verifica se lo stack ha raggiunto lo stato UPDATE_COMPLETE o UPDATE_ROLLBACK_COMPLETE.

```
Test-CFNStack -StackName MyStack -Status UPDATE_COMPLETE,UPDATE_ROLLBACK_COMPLETE
```

Output:

```
True
```

- Per i dettagli sull'API, vedere [CFNStackTest-in AWS Strumenti per PowerShell](#) Cmdlet Reference (V4).

Test-CFNTemplate

Il seguente esempio di codice mostra come utilizzare. Test-CFNTemplate

Strumenti per PowerShell V4

Esempio 1: convalida il contenuto del modello specificato. L'output riporta in dettaglio le funzionalità, la descrizione e i parametri del modello.

```
Test-CFNTemplate -TemplateBody "{TEMPLATE CONTENT HERE}"
```

Esempio 2: convalida il modello specificato a cui si accede tramite un URL Amazon S3. L'output riporta in dettaglio le funzionalità, la descrizione e i parametri del modello.

```
Test-CFNTemplate -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/  
templatefile.template
```

- Per i dettagli sull'API, vedere [ValidateTemplatein](#) AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-CFNStack

Il seguente esempio di codice mostra come utilizzare. Update-CFNStack

Strumenti per PowerShell V4

Esempio 1: aggiorna lo stack 'MyStack' con il modello e i parametri di personalizzazione specificati. 'PK1' rappresenta il nome di un parametro dichiarato nel modello e 'PV1' ne rappresenta il valore. I parametri di personalizzazione possono essere specificati anche utilizzando «Key» e «Value» anziché «ParameterKey» e «ParameterValue».

```
Update-CFNStack -StackName "myStack" `
    -TemplateBody "{Template Content Here}" `
    -Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

Esempio 2: aggiorna lo stack 'MyStack' con il modello e i parametri di personalizzazione specificati. 'PK1' e 'PK2' rappresentano i nomi dei parametri dichiarati nel modello, " e PV1 'PV2' rappresentano i valori richiesti. I parametri di personalizzazione possono anche essere specificati utilizzando «Key» e «Value» anziché «ParameterKey» e «ParameterValue».

```
Update-CFNStack -StackName "myStack" `
    -TemplateBody "{Template Content Here}" `
    -Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
    @{ ParameterKey="PK2"; ParameterValue="PV2" } )
```

Esempio 3: aggiorna lo stack 'MyStack' con il modello e i parametri di personalizzazione specificati. 'PK1' rappresenta il nome di un parametro dichiarato nel modello e 'PV2' ne rappresenta il valore. I parametri di personalizzazione possono essere specificati anche utilizzando «Key» e «Value» anziché «ParameterKey» e «ParameterValue».

```
Update-CFNStack -StackName "myStack" -TemplateBody "{Template Content Here}" -
Parameters @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

Esempio 4: aggiorna lo stack 'MyStack' con il modello specificato, ottenuto da Amazon S3, e i parametri di personalizzazione. 'PK1' e 'PK2' rappresentano i nomi dei parametri dichiarati nel modello, mentre " e 'PV1PV2' rappresentano i valori richiesti. I parametri di personalizzazione possono anche essere specificati utilizzando «Key» e «Value» anziché «ParameterKey» e «ParameterValue».

```
Update-CFNStack -StackName "myStack" `
    -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
    templatefile.template `
```

```
-Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
@{ ParameterKey="PK2"; ParameterValue="PV2" } )
```

Esempio 5: aggiorna lo stack 'MyStack', che in questo esempio si presume contenga risorse IAM, con il modello specificato, ottenuto da Amazon S3, e i parametri di personalizzazione. 'PK1' e 'PK2' rappresentano i nomi dei parametri dichiarati nel modello, mentre " e 'PV1PV2' rappresentano i valori richiesti. I parametri di personalizzazione possono anche essere specificati utilizzando «Key» e «Value» anziché «ParameterKey» e «ParameterValue». Gli stack contenenti risorse IAM richiedono di specificare il parametro -Capabilities «CAPABILITY_IAM», altrimenti l'aggiornamento avrà esito negativo con un errore ". InsufficientCapabilities

```
Update-CFNStack -StackName "myStack" `
-TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
-Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
@{ ParameterKey="PK2"; ParameterValue="PV2" } ) `
-Capabilities "CAPABILITY_IAM"
```

- Per i dettagli sull'API, vedere [UpdateStack](#) in Cmdlet Reference (V4).AWS Strumenti per PowerShell

Wait-CFNStack

Il seguente esempio di codice mostra come utilizzare. Wait-CFNStack

Strumenti per PowerShell V4

Esempio 1: verifica se lo stack ha raggiunto uno degli stati seguenti: UPDATE_ROLLBACK_COMPLETE, CREATE_COMPLETE, ROLLBACK_COMPLETE o UPDATE_COMPLETE. Se lo stack non è in uno di questi stati, il comando rimane inattivo per due secondi prima di testare nuovamente lo stato. Questa operazione viene ripetuta fino a quando lo stack raggiunge uno degli stati richiesti o fino allo scadere del periodo di timeout predefinito di 60 secondi. Se il periodo di timeout viene superato, viene generata un'eccezione. Se lo stack raggiunge uno degli stati richiesti entro il periodo di timeout, viene restituito alla pipeline.

```
$stack = Wait-CFNStack -StackName MyStack
```

Esempio 2: questo esempio attende un totale di 5 minuti (300 secondi) affinché lo stack raggiunga uno degli stati specificati. In questo esempio lo stato viene raggiunto prima del timeout e quindi l'oggetto stack viene restituito alla pipeline.

```
Wait-CFNStack -StackName MyStack -Timeout 300 -Status
CREATE_COMPLETE,ROLLBACK_COMPLETE
```

Output:

```
Capabilities      : {CAPABILITY_IAM}
ChangeSetId       :
CreationTime      : 6/1/2017 9:29:33 AM
Description       : AWS CloudFormation Sample Template
ec2_instance_with_instance_profile: Create an EC2 instance with an associated
instance profile. **WARNING** This template creates one or more Amazon EC2
instances and an Amazon SQS queue. You will be billed for the
AWS resources used if you create a stack from this template.
DisableRollback   : False
LastUpdatedTime   : 1/1/0001 12:00:00 AM
NotificationARNs  : {}
Outputs           : {}
Parameters        : {}
RoleARN           :
StackId           : arn:aws:cloudformation:us-west-2:123456789012:stack/
MyStack/7ea87b50-46e7-11e7-9c9b-503a90a9c4d1
StackName         : MyStack
StackStatus       : CREATE_COMPLETE
StackStatusReason :
Tags              : {}
TimeoutInMinutes  : 0
```

Esempio 3: questo esempio mostra l'errore generato quando uno stack non raggiunge uno degli stati richiesti entro il periodo di timeout (in questo caso il periodo predefinito di 60 secondi).

```
Wait-CFNStack -StackName MyStack -Status CREATE_COMPLETE,ROLLBACK_COMPLETE
```

Output:

```
Wait-CFNStack : Timed out after 60 seconds waiting for CloudFormation
stack MyStack in region us-west-2 to reach one of state(s):
UPDATE_ROLLBACK_COMPLETE,CREATE_COMPLETE,ROLLBACK_COMPLETE,UPDATE_COMPLETE
```

```
At line:1 char:1
+ Wait-CFNStack -StackName MyStack -State CREATE_COMPLETE,ROLLBACK_COMPLETE
+ ~~~~~
    + CategoryInfo          : InvalidOperation:
      (Amazon.PowerShe...tCFNStackCmdlet:WaitCFNStackCmdlet) [Wait-CFNStack],
      InvalidOperationException
    + FullyQualifiedErrorId :
      InvalidOperationException,Amazon.PowerShell.Cmdlets.CFN.WaitCFNStackCmdlet
```

- Per i dettagli sull'API, vedere [Wait- CFNStack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

CloudFront esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. CloudFront

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-CFCloudFrontOriginAccessIdentity

Il seguente esempio di codice mostra come utilizzare `Get-CFCloudFrontOriginAccessIdentity`.

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce un'identità di accesso all' CloudFront origine di Amazon specifica, specificata dal parametro `-Id`. Sebbene il parametro `-Id` non sia obbligatorio, se non lo si specifica non viene restituito alcun risultato.


```
Get-CFCloudFrontOriginAccessIdentity -Id E3XXXXXXXXXXRT
```

Output:

```

      CloudFrontOriginAccessIdentityConfig      Id
      S3CanonicalUserId
      -----
      -----
      Amazon.CloudFront.Model.CloudFrontOr... E3XXXXXXXXXXRT
      4b6e...
```

- Per i dettagli sull'API, vedere [GetCloudFrontOriginAccessIdentity](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFCloudFrontOriginAccessIdentityConfig

Il seguente esempio di codice mostra come utilizzare. Get-CFCloudFrontOriginAccessIdentityConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce informazioni di configurazione su una singola identità di accesso di CloudFront origine Amazon, specificata dal parametro -Id. Si verificano errori se non viene specificato alcun parametro -Id.

```
Get-CFCloudFrontOriginAccessIdentityConfig -Id E3XXXXXXXXXXRT
```

Output:

```

      CallerReference                                Comment
      -----
      mycallerreference: 2/1/2011 1:16:32 PM          Caller reference:
      2/1/2011 1:16:32 PM
```

- Per i dettagli sull'API, vedere [GetCloudFrontOriginAccessIdentityConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFCloudFrontOriginAccessIdentityList

Il seguente esempio di codice mostra come utilizzare. Get-CFCloudFrontOriginAccessIdentityList

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce un elenco di identità di accesso di CloudFront origine di Amazon. Poiché il MaxItem parametro - specifica il valore 2, i risultati includono due identità.

```
Get-CFCloudFrontOriginAccessIdentityList -MaxItem 2
```

Output:

```
IsTruncated : True
Items       : {E326XXXXXXXXXT, E1YWXXXXXXXX9B}
Marker      :
MaxItems    : 2
NextMarker  : E1YXXXXXXXXXX9B
Quantity    : 2
```

- Per i dettagli sull'API, vedere [ListCloudFrontOriginAccessIdentities](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFDistribution

Il seguente esempio di codice mostra come utilizzare. Get-CFDistribution

Strumenti per PowerShell V4

Esempio 1: recupera le informazioni relative a una distribuzione specifica.

```
Get-CFDistribution -Id EXAMPLE0000ID
```

- Per i dettagli sull'API, vedere [GetDistribution](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFDistributionConfig

Il seguente esempio di codice mostra come utilizzare. Get-CFDistributionConfig

Strumenti per PowerShell V4

Esempio 1: recupera la configurazione di una distribuzione specifica.

```
Get-CFDistributionConfig -Id EXAMPLE0000ID
```

- Per i dettagli sull'API, vedere [GetDistributionConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFDistributionList

Il seguente esempio di codice mostra come utilizzare. Get-CFDistributionList

Strumenti per PowerShell V4

Esempio 1: restituisce le distribuzioni.

```
Get-CFDistributionList
```

- Per i dettagli sull'API, vedere [ListDistributions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CFDistribution

Il seguente esempio di codice mostra come utilizzare. New-CFDistribution

Strumenti per PowerShell V4

Esempio 1: crea una CloudFront distribuzione di base, configurata con registrazione e memorizzazione nella cache.

```
$origin = New-Object Amazon.CloudFront.Model.Origin
$origin.DomainName = "amzn-s3-demo-bucket.s3.amazonaws.com"
$origin.Id = "UniqueOrigin1"
$origin.S3OriginConfig = New-Object Amazon.CloudFront.Model.S3OriginConfig
$origin.S3OriginConfig.OriginAccessIdentity = ""
New-CFDistribution `
    -DistributionConfig_Enabled $true `
    -DistributionConfig_Comment "Test distribution" `
    -Origins_Item $origin `
```

```

-Origins_Quantity 1 `
-Logging_Enabled $true `
-Logging_IncludeCookie $true `
-Logging_Bucket amzn-s3-demo-logging-bucket.s3.amazonaws.com `
-Logging_Prefix "help/" `
-DistributionConfig_CallerReference Client1 `
-DistributionConfig_DefaultRootObject index.html `
-DefaultCacheBehavior_TargetOriginId $origin.Id `
-ForwardedValues_QueryString $true `
-Cookies_Forward all `
-WhitelistedNames_Quantity 0 `
-TrustedSigners_Enabled $false `
-TrustedSigners_Quantity 0 `
-DefaultCacheBehavior_ViewerProtocolPolicy allow-all `
-DefaultCacheBehavior_MinTTL 1000 `
-DistributionConfig_PriceClass "PriceClass_All" `
-CacheBehaviors_Quantity 0 `
-Aliases_Quantity 0

```

- Per i dettagli sull'API, vedere [CreateDistribution](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CFInvalidation

Il seguente esempio di codice mostra come utilizzare. New-CFInvalidation

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova invalidazione per una distribuzione con l'ID EXAMPLNSTXAXE. CallerReference è un ID univoco scelto dall'utente; in questo caso, viene utilizzato un timestamp che rappresenta il 15 maggio 2019 alle 9:00. La variabile \$Paths archivia tre percorsi di immagini e file multimediali che l'utente non desidera vengano inseriti nella cache distribuita. Il valore del parametro -Paths_Quantity è il numero totale di percorsi specificati nel parametro -Paths_Item.

```

$Paths = "/images/*.gif", "/images/image1.jpg", "/videos/*.mp4"
New-CFInvalidation -DistributionId "EXAMPLNSTXAXE" -
InvalidationBatch_CallerReference 20190515090000 -Paths_Item $Paths -Paths_Quantity
3

```

Output:

Invalidation	Location
-----	-----
Amazon.CloudFront.Model.Invalidation https://cloudfront.amazonaws.com/2018-11-05/distribution/EXAMPLENSTXAXE/invalidation/EXAMPLE8N0K9H	

- Per i dettagli sull'API, vedere [CreateInvalidation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CFSignedCookie

Il seguente esempio di codice mostra come utilizzare. New-CFSignedCookie

Strumenti per PowerShell V4

Esempio 1: crea un cookie firmato per la risorsa specificata utilizzando una policy di accesso predefinita. Il cookie sarà valido per un anno.

```
$params = @{
  "ResourceUri"="http://xyz.cloudfront.net/image1.jpeg"
  "KeyPairId"="AKIAIOSFODNN7EXAMPLE"
  "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"
  "ExpiresOn"=(Get-Date).AddYears(1)
}
New-CFSignedCookie @params
```

Output:

```
Expires
-----
[CloudFront-Expires, 1472227284]
```

Esempio 2: crea un cookie firmato per le risorse specificate utilizzando una policy personalizzata. Il cookie sarà valido tra 24 ore e scadrà una settimana dopo.

```
$start = (Get-Date).AddHours(24)
$params = @{
  "ResourceUri"="http://xyz.cloudfront.net/content/*.jpeg"
  "KeyPairId"="AKIAIOSFODNN7EXAMPLE"
  "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"
```

```
"ExpiresOn"=$start.AddDays(7)
    "ActiveFrom"=$start
}

New-CFSignedCookie @params
```

Output:

```
Policy
-----
[CloudFront-Policy, eyJTd...wIjo...
```

Esempio 3: crea un cookie firmato per le risorse specificate utilizzando una policy personalizzata. Il cookie sarà valido tra 24 ore e scadrà una settimana dopo. L'accesso alle risorse è limitato all'intervallo IP specificato.

```
$start = (Get-Date).AddHours(24)
$params = @{
    "ResourceUri"="http://xyz.cloudfront.net/content/*.jpeg"
    "KeyPairId"="AKIAIOSFODNN7EXAMPLE"
    "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"
    "ExpiresOn"=$start.AddDays(7)
    "ActiveFrom"=$start
    "IpRange"="192.0.2.0/24"
}

New-CFSignedCookie @params
```

Output:

```
Policy
-----
[CloudFront-Policy, eyJTd...wIjo...
```

- Per i dettagli sull'API, vedere [CFSignedNew-Cookie](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CFSignedUrl

Il seguente esempio di codice mostra come utilizzare. New-CFSignedUrl

Strumenti per PowerShell V4

Esempio 1: crea un URL firmato per la risorsa specificata utilizzando una policy di accesso predefinita. L'URL sarà valido per un'ora. Un oggetto System.Uri contenente l'URL firmato viene emesso nella pipeline.

```
$params = @{  
  "ResourceUri"="https://cdn.example.com/index.html"  
  "KeyPairId"="AKIAIOSFODNN7EXAMPLE"  
  "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"  
  "ExpiresOn"=(Get-Date).AddHours(1)  
}  
New-CFSignedUrl @params
```

Esempio 2: crea un URL firmato per la risorsa specificata utilizzando una policy personalizzata. L'URL sarà valido tra 24 ore e scadrà una settimana dopo.

```
$start = (Get-Date).AddHours(24)  
$params = @{  
  "ResourceUri"="https://cdn.example.com/index.html"  
  "KeyPairId"="AKIAIOSFODNN7EXAMPLE"  
  "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"  
  "ExpiresOn"=(Get-Date).AddDays(7)  
  "ActiveFrom"=$start  
}  
New-CFSignedUrl @params
```

Esempio 3: crea un URL firmato per la risorsa specificata utilizzando una policy personalizzata. L'URL sarà valido tra 24 ore e scadrà una settimana dopo. L'accesso alla risorsa è limitato all'intervallo IP specificato.

```
$start = (Get-Date).AddHours(24)  
$params = @{  
  "ResourceUri"="https://cdn.example.com/index.html"  
  "KeyPairId"="AKIAIOSFODNN7EXAMPLE"  
  "PrivateKeyFile"="C:\pk-AKIAIOSFODNN7EXAMPLE.pem"  
  "ExpiresOn"=(Get-Date).AddDays(7)  
  "ActiveFrom"=$start  
  "IpRange"="192.0.2.0/24"  
}  
New-CFSignedUrl @params
```

- Per i dettagli sull'API, vedere [New- CFSigned Url](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

CloudTrail esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. CloudTrail

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Find-CTEvent

Il seguente esempio di codice mostra come utilizzare Find-CTEvent.

Strumenti per PowerShell V4

Esempio 1: restituisce tutti gli eventi che si sono verificati negli ultimi sette giorni. Per impostazione predefinita, il cmdlet effettua automaticamente più chiamate per distribuire tutti gli eventi, uscendo dal processo quando il servizio indica che non sono disponibili altri dati.

```
Find-CTEvent
```

Esempio 2: restituisce tutti gli eventi che si sono verificati negli ultimi sette giorni specificando una Regione diversa dall'impostazione predefinita della shell corrente.

```
Find-CTEvent -Region eu-central-1
```

Esempio 3: restituisce tutti gli eventi associati alla chiamata RunInstances API.


```
Find-CTEvent -LookupAttribute @{ AttributeKey="EventName";  
    AttributeValue="RunInstances" }
```

Esempio 4: restituisce i primi 5 eventi disponibili.

```
Find-CTEvent -MaxResult 5
```

- Per i dettagli sull'API, vedere [LookupEvents](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CTTrail

Il seguente esempio di codice mostra come utilizzare. Get-CTTrail

Strumenti per PowerShell V4

Esempio 1: restituisce le impostazioni di tutti i trail associati alla Regione corrente per l'account in uso.

```
Get-CTTrail
```

Esempio 2: restituisce le impostazioni dei trail specificati.

```
Get-CTTrail -TrailNameList trail1, trail2
```

Esempio 3: restituisce le impostazioni dei trail specificati creati in una Regione diversa dall'attuale impostazione predefinita della shell (in questo caso la Regione Francoforte (eu-central)).

```
Get-CTTrail -TrailNameList trailABC, trailDEF -Region eu-central-1
```

- Per i dettagli sull'API, vedere [DescribeTrails](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CTTrailStatus

Il seguente esempio di codice mostra come utilizzare. Get-CTTrailStatus

Strumenti per PowerShell V4

Esempio 1: restituisce informazioni sullo stato del percorso con il nome 'myExampleTrail'. I dati restituiti includono informazioni su errori di distribuzione, errori di Amazon SNS e Amazon S3 e tempi di avvio e arresto della registrazione per il trail. Questo esempio presuppone che il trail sia stato creato nella stessa Regione della shell predefinita corrente.

```
Get-CTTrailStatus -Name myExampleTrail
```

Esempio 2: restituisce le impostazioni sullo stato di un trail creato in una Regione diversa dall'attuale impostazione predefinita della shell (in questo caso la Regione Francoforte (eu-central)).

```
Get-CTTrailStatus -Name myExampleTrail -Region eu-central-1
```

- Per i dettagli sull'API, vedere [GetTrailStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CTTrail

Il seguente esempio di codice mostra come utilizzare. New-CTTrail

Strumenti per PowerShell V4

Esempio 1: crea un trail che utilizzerà il bucket "amzn-s3-demo-bucket" per l'archiviazione dei file di log.

```
New-CTTrail -Name "awscloudtrail-example" -S3BucketName "amzn-s3-demo-bucket"
```

Esempio 2: crea un trail che utilizzerà il bucket "amzn-s3-demo-bucket" per l'archiviazione dei file di log. Gli oggetti S3 che rappresentano i log saranno tutti associati al prefisso di chiave comune "mylogs". Quando vengono distribuiti nuovi log al bucket, verrà inviata una notifica all'argomento SNS "mlog-deliverytopic". In questo esempio viene utilizzato lo splatting per fornire i valori dei parametri al cmdlet.

```
$params = @{  
    Name="awscloudtrail-example"  
    S3BucketName="amzn-s3-demo-bucket"  
    S3KeyPrefix="mylogs"  
    SnsTopicName="mlog-deliverytopic"
```

```
}  
New-CTTrail @params
```

- Per i dettagli sull'API, vedere [CreateTrail](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CTTrail

Il seguente esempio di codice mostra come utilizzare. Remove-CTTrail

Strumenti per PowerShell V4

Esempio 1: elimina il trail specificato. Prima di eseguire il comando, verrà richiesta una conferma. Per disattivare la conferma, aggiungi il parametro -Force.

```
Remove-CTTrail -Name "awscloudtrail-example"
```

- Per i dettagli sull'API, vedere [DeleteTrail](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-CTLogging

Il seguente esempio di codice mostra come utilizzare. Start-CTLogging

Strumenti per PowerShell V4

Esempio 1: avvia la registrazione delle chiamate AWS API e la consegna dei file di registro per il percorso denominato 'myExampleTrail'. Questo esempio presuppone che il trail sia stato creato nella stessa Regione della shell predefinita corrente.

```
Start-CTLogging -Name myExampleTrail
```

Esempio 2: avvia la registrazione delle chiamate AWS API e la consegna dei file di registro per un percorso creato in una regione diversa dall'attuale impostazione predefinita della shell (in questo caso, la regione di Francoforte (eu-central-1)).

```
Start-CTLogging -Name myExampleTrail -Region eu-central-1
```

- Per i dettagli sull'API, vedere [StartLogging](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-CTLogging

Il seguente esempio di codice mostra come utilizzare. Stop-CTLogging

Strumenti per PowerShell V4

Esempio 1: sospende la registrazione delle chiamate AWS API e la consegna dei file di registro per il percorso denominato ". myExampleTrail Questo esempio presuppone che il trail sia stato creato nella stessa Regione della shell predefinita corrente.

```
Stop-CTLogging -Name myExampleTrail
```

Esempio 2: sospende la registrazione delle chiamate AWS API e la consegna dei file di registro per un trail creato in una regione diversa dall'attuale impostazione predefinita della shell (in questo caso, la regione di Francoforte (eu-central-1)).

```
Stop-CTLogging -Name myExampleTrail -Region eu-central-1
```

- Per i dettagli sull'API, vedere [StopLogging](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-CTTrail

Il seguente esempio di codice mostra come utilizzare. Update-CTTrail

Strumenti per PowerShell V4

Esempio 1: aggiorna il trail specificato in modo che gli eventi del servizio globale, ad esempio quelli di IAM, vengano registrati e modifica il prefisso di chiave comune per i successivi file di log in "globallogs".

```
Update-CTTrail -Name "awscloudtrail-example" -IncludeGlobalServiceEvents $true -  
S3KeyPrefix "globallogs"
```

Esempio 2: aggiorna il trail specificato in modo che le notifiche relative alle nuove distribuzioni di log vengano inviate all'argomento SNS specificato.

```
Update-CTTrail -Name "awscloudtrail-example" -SnsTopicName "mlog-deliverytopic2"
```

Esempio 3: aggiorna il trail specificato in modo che i log vengano distribuiti a un bucket diverso.

```
Update-CTTrail -Name "awscloudtrail-example" -S3BucketName "otherlogs"
```

- Per i dettagli sull'API, vedere [UpdateTrail](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

CloudWatch esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con CloudWatch.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-CWAlarm

Il seguente esempio di codice mostra come utilizzare `Get-CWAlarm`.

Strumenti per PowerShell V4

Esempio 1: restituisce tutti gli allarmi, inclusi gli allarmi composti e metrici di CloudWatch

```
Get-CWAlarm -MaxRecords 1
```

Output:

CompositeAlarms	MetricAlarms	NextToken
-----	-----	-----
	{MetricAlarms-01}	NextToken-01
	{MetricAlarms-02}	NextToken-02

```
{MetricAlarms-03}    NextToken-03
```

Esempio 2: restituisce solo i dati compositi degli allarmi ottenuti CloudWatch dopo l'impostazione del parametro `- AlarmType CompositeAlarms`

```
Get-CWAlarm -AlarmType 'CompositeAlarms'
```

Output:

```
CompositeAlarms      MetricAlarms NextToken
-----
{CompositeAlarms-01}
{CompositeAlarms-02}
{CompositeAlarms-03}
```

- Per i dettagli sull'API, vedere [DescribeAlarms](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CWDashboard

Il seguente esempio di codice mostra come utilizzare `Get-CWDashboard`

Strumenti per PowerShell V4

Esempio 1: restituisce l'ARN del corpo del pannello di controllo specificato.

```
Get-CWDashboard -DashboardName Dashboard1
```

Output:

```
DashboardArn          DashboardBody
-----
arn:aws:cloudwatch::123456789012:dashboard/Dashboard1 {...
```

- Per i dettagli sull'API, vedere [GetDashboard](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CWDashboardList

Il seguente esempio di codice mostra come utilizzare `Get-CWDashboardList`

Strumenti per PowerShell V4

Esempio 1: restituisce la raccolta di pannelli di controllo per l'account in uso.

```
Get-CWDashboardList
```

Output:

DashboardArn	DashboardName	LastModified	Size
-----	-----	-----	----
arn:...	Dashboard1	7/6/2017 8:14:15 PM	252

Esempio 2: restituisce la raccolta di pannelli di controllo per l'account i cui nomi iniziano con il prefisso "dev".

```
Get-CWDashboardList -DashboardNamePrefix dev
```

- Per i dettagli sull'API, vedere [ListDashboards](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CWDashboard

Il seguente esempio di codice mostra come utilizzare. Remove-CWDashboard

Strumenti per PowerShell V4

Esempio 1: elimina il pannello di controllo specificato, richiedendo la conferma prima di procedere. Per ignorare la conferma, aggiungi il parametro -Force al comando.

```
Remove-CWDashboard -DashboardName Dashboard1
```

- Per i dettagli sull'API, vedere [DeleteDashboards](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-CWDashboard

Il seguente esempio di codice mostra come utilizzare. Write-CWDashboard

Strumenti per PowerShell V4

Esempio 1: crea o aggiorna il pannello di controllo denominato "Dashboard1" per includere due widget per le metriche affiancati.

```
$dashBody = @"
{
  "widgets":[
    {
      "type":"metric",
      "x":0,
      "y":0,
      "width":12,
      "height":6,
      "properties":{
        "metrics":[
          [
            "AWS/EC2",
            "CPUUtilization",
            "InstanceId",
            "i-012345"
          ]
        ],
        "period":300,
        "stat":"Average",
        "region":"us-east-1",
        "title":"EC2 Instance CPU"
      }
    },
    {
      "type":"metric",
      "x":12,
      "y":0,
      "width":12,
      "height":6,
      "properties":{
        "metrics":[
          [
            "AWS/S3",
            "BucketSizeBytes",
            "BucketName",
            "amzn-s3-demo-bucket"
          ]
        ]
      }
    }
  ],

```



```

        "period":86400,
        "stat":"Maximum",
        "region":"us-east-1",
        "title":"amzn-s3-demo-bucket bytes"
    }
}
]
}
"@

Write-CWDashboard -DashboardName Dashboard1 -DashboardBody $dashBody

```

Esempio 2: crea o aggiorna il pannello di controllo inserendo il contenuto che lo descrive nel cmdlet.

```

$dashBody = @"
{
...
}
"@

$dashBody | Write-CWDashboard -DashboardName Dashboard1

```

- Per i dettagli sull'API, vedere [PutDashboard](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-CWMetricData

Il seguente esempio di codice mostra come utilizzare Write-CWMetricData

Strumenti per PowerShell V4

Esempio 1: crea un nuovo MetricDatum oggetto e lo scrive su Amazon Web Services CloudWatch Metrics.

```

### Create a MetricDatum .NET object
$Metric = New-Object -TypeName Amazon.CloudWatch.Model.MetricDatum
$Metric.Timestamp = [DateTime]::UtcNow
$Metric.MetricName = 'CPU'
$Metric.Value = 50

### Write the metric data to the CloudWatch service

```

```
Write-CWMetricData -Namespace instance1 -MetricData $Metric
```

- Per i dettagli sull'API, consulta [PutMetricData AWS Strumenti per PowerShell](#) Cmdlet Reference (V4).

CodeCommit esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. CodeCommit

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-CCBranch

Il seguente esempio di codice mostra come utilizzare `Get-CCBranch`.

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sulla ramificazione specificata per il repository specificato.

```
Get-CCBranch -RepositoryName MyDemoRepo -BranchName MyNewBranch
```

Output:

BranchName	CommitId
-----	-----
MyNewBranch	7763222d...561fc9c9

- Per i dettagli sull'API, vedere [GetBranch](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CCBranchList

Il seguente esempio di codice mostra come utilizzare. Get-CCBranchList

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene un elenco di nomi di ramificazione per il repository specificato.

```
Get-CCBranchList -RepositoryName MyDemoRepo
```

Output:

```
master
MyNewBranch
```

- Per i dettagli sull'API, vedere [ListBranches](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CCRepository

Il seguente esempio di codice mostra come utilizzare. Get-CCRepository

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sul repository specificato.

```
Get-CCRepository -RepositoryName MyDemoRepo
```

Output:

```
AccountId      : 80398EXAMPLE
Arn            : arn:aws:codecommit:us-east-1:80398EXAMPLE:MyDemoRepo
CloneUrlHttp   : https://git-codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo
CloneUrlSsh    : ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo
CreationDate   : 9/8/2015 3:21:33 PM
DefaultBranch  :
```

```

LastModifiedDate      : 9/8/2015 3:21:33 PM
RepositoryDescription : This is a repository for demonstration purposes.
RepositoryId          : c7d0d2b0-ce40-4303-b4c3-38529EXAMPLE
RepositoryName        : MyDemoRepo

```

- Per i dettagli sull'API, vedere [GetRepository](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CCRepositoryBatch

Il seguente esempio di codice mostra come utilizzare. Get-CCRepositoryBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio conferma quali dei repository specificati sono stati trovati o meno.

```
Get-CCRepositoryBatch -RepositoryName MyDemoRepo, MyNewRepo, AMissingRepo
```

Output:

Repositories	RepositoriesNotFound
-----	-----
{MyDemoRepo, MyNewRepo}	{AMissingRepo}

- Per i dettagli sull'API, vedere [BatchGetRepositories](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CCRepositoryList

Il seguente esempio di codice mostra come utilizzare. Get-CCRepositoryList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutti i repository in ordine crescente per nome di repository.

```
Get-CCRepositoryList -Order Ascending -SortBy RepositoryName
```

Output:

RepositoryId	RepositoryName

```
-----
c7d0d2b0-ce40-4303-b4c3-38529EXAMPLE    MyDemoRepo
05f30c66-e3e3-4f91-a0cd-1c84aEXAMPLE    MyNewRepo
```

- Per i dettagli sull'API, vedere [ListRepositories](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CCBranch

Il seguente esempio di codice mostra come utilizzare. New-CCBranch

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova ramificazione con il nome specificato per il repository e l'ID commit specificati.

```
New-CCBranch -RepositoryName MyDemoRepo -BranchName MyNewBranch -CommitId
7763222d...561fc9c9
```

- Per i dettagli sull'API, vedere [CreateBranch](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CCRepository

Il seguente esempio di codice mostra come utilizzare. New-CCRepository

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo repository con il nome e la descrizione specificati.

```
New-CCRepository -RepositoryName MyDemoRepo -RepositoryDescription "This is a
repository for demonstration purposes."
```

Output:

```
AccountId           : 80398EXAMPLE
Arn                 : arn:aws:codecommit:us-east-1:80398EXAMPLE:MyDemoRepo
CloneUrlHttp       : https://git-codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo
CloneUrlSsh        : ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo
```

```
CreationDate      : 9/18/2015 4:13:25 PM
DefaultBranch     :
LastModifiedDate  : 9/18/2015 4:13:25 PM
RepositoryDescription : This is a repository for demonstration purposes.
RepositoryId      : 43ef2443-3372-4b12-9e78-65c27EXAMPLE
RepositoryName    : MyDemoRepo
```

- Per i dettagli sull'API, vedere [CreateRepository](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CCRepository

Il seguente esempio di codice mostra come utilizzare. Remove-CCRepository

Strumenti per PowerShell V4

Esempio 1: questo esempio forza l'eliminazione del repository specificato. Il comando richiederà una conferma prima di procedere. Aggiungi il parametro -Force per eliminare il repository senza una richiesta di conferma.

```
Remove-CCRepository -RepositoryName MyDemoRepo
```

Output:

```
43ef2443-3372-4b12-9e78-65c27EXAMPLE
```

- Per i dettagli sull'API, vedere [DeleteRepository](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-CCDefaultBranch

Il seguente esempio di codice mostra come utilizzare. Update-CCDefaultBranch

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica la ramificazione predefinita per il repository specificato nella ramificazione specificata.

```
Update-CCDefaultBranch -RepositoryName MyDemoRepo -DefaultBranchName MyNewBranch
```

- Per i dettagli sull'API, vedere [UpdateDefaultBranch](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-CCRepositoryDescription

Il seguente esempio di codice mostra come utilizzare. Update-CCRepositoryDescription
Strumenti per PowerShell V4

Esempio 1: questo esempio modifica la descrizione del repository specificato.

```
Update-CCRepositoryDescription -RepositoryName MyDemoRepo -RepositoryDescription  
"This is an updated description."
```

- Per i dettagli sull'API, vedere [UpdateRepositoryDescription](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-CCRepositoryName

Il seguente esempio di codice mostra come utilizzare. Update-CCRepositoryName
Strumenti per PowerShell V4

Esempio 1: questo esempio modifica il nome del repository specificato.

```
Update-CCRepositoryName -NewName MyDemoRepo2 -OldName MyDemoRepo
```

- Per i dettagli sull'API, vedere [UpdateRepositoryName](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

CodeDeploy esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. CodeDeploy

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-CDOnPremiseInstanceTag

Il seguente esempio di codice mostra come utilizzare `Add-CDOnPremiseInstanceTag`.

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un'istanza on-premises con la chiave e il valore specificati per l'istanza on-premises specificata.

```
Add-CDOnPremiseInstanceTag -InstanceName AssetTag12010298EX -Tag @{"Key" = "Name";
"Value" = "CodeDeployDemo-OnPrem"}
```

- Per i dettagli sull'API, vedere [AddTagsToOnPremisesInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDApplication

Il seguente esempio di codice mostra come utilizzare `Get-CDApplication`.

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sull'applicazione specificata.

```
Get-CDApplication -ApplicationName CodeDeployDemoApplication
```

Output:

ApplicationId	ApplicationName	CreateTime
LinkedToGitHub		
-----	-----	-----

```
e07fb938-091e-4f2f-8963-4d3e8EXAMPLE    CodeDeployDemoApplication    7/20/2015
9:49:48 PM    False
```

- Per i dettagli sull'API, vedere [GetApplication](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDApplicationBatch

Il seguente esempio di codice mostra come utilizzare. Get-CDApplicationBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sulle applicazioni specificate.

```
Get-CDApplicationBatch -ApplicationName CodeDeployDemoApplication,
CodePipelineDemoApplication
```

Output:

ApplicationId LinkedToGitHub	ApplicationName	CreateTime
-----	-----	-----
e07fb938-091e-4f2f-8963-4d3e8EXAMPLE 9:49:48 PM False	CodeDeployDemoApplication	7/20/2015
1ecfd602-62f1-4038-8f0d-06688EXAMPLE 5:53:26 PM False	CodePipelineDemoApplication	8/13/2015

- Per i dettagli sull'API, vedere [BatchGetApplications](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDApplicationList

Il seguente esempio di codice mostra come utilizzare. Get-CDApplicationList

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene un elenco di applicazioni disponibili.

```
Get-CDApplicationList
```

Output:

```
CodeDeployDemoApplication
CodePipelineDemoApplication
```

- Per i dettagli sull'API, vedere [ListApplications](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDApplicationRevision

Il seguente esempio di codice mostra come utilizzare. Get-CDApplicationRevision

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sulla revisione dell'applicazione specificata.

```
$revision = Get-CDApplicationRevision -ApplicationName CodeDeployDemoApplication
-S3Location_Bucket amzn-s3-demo-bucket -Revision_RevisionType S3 -
S3Location_Key 5xd27EX.zip -S3Location_BundleType zip -S3Location_ETag
4565c1ac97187f190c1a90265EXAMPLE
Write-Output ("Description = " + $revision.RevisionInfo.Description + ",
RegisterTime = " + $revision.RevisionInfo.RegisterTime)
```

Output:

```
Description = Application revision registered by Deployment ID: d-CX9CHN3EX,
RegisterTime = 07/20/2015 23:46:42
```

- Per i dettagli sull'API, vedere [GetApplicationRevision](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDApplicationRevisionList

Il seguente esempio di codice mostra come utilizzare. Get-CDApplicationRevisionList

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sulle revisioni disponibili per l'applicazione specificata.

```

ForEach ($revision in (Get-CDApplicationRevisionList -ApplicationName
    CodeDeployDemoApplication -Deployed Ignore)) {
>>   If ($revision.RevisionType -Eq "S3") {
>>     Write-Output ("Type = S3, Bucket = " + $revision.S3Location.Bucket
+ ", BundleType = " + $revision.S3Location.BundleType + ", ETag = " +
    $revision.S3Location.ETag + ", Key = " + $revision.S3Location.Key)
>>   }
>>   If ($revision.RevisionType -Eq "GitHub") {
>>     Write-Output ("Type = GitHub, CommitId = " +
    $revision.GitHubLocation.CommitId + ", Repository = " +
    $revision.GitHubLocation.Repository)
>>   }
>> }
>> }
>> }

```

Output:

```

Type = S3, Bucket = amzn-s3-demo-bucket, BundleType = zip, ETag =
4565c1ac97187f190c1a90265EXAMPLE, Key = 5xd27EX.zip
Type = GitHub, CommitId = f48933c3...76405362, Repository = MyGitHubUser/
CodeDeployDemoRepo

```

- Per i dettagli sull'API, vedere [ListApplicationRevisions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDDeployment

Il seguente esempio di codice mostra come utilizzare. Get-CDDeployment

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni di riepilogo sull'implementazione specificata.

```
Get-CDDeployment -DeploymentId d-QZMRGSTEX
```

Output:

```

ApplicationName      : CodeDeployDemoApplication
CompleteTime         : 7/23/2015 11:26:04 PM
CreateTime           : 7/23/2015 11:24:43 PM

```

```

Creator                : user
DeploymentConfigName    : CodeDeployDefault.OneAtATime
DeploymentGroupName     : CodeDeployDemoFleet
DeploymentId            : d-QZMRGSTEX
DeploymentOverview      : Amazon.CodeDeploy.Model.DeploymentOverview
Description            :
ErrorInformation        :
IgnoreApplicationStopFailures : False
Revision               : Amazon.CodeDeploy.Model.RevisionLocation
StartTime              : 1/1/0001 12:00:00 AM
Status                 : Succeeded

```

Esempio 2: questo esempio ottiene informazioni sullo stato delle istanze che partecipano all'implementazione specificata.

```
(Get-CDDeployment -DeploymentId d-QZMRGSTEX).DeploymentOverview
```

Output:

```

Failed      : 0
InProgress  : 0
Pending     : 0
Skipped     : 0
Succeeded   : 3

```

Esempio 3: questo esempio ottiene informazioni sulla revisione dell'applicazione per l'implementazione specificata.

```
(Get-CDDeployment -DeploymentId d-QZMRGSTEX).Revision.S3Location
```

Output:

```

Bucket      : amzn-s3-demo-bucket
BundleType  : zip
ETag        : cfbb81b304ee5e27efc21adaed3EXAMPLE
Key         : clzfqEX
Version     :

```

- Per i dettagli sull'API, vedere [GetDeployment](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDDeploymentBatch

Il seguente esempio di codice mostra come utilizzare. Get-CDDeploymentBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sulle implementazioni specificate.

```
Get-CDDeploymentBatch -DeploymentId d-QZMRGSTEX, d-RR0T5KTEX
```

Output:

```

ApplicationName      : CodeDeployDemoApplication
CompleteTime         : 7/23/2015 11:26:04 PM
CreateTime           : 7/23/2015 11:24:43 PM
Creator              : user
DeploymentConfigName  : CodeDeployDefault.OneAtATime
DeploymentGroupName   : CodeDeployDemoFleet
DeploymentId          : d-QZMRGSTEX
DeploymentOverview    : Amazon.CodeDeploy.Model.DeploymentOverview
Description           :
ErrorInformation      :
IgnoreApplicationStopFailures : False
Revision             : Amazon.CodeDeploy.Model.RevisionLocation
StartTime            : 1/1/0001 12:00:00 AM
Status               : Succeeded

ApplicationName      : CodePipelineDemoApplication
CompleteTime         : 7/23/2015 6:07:30 PM
CreateTime           : 7/23/2015 6:06:29 PM
Creator              : user
DeploymentConfigName  : CodeDeployDefault.OneAtATime
DeploymentGroupName   : CodePipelineDemoFleet
DeploymentId          : d-RR0T5KTEX
DeploymentOverview    : Amazon.CodeDeploy.Model.DeploymentOverview
Description           :
ErrorInformation      :
IgnoreApplicationStopFailures : False
Revision             : Amazon.CodeDeploy.Model.RevisionLocation
StartTime            : 1/1/0001 12:00:00 AM
Status               : Succeeded

```

- Per i dettagli sull'API, vedere [BatchGetDeployments](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDDeploymentConfig

Il seguente esempio di codice mostra come utilizzare. Get-CDDeploymentConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni di riepilogo sulla configurazione di implementazione specificata.

```
Get-CDDeploymentConfig -DeploymentConfigName ThreeQuartersHealthy
```

Output:

CreateTime	DeploymentConfigId	DeploymentConfigName
MinimumHealthyHosts		
-----	-----	-----

10/3/2014 4:32:30 PM	518a3950-d034-46a1-9d2c-3c949EXAMPLE	ThreeQuartersHealthy
Amazon.CodeDeploy.Model.MinimumHealthyHosts		

Esempio 2: questo esempio ottiene informazioni sulla definizione della configurazione di implementazione specificata.

```
Write-Output ((Get-CDDeploymentConfig -DeploymentConfigName  
ThreeQuartersHealthy).MinimumHealthyHosts)
```

Output:

Type	Value
----	-----
FLEET_PERCENT	75

- Per i dettagli sull'API, vedere [GetDeploymentConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDDeploymentConfigList

Il seguente esempio di codice mostra come utilizzare. Get-CDDeploymentConfigList

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene un elenco di configurazioni di implementazione disponibili.

```
Get-CDDeploymentConfigList
```

Output:

```
ThreeQuartersHealthy
CodeDeployDefault.OneAtATime
CodeDeployDefault.AllAtOnce
CodeDeployDefault.HalfAtATime
```

- Per i dettagli sull'API, vedere [ListDeploymentConfigs](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDDeploymentGroup

Il seguente esempio di codice mostra come utilizzare. Get-CDDeploymentGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sul gruppo di distribuzione specificato.

```
Get-CDDeploymentGroup -ApplicationName CodeDeployDemoApplication -
DeploymentGroupName CodeDeployDemoFleet
```

Output:

```
ApplicationName           : CodeDeployDemoApplication
AutoScalingGroups         : {}
DeploymentConfigName       : CodeDeployDefault.OneAtATime
DeploymentGroupId          : 7d7c098a-b444-4b27-96ef-22791EXAMPLE
DeploymentGroupName        : CodeDeployDemoFleet
Ec2TagFilters              : {Name}
OnPremisesInstanceTagFilters : {}
ServiceRoleArn             : arn:aws:iam::80398EXAMPLE:role/
CodeDeploySampleStack-4ph6EX-CodeDeployTrustRole-09MWP7XTL8EX
```

```
TargetRevision : Amazon.CodeDeploy.Model.RevisionLocation
```

- Per i dettagli sull'API, vedere [GetDeploymentGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDDeploymentGroupList

Il seguente esempio di codice mostra come utilizzare. Get-CDDeploymentGroupList

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene un elenco di gruppi di distribuzione per l'applicazione specificata.

```
Get-CDDeploymentGroupList -ApplicationName CodeDeployDemoApplication
```

Output:

ApplicationName	DeploymentGroups
NextToken	
-----	-----

CodeDeployDemoApplication	{CodeDeployDemoFleet, CodeDeployProductionFleet}

- Per i dettagli sull'API, vedere [ListDeploymentGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDDeploymentInstance

Il seguente esempio di codice mostra come utilizzare. Get-CDDeploymentInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sull'istanza specificata dell'implementazione specificata.

```
Get-CDDeploymentInstance -DeploymentId d-QZMRGSTEX -InstanceId i-254e22EX
```

Output:


```
DeploymentId      : d-QZMRGSTEX
InstanceId       : arn:aws:ec2:us-east-1:80398EXAMPLE:instance/i-254e22EX
LastUpdatedAt    : 7/23/2015 11:25:24 PM
LifecycleEvents  : {ApplicationStop, DownloadBundle, BeforeInstall, Install...}
Status          : Succeeded
```

- Per i dettagli sull'API, vedere [GetDeploymentInstance](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDDeploymentInstanceList

Il seguente esempio di codice mostra come utilizzare. Get-CDDeploymentInstanceList

Strumenti per PowerShell V4

Esempio 1: Questo esempio ottiene un elenco di istanze IDs per la distribuzione specificata.

```
Get-CDDeploymentInstanceList -DeploymentId d-QZMRGSTEX
```

Output:

```
i-254e22EX
i-274e22EX
i-3b4e22EX
```

- Per i dettagli sull'API, vedere [ListDeploymentInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDDeploymentList

Il seguente esempio di codice mostra come utilizzare. Get-CDDeploymentList

Strumenti per PowerShell V4

Esempio 1: Questo esempio ottiene un elenco di distribuzioni IDs per l'applicazione e il gruppo di distribuzione specificati.

```
Get-CDDeploymentList -ApplicationName CodeDeployDemoApplication -DeploymentGroupName
CodeDeployDemoFleet
```

Output:

```
d-QZMRGSTEX  
d-RR0T5KTEX
```

- Per i dettagli sull'API, vedere [ListDeployments](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDOnPremiseInstance

Il seguente esempio di codice mostra come utilizzare. Get-CDOnPremiseInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sull'istanza on-premises specificata.

```
Get-CDOnPremiseInstance -InstanceName AssetTag12010298EX
```

Output:

```
DeregisterTime : 1/1/0001 12:00:00 AM  
IamUserArn      : arn:aws:iam::80398EXAMPLE:user/CodeDeployDemoUser  
InstanceArn     : arn:aws:codedeploy:us-east-1:80398EXAMPLE:instance/  
AssetTag12010298EX_rDH556dxEX  
InstanceName    : AssetTag12010298EX  
RegisterTime    : 4/3/2015 6:36:24 PM  
Tags            : {Name}
```

- Per i dettagli sull'API, vedere [GetOnPremisesInstance](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDOnPremiseInstanceBatch

Il seguente esempio di codice mostra come utilizzare. Get-CDOnPremiseInstanceBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni sulle istanze on-premises specificate.

```
Get-CDOnPremiseInstanceBatch -InstanceName AssetTag12010298EX, AssetTag12010298EX-2
```

Output:

```
DeregisterTime : 1/1/0001 12:00:00 AM
IamUserArn      : arn:aws:iam::80398EXAMPLE:user/CodeDeployFRWUser
InstanceArn     : arn:aws:codedeploy:us-east-1:80398EXAMPLE:instance/
AssetTag12010298EX-2_XmeSz18rEX
InstanceName    : AssetTag12010298EX-2
RegisterTime    : 4/3/2015 6:38:52 PM
Tags            : {Name}

DeregisterTime : 1/1/0001 12:00:00 AM
IamUserArn      : arn:aws:iam::80398EXAMPLE:user/CodeDeployDemoUser
InstanceArn     : arn:aws:codedeploy:us-east-1:80398EXAMPLE:instance/
AssetTag12010298EX_rDH556dxEX
InstanceName    : AssetTag12010298EX
RegisterTime    : 4/3/2015 6:36:24 PM
Tags            : {Name}
```

- Per i dettagli sull'API, vedere [BatchGetOnPremisesInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CDOnPremiseInstanceList

Il seguente esempio di codice mostra come utilizzare `Get-CDOnPremiseInstanceList`

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene un elenco di nomi di istanza on-premises disponibili.

```
Get-CDOnPremiseInstanceList
```

Output:

```
AssetTag12010298EX
AssetTag12010298EX-2
```

- Per i dettagli sull'API, vedere [ListOnPremisesInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CDApplication

Il seguente esempio di codice mostra come utilizzare. New-CDApplication

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova applicazione con il nome specificato.

```
New-CDApplication -ApplicationName MyNewApplication
```

Output:

```
f19e4b61-2231-4328-b0fd-e57f5EXAMPLE
```

- Per i dettagli sull'API, vedere [CreateApplication](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CDDeployment

Il seguente esempio di codice mostra come utilizzare. New-CDDeployment

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova implementazione per l'applicazione e il gruppo di distribuzione specificati con la configurazione di implementazione e la revisione dell'applicazione specificate.

```
New-CDDeployment -ApplicationName MyNewApplication -S3Location_Bucket amzn-s3-demo-bucket -S3Location_BundleType zip -DeploymentConfigName CodeDeployDefault.OneAtATime -DeploymentGroupName MyNewDeploymentGroup -IgnoreApplicationStopFailures $True -S3Location_Key aws-codedeploy_linux-master.zip -RevisionType S3
```

Output:

```
d-ZHR0G7UEX
```

Esempio 2: Questo esempio mostra come specificare i gruppi di tag di EC2 istanza in base ai quali un'istanza deve essere identificata per poter essere inclusa nell'ambiente sostitutivo di una blue/green distribuzione.

```
New-CDDeployment -ApplicationName MyNewApplication -S3Location_Bucket amzn-s3-demo-bucket -S3Location_BundleType zip -DeploymentConfigName CodeDeployDefault.OneAtATime -DeploymentGroupName MyNewDeploymentGroup -IgnoreApplicationStopFailures $True -S3Location_Key aws-codedeploy_linux-master.zip -RevisionType S3 -Ec2TagSetList @(@{Key="key1";Type="KEY_ONLY"},@{Key="Key2";Type="KEY_AND_VALUE";Value="Value2"}),@(@{Key="
```

Output:

```
d-ZHROG7UEX
```

- Per i dettagli sull'API, vedere [CreateDeployment](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CDDeploymentConfig

Il seguente esempio di codice mostra come utilizzare. New-CDDeploymentConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova configurazione di implementazione con il nome e il comportamento specificati.

```
New-CDDeploymentConfig -DeploymentConfigName AtLeastTwoHealthyHosts -MinimumHealthyHosts_Type HOST_COUNT -MinimumHealthyHosts_Value 2
```

Output:

```
0f3e8187-44ef-42da-aeeed-b6823EXAMPLE
```

- Per i dettagli sull'API, vedere [CreateDeploymentConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CDDeploymentGroup

Il seguente esempio di codice mostra come utilizzare. New-CDDeploymentGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un gruppo di distribuzione con il nome, il gruppo Auto Scaling, la configurazione di implementazione, i tag e il ruolo di servizio specificati, per l'applicazione specificata.

```
New-CDDeploymentGroup -ApplicationName MyNewApplication -AutoScalingGroup
CodeDeployDemo-ASG -DeploymentConfigName CodeDeployDefault.OneAtATime
-DeploymentGroupName MyNewDeploymentGroup -Ec2TagFilter @{Key="Name";
Type="KEY_AND_VALUE"; Value="CodeDeployDemo"} -ServiceRoleArn
arn:aws:iam::80398EXAMPLE:role/CodeDeployDemo
```

Output:

```
16bbf199-95fd-40fc-a909-0bbcfEXAMPLE
```

Esempio 2: Questo esempio mostra come specificare i gruppi di tag di EC2 istanza in base ai quali un'istanza deve essere identificata per poter essere inclusa nell'ambiente sostitutivo di una blue/green distribuzione.

```
New-CDDeploymentGroup -ApplicationName MyNewApplication -AutoScalingGroup
CodeDeployDemo-ASG -DeploymentConfigName CodeDeployDefault.OneAtATime
-DeploymentGroupName MyNewDeploymentGroup -Ec2TagFilter @{Key="Name";
Type="KEY_AND_VALUE"; Value="CodeDeployDemo"} -ServiceRoleArn
arn:aws:iam::80398EXAMPLE:role/CodeDeployDemo -Ec2TagSetList
@(@{Key="key1";Type="KEY_ONLY"},@{Key="Key2";Type="KEY_AND_VALUE";Value="Value2"}),@(@{Key=
```

Output:

```
16bbf199-95fd-40fc-a909-0bbcfEXAMPLE
```

- Per i dettagli sull'API, vedere [CreateDeploymentGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-CDApplicationRevision

Il seguente esempio di codice mostra come utilizzare. Register-CDApplicationRevision

Strumenti per PowerShell V4

Esempio 1: questo esempio registra una revisione dell'applicazione con la posizione Amazon S3 specificata, per l'applicazione specificata.

```
Register-CDApplicationRevision -ApplicationName MyNewApplication -S3Location_Bucket  
amzn-s3-demo-bucket -S3Location_BundleType zip -S3Location_Key aws-  
codedeploy_linux-master.zip -Revision_RevisionType S3
```

- Per i dettagli sull'API, vedere [RegisterApplicationRevision](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-CDOnPremiseInstance

Il seguente esempio di codice mostra come utilizzare. Register-CDOnPremiseInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio registra un'istanza on-premises con il nome e l'utente IAM specificati.

```
Register-CDOnPremiseInstance -IamUserArn arn:aws:iam::80398EXAMPLE:user/  
CodeDeployDemoUser -InstanceName AssetTag12010298EX
```

- Per i dettagli sull'API, vedere [RegisterOnPremisesInstance](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CDApplication

Il seguente esempio di codice mostra come utilizzare. Remove-CDApplication

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'applicazione con il nome specificato. Il comando richiederà una conferma prima di procedere. Aggiungi il parametro -Force per eliminare l'applicazione senza una richiesta di conferma.

```
Remove-CDApplication -ApplicationName MyNewApplication
```

- Per i dettagli sull'API, vedere [DeleteApplication](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CDDeploymentConfig

Il seguente esempio di codice mostra come utilizzare. Remove-CDDeploymentConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la configurazione di implementazione con il nome specificato. Il comando richiederà una conferma prima di procedere. Aggiungi il parametro -Force per eliminare la configurazione di implementazione senza una richiesta di conferma.

```
Remove-CDDeploymentConfig -DeploymentConfigName AtLeastTwoHealthyHosts
```

- Per i dettagli sull'API, vedere [DeleteDeploymentConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CDDeploymentGroup

Il seguente esempio di codice mostra come utilizzare. Remove-CDDeploymentGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il gruppo di distribuzione con il nome specificato per l'applicazione specificata. Il comando richiederà una conferma prima di procedere. Aggiungi il parametro -Force per eliminare il gruppo di distribuzione senza una richiesta di conferma.

```
Remove-CDDeploymentGroup -ApplicationName MyNewApplication -DeploymentGroupName  
MyNewDeploymentGroup
```

- Per i dettagli sull'API, vedere [DeleteDeploymentGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CDOnPremiseInstanceTag

Il seguente esempio di codice mostra come utilizzare. Remove-CDOnPremiseInstanceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il tag specificato per l'istanza on-premises con il nome specificato. Il comando richiederà una conferma prima di procedere. Aggiungi il parametro -Force per eliminare il tag senza una richiesta di conferma.


```
Remove-CDOnPremiseInstanceTag -InstanceName AssetTag12010298EX -Tag @{"Key" =  
"Name"; "Value" = "CodeDeployDemo-OnPrem"}
```

- Per i dettagli sull'API, vedere [RemoveTagsFromOnPremisesInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-CDDeployment

Il seguente esempio di codice mostra come utilizzare. Stop-CDDeployment

Strumenti per PowerShell V4

Esempio 1: questo esempio tenta di arrestare l'implementazione con l'ID specificato.

```
Stop-CDDeployment -DeploymentId d-LJQNREYEX
```

Output:

Status	StatusMessage
-----	-----
Pending	Stopping Pending. Stopping to schedule commands in the deployment instances

- Per i dettagli sull'API, vedere [StopDeployment](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-CDOnPremiseInstance

Il seguente esempio di codice mostra come utilizzare. Unregister-CDOnPremiseInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la registrazione dell'istanza on-premises con il nome specificato.

```
Unregister-CDOnPremiseInstance -InstanceName AssetTag12010298EX
```

- Per i dettagli sull'API, vedere [DeregisterOnPremisesInstance](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-CDApplication

Il seguente esempio di codice mostra come utilizzare. Update-CDApplication

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica il nome dell'applicazione specificata.

```
Update-CDApplication -ApplicationName MyNewApplication -NewApplicationName  
MyNewApplication-2
```

- Per i dettagli sull'API, vedere [UpdateApplication](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-CDDeploymentGroup

Il seguente esempio di codice mostra come utilizzare. Update-CDDeploymentGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica il nome del gruppo di distribuzione specificato per l'applicazione specificata.

```
Update-CDDeploymentGroup -ApplicationName MyNewApplication -  
CurrentDeploymentGroupName MyNewDeploymentGroup -NewDeploymentGroupName  
MyNewDeploymentGroup-2
```

Esempio 2: Questo esempio mostra come specificare i gruppi di tag di EC2 istanza in base ai quali un'istanza deve essere identificata per poter essere inclusa nell'ambiente sostitutivo di una blue/green distribuzione.

```
Update-CDDeploymentGroup -ApplicationName MyNewApplication -  
CurrentDeploymentGroupName MyNewDeploymentGroup -NewDeploymentGroupName  
MyNewDeploymentGroup-2 -Ec2TagSetList  
@(@{Key="key1";Type="KEY_ONLY"},@{Key="Key2";Type="KEY_AND_VALUE";Value="Value2"}),@(@{Key=
```

- Per i dettagli sull'API, vedere [UpdateDeploymentGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

CodePipeline esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. CodePipeline

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Confirm-CPJob

Il seguente esempio di codice mostra come utilizzare `Confirm-CPJob`.

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene lo stato di un processo specificato.

```
Confirm-CPJob -JobId f570dc12-5ef3-44bc-945a-6e133EXAMPLE -Nonce 3
```

Output:

```
Value
-----
InProgress
```

- Per i dettagli sull'API, vedere [AcknowledgeJob](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Disable-CPStageTransition

Il seguente esempio di codice mostra come utilizzare. `Disable-CPStageTransition`

Strumenti per PowerShell V4

Esempio 1: questo esempio disabilita la transizione in entrata per la fase specificata nella pipeline specificata.

```
Disable-CPStageTransition -PipelineName CodePipelineDemo -Reason "Disabling temporarily." -StageName Beta -TransitionType Inbound
```

- Per i dettagli sull'API, vedere [DisableStageTransition](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-CPStageTransition

Il seguente esempio di codice mostra come utilizzare. Enable-CPStageTransition

Strumenti per PowerShell V4

Esempio 1: questo esempio disabilita la transizione in entrata per lo stadio specificato nella pipeline specificata.

```
Enable-CPStageTransition -PipelineName CodePipelineDemo -StageName Beta -  
TransitionType Inbound
```

- Per i dettagli sull'API, vedere [EnableStageTransition](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CPActionType

Il seguente esempio di codice mostra come utilizzare. Get-CPActionType

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera le informazioni relative a tutte le azioni disponibili per il proprietario specificato.

```
ForEach ($actionType in (Get-CPActionType -ActionOwnerFilter AWS)) {  
    Write-Output ("For Category = " + $actionType.Id.Category + ", Owner = " +  
$actionType.Id.Owner + ", Provider = " + $actionType.Id.Provider + ", Version = " +  
$actionType.Id.Version + ":")  
    Write-Output ("  ActionConfigurationProperties:")  
}
```

```

ForEach ($acp in $actionType.ActionConfigurationProperties) {
    Write-Output ("    For " + $acp.Name + ":")
    Write-Output ("        Description = " + $acp.Description)
    Write-Output ("        Key = " + $acp.Key)
    Write-Output ("        Queryable = " + $acp.Queryable)
    Write-Output ("        Required = " + $acp.Required)
    Write-Output ("        Secret = " + $acp.Secret)
}
Write-Output ("  InputArtifactDetails:")
Write-Output ("    MaximumCount = " +
$actionType.InputArtifactDetails.MaximumCount)
Write-Output ("    MinimumCount = " +
$actionType.InputArtifactDetails.MinimumCount)
Write-Output ("  OutputArtifactDetails:")
Write-Output ("    MaximumCount = " +
$actionType.OutputArtifactDetails.MaximumCount)
Write-Output ("    MinimumCount = " +
$actionType.OutputArtifactDetails.MinimumCount)
Write-Output ("  Settings:")
Write-Output ("    EntityUrlTemplate = " + $actionType.Settings.EntityUrlTemplate)
Write-Output ("    ExecutionUrlTemplate = " +
$actionType.Settings.ExecutionUrlTemplate)
}

```

Output:

```

For Category = Deploy, Owner = AWS, Provider = ElasticBeanstalk, Version = 1:
ActionConfigurationProperties:
  For ApplicationName:
    Description = The AWS Elastic Beanstalk Application name
    Key = True
    Queryable = False
    Required = True
    Secret = False
  For EnvironmentName:
    Description = The AWS Elastic Beanstalk Environment name
    Key = True
    Queryable = False
    Required = True
    Secret = False
InputArtifactDetails:
  MaximumCount = 1
  MinimumCount = 1

```

```

OutputArtifactDetails:
  MaximumCount = 0
  MinimumCount = 0
Settings:
  EntityUrlTemplate = https://console.aws.amazon.com/elasticbeanstalk/r/
application/{Config:ApplicationName}
  ExecutionUrlTemplate = https://console.aws.amazon.com/elasticbeanstalk/r/
application/{Config:ApplicationName}
For Category = Deploy, Owner = AWS, Provider = CodeDeploy, Version = 1:
ActionConfigurationProperties:
  For ApplicationName:
    Description = The AWS CodeDeploy Application name
    Key = True
    Queryable = False
    Required = True
    Secret = False
  For DeploymentGroupName:
    Description = The AWS CodeDeploy Deployment Group name
    Key = True
    Queryable = False
    Required = True
    Secret = False
InputArtifactDetails:
  MaximumCount = 1
  MinimumCount = 1
OutputArtifactDetails:
  MaximumCount = 0
  MinimumCount = 0
Settings:
  EntityUrlTemplate = https://console.aws.amazon.com/codedeploy/home?#/
applications/{Config:ApplicationName}/deployment-groups/{Config:DeploymentGroupName}
  ExecutionUrlTemplate = https://console.aws.amazon.com/codedeploy/home?#/
deployments/{ExternalExecutionId}

```

- Per i dettagli sull'API, vedere [ListActionTypes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CPActionableJobList

Il seguente esempio di codice mostra come utilizzare. Get-CPActionableJobList

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni su tutti i processi con interazione per la categoria di azione, il proprietario, il provider, la versione e i parametri di query specificati.

```
Get-CPActionableJobList -ActionTypeId_Category Build -ActionTypeId_Owner Custom
-ActionTypeId_Provider MyCustomProviderName -ActionTypeId_Version 1 -QueryParam
@{"ProjectName" = "MyProjectName"}
```

Output:

AccountId	Data	Id
-----	-----	--
-----	-----	--
80398EXAMPLE	Amazon.CodePipeline.Model.JobData	0de392f5-712d-4f41-ace3-
f57a0EXAMPLE	3	

- Per i dettagli sull'API, vedere [PollForJobs](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CPJobDetail

Il seguente esempio di codice mostra come utilizzare. Get-CPJobDetail

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni generali sul processo specificato.

```
Get-CPJobDetail -JobId f570dc12-5ef3-44bc-945a-6e133EXAMPLE
```

Output:

AccountId	Data	Id
-----	-----	--
80398EXAMPLE	Amazon.CodePipeline.Model.JobData	
f570dc12-5ef3-44bc-945a-6e133EXAMPLE		

Esempio 2: questo esempio ottiene informazioni dettagliate sul processo specificato.

```

$jobDetails = Get-CPJobDetail -JobId f570dc12-5ef3-44bc-945a-6e133EXAMPLE
Write-Output ("For Job " + $jobDetails.Id + ":")
Write-Output ("  AccountId = " + $jobDetails.AccountId)
$jobData = $jobDetails.Data
Write-Output ("  Configuration:")
ForEach ($key in $jobData.ActionConfiguration.Keys) {
    $value = $jobData.ActionConfiguration.$key
    Write-Output ("    " + $key + " = " + $value)
}
Write-Output ("  ActionTypeId:")
Write-Output ("    Category = " + $jobData.ActionTypeId.Category)
Write-Output ("    Owner = " + $jobData.ActionTypeId.Owner)
Write-Output ("    Provider = " + $jobData.ActionTypeId.Provider)
Write-Output ("    Version = " + $jobData.ActionTypeId.Version)
Write-Output ("  ArtifactCredentials:")
Write-Output ("    AccessKeyId = " + $jobData.ArtifactCredentials.AccessKeyId)
Write-Output ("    SecretAccessKey = " +
    $jobData.ArtifactCredentials.SecretAccessKey)
Write-Output ("    SessionToken = " + $jobData.ArtifactCredentials.SessionToken)
Write-Output ("  InputArtifacts:")
ForEach ($ia in $jobData.InputArtifacts) {
    Write-Output ("    " + $ia.Name)
}
Write-Output ("  OutputArtifacts:")
ForEach ($oa in $jobData.OutputArtifacts) {
    Write-Output ("    " + $oa.Name)
}
Write-Output ("  PipelineContext:")
$context = $jobData.PipelineContext
Write-Output ("    Name = " + $context.Action.Name)
Write-Output ("    PipelineName = " + $context.PipelineName)
Write-Output ("    Stage = " + $context.Stage.Name)

```

Output:

```

For Job f570dc12-5ef3-44bc-945a-6e133EXAMPLE:
  AccountId = 80398EXAMPLE
  Configuration:
  ActionTypeId:
    Category = Build
    Owner = Custom
    Provider = MyCustomProviderName
    Version = 1

```



```
ArtifactCredentials:
  AccessKeyId = ASIAIEI3...IXI6YREX
  SecretAccessKey = cqAFDhEi...RdQyfa2u
  SessionToken = AQoDYXdz...5u+lsAU=
InputArtifacts:
  MyApp
OutputArtifacts:
  MyAppBuild
PipelineContext:
  Name = Build
  PipelineName = CodePipelineDemo
  Stage = Build
```

- Per i dettagli sull'API, vedere [GetJobDetails](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CPPipeline

Il seguente esempio di codice mostra come utilizzare. Get-CPPipeline

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni generali sulla pipeline specificata.

```
Get-CPPipeline -Name CodePipelineDemo -Version 1
```

Output:

```
ArtifactStore : Amazon.CodePipeline.Model.ArtifactStore
Name          : CodePipelineDemo
RoleArn       : arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole
Stages        : {Source, Build, Beta, TestStage}
Version       : 1
```

Esempio 2: questo esempio ottiene informazioni dettagliate sulla pipeline specificata.

```
$pipeline = Get-CPPipeline -Name CodePipelineDemo
Write-Output ("Name = " + $pipeline.Name)
Write-Output ("RoleArn = " + $pipeline.RoleArn)
Write-Output ("Version = " + $pipeline.Version)
Write-Output ("ArtifactStore:")
```

```

Write-Output ("  Location = " + $pipeline.ArtifactStore.Location)
Write-Output ("  Type = " + $pipeline.ArtifactStore.Type.Value)
Write-Output ("Stages:")
ForEach ($stage in $pipeline.Stages) {
  Write-Output ("  Name = " + $stage.Name)
  Write-Output ("    Actions:")
  ForEach ($action in $stage.Actions) {
    Write-Output ("      Name = " + $action.Name)
    Write-Output ("      Category = " + $action.ActionTypeId.Category)
    Write-Output ("      Owner = " + $action.ActionTypeId.Owner)
    Write-Output ("      Provider = " + $action.ActionTypeId.Provider)
    Write-Output ("      Version = " + $action.ActionTypeId.Version)
    Write-Output ("      Configuration:")
    ForEach ($key in $action.Configuration.Keys) {
      $value = $action.Configuration.$key
      Write-Output ("        " + $key + " = " + $value)
    }
    Write-Output ("      InputArtifacts:")
    ForEach ($ia in $action.InputArtifacts) {
      Write-Output ("        " + $ia.Name)
    }
    ForEach ($oa in $action.OutputArtifacts) {
      Write-Output ("        " + $oa.Name)
    }
    Write-Output ("      RunOrder = " + $action.RunOrder)
  }
}

```

Output:

```

Name = CodePipelineDemo
RoleArn = arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole
Version = 3
ArtifactStore:
  Location = amzn-s3-demo-bucket
  Type = S3
Stages:
  Name = Source
  Actions:
    Name = Source
    Category = Source
    Owner = ThirdParty
    Provider = GitHub

```

```
    Version = 1
    Configuration:
        Branch = master
        OAuthToken = ****
        Owner = my-user-name
        Repo = MyRepoName
    InputArtifacts:
        MyApp
    RunOrder = 1
Name = Build
    Actions:
        Name = Build
        Category = Build
        Owner = Custom
        Provider = MyCustomProviderName
        Version = 1
        Configuration:
            ProjectName = MyProjectName
        InputArtifacts:
            MyApp
            MyAppBuild
        RunOrder = 1
Name = Beta
    Actions:
        Name = CodePipelineDemoFleet
        Category = Deploy
        Owner = AWS
        Provider = CodeDeploy
        Version = 1
        Configuration:
            ApplicationName = CodePipelineDemoApplication
            DeploymentGroupName = CodePipelineDemoFleet
        InputArtifacts:
            MyAppBuild
        RunOrder = 1
Name = TestStage
    Actions:
        Name = MyJenkinsTestAction
        Category = Test
        Owner = Custom
        Provider = MyCustomTestProvider
        Version = 1
        Configuration:
            ProjectName = MyJenkinsProjectName
```

```
InputArtifacts:
  MyAppBuild
RunOrder = 1
```

- Per i dettagli sull'API, vedere [GetPipeline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CPPipelineList

Il seguente esempio di codice mostra come utilizzare. Get-CPPipelineList

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene un elenco delle pipeline disponibili.

```
Get-CPPipelineList
```

Output:

Created	Name	Updated	Version
-----	----	-----	-----
8/13/2015 10:17:54 PM	CodePipelineDemo	8/13/2015 10:17:54 PM	3
7/8/2015 2:41:53 AM	MyFirstPipeline	7/22/2015 9:06:37 PM	7

- Per i dettagli sull'API, vedere [ListPipelines](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CPPipelineState

Il seguente esempio di codice mostra come utilizzare. Get-CPPipelineState

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene informazioni generali sulle fasi della pipeline specificata.

```
Get-CPPipelineState -Name CodePipelineDemo
```

Output:

```
Created          : 8/13/2015 10:17:54 PM
```

```

PipelineName      : CodePipelineDemo
PipelineVersion   : 1
StageStates       : {Source, Build, Beta, TestStage}
Updated           : 8/13/2015 10:17:54 PM

```

Esempio 2: questo esempio ottiene informazioni dettagliate sullo stato della pipeline specificata.

```

ForEach ($stageState in (Get-CCPipelineState -Name $arg).StageStates) {
    Write-Output ("For " + $stageState.StageName + ":")
    Write-Output ("  InboundTransitionState:")
    Write-Output ("    DisabledReason = " +
$stageState.InboundTransitionState.DisabledReason)
    Write-Output ("    Enabled = " + $stageState.InboundTransitionState.Enabled)
    Write-Output ("    LastChangedAt = " +
$stageState.InboundTransitionState.LastChangedAt)
    Write-Output ("    LastChangedBy = " +
$stageState.InboundTransitionState.LastChangedBy)
    Write-Output ("  ActionStates:")
    ForEach ($actionState in $stageState.ActionStates) {
        Write-Output ("    For " + $actionState.ActionName + ":")
    Write-Output ("      CurrentRevision:")
        Write-Output ("        Created = " + $actionState.CurrentRevision.Created)
    Write-Output ("        RevisionChangeId = " +
$actionState.CurrentRevision.RevisionChangeId)
    Write-Output ("        RevisionId = " + $actionState.CurrentRevision.RevisionId)
    Write-Output ("        EntityUrl = " + $actionState.EntityUrl)
    Write-Output ("      LatestExecution:")
        Write-Output ("        ErrorDetails:")
        Write-Output ("          Code = " +
$actionState.LatestExecution.ErrorDetails.Code)
    Write-Output ("          Message = " +
$actionState.LatestExecution.ErrorDetails.Message)
    Write-Output ("        ExternalExecutionId = " +
$actionState.LatestExecution.ExternalExecutionId)
    Write-Output ("        ExternalExecutionUrl = " +
$actionState.LatestExecution.ExternalExecutionUrl)
    Write-Output ("        LastStatusChange = " +
$actionState.LatestExecution.LastStatusChange)
    Write-Output ("        PercentComplete = " +
$actionState.LatestExecution.PercentComplete)
    Write-Output ("        Status = " + $actionState.LatestExecution.Status)
    Write-Output ("        Summary = " + $actionState.LatestExecution.Summary)
    Write-Output ("        RevisionUrl = " + $actionState.RevisionUrl)

```

```
}
}
```

Output:

```
For Source:
  InboundTransitionState:
    DisabledReason =
    Enabled =
    LastChangedAt =
    LastChangedBy =
  ActionStates:
    For Source:
      CurrentRevision:
        Created =
        RevisionChangeId =
        RevisionId =
      EntityUrl = https://github.com/my-user-name/MyRepoName/tree/master
      LatestExecution:
        ErrorDetails:
          Code =
          Message =
          ExternalExecutionId =
          ExternalExecutionUrl =
          LastStatusChange = 07/20/2015 23:28:45
          PercentComplete = 0
          Status = Succeeded
          Summary =
        RevisionUrl =
For Build:
  InboundTransitionState:
    DisabledReason =
    Enabled = True
    LastChangedAt = 01/01/0001 00:00:00
    LastChangedBy =
  ActionStates:
    For Build:
      CurrentRevision:
        Created =
        RevisionChangeId =
        RevisionId =
      EntityUrl = http://54.174.131.1EX/job/MyJenkinsDemo
      LatestExecution:
```

```

        ErrorDetails:
            Code = TimeoutError
            Message = The action failed because a job worker exceeded its time limit.
If this is a custom action, make sure that the job worker is configured correctly.
            ExternalExecutionId =
            ExternalExecutionUrl =
            LastStatusChange = 07/21/2015 00:29:29
            PercentComplete = 0
            Status = Failed
            Summary =
            RevisionUrl =
For Beta:
    InboundTransitionState:
        DisabledReason =
        Enabled = True
        LastChangedAt = 01/01/0001 00:00:00
        LastChangedBy =
    ActionStates:
        For CodePipelineDemoFleet:
            CurrentRevision:
                Created =
                RevisionChangeId =
                RevisionId =
                EntityUrl = https://console.aws.amazon.com/codedeploy/home?#/applications/
CodePipelineDemoApplication/deployment-groups/CodePipelineDemoFleet
            LatestExecution:
                ErrorDetails:
                    Code =
                    Message =
                    ExternalExecutionId = d-D5LTCZXEX
                    ExternalExecutionUrl = https://console.aws.amazon.com/codedeploy/home?#/
deployments/d-D5LTCZXEX
                    LastStatusChange = 07/08/2015 22:07:42
                    PercentComplete = 0
                    Status = Succeeded
                    Summary = Deployment Succeeded
                RevisionUrl =
For TestStage:
    InboundTransitionState:
        DisabledReason =
        Enabled = True
        LastChangedAt = 01/01/0001 00:00:00
        LastChangedBy =
    ActionStates:

```

```

For MyJenkinsTestAction25:
  CurrentRevision:
    Created =
    RevisionChangeId =
    RevisionId =
  EntityUrl = http://54.174.131.1EX/job/MyJenkinsDemo
  LatestExecution:
    ErrorDetails:
      Code =
      Message =
    ExternalExecutionId = 5
    ExternalExecutionUrl = http://54.174.131.1EX/job/MyJenkinsDemo/5
    LastStatusChange = 07/08/2015 22:09:03
    PercentComplete = 0
    Status = Succeeded
    Summary = Finished
  RevisionUrl =

```

- Per i dettagli sull'API, vedere [GetPipelineState](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CPCustomActionType

Il seguente esempio di codice mostra come utilizzare. New-CPCustomActionType

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova azione personalizzata con le proprietà specificate.

```

New-CPCustomActionType -Category Build -ConfigurationProperty @{"Description"
= "The name of the build project must be provided when this action is added
to the pipeline."; "Key" = $True; "Name" = "ProjectName"; "Queryable"
= $False; "Required" = $True; "Secret" = $False; "Type" = "String"} -
Settings_EntityUrlTemplate "https://my-build-instance/job/{Config:ProjectName}/"
-Settings_ExecutionUrlTemplate "https://my-build-instance/job/mybuildjob/
lastSuccessfulBuild{ExternalExecutionId}/" -InputArtifactDetails_MaximumCount
1 -OutputArtifactDetails_MaximumCount 1 -InputArtifactDetails_MinimumCount 0 -
OutputArtifactDetails_MinimumCount 0 -Provider "MyBuildProviderName" -Version 1

```

Output:

```
ActionConfigurationProperties : {ProjectName}
```


Id	: Amazon.CodePipeline.Model.ActionTypeId
InputArtifactDetails	: Amazon.CodePipeline.Model.ArtifactDetails
OutputArtifactDetails	: Amazon.CodePipeline.Model.ArtifactDetails
Settings	: Amazon.CodePipeline.Model.ActionTypeSettings

- Per i dettagli sull'API, vedere [CreateCustomActionType](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CPPipeline

Il seguente esempio di codice mostra come utilizzare. New-CPPipeline

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova pipeline con le impostazioni specificate.

```
$pipeline = New-Object Amazon.CodePipeline.Model.PipelineDeclaration

$sourceStageAction = New-Object Amazon.CodePipeline.Model.ActionDeclaration
$deployStageAction = New-Object Amazon.CodePipeline.Model.ActionDeclaration

$sourceStageActionOutputArtifact = New-Object
    Amazon.CodePipeline.Model.OutputArtifact
$sourceStageActionOutputArtifact.Name = "MyApp"

$sourceStageAction.ActionTypeId = @{"Category" = "Source"; "Owner" = "AWS";
    "Provider" = "S3"; "Version" = 1}
$sourceStageAction.Configuration.Add("S3Bucket", "amzn-s3-demo-bucket")
$sourceStageAction.Configuration.Add("S3ObjectKey", "my-object-key-name.zip")
$sourceStageAction.OutputArtifacts.Add($sourceStageActionOutputArtifact)
$sourceStageAction.Name = "Source"

$deployStageActionInputArtifact = New-Object Amazon.CodePipeline.Model.InputArtifact
$deployStageActionInputArtifact.Name = "MyApp"

$deployStageAction.ActionTypeId = @{"Category" = "Deploy"; "Owner" = "AWS";
    "Provider" = "CodeDeploy"; "Version" = 1}
$deployStageAction.Configuration.Add("ApplicationName",
    "CodePipelineDemoApplication")
$deployStageAction.Configuration.Add("DeploymentGroupName", "CodePipelineDemoFleet")
$deployStageAction.InputArtifacts.Add($deployStageActionInputArtifact)
$deployStageAction.Name = "CodePipelineDemoFleet"
```

```

$sourceStage = New-Object Amazon.CodePipeline.Model.StageDeclaration
$deployStage = New-Object Amazon.CodePipeline.Model.StageDeclaration

$sourceStage.Name = "Source"
$deployStage.Name = "Beta"

$sourceStage.Actions.Add($sourceStageAction)
$deployStage.Actions.Add($deployStageAction)

$pipeline.ArtifactStore = @{"Location" = "amzn-s3-demo-bucket"; "Type" = "S3"}
$pipeline.Name = "CodePipelineDemo"
$pipeline.RoleArn = "arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole"
$pipeline.Stages.Add($sourceStage)
$pipeline.Stages.Add($deployStage)
$pipeline.Version = 1

New-CPPipeline -Pipeline $pipeline

```

Output:

```

ArtifactStore : Amazon.CodePipeline.Model.ArtifactStore
Name           : CodePipelineDemo
RoleArn        : arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole
Stages         : {Source, Beta}
Version        : 1

```

- Per i dettagli sull'API, vedere [CreatePipeline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CPCustomActionType

Il seguente esempio di codice mostra come utilizzare. Remove-CPCustomActionType

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'azione personalizzata specificata. Il comando richiederà una conferma prima di procedere. Aggiungi il parametro -Force per eliminare l'azione personalizzata senza una richiesta di conferma.

```
Remove-CPCustomActionType -Category Build -Provider MyBuildProviderName -Version 1
```

- Per i dettagli sull'API, vedere [DeleteCustomActionType](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CPPipeline

Il seguente esempio di codice mostra come utilizzare. Remove-CPPipeline

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la pipeline specificata. Il comando richiederà una conferma prima di procedere. Aggiungi il parametro -Force per eliminare la pipeline senza una richiesta di conferma.

```
Remove-CPPipeline -Name CodePipelineDemo
```

- Per i dettagli sull'API, vedere [DeletePipeline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-CPPipelineExecution

Il seguente esempio di codice mostra come utilizzare. Start-CPPipelineExecution

Strumenti per PowerShell V4

Esempio 1: questo esempio avvia l'esecuzione della pipeline specificata.

```
Start-CPPipelineExecution -Name CodePipelineDemo
```

- Per i dettagli sull'API, vedere [StartPipelineExecution](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-CPPipeline

Il seguente esempio di codice mostra come utilizzare. Update-CPPipeline

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la pipeline esistente specificata in base alle impostazioni specificate.

```
$pipeline = New-Object Amazon.CodePipeline.Model.PipelineDeclaration

$sourceStageAction = New-Object Amazon.CodePipeline.Model.ActionDeclaration
$deployStageAction = New-Object Amazon.CodePipeline.Model.ActionDeclaration

$sourceStageActionOutputArtifact = New-Object
    Amazon.CodePipeline.Model.OutputArtifact
$sourceStageActionOutputArtifact.Name = "MyApp"

$sourceStageAction.ActionTypeId = @{"Category" = "Source"; "Owner" = "AWS";
    "Provider" = "S3"; "Version" = 1}
$sourceStageAction.Configuration.Add("S3Bucket", "amzn-s3-demo-bucket")
$sourceStageAction.Configuration.Add("S3ObjectKey", "my-object-key-name.zip")
$sourceStageAction.OutputArtifacts.Add($sourceStageActionOutputArtifact)
$sourceStageAction.Name = "Source"

$deployStageActionInputArtifact = New-Object Amazon.CodePipeline.Model.InputArtifact
$deployStageActionInputArtifact.Name = "MyApp"

$deployStageAction.ActionTypeId = @{"Category" = "Deploy"; "Owner" = "AWS";
    "Provider" = "CodeDeploy"; "Version" = 1}
$deployStageAction.Configuration.Add("ApplicationName",
    "CodePipelineDemoApplication")
$deployStageAction.Configuration.Add("DeploymentGroupName", "CodePipelineDemoFleet")
$deployStageAction.InputArtifacts.Add($deployStageActionInputArtifact)
$deployStageAction.Name = "CodePipelineDemoFleet"

$sourceStage = New-Object Amazon.CodePipeline.Model.StageDeclaration
$deployStage = New-Object Amazon.CodePipeline.Model.StageDeclaration

$sourceStage.Name = "MyInputFiles"
$deployStage.Name = "MyTestDeployment"

$sourceStage.Actions.Add($sourceStageAction)
$deployStage.Actions.Add($deployStageAction)

$pipeline.ArtifactStore = @{"Location" = "amzn-s3-demo-bucket"; "Type" = "S3"}
$pipeline.Name = "CodePipelineDemo"
$pipeline.RoleArn = "arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole"
$pipeline.Stages.Add($sourceStage)
$pipeline.Stages.Add($deployStage)
$pipeline.Version = 1
```

```
Update-CPPipeline -Pipeline $pipeline
```

Output:

```
ArtifactStore : Amazon.CodePipeline.Model.ArtifactStore
Name          : CodePipelineDemo
RoleArn       : arn:aws:iam::80398EXAMPLE:role/CodePipelineServiceRole
Stages        : {InputFiles, TestDeployment}
Version       : 2
```

- Per i dettagli sull'API, vedere [UpdatePipeline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di identità di Amazon Cognito con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon Cognito Identity.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-CGIIdentityPool

Il seguente esempio di codice mostra come usare. Get-CGIIdentityPool

Strumenti per PowerShell V4

Esempio 1: recupera informazioni su uno specifico pool di identità in base al relativo ID.

```
Get-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
```

Output:

```

LoggedAt                : 8/12/2015 4:29:40 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName    :
IdentityPoolId           : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName         : CommonTests1
OpenIdConnectProviderARNs : {}
SupportedLoginProviders  : {}
ResponseMetadata         : Amazon.Runtime.ResponseMetadata
ContentLength            : 142
HttpStatusCode           : OK

```

- Per i dettagli sull'API, vedere [DescribeIdentityPool](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CGIIIdentityPoolList

Il seguente esempio di codice mostra come utilizzare. `Get-CGIIIdentityPoolList`

Strumenti per PowerShell V4

Esempio 1: recupera un elenco di pool di identità esistenti.

```
Get-CGIIIdentityPoolList
```

Output:

IdentityPoolId	IdentityPoolName
-----	-----
us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1	CommonTests1
us-east-1:118d242d-204e-4b88-b803-EXAMPLEGUID2	Tests2
us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3	CommonTests13

- Per i dettagli sull'API, vedere [ListIdentityPools](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CGIIIdentityPoolRole

Il seguente esempio di codice mostra come utilizzare. `Get-CGIIIdentityPoolRole`

Strumenti per PowerShell V4

Esempio 1: ottiene informazioni sui ruoli per un pool di identità specifico.

```
Get-CGIIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
```

Output:

```
LoggedAt           : 8/12/2015 4:33:51 PM
IdentityPoolId     : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
Roles              : {[unauthenticated, arn:aws:iam::123456789012:role/
CommonTests1Role]}
ResponseMetadata   : Amazon.Runtime.ResponseMetadata
ContentLength      : 165
HttpStatusCode     : OK
```

- Per i dettagli sull'API, vedere [GetIdentityPoolRoles](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-CGIIIdentityPool

Il seguente esempio di codice mostra come utilizzare. New-CGIIIdentityPool

Strumenti per PowerShell V4

Esempio 1: crea un nuovo pool di identità che consente di utilizzare identità non autenticate.

```
New-CGIIIdentityPool -AllowUnauthenticatedIdentities $true -IdentityPoolName
CommonTests13
```

Output:

```
LoggedAt           : 8/12/2015 4:56:07 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName          :
IdentityPoolId               : us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3
IdentityPoolName              : CommonTests13
OpenIdConnectProviderARNs     : {}
SupportedLoginProviders       : {}
```

```
ResponseMetadata      : Amazon.Runtime.ResponseMetadata
ContentLength          : 136
HttpStatusCode         : OK
```

- Per i dettagli sull'API, vedere [CreateIdentityPool](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CGIIdentityPool

Il seguente esempio di codice mostra come utilizzare. Remove-CGIIdentityPool

Strumenti per PowerShell V4

Esempio 1: elimina un pool di identità specifico.

```
Remove-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

- Per i dettagli sull'API, vedere [DeleteIdentityPool](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-CGIIdentityPoolRole

Il seguente esempio di codice mostra come utilizzare. Set-CGIIdentityPoolRole

Strumenti per PowerShell V4

Esempio 1: configura un pool di identità specifico in modo che abbia un ruolo IAM non autenticato.

```
Set-CGIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1 -Role @{ "unauthenticated" = "arn:aws:iam::123456789012:role/
CommonTests1Role" }
```

- Per i dettagli sull'API, vedere [SetIdentityPoolRoles](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-CGIIdentityPool

Il seguente esempio di codice mostra come utilizzare. Update-CGIIdentityPool

Strumenti per PowerShell V4

Esempio 1: aggiorna alcune proprietà del pool di identità (in questo caso il nome del pool di identità).

```
Update-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1 -IdentityPoolName NewPoolName
```

Output:

```
LoggedAt                : 8/12/2015 4:53:33 PM
AllowUnauthenticatedIdentities : False
DeveloperProviderName    :
IdentityPoolId           : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName         : NewPoolName
OpenIdConnectProviderARNs : {}
SupportedLoginProviders  : {}
ResponseMetadata         : Amazon.Runtime.ResponseMetadata
ContentLength            : 135
HttpStatusCode           : OK
```

- Per i dettagli sull'API, vedere [UpdateIdentityPool](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

AWS Config esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. AWS Config

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-CFGResourceTag

Il seguente esempio di codice mostra come utilizzare `Add-CFGResourceTag`.

Strumenti per PowerShell V4

Esempio 1: questo esempio associa il tag specificato all'ARN di risorsa, che in questo caso è `config-rule/config-rule-16iyn0`.

```
Add-CFGResourceTag -ResourceArn arn:aws:config:eu-west-1:123456789012:config-rule/
config-rule-16iyn0 -Tag @{Key="Release";Value="Beta"}
```

- Per i dettagli sull'API, vedere [TagResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGAggregateComplianceByConfigRuleList

Il seguente esempio di codice mostra come utilizzare `Get-CFGAggregateComplianceByConfigRuleList`.

Strumenti per PowerShell V4

Esempio 1: Questo esempio recupera i dettagli dal filtro `ConfigurationAggregator` «kaju» per la regola di configurazione specificata e la «Conformità» expands/returns della regola.

```
Get-CFGAggregateComplianceByConfigRuleList -ConfigurationAggregatorName kaju
-Filters_ConfigRuleName ALB_HTTP_TO_HTTPS_REDIRECTION_CHECK | Select-Object -
ExpandProperty Compliance
```

Output:

ComplianceContributorCount	ComplianceType
-----	-----
Amazon.ConfigService.Model.ComplianceContributorCount	NON_COMPLIANT

Esempio 2: questo esempio recupera i dettagli da quanto fornito `ConfigurationAggregator`, li filtra per l'account specificato per tutte le regioni incluse nell'aggregatore e restituisce ulteriormente la conformità di tutte le regole.

```
Get-CFGAggregateComplianceByConfigRuleList -ConfigurationAggregatorName
kaju -Filters_AccountId 123456789012 | Select-Object ConfigRuleName,
@{N="Compliance";E={$_.Compliance.ComplianceType}}
```

Output:

ConfigRuleName	Compliance
-----	-----
ALB_HTTP_TO_HTTPS_REDIRECTION_CHECK	NON_COMPLIANT
ec2-instance-no-public-ip	NON_COMPLIANT
desired-instance-type	NON_COMPLIANT

- Per i dettagli sull'API, vedere [DescribeAggregateComplianceByConfigRules](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGAggregateComplianceDetailsByConfigRule

Il seguente esempio di codice mostra come utilizzare. Get - CFGAggregateComplianceDetailsByConfigRule

Strumenti per PowerShell V4

Esempio 1: Questo esempio restituisce i risultati della valutazione selezionando l'output con resource-id e resource-type per la regola AWS Config " che si trovano nello stato 'COMPLIANTdesired-instance-type' per l'account, l'aggregatore, la regione e la regola di configurazione specificati

```
Get-CFGAggregateComplianceDetailsByConfigRule -AccountId 123456789012 -
AwsRegion eu-west-1 -ComplianceType COMPLIANT -ConfigRuleName desired-
instance-type -ConfigurationAggregatorName raju | Select-Object -
ExpandProperty EvaluationResultIdentifier | Select-Object -ExpandProperty
EvaluationResultQualifier
```

Output:

ConfigRuleName	ResourceId	ResourceType
-----	-----	-----
desired-instance-type	i-0f1bf2f34c5678d12	AWS::EC2::Instance
desired-instance-type	i-0fd12dd3456789123	AWS::EC2::Instance

- Per [GetAggregateComplianceDetailsByConfigRule](#) AWS Strumenti per PowerShell i dettagli sull'API, vedere in Cmdlet Reference (V4).

Get-CFGAggregateConfigRuleComplianceSummary

Il seguente esempio di codice mostra come utilizzare. Get - CFGAggregateConfigRuleComplianceSummary

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il numero di regole non conformi per l'aggregatore specificato.

```
(Get-CFGAggregateConfigRuleComplianceSummary -ConfigurationAggregatorName
  raju).AggregateComplianceCounts.ComplianceSummary.NonCompliantResourceCount
```

Output:

```
CapExceeded  CappedCount
-----
False        5
```

- Per i dettagli sull'API, vedere [GetAggregateConfigRuleComplianceSummary](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGAggregateDiscoveredResourceCount

Il seguente esempio di codice mostra come utilizzare. Get - CFGAggregateDiscoveredResourceCount

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il conteggio delle risorse per l'aggregatore specificato filtrate in base alla Regione us-east-1.

```
Get-CFGAggregateDiscoveredResourceCount -ConfigurationAggregatorName Master -
  Filters_Region us-east-1
```

Output:

```

GroupByKey GroupedResourceCounts NextToken TotalDiscoveredResources
-----
{} 455

```

Esempio 2: questo esempio restituisce il conteggio delle risorse raggruppate per RESOURCE_TYPE per la Regione filtrata per l'aggregatore specificato.

```

Get-CFGAggregateDiscoveredResourceCount -ConfigurationAggregatorName Master -
Filters_Region us-east-1 -GroupByKey RESOURCE_TYPE |
Select-Object -ExpandProperty GroupedResourceCounts

```

Output:

GroupName	ResourceCount
-----	-----
AWS::CloudFormation::Stack	12
AWS::CloudFront::Distribution	1
AWS::CloudTrail::Trail	1
AWS::DynamoDB::Table	1
AWS::EC2::EIP	2
AWS::EC2::FlowLog	2
AWS::EC2::InternetGateway	4
AWS::EC2::NatGateway	2
AWS::EC2::NetworkAcl	4
AWS::EC2::NetworkInterface	12
AWS::EC2::RouteTable	13
AWS::EC2::SecurityGroup	18
AWS::EC2::Subnet	16
AWS::EC2::VPC	4
AWS::EC2::VPCEndpoint	2
AWS::EC2::VPCPeeringConnection	1
AWS::IAM::Group	2
AWS::IAM::Policy	51
AWS::IAM::Role	78
AWS::IAM::User	7
AWS::Lambda::Function	3
AWS::RDS::DBSecurityGroup	1
AWS::S3::Bucket	3
AWS::SSM::AssociationCompliance	107
AWS::SSM::ManagedInstanceInventory	108

- Per i dettagli sull'API, vedere [GetAggregateDiscoveredResourceCounts](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGAggregateDiscoveredResourceList

Il seguente esempio di codice mostra come utilizzare. Get - CFGAggregateDiscoveredResourceList

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce gli identificatori di risorsa per il tipo di risorsa specificato aggregati nell'aggregatore "Ireland". Per l'elenco dei tipi di risorse, consulta https://docs.aws.amazon.com/sdkfornet/v3/apidocs/index.html?page=ConfigService/TConfigServiceResourceType.html&tocid=Amazon_._.ConfigService.ResourceType

```
Get-CFGAggregateDiscoveredResourceList -ConfigurationAggregatorName Ireland -
ResourceType ([Amazon.ConfigService.ResourceType]::AWSAutoScalingAutoScalingGroup)
```

Output:

```
ResourceId      : arn:aws:autoscaling:eu-
west-1:123456789012:autoScalingGroup:12e3b4fc-1234-1234-
a123-1d2ba3c45678:autoScalingGroupName/asg-1
ResourceName    : asg-1
ResourceType    : AWS::AutoScaling::AutoScalingGroup
SourceAccountId : 123456789012
SourceRegion    : eu-west-1
```

Esempio 2: questo esempio restituisce il tipo di risorsa **AwsEC2SecurityGroup** denominato "default" per l'aggregatore specificato filtrato in base alla Regione us-east-1.

```
Get-CFGAggregateDiscoveredResourceList -ConfigurationAggregatorName raju -
ResourceType ([Amazon.ConfigService.ResourceType]::AWSEC2SecurityGroup) -
Filters_Region us-east-1 -Filters_ResourceName default
```

Output:

```
ResourceId      : sg-01234bd5dbfa67c89
ResourceName    : default
ResourceType    : AWS::EC2::SecurityGroup
```

```
SourceAccountId : 123456789102
SourceRegion    : us-east-1

ResourceId      : sg-0123a4ebbf56789be
ResourceName    : default
ResourceType    : AWS::EC2::SecurityGroup
SourceAccountId : 123456789102
SourceRegion    : us-east-1

ResourceId      : sg-4fc1d234
ResourceName    : default
ResourceType    : AWS::EC2::SecurityGroup
SourceAccountId : 123456789102
SourceRegion    : us-east-1
```

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4).
[ListAggregateDiscoveredResources](#) AWS Strumenti per PowerShell

Get-CFGAggregateResourceConfig

Il seguente esempio di codice mostra come utilizzare. Get-CFGAggregateResourceConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce l'elemento di configurazione per la risorsa specificata aggregata ed espande la configurazione.

```
(Get-CFGAggregateResourceConfig -ResourceIdentifier_SourceRegion
us-east-1 -ResourceIdentifier_SourceAccountId 123456789012 -
ResourceIdentifier_ResourceId sg-4fc1d234 -ResourceIdentifier_ResourceType
([Amazon.ConfigService.ResourceType]::AWSEC2SecurityGroup) -
ConfigurationAggregatorName raju).Configuration | ConvertFrom-Json
```

Output:

```
{"description":"default VPC security group","groupName":"default","ipPermissions":
[{"ipProtocol":"-1","ipv6Ranges":[],"prefixListIds":[],"userIdGroupPairs":
[{"groupId":"sg-4fc1d234","userId":"123456789012"}],"ipv4Ranges":
[{"ipRanges":[]},{ "fromPort":3389,"ipProtocol":"tcp","ipv6Ranges":
[{"prefixListIds":[],"toPort":3389,"userIdGroupPairs":[],"ipv4Ranges":
[{"cidrIp":"54.240.197.224/29","description":"office subnet"},
{"cidrIp":"72.21.198.65/32","description":"home pc"}],"ipRanges":
```

```
[["54.240.197.224/29", "72.21.198.65/32"]], "ownerId": "123456789012", "groupId": "sg-4fc1d234",
[{"ipProtocol": "-1", "ipv6Ranges": [], "prefixListIds": [], "userIdGroupPairs":
[], "ipv4Ranges": [{"cidrIp": "0.0.0.0/0"}], "ipRanges": ["0.0.0.0/0"]}], "tags":
[], "vpcId": "vpc-2d1c2e34"}
```

- Per i dettagli sull'API, vedere [GetAggregateResourceconfig-service](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGAggregateResourceConfigBatch

Il seguente esempio di codice mostra come utilizzare. Get-CFGAggregateResourceConfigBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera l'elemento di configurazione corrente per la risorsa (identificata) presente nell'aggregatore specificato.

```
$resIdentifier=[Amazon.ConfigService.Model.AggregateResourceIdentifier]@{
    ResourceId= "i-012e3cb4df567e8aa"
    ResourceName = "arn:aws:ec2:eu-west-1:123456789012:instance/i-012e3cb4df567e8aa"
    ResourceType = [Amazon.ConfigService.ResourceType]::AWSEC2Instance
    SourceAccountId = "123456789012"
    SourceRegion = "eu-west-1"
}

Get-CFGAggregateResourceConfigBatch -ResourceIdentifier $resIdentifier -
ConfigurationAggregatorName raju
```

Output:

```
BaseConfigurationItems  UnprocessedResourceIdentifiers
-----
{}                      {arn:aws:ec2:eu-west-1:123456789012:instance/
i-012e3cb4df567e8aa}
```

- Per i dettagli sull'API, vedere [BatchGetAggregateResourceconfig-service](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGAggregationAuthorizationList

Il seguente esempio di codice mostra come utilizzare. Get-CFGAggregationAuthorizationList

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera le autorizzazioni concesse agli aggregatori.

```
Get-CFGAggregationAuthorizationList
```

Output:

```
AggregationAuthorizationArn
  AuthorizedAccountId AuthorizedAwsRegion CreationTime
-----
-----
arn:aws:config-service:eu-west-1:123456789012:aggregation-
authorization/123456789012/eu-west-1 123456789012 eu-west-1
8/26/2019 12:55:27 AM
```

- Per i dettagli sull'API, vedere [DescribeAggregationAuthorizations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGComplianceByConfigRule

Il seguente esempio di codice mostra come utilizzare. Get-CFGComplianceByConfigRule

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera i dettagli di conformità per la regola ebs-optimized-instance, per la quale non esistono risultati di valutazione correnti per la regola, quindi restituisce INSUFFICIENT_DATA

```
(Get-CFGComplianceByConfigRule -ConfigRuleName ebs-optimized-instance).Compliance
```

Output:

```
ComplianceContributorCount ComplianceType
-----
INSUFFICIENT_DATA
```

Esempio 2: questo esempio restituisce il numero di risorse non conformi per la regola ALB_HTTP_TO_HTTPS_REDIRECTION_CHECK.

```
(Get-CFGComplianceByConfigRule -ConfigRuleName ALB_HTTP_TO_HTTPS_REDIRECTION_CHECK -
ComplianceType NON_COMPLIANT).Compliance.ComplianceContributorCount
```

Output:

```
CapExceeded CappedCount
-----
False      2
```

- Per i dettagli sull'API, vedere [DescribeComplianceByConfigRule](#) in Cmdlet Reference (V4). AWS Strumenti per PowerShell

Get-CFGComplianceByResource

Il seguente esempio di codice mostra come utilizzare. Get-CFGComplianceByResource

Strumenti per PowerShell V4

Esempio 1: questo esempio controlla il tipo di conformità "COMPLIANT" nel tipo di risorsa **AWS::SSM::ManagedInstanceInventory**.

```
Get-CFGComplianceByResource -ComplianceType COMPLIANT -ResourceType
AWS::SSM::ManagedInstanceInventory
```

Output:

```
Compliance                                ResourceId                                ResourceType
-----
Amazon.ConfigService.Model.Compliance i-0123bcf4b567890e3
AWS::SSM::ManagedInstanceInventory
Amazon.ConfigService.Model.Compliance i-0a1234f6f5d6b78f7
AWS::SSM::ManagedInstanceInventory
```

- Per i dettagli sull'API, vedere [DescribeComplianceByResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGComplianceDetailsByConfigRule

Il seguente esempio di codice mostra come utilizzare. Get-CFGComplianceDetailsByConfigRule

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene i risultati della valutazione per la regola access-keys-rotated e restituisce l'output raggruppato per tipo di conformità

```
Get-CFGComplianceDetailsByConfigRule -ConfigRuleName access-keys-rotated | Group-Object ComplianceType
```

Output:

```
Count Name Group
-----
2 COMPLIANT {Amazon.ConfigService.Model.EvaluationResult,
Amazon.ConfigService.Model.EvaluationResult}
5 NON_COMPLIANT {Amazon.ConfigService.Model.EvaluationResult,
Amazon.ConfigService.Model.EvaluationResult,
Amazon.ConfigService.Model.EvaluationRes...
```

Esempio 2: questo esempio richiede i dettagli di conformità per la regola per le risorse COMPLIANT. access-keys-rotated

```
Get-CFGComplianceDetailsByConfigRule -ConfigRuleName access-
keys-rotated -ComplianceType COMPLIANT | ForEach-Object
{$_ .EvaluationResultIdentifier.EvaluationResultQualifier}
```

Output:

```
ConfigRuleName      ResourceId      ResourceType
-----
access-keys-rotated BCAB1CDJ2LITAPVEW3JAH AWS::IAM::User
access-keys-rotated BCAB1CDJ2LITL3EHREM4Q AWS::IAM::User
```

- Per i dettagli sull'API, vedere [GetComplianceDetailsByConfigRule](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGComplianceDetailsByResource

Il seguente esempio di codice mostra come utilizzare. Get-CFGComplianceDetailsByResource

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra risultati di valutazione per una risorsa specificata.

```
Get-CFGComplianceDetailsByResource -ResourceId ABCD5STJ4EFGHIVEW6JAH -ResourceType
'AWS::IAM::User'
```

Output:

```
Annotation           :
ComplianceType       : COMPLIANT
ConfigRuleInvokedTime : 8/25/2019 11:34:56 PM
EvaluationResultIdentifier : Amazon.ConfigService.Model.EvaluationResultIdentifier
ResultRecordedTime    : 8/25/2019 11:34:56 PM
ResultToken          :
```

- Per i dettagli sull'API, vedere [GetComplianceDetailsByResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGComplianceSummaryByConfigRule

Il seguente esempio di codice mostra come utilizzare. Get - CFGComplianceSummaryByConfigRule

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il numero di regole di configurazione non conformi.

```
Get-CFGComplianceSummaryByConfigRule -Select
ComplianceSummary.NonCompliantResourceCount
```

Output:

```
CapExceeded CappedCount
-----
False      9
```

- Per i dettagli sull'API, vedere [GetComplianceSummaryByConfigRule](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGComplianceSummaryByResourceType

Il seguente esempio di codice mostra come utilizzare. Get - CFGComplianceSummaryByResourceType

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il numero di risorse conformi o non conformi e converte l'output in json.

```
Get-CFGComplianceSummaryByResourceType -Select
ComplianceSummariesByResourceType.ComplianceSummary | ConvertTo-Json
{
  "ComplianceSummaryTimestamp": "2019-12-14T06:14:49.778Z",
  "CompliantResourceCount": {
    "CapExceeded": false,
    "CappedCount": 2
  },
  "NonCompliantResourceCount": {
    "CapExceeded": true,
    "CappedCount": 100
  }
}
```

- Per i dettagli sull'API, vedere [GetComplianceSummaryByResourceType](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGConfigRule

Il seguente esempio di codice mostra come utilizzare. Get - CFGConfigRule

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le regole di configurazione per l'account, con le proprietà selezionate.

```
Get-CFGConfigRule | Select-Object ConfigRuleName, ConfigRuleId, ConfigRuleArn,
ConfigRuleState
```

Output:

ConfigRuleName	ConfigRuleId	ConfigRuleArn
	ConfigRuleState	
-----	-----	-----
ALB_REDIRECTION_CHECK	config-rule-12iyn3	arn:aws:config-
service:eu-west-1:123456789012:config-rule/config-rule-12iyn3	ACTIVE	
access-keys-rotated	config-rule-aospfr	arn:aws:config-
service:eu-west-1:123456789012:config-rule/config-rule-aospfr	ACTIVE	
autoscaling-group-elb-healthcheck-required	config-rule-cn1f2x	arn:aws:config-
service:eu-west-1:123456789012:config-rule/config-rule-cn1f2x	ACTIVE	

- Per i dettagli sull'API, vedere [DescribeConfigRules](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGConfigRuleEvaluationStatus

Il seguente esempio di codice mostra come utilizzare `Get-CFGConfigRuleEvaluationStatus`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce le informazioni sullo stato delle regole di configurazione specificate.

```
Get-CFGConfigRuleEvaluationStatus -ConfigRuleName root-account-mfa-enabled, vpc-
flow-logs-enabled
```

Output:

```
ConfigRuleArn      : arn:aws:config:eu-west-1:123456789012:config-rule/
config-rule-kvq1wk
ConfigRuleId       : config-rule-kvq1wk
ConfigRuleName     : root-account-mfa-enabled
FirstActivatedTime : 8/27/2019 8:05:17 AM
FirstEvaluationStarted : True
LastErrorCode      :
LastErrorMessage   :
LastFailedEvaluationTime : 1/1/0001 12:00:00 AM
LastFailedInvocationTime : 1/1/0001 12:00:00 AM
LastSuccessfulEvaluationTime : 12/13/2019 8:12:03 AM
LastSuccessfulInvocationTime : 12/13/2019 8:12:03 AM
```

```

ConfigRuleArn      : arn:aws:config:eu-west-1:123456789012:config-rule/
config-rule-z1s23b
ConfigRuleId       : config-rule-z1s23b
ConfigRuleName     : vpc-flow-logs-enabled
FirstActivatedTime  : 8/14/2019 6:23:44 AM
FirstEvaluationStarted : True
LastErrorCode      :
LastErrorMessage    :
LastFailedEvaluationTime : 1/1/0001 12:00:00 AM
LastFailedInvocationTime : 1/1/0001 12:00:00 AM
LastSuccessfulEvaluationTime : 12/13/2019 7:12:01 AM
LastSuccessfulInvocationTime : 12/13/2019 7:12:01 AM

```

- Per i dettagli sull'API, vedere [DescribeConfigRuleEvaluationStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGConfigurationAggregatorList

Il seguente esempio di codice mostra come utilizzare `Get-CFGConfigurationAggregatorList` Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce tutti gli aggregatori per la Regione/l'account.

```
Get-CFGConfigurationAggregatorList
```

Output:

```

AccountAggregationSources      :
  {Amazon.ConfigService.Model.AccountAggregationSource}
ConfigurationAggregatorArn     : arn:aws:config-service:eu-
west-1:123456789012:config-aggregator/config-aggregator-xabcalme
ConfigurationAggregatorName    : IrelandMaster
CreationTime                   : 8/25/2019 11:42:39 PM
LastUpdatedTime               : 8/25/2019 11:42:39 PM
OrganizationAggregationSource  :

AccountAggregationSources      : {}
ConfigurationAggregatorArn     : arn:aws:config-service:eu-
west-1:123456789012:config-aggregator/config-aggregator-qubqabcd
ConfigurationAggregatorName    : raju
CreationTime                   : 8/11/2019 8:39:25 AM

```

```
LastUpdatedTime           : 8/11/2019 8:39:25 AM
OrganizationAggregationSource :
    Amazon.ConfigService.Model.OrganizationAggregationSource
```

- Per i dettagli sull'API, vedere [DescribeConfigurationAggregators](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGConfigurationAggregatorSourcesStatus

Il seguente esempio di codice mostra come utilizzare. Get - CFGConfigurationAggregatorSourcesStatus

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza i campi richiesti per le fonti nell'aggregatore specificato.

```
Get-CFGConfigurationAggregatorSourcesStatus -ConfigurationAggregatorName raju |
select SourceType, LastUpdateStatus, LastUpdateTime, SourceId
```

Output:

SourceType	LastUpdateStatus	LastUpdateTime	SourceId
-----	-----	-----	-----
ORGANIZATION	SUCCEEDED	12/31/2019 7:45:06 AM	Organization
ACCOUNT	SUCCEEDED	12/31/2019 7:09:38 AM	612641234567
ACCOUNT	SUCCEEDED	12/31/2019 7:12:53 AM	933301234567
ACCOUNT	SUCCEEDED	12/31/2019 7:18:10 AM	933301234567
ACCOUNT	SUCCEEDED	12/31/2019 7:25:17 AM	933301234567
ACCOUNT	SUCCEEDED	12/31/2019 7:25:49 AM	612641234567
ACCOUNT	SUCCEEDED	12/31/2019 7:26:11 AM	612641234567

- Per i dettagli sull'API, vedere [DescribeConfigurationAggregatorSourcesStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGConfigurationRecorder

Il seguente esempio di codice mostra come utilizzare. Get - CFGConfigurationRecorder

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce i dettagli dei registratori di configurazione.


```
Get-CFGConfigurationRecorder | Format-List
```

Output:

```
Name           : default
RecordingGroup  : Amazon.ConfigService.Model.RecordingGroup
RoleARN         : arn:aws:iam::123456789012:role/aws-service-role/
config.amazonaws.com/AWSServiceRoleForConfig
```

- Per i dettagli sull'API, vedere [DescribeConfigurationRecorders](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGConfigurationRecorderStatus

Il seguente esempio di codice mostra come utilizzare. Get-CFGConfigurationRecorderStatus
Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce lo stato dei valori restituiti dei registratori di configurazione.

```
Get-CFGConfigurationRecorderStatus
```

Output:

```
LastErrorCode      :
LastErrorMessage   :
LastStartTime      : 10/11/2019 10:13:51 AM
LastStatus         : Success
LastStatusChangeTime : 12/31/2019 6:14:12 AM
LastStopTime       : 10/11/2019 10:13:46 AM
Name               : default
Recording          : True
```

- Per i dettagli sull'API, vedere [DescribeConfigurationRecorderStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGConformancePack

Il seguente esempio di codice mostra come utilizzare. Get-CFGConformancePack

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutti i pacchetti di conformità.

```
Get-CFGConformancePack
```

Output:

```
ConformancePackArn      : arn:aws:config:eu-west-1:123456789012:conformance-
conformance-pack/dono/conformance-pack-p0acq8bpz
ConformancePackId       : conformance-pack-p0acabcde
ConformancePackInputParameters : {}
ConformancePackName     : dono
CreatedBy                :
DeliveryS3Bucket        : kt-ps-examples
DeliveryS3KeyPrefix     :
LastUpdateRequestedTime : 12/31/2019 8:45:31 AM
```

- Per i dettagli sull'API, vedere [DescribeConformancePacks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGDeliveryChannel

Il seguente esempio di codice mostra come utilizzare. Get-CFGDeliveryChannel

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera il canale di distribuzione per la Regione e visualizza i dettagli.

```
Get-CFGDeliveryChannel -Region eu-west-1 | Select-Object Name, S3BucketName,
S3KeyPrefix,
@{N="DeliveryFrequency";E={$_.ConfigSnapshotDeliveryProperties.DeliveryFrequency}}
```

Output:

Name	S3BucketName	S3KeyPrefix	DeliveryFrequency
----	-----	-----	-----
default	config-bucket-NA	my	TwentyFour_Hours

- Per i dettagli sull'API, vedere [DescribeDeliveryChannels](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-CFGResourceTag

Il seguente esempio di codice mostra come utilizzare. Get-CFGResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i tag associati per la risorsa specificata

```
Get-CFGResourceTag -ResourceArn $rules[0].ConfigRuleArn
```

Output:

```
Key      Value
---      -
Version  1.3
```

- Per i dettagli sull'API, vedere [ListTagsForResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-CFGConformancePack

Il seguente esempio di codice mostra come utilizzare. Remove-CFGConformancePack

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il pacchetto di conformità specificato, insieme a tutte le regole, le azioni correttive e i risultati della valutazione a esso associati.

```
Remove-CFGConformancePack -ConformancePackName dono
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-CFGConformancePack (DeleteConformancePack)" on
target "dono".
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): Y
```

- Per i dettagli sull'API, vedere [DeleteConformancePack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-CFGConformancePack

Il seguente esempio di codice mostra come utilizzare Write-CFGConformancePack

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un pacchetto di conformità, recuperando il modello dal file yaml specificato.

```
Write-CFGConformancePack -ConformancePackName dono -DeliveryS3Bucket amzn-s3-demo-bucket -TemplateBody (Get-Content C:\windows\temp\template.yaml -Raw)
```

- Per i dettagli sull'API, vedere [PutConformancePack](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-CFGDeliveryChannel

Il seguente esempio di codice mostra come utilizzare Write-CFGDeliveryChannel

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica la proprietà deliveryFrequency di un canale di distribuzione esistente.

```
Write-CFGDeliveryChannel -ConfigSnapshotDeliveryProperties_DeliveryFrequency TwentyFour_Hours -DeliveryChannelName default -DeliveryChannel_S3BucketName amzn-s3-demo-bucket -DeliveryChannel_S3KeyPrefix my
```

- Per i dettagli sull'API, vedere [PutDeliveryChannel](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Device Farm con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Device Farm.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

New-DFUpload

Il seguente esempio di codice mostra come utilizzare New-DFUpload.

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un caricamento AWS Device Farm per un'app Android. È possibile ottenere l'ARN del progetto dall'output di New-DFProject o Get-DFProject List. Utilizzate l'URL firmato nell'DFUpload output New- per caricare un file su Device Farm.

```
New-DFUpload -ContentType "application/octet-stream" -ProjectArn  
"arn:aws:devicefarm:us-west-2:123456789012:project:EXAMPLEa-7ec1-4741-9c1f-  
d3e04EXAMPLE" -Name "app.apk" -Type ANDROID_APP
```

- Per i dettagli sull'API, vedere [CreateUpload](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Directory Service esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Directory Service

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-DSIpRoute

Il seguente esempio di codice mostra come utilizzare Add-DSIpRoute.

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove il tag di risorsa assegnato all'ID directory specificato

```
Add-DSIpRoute -DirectoryId d-123456ijkl -IpRoute @{CidrIp ="203.0.113.5/32"} -  
UpdateSecurityGroupForDirectoryController $true
```

- Per i dettagli sull'API, vedere [AddIpRoutes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-DSResourceTag

Il seguente esempio di codice mostra come utilizzare Add-DSResourceTag

Strumenti per PowerShell V4

Esempio 1: questo comando aggiunge il tag di risorsa all'ID di directory specificato

```
Add-DSResourceTag -ResourceId d-123456ijkl -Tag @{Key="myTag"; Value="mytgValue"}
```

- Per i dettagli sull'API, vedere [AddTagsToResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Approve-DSTrust

Il seguente esempio di codice mostra come utilizzare. Approve-DSTrust

Strumenti per PowerShell V4

Esempio 1: questo esempio chiama l'operazione dell' VerifyTrust API AWS Directory Service per il Trustid specificato.

```
Approve-DSTrust -TrustId t-9067157123
```

- Per i dettagli sull'API, vedere [VerifyTrust](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Confirm-DSSharedDirectory

Il seguente esempio di codice mostra come utilizzare. Confirm-DSSharedDirectory

Strumenti per PowerShell V4

Esempio 1: Questo esempio accetta una richiesta di condivisione della directory inviata dal proprietario Account AWS della directory.

```
Confirm-DSSharedDirectory -SharedDirectoryId d-9067012345
```

Output:

```
CreatedDateTime      : 12/30/2019 4:20:27 AM
LastUpdatedDateTime : 12/30/2019 4:21:40 AM
OwnerAccountId       : 123456781234
OwnerDirectoryId     : d-123456ijkl
SharedAccountId      : 123456784321
SharedDirectoryId    : d-9067012345
ShareMethod          :
ShareNotes           : This is test sharing
ShareStatus          : Sharing
```

- Per i dettagli sull'API, vedere [AcceptSharedDirectory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Connect-DSDirectory

Il seguente esempio di codice mostra come utilizzare. Connect-DSDirectory

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un AD Connector per la connessione a una directory on-premises.

```
Connect-DSDirectory -Name contoso.com -ConnectSettings_CustomerUserName  
Administrator -Password $Password -ConnectSettings_CustomerDnsIp 172.31.36.96  
-ShortName CONTOSO -Size Small -ConnectSettings_VpcId vpc-123459da -  
ConnectSettings_SubnetId subnet-1234ccaa, subnet-5678ffbb
```

- Per i dettagli sull'API, vedere [ConnectDirectory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Deny-DSSharedDirectory

Il seguente esempio di codice mostra come utilizzare. Deny-DSSharedDirectory

Strumenti per PowerShell V4

Esempio 1: questo esempio rifiuta una richiesta di condivisione della directory inviata dall'account del proprietario della directory.

```
Deny-DSSharedDirectory -SharedDirectoryId d-9067012345
```

Output:

```
d-9067012345
```

- Per i dettagli sull'API, vedere [RejectSharedDirectory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Disable-DSDirectoryShare

Il seguente esempio di codice mostra come utilizzare. Disable-DSDirectoryShare

Strumenti per PowerShell V4

Esempio 1: questo esempio arresta la condivisione della directory tra il proprietario della directory e gli account consumer.

```
Disable-DSDirectoryShare -DirectoryId d-123456ijkl -UnshareTarget_Id 123456784321 -  
UnshareTarget_Type ACCOUNT
```

Output:

```
d-9067012345
```

- Per i dettagli sull'API, vedere [UnshareDirectory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Disable-DSLDPAPS

Il seguente esempio di codice mostra come utilizzare. Disable-DSLDPAPS

Strumenti per PowerShell V4

Esempio 1: questo esempio disattiva le chiamate LDAP protette per la directory specificata.

```
Disable-DSLDPAPS -DirectoryId d-123456ijkl -Type Client
```

- Per informazioni dettagliate sull'API, consulta [DisableLDAPPS](#) nella documentazione di riferimento dei cmdlet di AWS Strumenti per PowerShell (V4).

Disable-DSRadius

Il seguente esempio di codice mostra come usare Disable-DSRadius.

Strumenti per PowerShell V4

Esempio 1: questo esempio disabilita il server RADIUS configurato per una directory AD Connector o Microsoft AD.

```
Disable-DSRadius -DirectoryId d-123456ijkl
```

- Per i dettagli sull'API, vedere [DisableRadius](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Disable-DSSso

Il seguente esempio di codice mostra come utilizzare. Disable-DSSso

Strumenti per PowerShell V4

Esempio 1: questo esempio disabilita l'autenticazione Single Sign-On (SSO) per una directory.

```
Disable-DSSso -DirectoryId d-123456ijkl
```

- Per i dettagli sull'API, vedere [DisableSso](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-DSDirectoryShare

Il seguente esempio di codice mostra come utilizzare. Enable-DSDirectoryShare

Strumenti per PowerShell V4

Esempio 1: Questo esempio condivide una directory specificata nel tuo AWS account con un altro AWS account utilizzando il metodo Handshake.

```
Enable-DSDirectoryShare -DirectoryId d-123456ijkl -ShareTarget_Id 123456784321 -  
ShareMethod HANDSHAKE -ShareTarget_Type ACCOUNT
```

Output:

```
d-9067012345
```

- Per i dettagli sull'API, vedere [ShareDirectory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-DSLDPAPS

Il seguente esempio di codice mostra come utilizzare. Enable-DSLDPAPS

Strumenti per PowerShell V4

Esempio 1: questo esempio attiva l'opzione per la directory specifica in modo da utilizzare sempre le chiamate LDAP protette.

```
Enable-DSLDAPS -DirectoryId d-123456ijkl -Type Client
```

- Per informazioni dettagliate sull'API, consulta [EnableLDAPS](#) nella documentazione di riferimento dei cmdlet di AWS Strumenti per PowerShell (V4).

Enable-DSRadius

Il seguente esempio di codice mostra come usare `Enable-DSRadius`.

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita l'autenticazione a più fattori (MFA) con la configurazione server RADIUS fornita per una directory AD Connector o Microsoft AD.

```
Enable-DSRadius -DirectoryId d-123456ijkl  
-RadiusSettings_AuthenticationProtocol PAP  
-RadiusSettings_DisplayLabel Radius  
-RadiusSettings_RadiusPort 1812  
-RadiusSettings_RadiusRetry 4  
-RadiusSettings_RadiusServer 10.4.185.113  
-RadiusSettings_RadiusTimeout 50  
-RadiusSettings_SharedSecret wJalrXUtnFEMI
```

- Per i dettagli sull'API, vedere [EnableRadius](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-DSSso

Il seguente esempio di codice mostra come utilizzare `Enable-DSSso`.

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita l'autenticazione Single Sign-On (SSO) per una directory.

```
Enable-DSSso -DirectoryId d-123456ijkl
```

- Per i dettagli sull'API, vedere [EnableSso](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSCertificate

Il seguente esempio di codice mostra come utilizzare. Get-DSCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza informazioni sul certificato registrato per una connessione LDAP protetta.

```
Get-DSCertificate -DirectoryId d-123456ijkl -CertificateId c-906731e34f
```

Output:

```
CertificateId      : c-906731e34f
CommonName         : contoso-EC2AMAZ-CTGG2NM-CA
ExpiryDateTime     : 4/15/2025 6:34:15 PM
RegisteredDateTime : 4/15/2020 6:38:56 PM
State              : Registered
StateReason        : Certificate registered successfully.
```

- Per i dettagli sull'API, vedere [DescribeCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSCertificateList

Il seguente esempio di codice mostra come utilizzare. Get-DSCertificateList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutti i certificati registrati per una connessione LDAP protetta per la directory specificata.

```
Get-DSCertificateList -DirectoryId d-123456ijkl
```

Output:

CertificateId	CommonName	ExpiryDateTime	State

```
-----
c-906731e34f  contoso-EC2AMAZ-CTGG2NM-CA  4/15/2025  6:34:15 PM  Registered
```

- Per i dettagli sull'API, vedere [ListCertificates](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSConditionalForwarder

Il seguente esempio di codice mostra come utilizzare. Get-DSConditionalForwarder

Strumenti per PowerShell V4

Esempio 1: questo comando ottiene tutti i server d'inoltro condizionale per l'ID directory specificato.

```
Get-DSConditionalForwarder -DirectoryId d-123456ijkl
```

Output:

DnsIpAddress	RemoteDomainName	ReplicationScope
-----	-----	-----
{172.31.77.239}	contoso.com	Domain

- Per i dettagli sull'API, vedere [DescribeConditionalForwarders](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSDirectory

Il seguente esempio di codice mostra come utilizzare. Get-DSDirectory

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce le informazioni sulle directory appartenenti a questo account.

```
Get-DSDirectory | Select-Object DirectoryId, Name, DnsIpAddress, Type
```

Output:

DirectoryId	Name	DnsIpAddress	Type

```

-----
d-123456abcd abcd.example.com {172.31.74.189, 172.31.13.145} SimpleAD
d-123456efgh wifi.example.com {172.31.16.108, 172.31.10.56} ADConnector
d-123456ijkl lan2.example.com {172.31.10.56, 172.31.16.108} MicrosoftAD

```

- Per i dettagli sull'API, vedere [DescribeDirectories](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSDirectoryLimit

Il seguente esempio di codice mostra come utilizzare. Get-DSDirectoryLimit

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza le informazioni sui limiti delle directory per la Regione us-east-1.

```
Get-DSDirectoryLimit -Region us-east-1
```

Output:

```

CloudOnlyDirectoriesCurrentCount : 1
CloudOnlyDirectoriesLimit         : 10
CloudOnlyDirectoriesLimitReached : False
CloudOnlyMicrosoftADCurrentCount : 1
CloudOnlyMicrosoftADLimit        : 20
CloudOnlyMicrosoftADLimitReached : False
ConnectedDirectoriesCurrentCount : 1
ConnectedDirectoriesLimit        : 10

```

- Per i dettagli sull'API, vedere [GetDirectoryLimits](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSDomainControllerList

Il seguente esempio di codice mostra come utilizzare. Get-DSDomainControllerList

Strumenti per PowerShell V4

Esempio 1: questo comando ottiene l'elenco dettagliato dei controller di dominio avviati per l'ID directory specificato

```
Get-DSDomainControllerList -DirectoryId d-123456ijkl
```

Output:

```
AvailabilityZone      : us-east-1b
DirectoryId           : d-123456ijkl
DnsIpAddr             : 172.31.16.108
DomainControllerId    : dc-1234567aa6
LaunchTime            : 4/4/2019 4:53:43 AM
Status                : Active
StatusLastUpdatedDateTime : 4/24/2019 1:37:54 PM
StatusReason          :
SubnetId              : subnet-1234kkaa
VpcId                 : vpc-123459d

AvailabilityZone      : us-east-1d
DirectoryId           : d-123456ijkl
DnsIpAddr             : 172.31.10.56
DomainControllerId    : dc-1234567aa7
LaunchTime            : 4/4/2019 4:53:43 AM
Status                : Active
StatusLastUpdatedDateTime : 4/4/2019 5:14:31 AM
StatusReason          :
SubnetId              : subnet-5678ffbb
VpcId                 : vpc-123459d
```

- Per i dettagli sull'API, vedere [DescribeDomainControllers](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSEventTopic

Il seguente esempio di codice mostra come utilizzare. `Get-DSEventTopic`

Strumenti per PowerShell V4

Esempio 1: questo comando visualizza le informazioni relative all'argomento SNS configurato per la notifica quando lo stato della directory cambia.

```
Get-DSEventTopic -DirectoryId d-123456ijkl
```

Output:

```
CreatedDateTime : 12/13/2019 11:15:32 AM
DirectoryId     : d-123456ijkl
Status         : Registered
TopicArn       : arn:aws:sns:us-east-1:123456781234:snstopicname
TopicName      : snstopicname
```

- Per i dettagli sull'API, vedere [DescribeEventTopics](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSIpRouteList

Il seguente esempio di codice mostra come utilizzare. Get-DSIpRouteList

Strumenti per PowerShell V4

Esempio 1: questo comando ottiene i blocchi di indirizzi IP pubblici configurati nel routing di IP delle directory

```
Get-DSIpRouteList -DirectoryId d-123456ijkl
```

Output:

```
AddedDateTime      : 12/13/2019 12:27:22 PM
CidrIp             : 203.0.113.5/32
Description        : Public IP of On-Prem DNS Server
DirectoryId        : d-123456ijkl
IpRouteStatusMsg   : Added
IpRouteStatusReason :
```

- Per i dettagli sull'API, vedere [ListIpRoutes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSLdapSetting

Il seguente esempio di codice mostra come utilizzare. Get-DSLdapSetting

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive lo stato della sicurezza LDAP per la directory specificata.


```
Get-DSLdapSSetting -DirectoryId d-123456ijkl
```

Output:

```
LastUpdatedDateTime  LDAPSStatus LDAPSStatusReason
-----
4/15/2020 6:51:03 PM Enabled      LDAPS is enabled successfully.
```

- Per i dettagli sull'API, vedere [Descrivi LDAPSSettings](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSLogSubscriptionList

Il seguente esempio di codice mostra come utilizzare. Get-DSLogSubscriptionList

Strumenti per PowerShell V4

Esempio 1: questo comando ottiene le informazioni sugli abbonamenti ai log per l'ID directory specificato

```
Get-DSLogSubscriptionList -DirectoryId d-123456ijkl
```

Output:

```
DirectoryId  LogGroupName
SubscriptionCreatedDateTime
-----
d-123456ijkl /aws/directoryservice/d-123456ijkl-lan2.example.com 12/14/2019 9:05:23 AM
```

- Per i dettagli sull'API, vedere [ListLogSubscriptions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSResourceTag

Il seguente esempio di codice mostra come utilizzare. Get-DSResourceTag

Strumenti per PowerShell V4

Esempio 1: questo comando ottiene tutti i tag della directory specificata.

```
Get-DSResourceTag -ResourceId d-123456ijkl
```

Output:

Key	Value
---	-----
myTag	myTagValue

- Per i dettagli sull'API, vedere [ListTagsForResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSSchemaExtension

Il seguente esempio di codice mostra come utilizzare. Get-DSSchemaExtension

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le estensioni dello schema applicate a una directory Microsoft AD.

```
Get-DSSchemaExtension -DirectoryId d-123456ijkl
```

Output:

Description	: ManagedADSchemaExtension
DirectoryId	: d-123456ijkl
EndDateTime	: 4/12/2020 10:30:49 AM
SchemaExtensionId	: e-9067306643
SchemaExtensionStatus	: Completed
SchemaExtensionStatusReason	: Schema updates are complete.
StartDateTime	: 4/12/2020 10:28:42 AM

- Per i dettagli sull'API, vedere [ListSchemaExtensions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSSharedDirectory

Il seguente esempio di codice mostra come utilizzare. Get-DSSharedDirectory

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene le directory condivise del tuo account AWS

```
Get-DSSharedDirectory -OwnerDirectoryId d-123456ijkl -SharedDirectoryId d-9067012345
```

Output:

```
CreatedDateTime      : 12/30/2019 4:34:37 AM
LastUpdatedDateTime : 12/30/2019 4:35:22 AM
OwnerAccountId       : 123456781234
OwnerDirectoryId     : d-123456ijkl
SharedAccountId      : 123456784321
SharedDirectoryId    : d-9067012345
ShareMethod          : HANDSHAKE
ShareNotes           : This is a test Sharing
ShareStatus          : Shared
```

- Per i dettagli sull'API, vedere [DescribeSharedDirectories](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSSnapshot

Il seguente esempio di codice mostra come utilizzare. Get-DSSnapshot

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce le informazioni sugli snapshot di directory specificati appartenenti a questo account.

```
Get-DSSnapshot -DirectoryId d-123456ijkl
```

Output:

```
DirectoryId : d-123456ijkl
Name        :
SnapshotId  : s-9064bd1234
StartTime   : 12/13/2019 6:33:01 PM
```

```
Status      : Completed
Type        : Auto

DirectoryId : d-123456ijkl
Name        :
SnapshotId  : s-9064bb4321
StartTime   : 12/9/2019 9:48:11 PM
Status      : Completed
Type        : Auto
```

- Per i dettagli sull'API, vedere [DescribeSnapshots](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSSnapshotLimit

Il seguente esempio di codice mostra come utilizzare. Get-DSSnapshotLimit

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i limiti degli snapshot manuali per una directory specificata.

```
Get-DSSnapshotLimit -DirectoryId d-123456ijkl
```

Output:

```
ManualSnapshotsCurrentCount ManualSnapshotsLimit ManualSnapshotsLimitReached
-----
0                           5                           False
```

- Per i dettagli sull'API, vedere [GetSnapshotLimits](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DSTrust

Il seguente esempio di codice mostra come utilizzare. Get-DSTrust

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce le informazioni sulle relazioni di attendibilità create per l'ID directory specificato.

```
Get-DSTrust -DirectoryId d-123456abcd
```

Output:

```
CreatedDateTime      : 7/5/2019 4:55:42 AM
DirectoryId          : d-123456abcd
LastUpdatedDateTime  : 7/5/2019 4:56:04 AM
RemoteDomainName     : contoso.com
SelectiveAuth        : Disabled
StateLastUpdatedDateTime : 7/5/2019 4:56:04 AM
TrustDirection       : One-Way: Incoming
TrustId              : t-9067157123
TrustState           : Created
TrustStateReason      :
TrustType            : Forest
```

- Per i dettagli sull'API, vedere [DescribeTrusts](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-DSAlias

Il seguente esempio di codice mostra come utilizzare. New-DSAlias

Strumenti per PowerShell V4

Esempio 1: questo comando crea un alias per una directory e assegna tale alias all'ID directory specificato.

```
New-DSAlias -DirectoryId d-123456ijkl -Alias MyOrgName
```

Output:

```
Alias      DirectoryId
-----
myorgname d-123456ijkl
```

- Per i dettagli sull'API, vedere [CreateAlias](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-DSComputer

Il seguente esempio di codice mostra come utilizzare. New-DSComputer

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo oggetto computer Active Directory.

```
New-DSComputer -DirectoryId d-123456ijkl -ComputerName ADMemberServer -Password
$password
```

Output:

ComputerAttributes	ComputerId
ComputerName	
-----	-----

{WindowsSamName, DistinguishedName}	S-1-5-21-1191241402-978882507-2717148213-1662
ADMemberServer	

- Per i dettagli sull'API, vedere [CreateComputer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-DSConditionalForwarder

Il seguente esempio di codice mostra come utilizzare. New-DSConditionalForwarder

Strumenti per PowerShell V4

Esempio 1: Questo esempio crea un server d'inoltro condizionale nell'ID di directory specificato.
AWS

```
New-DSConditionalForwarder -DirectoryId d-123456ijkl -DnsIpAddress
172.31.36.96,172.31.10.56 -RemoteDomainName contoso.com
```

- Per i dettagli sull'API, vedere [CreateConditionalForwarder](#) in Cmdlet Reference (V4).AWS Strumenti per PowerShell

New-DSDirectory

Il seguente esempio di codice mostra come utilizzare. New-DSDirectory

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova directory Simple AD.

```
New-DSDirectory -Name corp.example.com -Password $Password -Size Small -  
VpcSettings_VpcId vpc-123459d -VpcSettings_SubnetIds subnet-1234kkaa,subnet-5678ffbb
```

- Per i dettagli sull'API, vedere [CreateDirectory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-DSLogSubscription

Il seguente esempio di codice mostra come utilizzare. New-DSLogSubscription

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un abbonamento per inoltrare in tempo reale i log di sicurezza dei controller di dominio Directory Service al gruppo di CloudWatch log Amazon specificato nel tuo Account AWS.

```
New-DSLogSubscription -DirectoryId d-123456ijkl -LogGroupName /aws/directoryservice/  
d-123456ijkl-lan2.example.com
```

- Per i dettagli sull'API, consulta [CreateLogSubscription AWS Strumenti per PowerShell](#) Cmdlet Reference (V4).

New-DSMicrosoftAD

Il seguente esempio di codice mostra come utilizzare. New-DSMicrosoftAD

Strumenti per PowerShell V4

Esempio 1: Questo esempio crea una nuova directory Microsoft AD in Cloud AWS.

```
New-DSMicrosoftAD -Name corp.example.com -Password $Password -edition Standard -  
VpcSettings_VpcId vpc-123459d -VpcSettings_SubnetIds subnet-1234kkaa,subnet-5678ffbb
```

- Per i dettagli sull'API, vedere [CreateMicrosoftAD](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-DSSnapshot

Il seguente esempio di codice mostra come utilizzare. New-DSSnapshot

Strumenti per PowerShell V4

Esempio 1: questo esempio crea uno snapshot di directory

```
New-DSSnapshot -DirectoryId d-123456ijkl
```

- Per i dettagli sull'API, vedere [CreateSnapshot](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-DSTrust

Il seguente esempio di codice mostra come utilizzare. New-DSTrust

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un trust bidirezionale a livello di foresta tra la directory AWS Managed Microsoft AD e Microsoft Active Directory esistente in locale.

```
New-DSTrust -DirectoryId d-123456ijkl -RemoteDomainName contoso.com -TrustDirection  
Two-Way -TrustType Forest -TrustPassword $Password -ConditionalForwarderIpAddr  
172.31.36.96
```

Output:

```
t-9067157123
```

- Per i dettagli sull'API, vedere [CreateTrust](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-DSCertificate

Il seguente esempio di codice mostra come utilizzare. Register-DSCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio registra un certificato per la connessione LDAP protetta.


```
$Certificate = Get-Content contoso.cer -Raw
Register-DSCertificate -DirectoryId d-123456ijkl -CertificateData $Certificate
```

Output:

```
c-906731e350
```

- Per i dettagli sull'API, vedere [RegisterCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-DSEventTopic

Il seguente esempio di codice mostra come utilizzare. Register-DSEventTopic

Strumenti per PowerShell V4

Esempio 1: questo esempio associa una directory come publisher a un argomento SNS.

```
Register-DSEventTopic -DirectoryId d-123456ijkl -TopicName snstopicname
```

- Per i dettagli sull'API, vedere [RegisterEventTopic](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-DSConditionalForwarder

Il seguente esempio di codice mostra come utilizzare. Remove-DSConditionalForwarder

Strumenti per PowerShell V4

Esempio 1: Questo esempio rimuove il server d'inoltro condizionale che è stato impostato per Direcotry. AWS

```
Remove-DSConditionalForwarder -DirectoryId d-123456ijkl -RemoteDomainName
contoso.com
```

- Per i dettagli sull'API, vedere [DeleteConditionalForwarder](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-DSDirectory

Il seguente esempio di codice mostra come utilizzare. Remove-DSDirectory

Strumenti per PowerShell V4

Esempio 1: Questo esempio elimina una AWS directory del servizio Directory (Simple AD/Microsoft AD/AD Connector)

```
Remove-DSDirectory -DirectoryId d-123456ijkl
```

- Per i dettagli sull'API, vedere [DeleteDirectory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-DSIpRoute

Il seguente esempio di codice mostra come utilizzare. Remove-DSIpRoute

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove l'IP specificato dalle route IP configurate di Directory-id.

```
Remove-DSIpRoute -DirectoryId d-123456ijkl -CidrIp 203.0.113.5/32
```

- Per i dettagli sull'API, vedere [RemoveIpRoutes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-DSLogSubscription

Il seguente esempio di codice mostra come utilizzare. Remove-DSLogSubscription

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove l'abbonamento ai log per l'ID directory specificato

```
Remove-DSLogSubscription -DirectoryId d-123456ijkl
```

- Per i dettagli sull'API, vedere [DeleteLogSubscription](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-DSResourceTag

Il seguente esempio di codice mostra come utilizzare. Remove-DSResourceTag

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove il tag di risorsa assegnato all'ID directory specificato

```
Remove-DSResourceTag -ResourceId d-123456ijkl -TagKey myTag
```

- Per i dettagli sull'API, vedere [RemoveTagsFromResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-DSSnapshot

Il seguente esempio di codice mostra come utilizzare. Remove-DSSnapshot

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove lo snapshot creato manualmente.

```
Remove-DSSnapshot -SnapshotId s-9068b488kc
```

- Per i dettagli sull'API, vedere [DeleteSnapshot](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-DSTrust

Il seguente esempio di codice mostra come utilizzare. Remove-DSTrust

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la relazione di trust esistente tra la directory AD AWS gestita e un dominio esterno.

```
Get-DSTrust -DirectoryId d-123456ijkl -Select Trusts.TrustId | Remove-DSTrust
```

Output:

```
t-9067157123
```

- Per i dettagli sull'API, vedere [DeleteTrust](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Reset-DSUserPassword

Il seguente esempio di codice mostra come utilizzare. Reset-DSUserPassword

Strumenti per PowerShell V4

Esempio 1: Questo esempio reimposta la password dell'utente di Active Directory denominato ADUser in AWS Managed microsoft AD o Simple AD Directory

```
Reset-DSUserPassword -UserName ADUser -DirectoryId d-123456ijkl -NewPassword  
$Password
```

- Per i dettagli sull'API, vedere [ResetUserPassword](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Restore-DSFromSnapshot

Il seguente esempio di codice mostra come utilizzare. Restore-DSFromSnapshot

Strumenti per PowerShell V4

Esempio 1: questo esempio ripristina una directory utilizzando uno snapshot della directory esistente.

```
Restore-DSFromSnapshot -SnapshotId s-9068b488kc
```

- Per i dettagli sull'API, vedere [RestoreFromSnapshot](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-DSDomainControllerCount

Il seguente esempio di codice mostra come utilizzare. Set-DSDomainControllerCount

Strumenti per PowerShell V4

Esempio 1: questo esempio imposta il numero di controller di dominio su 3 per l'ID directory specificato.

```
Set-DSDomainControllerCount -DirectoryId d-123456ijkl -DesiredNumber 3
```

- Per i dettagli sull'API, vedere [UpdateNumberOfDomainControllers](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-DSSchemaExtension

Il seguente esempio di codice mostra come utilizzare. Start-DSSchemaExtension

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca un'estensione dello schema a una directory Microsoft AD.

```
$ldif = Get-Content D:\Users\Username\Downloads\ExtendedSchema.ldf -Raw
Start-DSSchemaExtension -DirectoryId d-123456ijkl -
CreateSnapshotBeforeSchemaExtension $true -Description ManagedADSchemaExtension -
LdifContent $ldif
```

Output:

```
e-9067306643
```

- Per i dettagli sull'API, vedere [StartSchemaExtension](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-DSSchemaExtension

Il seguente esempio di codice mostra come utilizzare. Stop-DSSchemaExtension

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla un'estensione dello schema in esecuzione su una directory Microsoft AD.

```
Stop-DSSchemaExtension -DirectoryId d-123456ijkl -SchemaExtensionId e-9067306643
```

- Per i dettagli sull'API, vedere [CancelSchemaExtension](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-DSCertificate

Il seguente esempio di codice mostra come utilizzare. Unregister-DSCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina dal sistema il certificato registrato per una connessione LDAP protetta.

```
Unregister-DSCertificate -DirectoryId d-123456ijkl -CertificateId c-906731e34f
```

- Per i dettagli sull'API, vedere [DeregisterCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-DSEventTopic

Il seguente esempio di codice mostra come utilizzare. Unregister-DSEventTopic

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la directory specificata come publisher per l'argomento SNS specificato.

```
Unregister-DSEventTopic -DirectoryId d-123456ijkl -TopicName snstopicname
```

- Per i dettagli sull'API, vedere [DeregisterEventTopic](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-DSConditionalForwarder

Il seguente esempio di codice mostra come utilizzare. Update-DSConditionalForwarder

Strumenti per PowerShell V4

Esempio 1: Questo esempio aggiorna un server d'inoltro condizionale che è stato impostato per la directory. AWS

```
Update-DSConditionalForwarder -DirectoryId d-123456ijkl -DnsIpAddress 172.31.36.96,172.31.16.108 -RemoteDomainName contoso.com
```

- Per i dettagli sull'API, vedere [UpdateConditionalForwarder](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-DSRadius

Il seguente esempio di codice mostra come utilizzare. Update-DSRadius

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna le informazioni del server RADIUS per una directory AD Connector o Microsoft AD.

```
Update-DSRadius -DirectoryId d-123456ijkl -RadiusSettings_RadiusRetry 3
```

- Per i dettagli sull'API, vedere [UpdateRadius](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-DSTrust

Il seguente esempio di codice mostra come utilizzare. Update-DSTrust

Strumenti per PowerShell V4

Esempio 1: Questo esempio aggiorna il SelectiveAuth parametro del trust-id specificato da Disabled a Enabled.

```
Update-DSTrust -TrustId t-9067157123 -SelectiveAuth Enabled
```

Output:

RequestId	TrustId
-----	-----
138864a7-c9a8-4ad1-a828-eae479e85b45	t-9067157123

- Per i dettagli sull'API, vedere [UpdateTrust](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

AWS DMS esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con AWS DMS.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

New-DMSReplicationTask

Il seguente esempio di codice mostra come utilizzare `New-DMSReplicationTask`.

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova attività di replica del AWS Database Migration Service che utilizza `CdcStartTime` invece di `CdcStartPosition`. `MigrationType` è impostato su "full-load-and-cdc", il che significa che la tabella di destinazione deve essere vuota. La nuova attività è taggata con un tag la cui chiave è `Stage` e il cui valore è `Test`. Per ulteriori informazioni sui valori utilizzati da questo cmdlet, vedere `Creating a Task` (https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Tasks.Creating.html) nella Database Migration Service User Guide. AWS

```
New-DMSReplicationTask -ReplicationInstanceArn "arn:aws:dms:us-east-1:123456789012:rep:EXAMPLE66XFJUWATDJGBEXAMPLE" `
  -CdcStartTime "2019-08-08T12:12:12" `
  -CdcStopPosition "server_time:2019-08-09T12:12:12" `
  -MigrationType "full-load-and-cdc" `
  -ReplicationTaskIdentifier "task1" `
  -ReplicationTaskSetting "" `
  -SourceEndpointArn "arn:aws:dms:us-east-1:123456789012:endpoint:EXAMPLEW5UANC7Y3P4EEXAMPLE" `
  -TableMapping "file:///home/testuser/table-mappings.json"
```



```
-Tag @{"Key"="Stage";"Value"="Test"}`
-TargetEndpointArn "arn:aws:dms:us-
east-1:123456789012:endpoint:EXAMPLEJZASXWHTWCLNEXAMPLE"
```

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4). [CreateReplicationTask](#) AWS Strumenti per PowerShell

Esempi di DynamoDB con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con DynamoDB.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-DDBIndexSchema

Il seguente esempio di codice mostra come utilizzare. Add-DDBIndexSchema

Strumenti per PowerShell V4

Esempio 1: crea un TableSchema oggetto vuoto e vi aggiunge una nuova definizione di indice secondario locale prima di scrivere l' TableSchema oggetto nella pipeline.

```
$schema | Add-DDBIndexSchema -IndexName "LastPostIndex" -RangeKeyName
"LastPostDateTime" -RangeKeyDataType "S" -ProjectionType "keys_only"
$schema = New-DDBTableSchema
```

Output:

AttributeSchema	KeySchema
LocalSecondaryIndexSchema	
-----	-----

{LastPostDateTime}	{}
{LastPostIndex}	

Esempio 2: aggiunge una nuova definizione di indice secondario locale all' TableSchema oggetto fornito prima di riscrivere l' TableSchema oggetto nella pipeline. L' TableSchema oggetto può essere fornito anche utilizzando il parametro -Schema.

```
New-DDBTableSchema | Add-DDBIndexSchema -IndexName "LastPostIndex" -RangeKeyName
"LastPostDateTime" -RangeKeyDataType "S" -ProjectionType "keys_only"
```

Output:

AttributeSchema	KeySchema
LocalSecondaryIndexSchema	
-----	-----

{LastPostDateTime}	{}
{LastPostIndex}	

- Per i dettagli sull'API, vedere [Add- DDBIndex Schema](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-DDBKeySchema

Il seguente esempio di codice mostra come utilizzare. Add-DDBKeySchema

Strumenti per PowerShell V4

Esempio 1: crea un TableSchema oggetto vuoto e vi aggiunge voci di definizione di chiavi e attributi utilizzando i dati chiave specificati prima di scrivere l' TableSchema oggetto nella pipeline. Il tipo di chiave è dichiarato «HASH» per impostazione predefinita; utilizzate il KeyType parametro - con il valore «RANGE» per dichiarare una chiave di intervallo.

```
$schema = New-DDBTableSchema
$schema | Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S"
```

Output:

AttributeSchema	KeySchema
LocalSecondaryIndexSchema	
-----	-----

{ForumName}	{ForumName}
{}	

Esempio 2: aggiunge nuove voci di definizione di chiavi e attributi all' TableSchema oggetto fornito prima di scrivere l' TableSchema oggetto nella pipeline. Il tipo di chiave è dichiarato «HASH» per impostazione predefinita; utilizzate il KeyType parametro - con il valore «RANGE» per dichiarare una chiave di intervallo. L' TableSchema oggetto può essere fornito anche utilizzando il parametro -Schema.

```
New-DDBTableSchema | Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S"
```

Output:

AttributeSchema	KeySchema
LocalSecondaryIndexSchema	
-----	-----

{ForumName}	{ForumName}
{}	

- Per i dettagli sull'API, vedere [Add- DDBKey Schema](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

ConvertFrom-DDBItem

Il seguente esempio di codice mostra come utilizzare. ConvertFrom-DDBItem

Strumenti per PowerShell V4

Esempio 1: ConvertFrom - DDBItem viene utilizzato per convertire il risultato di Get-DDBItem da una tabella hash di AttributeValues DynamoDB a una tabella hash di tipi comuni come string e double.

```
@{
```

```

    SongTitle = 'Somewhere Down The Road'
    Artist     = 'No One You Know'
} | ConvertTo-DDBItem

Get-DDBItem -TableName 'Music' -Key $key | ConvertFrom-DDBItem

```

Output:

Name	Value
----	-----
Genre	Country
Artist	No One You Know
Price	1.94
CriticRating	9
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous

- Per i dettagli sull'API, vedere [ConvertFrom- DDBItem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

ConvertTo-DDBItem

Il seguente esempio di codice mostra come utilizzare. ConvertTo-DDBItem

Strumenti per PowerShell V4

Esempio 1: un esempio di conversione di una tabella hash in un dizionario di valori di attributi DynamoDB.

```

@{
    SongTitle = 'Somewhere Down The Road'
    Artist     = 'No One You Know'
} | ConvertTo-DDBItem

Key      Value
---      -
SongTitle Amazon.DynamoDBv2.Model.AttributeValue
Artist    Amazon.DynamoDBv2.Model.AttributeValue

```

Esempio 2: un esempio di conversione di una tabella hash in un dizionario di valori di attributi DynamoDB.

```
@{
    MyMap          = @{
        MyString = 'my string'
    }
    MyStringSet    = [System.Collections.Generic.HashSet[String]]@('my', 'string')
    MyNumericSet   = [System.Collections.Generic.HashSet[Int]]@(1, 2, 3)
    MyBinarySet    = [System.Collections.Generic.HashSet[System.IO.MemoryStream]]@(
        ([IO.MemoryStream]::new([Text.Encoding]::UTF8.GetBytes('my'))),
        ([IO.MemoryStream]::new([Text.Encoding]::UTF8.GetBytes('string'))))
    )
    MyList1        = @('my', 'string')
    MyList2        = [System.Collections.Generic.List[Int]]@(1, 2)
    MyList3        = [System.Collections.ArrayList]@('one', 2, $true)
} | ConvertTo-DDBItem
```

Output:

Key	Value
---	----
MyStringSet	Amazon.DynamoDBv2.Model.AttributeValue
MyList1	Amazon.DynamoDBv2.Model.AttributeValue
MyNumericSet	Amazon.DynamoDBv2.Model.AttributeValue
MyList2	Amazon.DynamoDBv2.Model.AttributeValue
MyBinarySet	Amazon.DynamoDBv2.Model.AttributeValue
MyMap	Amazon.DynamoDBv2.Model.AttributeValue
MyList3	Amazon.DynamoDBv2.Model.AttributeValue

- Per i dettagli sull'API, vedere [ConvertTo- DDBItem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DDBBatchItem

Il seguente esempio di codice mostra come utilizzare. Get-DDBBatchItem

Strumenti per PowerShell V4

Esempio 1: ottiene l'elemento con SongTitle «Somewhere Down The Road» dalle tabelle «Music» e «Songs» di DynamoDB.

```
$key = @{
    SongTitle = 'Somewhere Down The Road'
```

```

    Artist = 'No One You Know'
} | ConvertTo-DDBItem

$keysAndAttributes = New-Object Amazon.DynamoDBv2.Model.KeysAndAttributes
$list = New-Object
'System.Collections.Generic.List[System.Collections.Generic.Dictionary[String,
Amazon.DynamoDBv2.Model.AttributeValue]]'
$list.Add($key)
$keysAndAttributes.Keys = $list

$requestItem = @{
    'Music' = [Amazon.DynamoDBv2.Model.KeysAndAttributes]$keysAndAttributes
    'Songs' = [Amazon.DynamoDBv2.Model.KeysAndAttributes]$keysAndAttributes
}

$batchItems = Get-DDBBatchItem -RequestItem $requestItem
$batchItems.GetEnumerator() | ForEach-Object {$PSItem.Value} | ConvertFrom-DDBItem

```

Output:

Name	Value
----	-----
Artist	No One You Know
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous
CriticRating	10
Genre	Country
Price	1.94
Artist	No One You Know
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous
CriticRating	10
Genre	Country
Price	1.94

- Per i dettagli sull'API, vedere [BatchGetItem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DDBItem

Il seguente esempio di codice mostra come utilizzare. Get-DDBItem

Strumenti per PowerShell V4

Esempio 1: restituisce l'elemento DynamoDB con la chiave di partizione e la SongTitle chiave di ordinamento Artist.

```
$key = @{  
    SongTitle = 'Somewhere Down The Road'  
    Artist = 'No One You Know'  
} | ConvertTo-DDBItem  
  
Get-DDBItem -TableName 'Music' -Key $key | ConvertFrom-DDBItem
```

Output:

Name	Value
----	-----
Genre	Country
SongTitle	Somewhere Down The Road
Price	1.94
Artist	No One You Know
CriticRating	9
AlbumTitle	Somewhat Famous

- Per i dettagli sull'API, vedere [GetItem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DDBTable

Il seguente esempio di codice mostra come utilizzare. Get-DDBTable

Strumenti per PowerShell V4

Esempio 1: restituisce i dettagli della tabella specificata.

```
Get-DDBTable -TableName "myTable"
```

- Per i dettagli sull'API, vedere [DescribeTable](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-DDBTableList

Il seguente esempio di codice mostra come utilizzare. Get-DDBTableList

Strumenti per PowerShell V4

Esempio 1: restituisce i dettagli di tutte le tabelle, eseguendo un'iterazione automatica finché il servizio non indica che non esistono altre tabelle.

```
Get-DDBTableList
```

- Per i dettagli sull'API, vedere [ListTables](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Invoke-DDBQuery

Il seguente esempio di codice mostra come utilizzare. Invoke-DDBQuery

Strumenti per PowerShell V4

Esempio 1: richiama una query che restituisce elementi DynamoDB con l'elemento Artist specificato. SongTitle

```
$invokeDDBQuery = @{
    TableName = 'Music'
    KeyConditionExpression = ' SongTitle = :SongTitle and Artist = :Artist'
    ExpressionAttributeValues = @{
        ':SongTitle' = 'Somewhere Down The Road'
        ':Artist' = 'No One You Know'
    } | ConvertTo-DDBItem
}
Invoke-DDBQuery @invokeDDBQuery | ConvertFrom-DDBItem
```

Output:

Name	Value
----	-----
Genre	Country
Artist	No One You Know
Price	1.94
CriticRating	9
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous

- Per informazioni dettagliate sull'API, consulta [Query](#) nella documentazione di riferimento dei cmdlet di AWS Strumenti per PowerShell (V4).

Invoke-DDBScan

Il seguente esempio di codice mostra come utilizzare. Invoke-DDBScan

Strumenti per PowerShell V4

Esempio 1: restituisce tutti gli elementi nella tabella Music.

```
Invoke-DDBScan -TableName 'Music' | ConvertFrom-DDBItem
```

Output:

Name	Value
----	----
Genre	Country
Artist	No One You Know
Price	1.94
CriticRating	9
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous
Genre	Country
Artist	No One You Know
Price	1.98
CriticRating	8.4
SongTitle	My Dog Spot
AlbumTitle	Hey Now

Esempio 2: restituisce gli elementi nella tabella Music con un CriticRating valore maggiore o uguale a nove.

```
$scanFilter = @{
    CriticRating = [Amazon.DynamoDBv2.Model.Condition]@{
        AttributeValueList = @(@{N = '9'})
        ComparisonOperator = 'GE'
    }
}
Invoke-DDBScan -TableName 'Music' -ScanFilter $scanFilter | ConvertFrom-DDBItem
```

Output:

Name	Value
----	-----
Genre	Country
Artist	No One You Know
Price	1.94
CriticRating	9
SongTitle	Somewhere Down The Road
AlbumTitle	Somewhat Famous

- Per informazioni dettagliate sull'API, consulta [Scan](#) nella documentazione di riferimento dei cmdlet di AWS Strumenti per PowerShell (V4).

New-DDBTable

Il seguente esempio di codice mostra come utilizzare `New-DDBTable`.

Strumenti per PowerShell V4

Esempio 1: Questo esempio crea una tabella denominata `Thread` con una chiave primaria composta da `'ForumName'` (hash del tipo di chiave) e `'Subject'` (intervallo dei tipi di chiave). Lo schema utilizzato per costruire la tabella può essere inserito in ogni cmdlet come mostrato o specificato mediante il parametro `-Schema`.

```
$schema = New-DDBTableSchema
$schema | Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S"
$schema | Add-DDBKeySchema -KeyName "Subject" -KeyType RANGE -KeyDataType "S"
$schema | New-DDBTable -TableName "Thread" -ReadCapacity 10 -WriteCapacity 5
```

Output:

```
AttributeDefinitions : {ForumName, Subject}
TableName            : Thread
KeySchema            : {ForumName, Subject}
TableStatus          : CREATING
CreationDateTime      : 10/28/2013 4:39:49 PM
ProvisionedThroughput : Amazon.DynamoDBv2.Model.ProvisionedThroughputDescription
TableSizeBytes       : 0
ItemCount            : 0
```

```
LocalSecondaryIndexes : {}
```

Esempio 2: Questo esempio crea una tabella denominata Thread con una chiave primaria composta da 'ForumName' (hash del tipo di chiave) e 'Subject' (intervallo dei tipi di chiave). Viene inoltre definito un indice secondario locale. La chiave dell'indice secondario locale verrà impostata automaticamente dalla chiave hash primaria sulla tabella ()ForumName. Lo schema utilizzato per costruire la tabella può essere inserito in ogni cmdlet come mostrato o specificato mediante il parametro -Schema.

```
$schema = New-DDBTableSchema
$schema | Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S"
$schema | Add-DDBKeySchema -KeyName "Subject" -KeyDataType "S"
$schema | Add-DDBIndexSchema -IndexName "LastPostIndex" -RangeKeyName
    "LastPostDateTime" -RangeKeyDataType "S" -ProjectionType "keys_only"
$schema | New-DDBTable -TableName "Thread" -ReadCapacity 10 -WriteCapacity 5
```

Output:

```
AttributeDefinitions : {ForumName, LastPostDateTime, Subject}
TableName             : Thread
KeySchema              : {ForumName, Subject}
TableStatus           : CREATING
CreationDateTime       : 10/28/2013 4:39:49 PM
ProvisionedThroughput : Amazon.DynamoDBv2.Model.ProvisionedThroughputDescription
TableSizeBytes        : 0
ItemCount             : 0
LocalSecondaryIndexes : {LastPostIndex}
```

Esempio 3: Questo esempio mostra come utilizzare una singola pipeline per creare una tabella denominata Thread con una chiave primaria composta da " ForumName (hash del tipo di chiave) e 'Subject' (intervallo dei tipi di chiave) e un indice secondario locale. Add- DDBKey Schema e Add- DDBIndex Schema creano automaticamente un nuovo TableSchema oggetto se uno non viene fornito dalla pipeline o dal parametro -Schema.

```
New-DDBTableSchema |
    Add-DDBKeySchema -KeyName "ForumName" -KeyDataType "S" |
    Add-DDBKeySchema -KeyName "Subject" -KeyDataType "S" |
    Add-DDBIndexSchema -IndexName "LastPostIndex" `
        -RangeKeyName "LastPostDateTime" `
        -RangeKeyDataType "S" `
        -ProjectionType "keys_only" |
```

```
New-DDBTable -TableName "Thread" -ReadCapacity 10 -WriteCapacity 5
```

Output:

```
AttributeDefinitions : {ForumName, LastPostDateTime, Subject}
TableName            : Thread
KeySchema            : {ForumName, Subject}
TableStatus          : CREATING
CreationDateTime      : 10/28/2013 4:39:49 PM
ProvisionedThroughput : Amazon.DynamoDBv2.Model.ProvisionedThroughputDescription
TableSizeBytes        : 0
ItemCount            : 0
LocalSecondaryIndexes : {LastPostIndex}
```

- Per i dettagli sull'API, vedere [CreateTable](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-DDBTableSchema

Il seguente esempio di codice mostra come utilizzare. New-DDBTableSchema

Strumenti per PowerShell V4

Esempio 1: crea un TableSchema oggetto vuoto pronto ad accettare definizioni di chiavi e indici da utilizzare nella creazione di una nuova tabella Amazon DynamoDB. L'oggetto restituito può essere reindirizzato ai cmdlet Add- DDBKey Schema, Add- DDBIndex Schema e New- o passato a essi utilizzando il parametro -Schema su ogni DDBTable cmdlet.

```
New-DDBTableSchema
```

Output:

AttributeSchema	KeySchema
LocalSecondaryIndexSchema	
-----	-----

{}	{}
{}	

- Per i dettagli sull'API, vedere [New- DDBTable Schema](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-DDBItem

Il seguente esempio di codice mostra come utilizzare. Remove-DDBItem

Strumenti per PowerShell V4

Esempio 1: rimuove l'elemento DynamoDB corrispondente alla chiave specificata.

```
$key = @{  
    SongTitle = 'Somewhere Down The Road'  
    Artist = 'No One You Know'  
} | ConvertTo-DDBItem  
Remove-DDBItem -TableName 'Music' -Key $key -Confirm:$false
```

- Per i dettagli sull'API, vedere [DeleteItem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-DDBTable

Il seguente esempio di codice mostra come utilizzare. Remove-DDBTable

Strumenti per PowerShell V4

Esempio 1: elimina la tabella specificata. Prima di procedere con l'operazione, viene richiesta una conferma.

```
Remove-DDBTable -TableName "myTable"
```

Esempio 2: elimina la tabella specificata. Prima di procedere con l'operazione, non viene richiesta una conferma.

```
Remove-DDBTable -TableName "myTable" -Force
```

- Per i dettagli sull'API, vedere [DeleteTable](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-DDBBatchItem

Il seguente esempio di codice mostra come utilizzare. Set-DDBBatchItem

Strumenti per PowerShell V4

Esempio 1: crea un nuovo elemento o sostituisce un elemento esistente con un nuovo elemento nelle tabelle DynamoDB Music e Songs.

```
$item = @{
    SongTitle = 'Somewhere Down The Road'
    Artist = 'No One You Know'
    AlbumTitle = 'Somewhat Famous'
    Price = 1.94
    Genre = 'Country'
    CriticRating = 10.0
} | ConvertTo-DDBItem

$writeRequest = New-Object Amazon.DynamoDBv2.Model.WriteRequest
$writeRequest.PutRequest = [Amazon.DynamoDBv2.Model.PutRequest]$item
```

Output:

```
$requestItem = @{
    'Music' = [Amazon.DynamoDBv2.Model.WriteRequest]($writeRequest)
    'Songs' = [Amazon.DynamoDBv2.Model.WriteRequest]($writeRequest)
}

Set-DDBBatchItem -RequestItem $requestItem
```

- Per i dettagli sull'API, vedere [BatchWriteItem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-DDBItem

Il seguente esempio di codice mostra come utilizzare. Set-DDBItem

Strumenti per PowerShell V4

Esempio 1: crea un nuovo elemento o sostituisce un elemento esistente con uno nuovo.

```
$item = @{
    SongTitle = 'Somewhere Down The Road'
    Artist = 'No One You Know'
    AlbumTitle = 'Somewhat Famous'
```

```

        Price = 1.94
        Genre = 'Country'
        CriticRating = 9.0
    } | ConvertTo-DDBItem
Set-DDBItem -TableName 'Music' -Item $item

```

- Per i dettagli sull'API, vedere [PutItem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-DDBItem

Il seguente esempio di codice mostra come utilizzare. Update-DDBItem

Strumenti per PowerShell V4

Esempio 1: imposta l'attributo genere su 'Rap' sull'elemento DynamoDB con la chiave di partizione e la SongTitle chiave di ordinamento Artist.

```

$key = @{
    SongTitle = 'Somewhere Down The Road'
    Artist = 'No One You Know'
} | ConvertTo-DDBItem

$updateDdbItem = @{
    TableName = 'Music'
    Key = $key
    UpdateExpression = 'set Genre = :val1'
    ExpressionAttributeValue = (@{
        ':val1' = ([Amazon.DynamoDBv2.Model.AttributeValue]'Rap')
    })
}
Update-DDBItem @updateDdbItem

```

Output:

Name	Value
----	-----
Genre	Rap

- Per i dettagli sull'API, vedere [UpdateItem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-DDBTable

Il seguente esempio di codice mostra come utilizzare. Update-DDBTable

Strumenti per PowerShell V4

Esempio 1: aggiorna il throughput con provisioning per la tabella specificata.

```
Update-DDBTable -TableName "myTable" -ReadCapacity 10 -WriteCapacity 5
```

- Per i dettagli sull'API, vedere [UpdateTable](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

EC2 Esempi di Amazon con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon EC2.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-EC2CapacityReservation

Il seguente esempio di codice mostra come usare Add-EC2CapacityReservation.

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova prenotazione della capacità con gli attributi specificati

```
Add-EC2CapacityReservation -InstanceType m4.xlarge -InstanceCount 2 -  
AvailabilityZone eu-west-1b -EbsOptimized True -InstancePlatform Windows
```


Output:

```
AvailabilityZone      : eu-west-1b
AvailableInstanceCount : 2
CapacityReservationId : cr-0c1f2345db6f7cdba
CreateDate            : 3/28/2019 9:29:41 AM
EbsOptimized          : True
EndDate               : 1/1/0001 12:00:00 AM
EndDateType           : unlimited
EphemeralStorage      : False
InstanceMatchCriteria : open
InstancePlatform      : Windows
InstanceType          : m4.xlarge
State                 : active
Tags                  : {}
Tenancy                : default
TotalInstanceCount    : 2
```

- Per i dettagli sull'API, vedere [CreateCapacityReservation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-EC2InternetGateway

Il seguente esempio di codice mostra come utilizzare. Add-EC2InternetGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio collega il gateway Internet specificato al VPC specificato.

```
Add-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d -VpcId vpc-12345678
```

Esempio 2: questo esempio crea un VPC e un gateway Internet, quindi collega il gateway Internet al VPC.

```
$vpc = New-EC2Vpc -CidrBlock 10.0.0.0/16
New-EC2InternetGateway | Add-EC2InternetGateway -VpcId $vpc.VpcId
```

- Per i dettagli sull'API, vedere [AttachInternetGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-EC2NetworkInterface

Il seguente esempio di codice mostra come utilizzare Add-EC2NetworkInterface

Strumenti per PowerShell V4

Esempio 1: questo esempio collega l'interfaccia di rete specificata all'istanza specificata.

```
Add-EC2NetworkInterface -NetworkInterfaceId eni-12345678 -InstanceId i-1a2b3c4d -
DeviceIndex 1
```

Output:

```
eni-attach-1a2b3c4d
```

- Per i dettagli sull'API, vedere [AttachNetworkInterface](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-EC2Volume

Il seguente esempio di codice mostra come utilizzare Add-EC2Volume

Strumenti per PowerShell V4

Esempio 1: questo esempio collega il volume specificato all'istanza specificata e lo espone con il nome di dispositivo specificato.

```
Add-EC2Volume -VolumeId vol-12345678 -InstanceId i-1a2b3c4d -Device /dev/sdh
```

Output:

```
AttachTime       : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device           : /dev/sdh
InstanceId       : i-1a2b3c4d
State            : attaching
VolumeId        : vol-12345678
```

- Per i dettagli sull'API, vedere [AttachVolume](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-EC2VpnGateway

Il seguente esempio di codice mostra come utilizzare. Add-EC2VpnGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio collega il gateway privato virtuale specificato al VPC specificato.

```
Add-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d -VpcId vpc-12345678
```

Output:

State	VpcId
-----	-----
attaching	vpc-12345678

- Per i dettagli sull'API, vedere [AttachVpnGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Approve-EC2VpcPeeringConnection

Il seguente esempio di codice mostra come utilizzare. Approve-EC2VpcPeeringConnection

Strumenti per PowerShell V4

Esempio 1: Questo esempio approva il pcx-1dfad234b56ff78be richiesto VpcPeeringConnectionId

```
Approve-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-1dfad234b56ff78be
```

Output:

AccepterVpcInfo	: Amazon.EC2.Model.VpcPeeringConnectionVpcInfo
ExpirationTime	: 1/1/0001 12:00:00 AM
RequesterVpcInfo	: Amazon.EC2.Model.VpcPeeringConnectionVpcInfo
Status	: Amazon.EC2.Model.VpcPeeringConnectionStateReason
Tags	: {}
VpcPeeringConnectionId	: pcx-1dfad234b56ff78be

- Per i dettagli AWS Strumenti per PowerShell sull'[AcceptVpcPeeringConnection](#) API, vedere in Cmdlet Reference (V4).

Confirm-EC2ProductInstance

Il seguente esempio di codice mostra come utilizzare. Confirm-EC2ProductInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio determina se il codice prodotto specificato è associato all'istanza specificata.

```
Confirm-EC2ProductInstance -ProductCode 774F4FF8 -InstanceId i-12345678
```

- Per i dettagli sull'API, vedere [ConfirmProductInstance](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Copy-EC2Image

Il seguente esempio di codice mostra come utilizzare. Copy-EC2Image

Strumenti per PowerShell V4

Esempio 1: questo esempio copia l'AMI specificato nella Regione "Europa (Irlanda)" nella Regione "Stati Uniti occidentali (Oregon)". Se il parametro -Region non è specificato, come Regione di destinazione viene utilizzata la Regione predefinita corrente.

```
Copy-EC2Image -SourceRegion eu-west-1 -SourceImageId ami-12345678 -Region us-west-2  
-Name "Copy of ami-12345678"
```

Output:

```
ami-87654321
```

- Per i dettagli sull'API, vedere [CopyImage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Copy-EC2Snapshot

Il seguente esempio di codice mostra come utilizzare. Copy-EC2Snapshot

Strumenti per PowerShell V4

Esempio 1: questo esempio copia lo snapshot specificato dalla Regione “Europa (Irlanda)” alla Regione “Stati Uniti occidentali (Oregon)”.

```
Copy-EC2Snapshot -SourceRegion eu-west-1 -SourceSnapshotId snap-12345678 -Region us-west-2
```

Esempio 2: se si imposta una Regione predefinita e si omette il parametro Region, la Regione di destinazione predefinita è la Regione predefinita.

```
Set-DefaultAWSRegion us-west-2  
Copy-EC2Snapshot -SourceRegion eu-west-1 -SourceSnapshotId snap-12345678
```

- Per i dettagli sull'API, vedere [CopySnapshot](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Deny-EC2VpcPeeringConnection

Il seguente esempio di codice mostra come utilizzare Deny-EC2VpcPeeringConnection

Strumenti per PowerShell V4

Esempio 1: L'esempio precedente nega la richiesta per l'id della richiesta VpcPeering pcx-01a2b3ce45fe67eb8

```
Deny-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-01a2b3ce45fe67eb8
```

- Per i dettagli [RejectVpcPeeringConnection](#) sull'AWS Strumenti per PowerShell API, vedere in Cmdlet Reference (V4).

Disable-EC2VgwRoutePropagation

Il seguente esempio di codice mostra come utilizzare Disable-EC2VgwRoutePropagation

Strumenti per PowerShell V4

Esempio 1: questo esempio disabilita il gateway privato virtuale (VGW) per evitare la propagazione automatica delle route alla tabella di routing specificata.

```
Disable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- Per i dettagli sull'API, vedere [DisableVgwRoutePropagation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Disable-EC2VpcClassicLink

Il seguente esempio di codice mostra come utilizzare. `Disable-EC2VpcClassicLink`

Strumenti per PowerShell V4

Esempio 1: Questo esempio disabilita EC2 VpcClassicLink per vpc-01e23c4a5d6db78e9. Restituisce True o False

```
Disable-EC2VpcClassicLink -VpcId vpc-01e23c4a5d6db78e9
```

- AWS Strumenti per PowerShell Per i dettagli sull'[DisableVpcClassicLink](#) API, vedere in Cmdlet Reference (V4).

Disable-EC2VpcClassicLinkDnsSupport

Il seguente esempio di codice mostra come utilizzare. `Disable-EC2VpcClassicLinkDnsSupport`

Strumenti per PowerShell V4

Esempio 1: Questo esempio disabilita il supporto ClassicLink DNS per vpc-0b12d3456a7e8910d

```
Disable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d
```

- Per i dettagli sull'[DisableVpcClassicLinkDnsSupport](#) API, AWS Strumenti per PowerShell vedere in Cmdlet Reference (V4).

Dismount-EC2InternetGateway

Il seguente esempio di codice mostra come utilizzare. `Dismount-EC2InternetGateway`

Strumenti per PowerShell V4

Esempio 1: questo esempio scollega il gateway Internet specificato dal VPC specificato.

```
Dismount-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d -VpcId vpc-12345678
```

- Per i dettagli sull'API, vedere [DetachInternetGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Dismount-EC2NetworkInterface

Il seguente esempio di codice mostra come utilizzare Dismount-EC2NetworkInterface

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il collegamento specificato tra un'interfaccia di rete e un'istanza.

```
Dismount-EC2NetworkInterface -AttachmentId eni-attach-1a2b3c4d -Force
```

- Per i dettagli sull'API, vedere [DetachNetworkInterface](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Dismount-EC2Volume

Il seguente esempio di codice mostra come utilizzare Dismount-EC2Volume

Strumenti per PowerShell V4

Esempio 1: questo esempio scollega il volume specificato.

```
Dismount-EC2Volume -VolumeId vol-12345678
```

Output:

```
AttachTime       : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device           : /dev/sdh
InstanceId       : i-1a2b3c4d
State            : detaching
VolumeId        : vol-12345678
```

Esempio 2: è anche possibile specificare l'ID dell'istanza e il nome del dispositivo per scollegare il volume corretto.

```
Dismount-EC2Volume -VolumeId vol-12345678 -InstanceId i-1a2b3c4d -Device /dev/sdh
```

- Per i dettagli sull'API, vedere [DetachVolume](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Dismount-EC2VpnGateway

Il seguente esempio di codice mostra come utilizzare. Dismount-EC2VpnGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio scollega il gateway privato virtuale specificato dal VPC specificato.

```
Dismount-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d -VpcId vpc-12345678
```

- Per i dettagli sull'API, vedere [DetachVpnGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2CapacityReservation

Il seguente esempio di codice mostra come utilizzare. Edit-EC2CapacityReservation

Strumenti per PowerShell V4

Esempio 1: Questo esempio modifica CapacityReservationId cr-0c1f2345db6f7cdba cambiando il conteggio delle istanze su 1

```
Edit-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba -  
InstanceCount 1
```

Output:

```
True
```

- Per [ModifyCapacityReservation](#) i dettagli AWS Strumenti per PowerShell sull'API, vedere in Cmdlet Reference (V4).

Edit-EC2Host

Il seguente esempio di codice mostra come utilizzare. Edit-EC2Host

Strumenti per PowerShell V4

Esempio 1: Questo esempio modifica le AutoPlacement impostazioni su off per l'host dedicato h-01e23f4cd567890f3

```
Edit-EC2Host -HostId h-03e09f8cd681609f3 -AutoPlacement off
```

Output:

Successful	Unsuccessful
-----	-----
{h-01e23f4cd567890f3}	{}

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4). [ModifyHosts](#) AWS Strumenti per PowerShell

Edit-EC2IdFormat

Il seguente esempio di codice mostra come utilizzare. Edit-EC2IdFormat

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita il formato di ID più lungo per il tipo di risorsa specificato.

```
Edit-EC2IdFormat -Resource instance -UseLongId $true
```

Esempio 2: questo esempio disabilita il formato di ID più lungo per il tipo di risorsa specificato.

```
Edit-EC2IdFormat -Resource instance -UseLongId $false
```

- Per i dettagli sull'API, vedere [ModifyIdFormat](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2ImageAttribute

Il seguente esempio di codice mostra come utilizzare. Edit-EC2ImageAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la descrizione dell'AMI specificata.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Description "New description"
```

Esempio 2: questo esempio rende l'AMI pubblico (ad esempio, in modo che chiunque Account AWS possa utilizzarlo).

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType add -UserGroup all
```

Esempio 3: questo esempio rende privata l'AMI (ad esempio, in modo che solo il proprietario possa utilizzarla).

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType remove -UserGroup all
```

Esempio 4: Questo esempio concede il permesso di avvio all'oggetto specificato Account AWS.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType add -UserId 111122223333
```

Esempio 5: Questo esempio rimuove l'autorizzazione di avvio da quella specificata Account AWS.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType remove -UserId 111122223333
```

- Per i dettagli sull'API, vedere [ModifyImageAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2InstanceAttribute

Il seguente esempio di codice mostra come utilizzare. Edit-EC2InstanceAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica il tipo di istanza dell'istanza specificata.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -InstanceType m3.medium
```

Esempio 2: Questo esempio abilita una rete avanzata per l'istanza specificata, specificando «simple» come valore del parametro di supporto della rete Single Root I/O Virtualization (SR-IOV), -.. SriovNetSupport

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SriovNetSupport "simple"
```

Esempio 3: questo esempio modifica i gruppi di sicurezza per l'istanza specificata. L'istanza deve trovarsi in un VPC. È necessario specificare l'ID (e non il nome) di ogni gruppo di sicurezza.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -Group @( "sg-12345678",  
"sg-45678901" )
```

Esempio 4: questo esempio abilita l'ottimizzazione EBS I/O per l'istanza specificata. Questa funzionalità non è disponibile con tutti i tipi di istanza. Si applicano altri costi di utilizzo quando si usa un'istanza ottimizzata per EBS.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -EbsOptimized $true
```

Esempio 5: questo esempio consente di source/destination verificare l'istanza specificata. Affinché un'istanza NAT esegua NAT, il valore deve essere “false”.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SourceDestCheck $true
```

Esempio 6: questo esempio disabilita la terminazione dell'istanza specificata.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -DisableApiTermination $true
```

Esempio 7: questo esempio modifica l'istanza specificata in modo che venga terminata quando viene avviato l'arresto dall'istanza.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -InstanceInitiatedShutdownBehavior  
terminate
```

- Per i dettagli sull'API, vedere [ModifyInstanceAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2InstanceCreditSpecification

Il seguente esempio di codice mostra come utilizzare. Edit-EC2InstanceCreditSpecification

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita crediti per istanze T2 in modalità illimitata per l'istanza i-01234567890abcdef.

```
$Credit = New-Object -TypeName Amazon.EC2.Model.InstanceCreditSpecificationRequest
$Credit.InstanceId = "i-01234567890abcdef"
$Credit.CpuCredits = "unlimited"
Edit-EC2InstanceCreditSpecification -InstanceCreditSpecification $Credit
```

- Per i dettagli sull'API, vedere [ModifyInstanceCreditSpecification](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2NetworkInterfaceAttribute

Il seguente esempio di codice mostra come utilizzare. Edit-EC2NetworkInterfaceAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica l'interfaccia di rete specificata in modo che l'allegato specificato venga eliminato al terminazione dell'istanza.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
Attachment_AttachmentId eni-attach-1a2b3c4d -Attachment_DeleteOnTermination $true
```

Esempio 2: questo esempio modifica la descrizione dell'interfaccia di rete specificata.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Description "my
description"
```

Esempio 3: questo esempio modifica il gruppo di sicurezza per l'interfaccia di rete specificata.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Groups
sg-1a2b3c4d
```

Esempio 4: Questo esempio disabilita il source/destination controllo dell'interfaccia di rete specificata.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -SourceDestCheck $false
```

- Per i dettagli sull'API, vedere [ModifyNetworkInterfaceAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2ReservedInstance

Il seguente esempio di codice mostra come utilizzare. Edit-EC2ReservedInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica la zona di disponibilità, il numero di istanze e la piattaforma per le istanze riservate specificate.

```
$config = New-Object Amazon.EC2.Model.ReservedInstancesConfiguration
$config.AvailabilityZone = "us-west-2a"
$config.InstanceCount = 1
$config.Platform = "EC2-VPC"

Edit-EC2ReservedInstance `
-ReservedInstancesId @"FE32132D-70D5-4795-B400-AE435EXAMPLE", "0CC556F3-7AB8-4C00-B0E5-98666EXAMPLE" `
-TargetConfiguration $config
```

- Per i dettagli sull'API, vedere [ModifyReservedInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2SnapshotAttribute

Il seguente esempio di codice mostra come utilizzare. Edit-EC2SnapshotAttribute

Strumenti per PowerShell V4

Esempio 1: Questo esempio rende pubblica l'istantanea specificata impostandone CreateVolumePermission l'attributo.

```
Edit-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute CreateVolumePermission -OperationType Add -GroupName all
```

- Per i dettagli sull'API, vedere [ModifySnapshotAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2SpotFleetRequest

Il seguente esempio di codice mostra come utilizzare. Edit-EC2SpotFleetRequest

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la capacità di destinazione della richiesta del parco istanze spot specificata.

```
Edit-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE -TargetCapacity 10
```

Output:

```
True
```

- Per i dettagli sull'API, vedere [ModifySpotFleetRequest](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2SubnetAttribute

Il seguente esempio di codice mostra come utilizzare. Edit-EC2SubnetAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita gli indirizzi IP pubblici per la sottorete specificata.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $true
```

Esempio 2: questo esempio disabilita gli indirizzi IP pubblici per la sottorete specificata.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $false
```

- Per i dettagli sull'API, vedere [ModifySubnetAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2VolumeAttribute

Il seguente esempio di codice mostra come utilizzare. Edit-EC2VolumeAttribute

Strumenti per PowerShell V4

Esempio 1: Questo esempio modifica l'attributo specificato del volume specificato. I/O le operazioni per il volume vengono riprese automaticamente dopo essere state sospese a causa di dati potenzialmente incoerenti.

```
Edit-EC2VolumeAttribute -VolumeId vol-12345678 -AutoEnableIO $true
```

- Per i dettagli sull'API, vedere [ModifyVolumeAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-EC2VpcAttribute

Il seguente esempio di codice mostra come utilizzare. Edit-EC2VpcAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita il supporto dei nomi host DNS per il VPC specificato.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $true
```

Esempio 2: questo esempio disabilita il supporto dei nomi host DNS per il VPC specificato.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $false
```

Esempio 3: questo esempio abilita il supporto della risoluzione DNS per il VPC specificato.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $true
```

Esempio 4: questo esempio disabilita il supporto della risoluzione DNS per il VPC specificato.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $false
```

- Per i dettagli sull'API, vedere [ModifyVpcAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-EC2VgwRoutePropagation

Il seguente esempio di codice mostra come utilizzare. Enable-EC2VgwRoutePropagation

Strumenti per PowerShell V4

Esempio 1: questo esempio consente al gateway privato virtuale (VGW) specificato di propagare automaticamente le route alla tabella di routing specificata.

```
Enable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- Per i dettagli sull'API, vedere [EnableVgwRoutePropagation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-EC2VolumeIO

Il seguente esempio di codice mostra come utilizzare. Enable-EC2VolumeIO

Strumenti per PowerShell V4

Esempio 1: Questo esempio abilita I/O le operazioni per il volume specificato, se I/O le operazioni sono state disabilitate.

```
Enable-EC2VolumeIO -VolumeId vol-12345678
```

- Per i dettagli sull'API, vedere [EnableVolumeIO](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-EC2VpcClassicLink

Il seguente esempio di codice mostra come utilizzare. Enable-EC2VpcClassicLink

Strumenti per PowerShell V4

Esempio 1: Questo esempio abilita VPC vpc-0123456b789b0d12f per ClassicLink

```
Enable-EC2VpcClassicLink -VpcId vpc-0123456b789b0d12f
```

Output:


```
True
```

- Per i dettagli sull'[EnableVpcClassicLink](#) API, AWS Strumenti per PowerShell vedere in Cmdlet Reference (V4).

Enable-EC2VpcClassicLinkDnsSupport

Il seguente esempio di codice mostra come utilizzare. Enable-EC2VpcClassicLinkDnsSupport
Strumenti per PowerShell V4

Esempio 1: Questo esempio consente a vpc-0b12d3456a7e8910d di supportare la risoluzione dei nomi host DNS per ClassicLink

```
Enable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

- Per i dettagli [EnableVpcClassicLinkDnsSupport](#) sull'AWS Strumenti per PowerShell API, vedere in Cmdlet Reference (V4).

Get-EC2AccountAttribute

Il seguente esempio di codice mostra come utilizzare. Get-EC2AccountAttribute

Strumenti per PowerShell V4

Esempio 1: Questo esempio descrive se è possibile avviare istanze in EC2 -Classic e EC2 -VPC nella regione o solo in -VPC. EC2

```
(Get-EC2AccountAttribute -AttributeName supported-platforms).AttributeValues
```

Output:

```
AttributeValue
-----
EC2
VPC
```

Esempio 2: questo esempio descrive il VPC predefinito o restituisce “none” se non si dispone di un VPC predefinito nella Regione.

```
(Get-EC2AccountAttribute -AttributeName default-vpc).AttributeValues
```

Output:

```
AttributeValue
-----
vpc-12345678
```

Esempio 3: questo esempio descrive il numero massimo di istanze on demand che è possibile eseguire.

```
(Get-EC2AccountAttribute -AttributeName max-instances).AttributeValues
```

Output:

```
AttributeValue
-----
20
```

- Per i dettagli sull'API, vedere [DescribeAccountAttributes](#) in Cmdlet Reference (V4).AWS Strumenti per PowerShell

Get-EC2Address

Il seguente esempio di codice mostra come utilizzare. Get-EC2Address

Strumenti per PowerShell V4

Esempio 1: Questo esempio descrive l'indirizzo IP elastico specificato per le istanze in EC2 - Classic.

```
Get-EC2Address -AllocationId eipalloc-12345678
```

Output:

```
AllocationId      : eipalloc-12345678
AssociationId     : eipassoc-12345678
Domain           : vpc
InstanceId        : i-87654321
```

```
NetworkInterfaceId      : eni-12345678
NetworkInterfaceOwnerId : 12345678
PrivateIpAddress        : 10.0.2.172
PublicIp                : 198.51.100.2
```

Esempio 2: questo esempio descrive gli indirizzi IP elastici in uso per le istanze in un VPC. Questa sintassi richiede la PowerShell versione 3 o successiva.

```
Get-EC2Address -Filter @{ Name="domain";Values="vpc" }
```

Esempio 3: questo esempio descrive l'indirizzo IP elastico specificato per le istanze in EC2 - Classic.

```
Get-EC2Address -PublicIp 203.0.113.17
```

Output:

```
AllocationId      :
AssociationId      :
Domain            : standard
InstanceId         : i-12345678
NetworkInterfaceId :
NetworkInterfaceOwnerId :
PrivateIpAddress   :
PublicIp          : 203.0.113.17
```

Esempio 4: questo esempio descrive gli indirizzi IP elastici per le istanze in -Classic. EC2 Questa sintassi richiede la PowerShell versione 3 o successiva.

```
Get-EC2Address -Filter @{ Name="domain";Values="standard" }
```

Esempio 5: questo esempio descrive tutti gli indirizzi IP elastici.

```
Get-EC2Address
```

Esempio 6: questo esempio restituisce l'IP pubblico e privato per l'ID istanza specificata nel filtro

```
Get-EC2Address -Region eu-west-1 -Filter @{Name="instance-
id";Values="i-0c12d3f4f567ffb89"} | Select-Object PrivateIpAddress, PublicIp
```

Output:

PrivateIpAddress	PublicIp
-----	-----
10.0.0.99	63.36.5.227

Esempio 7: questo esempio recupera tutto l'Elastic IPs con il relativo ID di allocazione, l'ID di associazione e gli ID di istanza

```
Get-EC2Address -Region eu-west-1 | Select-Object InstanceId, AssociationId,
AllocationId, PublicIp
```

Output:

InstanceId	AssociationId	AllocationId	PublicIp
-----	-----	-----	-----
		eipalloc-012e3b456789e1fad	
17.212.120.178			
i-0c123dfd3415bac67	eipassoc-0e123456bb7890bdb	eipalloc-01cd23ebf45f7890c	
17.212.124.77			
		eipalloc-012345678eeabcfad	
17.212.225.7			
i-0123d405c67e89a0c	eipassoc-0c123b456783966ba	eipalloc-0123cdd456a8f7892	
37.216.52.173			
i-0f1bf2f34c5678d09	eipassoc-0e12934568a952d96	eipalloc-0e1c23e4d5e6789e4	
37.218.222.278			
i-012e3cb4df567e8aa	eipassoc-0d1b2fa4d67d03810	eipalloc-0123f456f78a01b58	
37.210.82.27			
i-0123bcf4b567890e1	eipassoc-01d2345f678903fb1	eipalloc-0e1db23cfef5c45c7	
37.215.222.270			

Esempio 8: Questo esempio recupera l'elenco di indirizzi EC2 IP che corrispondono alla chiave di tag «Category» con il valore «Prod»

```
Get-EC2Address -Filter @{Name="tag:Category";Values="Prod"}
```

Output:

AllocationId	: eipalloc-0123f456f81a01b58
AssociationId	: eipassoc-0d1b23a456d103810
CustomerOwnedIp	:

```

CustomerOwnedIpv4Pool    :
Domain                   : vpc
InstanceId               : i-012e3cb4df567e1aa
NetworkBorderGroup      : eu-west-1
NetworkInterfaceId      : eni-0123f41d5a60d5f40
NetworkInterfaceOwnerId : 123456789012
PrivateIpAddress        : 192.168.1.84
PublicIp                : 34.250.81.29
PublicIpv4Pool          : amazon
Tags                    : {Category, Name}

```

- Per i dettagli sull'API, vedere [DescribeAddresses](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2AvailabilityZone

Il seguente esempio di codice mostra come utilizzare `Get-EC2AvailabilityZone`

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le zone di disponibilità disponibili per la Regione corrente.

```
Get-EC2AvailabilityZone
```

Output:

Messages	RegionName	State	ZoneName
-----	-----	-----	-----
{}	us-west-2	available	us-west-2a
{}	us-west-2	available	us-west-2b
{}	us-west-2	available	us-west-2c

Esempio 2: questo esempio descrive tutte le zone di disponibilità il cui stato è "impaired". La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
Get-EC2AvailabilityZone -Filter @{ Name="state";Values="impaired" }
```

Esempio 3: con PowerShell la versione 2, è necessario utilizzare `New-Object` per creare il filtro.

```

$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"

```

```
$filter.Values = "impaired"

Get-EC2AvailabilityZone -Filter $filter
```

- Per i dettagli sull'API, vedere [DescribeAvailabilityZones](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2BundleTask

Il seguente esempio di codice mostra come utilizzare. Get-EC2BundleTask

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'attività di creazione di bundle specificata.

```
Get-EC2BundleTask -BundleId bun-12345678
```

Esempio 2: questo esempio descrive le attività di creazione di bundle il cui stato è “complete” o “failed”.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "complete", "failed" )

Get-EC2BundleTask -Filter $filter
```

- Per i dettagli sull'API, vedere [DescribeBundleTasks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2CapacityReservation

Il seguente esempio di codice mostra come utilizzare. Get-EC2CapacityReservation

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive una o più prenotazioni della capacità per la Regione

```
Get-EC2CapacityReservation -Region eu-west-1
```

Output:

```

AvailabilityZone      : eu-west-1b
AvailableInstanceCount : 2
CapacityReservationId : cr-0c1f2345db6f7cdba
CreateDate            : 3/28/2019 9:29:41 AM
EbsOptimized          : True
EndDate               : 1/1/0001 12:00:00 AM
EndDateType           : unlimited
EphemeralStorage      : False
InstanceMatchCriteria : open
InstancePlatform      : Windows
InstanceType          : m4.xlarge
State                 : active
Tags                  : {}
Tenancy               : default
TotalInstanceCount    : 2

```

- Per i dettagli sull'API, vedere [DescribeCapacityReservations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2ConsoleOutput

Il seguente esempio di codice mostra come utilizzare `Get-EC2ConsoleOutput`

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene l'output della console per l'istanza Linux specificata. L'output della console è codificato.

```
Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456
```

Output:

InstanceId	Output
-----	-----
i-0e194d3c47c123637	WyAgICAwLjAwMDAwMF0gQ29tbW...bGU9dHR5UzAgc2Vs

Esempio 2: questo esempio memorizza l'output codificato della console in una variabile e quindi lo decodifica.

```
$Output_encoded = (Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456).Output
```

```
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($Output_encoded))
```

- Per i dettagli sull'API, vedere [GetConsoleOutput](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2CustomerGateway

Il seguente esempio di codice mostra come utilizzare. Get-EC2CustomerGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il gateway del cliente specificato.

```
Get-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

Output:

```
BgpAsn           : 65534
CustomerGatewayId : cgw-1a2b3c4d
IpAddress        : 203.0.113.12
State            : available
Tags             : {}
Type             : ipsec.1
```

Esempio 2: questo esempio descrive qualsiasi gateway del cliente il cui stato è “pending” (In sospenso) o “available” (Disponibile).

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2CustomerGateway -Filter $filter
```

Esempio 3: questo esempio descrive tutti i gateway del cliente specificato.

```
Get-EC2CustomerGateway
```

- Per i dettagli sull'API, vedere [DescribeCustomerGateways](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2DhcpOption

Il seguente esempio di codice mostra come utilizzare. Get-EC2DhcpOption

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i set di opzioni DHCP.

```
Get-EC2DhcpOption
```

Output:

DhcpConfigurations	DhcpOptionsId	Tag
-----	-----	---
{domain-name, domain-name-servers}	dopt-1a2b3c4d	{}
{domain-name, domain-name-servers}	dopt-2a3b4c5d	{}
{domain-name-servers}	dopt-3a4b5c6d	{}

Esempio 2: questo esempio ottiene i dettagli di configurazione per il set di opzioni DHCP specificato.

```
(Get-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d).DhcpConfigurations
```

Output:

Key	Values
---	-----
domain-name	{abc.local}
domain-name-servers	{10.0.0.101, 10.0.0.102}

- Per i dettagli sull'API, vedere [DescribeDhcpOptions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2FlowLog

Il seguente esempio di codice mostra come utilizzare. Get-EC2FlowLog

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive uno o più log di flusso con il tipo di destinazione log "s3"

```
Get-EC2FlowLog -Filter @{"Name="log-destination-type";Values="s3"}
```

Output:

```
CreationTime      : 2/25/2019 9:07:36 PM
DeliverLogsErrorMessage :
DeliverLogsPermissionArn :
DeliverLogsStatus : SUCCESS
FlowLogId         : fl-01b2e3d45f67f8901
FlowLogStatus     : ACTIVE
LogDestination    : arn:aws:s3:::amzn-s3-demo-bucket-dd-tata
LogDestinationType : s3
LogGroupName      :
ResourceId        : eni-01d2dda3456b7e890
TrafficType       : ALL
```

- Per i dettagli sull'API, vedere [DescribeFlowLogs](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2Host

Il seguente esempio di codice mostra come utilizzare. Get-EC2Host

Strumenti per PowerShell V4

Esempio 1: Questo esempio restituisce i dettagli dell' EC2 host

```
Get-EC2Host
```

Output:

```
AllocationTime    : 3/23/2019 4:55:22 PM
AutoPlacement     : off
AvailabilityZone   : eu-west-1b
AvailableCapacity : Amazon.EC2.Model.AvailableCapacity
ClientToken       :
HostId            : h-01e23f4cd567890f1
HostProperties     : Amazon.EC2.Model.HostProperties
HostReservationId :
Instances         : {}
ReleaseTime       : 1/1/0001 12:00:00 AM
```

```
State           : available
Tags            : {}
```

Esempio 2: Questo esempio esegue una query `AvailableInstanceCapacity` per l'host `h-01e23f4cd567899f1`

```
Get-EC2Host -HostId h-01e23f4cd567899f1 | Select-Object -ExpandProperty
AvailableCapacity | Select-Object -expand AvailableInstanceCapacity
```

Output:

```
AvailableCapacity InstanceType TotalCapacity
-----
11                m4.xlarge      11
```

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4). [DescribeHosts](#) AWS Strumenti per PowerShell

Get-EC2HostReservationOffering

Il seguente esempio di codice mostra come utilizzare `Get-EC2HostReservationOffering`

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le prenotazioni di host dedicati disponibili per l'acquisto per il filtro specificato «instance-family» where is «» `PaymentOption NoUpfront`

```
Get-EC2HostReservationOffering -Filter @{Name="instance-family";Values="m4"} |
Where-Object PaymentOption -eq NoUpfront
```

Output:

```
CurrencyCode      :
Duration          : 94608000
HourlyPrice       : 1.307
InstanceFamily    : m4
OfferingId        : hro-0c1f234567890d9ab
PaymentOption     : NoUpfront
UpfrontPrice      : 0.000
```

```

CurrencyCode   :
Duration       : 31536000
HourlyPrice    : 1.830
InstanceFamily : m4
OfferingId     : hro-04ad12aaaf34b5a67
PaymentOption  : NoUpfront
UpfrontPrice   : 0.000

```

- Per i dettagli sull'API, vedere [DescribeHostReservationOfferings](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2HostReservationPurchasePreview

Il seguente esempio di codice mostra come utilizzare. Get-EC2HostReservationPurchasePreview

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza l'anteprima di un acquisto di prenotazioni con configurazioni corrispondenti a quelle dell'host dedicato h-01e23f4cd567890f1

```

Get-EC2HostReservationPurchasePreview -OfferingId hro-0c1f23456789d0ab -HostIdSet
h-01e23f4cd567890f1

```

Output:

```

CurrencyCode Purchase TotalHourlyPrice TotalUpfrontPrice
-----
{}          1.307          0.000

```

- Per i dettagli sull'API, vedere [GetHostReservationPurchasePreview](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2IdFormat

Il seguente esempio di codice mostra come utilizzare. Get-EC2IdFormat

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il formato di ID del tipo di risorsa specificato.

```
Get-EC2IdFormat -Resource instance
```

Output:

Resource	UseLongIds
-----	-----
instance	False

Esempio 2: Questo esempio descrive i formati ID per tutti i tipi di risorse che supportano il formato IDs Longer.

```
Get-EC2IdFormat
```

Output:

Resource	UseLongIds
-----	-----
reservation	False
instance	False

- Per i dettagli sull'API, vedere [DescribeIdFormat](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2IdentityIdFormat

Il seguente esempio di codice mostra come utilizzare. Get-EC2IdentityIdFormat

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il formato di ID per la risorsa “image” per il ruolo specificato

```
Get-EC2IdentityIdFormat -PrincipalArn arn:aws:iam::123456789511:role/JDBC -Resource image
```

Output:

Deadline	Resource	UseLongIds
-----	-----	-----

```
8/2/2018 11:30:00 PM image True
```

- Per i dettagli sull'API, vedere [DescribeIdentityIdFormat](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2Image

Il seguente esempio di codice mostra come utilizzare. Get-EC2Image

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'AMI specificata.

```
Get-EC2Image -ImageId ami-12345678
```

Output:

```
Architecture      : x86_64
BlockDeviceMappings : {/dev/xvda}
CreationDate      : 2014-10-20T00:56:28.000Z
Description       : My image
Hypervisor        : xen
ImageId           : ami-12345678
ImageLocation     : 123456789012/my-image
ImageOwnerAlias   :
ImageType         : machine
KernelId          :
Name              : my-image
OwnerId           : 123456789012
Platform          :
ProductCodes      : {}
Public            : False
RamdiskId         :
RootDeviceName    : /dev/xvda
RootDeviceType    : ebs
SriovNetSupport   : simple
State             : available
StateReason       :
Tags              : {Name}
VirtualizationType : hvm
```

Esempio 2: Questo esempio descrive quello AMIs che possiedi.

```
Get-EC2Image -owner self
```

Esempio 3: Questo esempio descrive il pubblico AMIs che esegue Microsoft Windows Server.

```
Get-EC2Image -Filter @{ Name="platform"; Values="windows" }
```

Esempio 4: Questo esempio descrive tutto il pubblico AMIs nella regione 'us-west-2'.

```
Get-EC2Image -Region us-west-2
```

- Per i dettagli sull'API, vedere [DescribeImages](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2ImageAttribute

Il seguente esempio di codice mostra come utilizzare Get-EC2ImageAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene la descrizione dell'AMI specificata.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute description
```

Output:

```
BlockDeviceMappings : {}  
Description           : My image description  
ImageId               : ami-12345678  
KernelId              :  
LaunchPermissions     : {}  
ProductCodes          : {}  
RamdiskId             :  
SriovNetSupport       :
```

Esempio 2: questo esempio concede le autorizzazioni di avvio per l'AMI specificata.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

Output:

```
BlockDeviceMappings : {}  
Description          :  
ImageId              : ami-12345678  
KernelId             :  
LaunchPermissions    : {all}  
ProductCodes         : {}  
RamdiskId            :  
SriovNetSupport      :
```

Esempio 3: questo esempio verifica se la funzione Reti avanzate è abilitata.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute sriovNetSupport
```

Output:

```
BlockDeviceMappings : {}  
Description          :  
ImageId              : ami-12345678  
KernelId             :  
LaunchPermissions    : {}  
ProductCodes         : {}  
RamdiskId            :  
SriovNetSupport      : simple
```

- Per i dettagli sull'API, vedere [DescribeImageAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2ImageByName

Il seguente esempio di codice mostra come utilizzare. Get-EC2ImageByName

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il set completo di nomi dei filtri attualmente supportati.

```
Get-EC2ImageByName
```

Output:

```
WINDOWS_2016_BASE
```



```
WINDOWS_2016_NANO
WINDOWS_2016_CORE
WINDOWS_2016_CONTAINER
WINDOWS_2016_SQL_SERVER_ENTERPRISE_2016
WINDOWS_2016_SQL_SERVER_STANDARD_2016
WINDOWS_2016_SQL_SERVER_WEB_2016
WINDOWS_2016_SQL_SERVER_EXPRESS_2016
WINDOWS_2012R2_BASE
WINDOWS_2012R2_CORE
WINDOWS_2012R2_SQL_SERVER_EXPRESS_2016
WINDOWS_2012R2_SQL_SERVER_STANDARD_2016
WINDOWS_2012R2_SQL_SERVER_WEB_2016
WINDOWS_2012R2_SQL_SERVER_EXPRESS_2014
WINDOWS_2012R2_SQL_SERVER_STANDARD_2014
WINDOWS_2012R2_SQL_SERVER_WEB_2014
WINDOWS_2012_BASE
WINDOWS_2012_SQL_SERVER_EXPRESS_2014
WINDOWS_2012_SQL_SERVER_STANDARD_2014
WINDOWS_2012_SQL_SERVER_WEB_2014
WINDOWS_2012_SQL_SERVER_EXPRESS_2012
WINDOWS_2012_SQL_SERVER_STANDARD_2012
WINDOWS_2012_SQL_SERVER_WEB_2012
WINDOWS_2012_SQL_SERVER_EXPRESS_2008
WINDOWS_2012_SQL_SERVER_STANDARD_2008
WINDOWS_2012_SQL_SERVER_WEB_2008
WINDOWS_2008R2_BASE
WINDOWS_2008R2_SQL_SERVER_EXPRESS_2012
WINDOWS_2008R2_SQL_SERVER_STANDARD_2012
WINDOWS_2008R2_SQL_SERVER_WEB_2012
WINDOWS_2008R2_SQL_SERVER_EXPRESS_2008
WINDOWS_2008R2_SQL_SERVER_STANDARD_2008
WINDOWS_2008R2_SQL_SERVER_WEB_2008
WINDOWS_2008RTM_BASE
WINDOWS_2008RTM_SQL_SERVER_EXPRESS_2008
WINDOWS_2008RTM_SQL_SERVER_STANDARD_2008
WINDOWS_2008_BEANSTALK_IIS75
WINDOWS_2012_BEANSTALK_IIS8
VPC_NAT
```

Esempio 2: questo esempio descrive l'AMI specificata. L'utilizzo di questo comando per individuare un'AMI è utile perché AWS rilascia ogni mese nuove Windows AMIs con gli ultimi aggiornamenti. È possibile specificare 'ImageId' per New-EC2Instance avviare un'istanza utilizzando l'AMI corrente per il filtro specificato.

```
Get-EC2ImageByName -Names WINDOWS_2016_BASE
```

Output:

```
Architecture      : x86_64
BlockDeviceMappings : {/dev/sda1, xvdca, xvdcb, xvdcc...}
CreationDate       : yyyy.mm.ddThh:mm:ss.000Z
Description        : Microsoft Windows Server 2016 with Desktop Experience Locale
                    English AMI provided by Amazon
Hypervisor         : xen
ImageId            : ami-xxxxxxxxx
ImageLocation       : amazon/Windows_Server-2016-English-Full-Base-yyyy.mm.dd
ImageOwnerAlias     : amazon
ImageType          : machine
KernelId           :
Name               : Windows_Server-2016-English-Full-Base-yyyy.mm.dd
OwnerId            : 801119661308
Platform           : Windows
ProductCodes       : {}
Public             : True
RamdiskId          :
RootDeviceName     : /dev/sda1
RootDeviceType     : ebs
SriovNetSupport     : simple
State              : available
StateReason        :
Tags               : {}
VirtualizationType : hvm
```

- Per i dettagli sull'API, vedere [Get-EC2ImageByName](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2ImportImageTask

Il seguente esempio di codice mostra come utilizzare. Get-EC2ImportImageTask

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'attività di importazione di immagini specificata.

```
Get-EC2ImportImageTask -ImportTaskId import-ami-hgfedcba
```

Output:

```
Architecture      : x86_64
Description       : Windows Image 2
Hypervisor        :
ImageId           : ami-1a2b3c4d
ImportTaskId      : import-ami-hgfedcba
LicenseType       : AWS
Platform          : Windows
Progress          :
SnapshotDetails   : {/dev/sda1}
Status            : completed
StatusMessage     :
```

Esempio 2: questo esempio descrive tutte le attività di importazione di immagini specificate.

```
Get-EC2ImportImageTask
```

Output:

```
Architecture      :
Description       : Windows Image 1
Hypervisor        :
ImageId           :
ImportTaskId      : import-ami-abcdefgh
LicenseType       : AWS
Platform          : Windows
Progress          :
SnapshotDetails   : {}
Status            : deleted
StatusMessage     : User initiated task cancelation

Architecture      : x86_64
Description       : Windows Image 2
Hypervisor        :
ImageId           : ami-1a2b3c4d
ImportTaskId      : import-ami-hgfedcba
LicenseType       : AWS
Platform          : Windows
Progress          :
SnapshotDetails   : {/dev/sda1}
Status            : completed
```

```
StatusMessage      :
```

- Per i dettagli sull'API, vedere [DescribeImportImageTasks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2ImportSnapshotTask

Il seguente esempio di codice mostra come utilizzare. Get-EC2ImportSnapshotTask

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'attività di importazione di snapshot specificata.

```
Get-EC2ImportSnapshotTask -ImportTaskId import-snap-abcdefgh
```

Output:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----
Disk Image Import 1	import-snap-abcdefgh	Amazon.EC2.Model.SnapshotTaskDetail

Esempio 2: questo esempio descrive tutte le attività di importazione di snapshot specificate.

```
Get-EC2ImportSnapshotTask
```

Output:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----
Disk Image Import 1	import-snap-abcdefgh	Amazon.EC2.Model.SnapshotTaskDetail
Disk Image Import 2	import-snap-hgfedcba	Amazon.EC2.Model.SnapshotTaskDetail

- Per i dettagli sull'API, vedere [DescribeImportSnapshotTasks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2Instance

Il seguente esempio di codice mostra come utilizzare. Get-EC2Instance

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'istanza specificata.

```
(Get-EC2Instance -InstanceId i-12345678).Instances
```

Output:

```
AmiLaunchIndex      : 0
Architecture        : x86_64
BlockDeviceMappings : {/dev/sda1}
ClientToken          : TleEy1448154045270
EbsOptimized         : False
Hypervisor           : xen
IamInstanceProfile   : Amazon.EC2.Model.IamInstanceProfile
ImageId              : ami-12345678
InstanceId           : i-12345678
InstanceLifecycle    :
InstanceType         : t2.micro
KernelId             :
KeyName              : my-key-pair
LaunchTime           : 12/4/2015 4:44:40 PM
Monitoring           : Amazon.EC2.Model.Monitoring
NetworkInterfaces    : {ip-10-0-2-172.us-west-2.compute.internal}
Placement            : Amazon.EC2.Model.Placement
Platform             : Windows
PrivateDnsName       : ip-10-0-2-172.us-west-2.compute.internal
PrivateIpAddress     : 10.0.2.172
ProductCodes         : {}
PublicDnsName        :
PublicIpAddress       :
RamdiskId            :
RootDeviceName       : /dev/sda1
RootDeviceType       : ebs
SecurityGroups       : {default}
SourceDestCheck      : True
SpotInstanceRequestId :
SriovNetSupport      :
State                : Amazon.EC2.Model.InstanceState
```

```

StateReason      :
StateTransitionReason :
SubnetId         : subnet-12345678
Tags             : {Name}
VirtualizationType : hvm
VpcId            : vpc-12345678

```

Esempio 2: questo esempio descrive tutte le istanze nella Regione corrente, raggruppate per prenotazione. Per visualizzare i dettagli dell'istanza, espandi la raccolta Instances all'interno di ciascun oggetto di prenotazione.

```
Get-EC2Instance
```

Output:

```

GroupNames      : {}
Groups          : {}
Instances       : {}
OwnerId         : 123456789012
RequesterId     : 226008221399
ReservationId   : r-c5df370c

GroupNames      : {}
Groups          : {}
Instances       : {}
OwnerId         : 123456789012
RequesterId     : 854251627541
ReservationId   : r-63e65bab
...

```

Esempio 3: Questo esempio illustra l'utilizzo di un filtro per interrogare EC2 le istanze in una sottorete specifica di un VPC.

```
(Get-EC2Instance -Filter @{Name="vpc-id";Values="vpc-1a2bc34d"},@{Name="subnet-id";Values="subnet-1a2b3c4d"}).Instances
```

Output:

```

InstanceId      InstanceType Platform PrivateIpAddress PublicIpAddress
SecurityGroups SubnetId      VpcId

```

```

-----
-----
i-01af...82cf180e19 t2.medium    Windows  10.0.0.98      ...
      subnet-1a2b3c4d vpc-1a2b3c4d
i-0374...7e9d5b0c45 t2.xlarge  Windows  10.0.0.53      ...
      subnet-1a2b3c4d vpc-1a2b3c4d

```

Esempio 4: Questo esempio illustra l'utilizzo di un filtro con più valori per ricercare EC2 istanze sia in esecuzione che interrotte

```

$instanceParams = @{
    Filter = @(
        @{'Name' = 'instance-state-name'; 'Values' = @("running","stopped")}
    )
}

(Get-EC2Instance @instanceParams).Instances

```

Output:

```

InstanceId      InstanceType Platform PrivateIpAddress PublicIpAddress
SecurityGroups SubnetId      VpcId
-----
-----
i-05a9...f6c5f46e18 t3.medium    10.0.1.7      ...
      subnet-1a2b3c4d vpc-1a2b3c4d
i-02cf...945c4fdd07 t3.medium    Windows  10.0.1.8      ...
      subnet-1a2b3c4d vpc-1a2b3c4d
i-0ac0...c037f9f3a1 t3.xlarge    Windows  10.0.1.10     ...
      subnet-1a2b3c4d vpc-1a2b3c4d
i-066b...57b7b08888 t3.medium    Windows  10.0.1.11     ...
      subnet-1a2b3c4d vpc-1a2b3c4d
i-0fee...82e83ccd72 t3.medium    Windows  10.0.1.5      ...
      subnet-1a2b3c4d vpc-1a2b3c4d
i-0a68...274cc5043b t3.medium    Windows  10.0.1.6      ...
      subnet-1a2b3c4d vpc-1a2b3c4d

```

Esempio 5: questo esempio illustra l'utilizzo di un filtro con più valori per interrogare EC2 le istanze sia in esecuzione che interrotte e l'utilizzo del cmdlet Select-Object per la scelta di valori specifici da generare.

```

$instanceParams = @{

```

```

    Filter = @(
        @{'Name' = 'instance-state-name'; 'Values' = @("running","stopped")}
    )
}

$SelectParams = @{
    Property = @(
        "InstanceId", "InstanceType", "Platform", "PrivateIpAddress",
        @{'Name'="Name";Expression={$_.Tags[$_Tags.Key.IndexOf("Name")].Value}},
        @{'Name'="State";Expression={$_.State.Name}}
    )
}

$result = Get-EC2Instance @InstanceParams
$result.Instances | Select-Object @SelectParams | Format-Table -AutoSize

```

Output:

InstanceId	InstanceType	Platform	PrivateIpAddress	Name	State
-----	-----	-----	-----	----	-----
i-05a9...f6c5f46e18	t3.medium		10.0.1.7	ec2-name-01	running
i-02cf...945c4fdd07	t3.medium	Windows	10.0.1.8	ec2-name-02	stopped
i-0ac0...c037f9f3a1	t3.xlarge	Windows	10.0.1.10	ec2-name-03	running
i-066b...57b7b08888	t3.medium	Windows	10.0.1.11	ec2-name-04	stopped
i-0fee...82e83ccd72	t3.medium	Windows	10.0.1.5	ec2-name-05	running
i-0a68...274cc5043b	t3.medium	Windows	10.0.1.6	ec2-name-06	stopped

- Per i dettagli sull'API, vedere [DescribeInstances](#) in Cmdlet Reference (V4). AWS Strumenti per PowerShell

Get-EC2InstanceAttribute

Il seguente esempio di codice mostra come utilizzare. Get-EC2InstanceAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il tipo di istanza dell'istanza specificata.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute instanceType
```

Output:


```
InstanceType                : t2.micro
```

Esempio 2: questo esempio descrive se la funzione Reti avanzate è abilitata per l'istanza specificata.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

Output:

```
SriovNetSupport             : simple
```

Esempio 3: questo esempio crea i gruppi di sicurezza per l'istanza specificata.

```
(Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute groupSet).Groups
```

Output:

```
GroupId
-----
sg-12345678
sg-45678901
```

Esempio 4: questo esempio descrive se la funzione Ottimizzazione per EBS è abilitata per l'istanza specificata.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

Output:

```
EbsOptimized                : False
```

Esempio 5: Questo esempio descrive l'attributo disableApiTermination " dell'istanza specificata.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute disableApiTermination
```

Output:

```
DisableApiTermination       : False
```

Esempio 6: Questo esempio descrive l'attributo 'instanceInitiatedShutdownComportamento' dell'istanza specificata.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute  
instanceInitiatedShutdownBehavior
```

Output:

```
InstanceInitiatedShutdownBehavior : stop
```

- Per i dettagli sull'API, vedere [DescribeInstanceAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2InstanceMetadata

Il seguente esempio di codice mostra come utilizzare. Get-EC2InstanceMetadata

Strumenti per PowerShell V4

Esempio 1: elenca le categorie disponibili di metadati delle istanze su cui è possibile eseguire query.

```
Get-EC2InstanceMetadata -ListCategory
```

Output:

```
AmiId  
LaunchIndex  
ManifestPath  
AncestorAmiId  
BlockDeviceMapping  
InstanceId  
InstanceType  
LocalHostname  
LocalIpv4  
KernelId  
AvailabilityZone  
ProductCode  
PublicHostname  
PublicIpv4  
PublicKey
```

```
RamdiskId  
Region  
ReservationId  
SecurityGroup  
UserData  
InstanceMonitoring  
IdentityDocument  
IdentitySignature  
IdentityPkcs7
```

Esempio 2: restituisce l'ID dell'Amazon Machine Image (AMI) utilizzato per avviare l'istanza.

```
Get-EC2InstanceMetadata -Category AmiId
```

Output:

```
ami-b2e756ca
```

Esempio 3: questo esempio esegue query sul documento di identità in formato JSON per l'istanza.

```
Get-EC2InstanceMetadata -Category IdentityDocument  
{  
  "availabilityZone" : "us-west-2a",  
  "devpayProductCodes" : null,  
  "marketplaceProductCodes" : null,  
  "version" : "2017-09-30",  
  "instanceId" : "i-01ed50f7e2607f09e",  
  "billingProducts" : [ "bp-6ba54002" ],  
  "instanceType" : "t2.small",  
  "pendingTime" : "2018-03-07T16:26:04Z",  
  "imageId" : "ami-b2e756ca",  
  "privateIp" : "10.0.0.171",  
  "accountId" : "111122223333",  
  "architecture" : "x86_64",  
  "kernelId" : null,  
  "ramdiskId" : null,  
  "region" : "us-west-2"  
}
```

Esempio 4: questo esempio utilizza una query a livello di percorso per ottenere gli indirizzi MAC dell'interfaccia di rete per l'istanza.

```
Get-EC2InstanceMetadata -Path "/network/interfaces/macs"
```

Output:

```
02:80:7f:ef:4c:e0/
```

Esempio 5: se all'istanza è associato un ruolo IAM, restituisce informazioni sull'ultima volta in cui il profilo dell'istanza è stato aggiornato, inclusa la LastUpdated data dell'istanza InstanceProfileArn, e InstanceProfileId.

```
Get-EC2InstanceMetadata -Path "/iam/info"
```

Output:

```
{
  "Code" : "Success",
  "LastUpdated" : "2018-03-08T03:38:40Z",
  "InstanceProfileArn" : "arn:aws:iam::111122223333:instance-profile/
MyLaunchRole_Profile",
  "InstanceProfileId" : "AIPAI4...WVK2RW"
}
```

- Per i dettagli sull'API, vedere [Get-EC2InstanceMetadata](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2InstanceStatus

Il seguente esempio di codice mostra come utilizzare. Get-EC2InstanceStatus

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive lo stato dell'istanza specificata.

```
Get-EC2InstanceStatus -InstanceId i-12345678
```

Output:

```
AvailabilityZone : us-west-2a
Events           : {}
InstanceId       : i-12345678
```

```

InstanceState      : Amazon.EC2.Model.InstanceState
Status             : Amazon.EC2.Model.InstanceStatusSummary
SystemStatus       : Amazon.EC2.Model.InstanceStatusSummary

```

```

$status = Get-EC2InstanceStatus -InstanceId i-12345678
$status.InstanceState

```

Output:

```

Code      Name
----      -
16        running

```

```
$status.Status
```

Output:

```

Details      Status
-----      -
{reachability} ok

```

```
$status.SystemStatus
```

Output:

```

Details      Status
-----      -
{reachability} ok

```

- Per i dettagli sull'API, vedere [DescribeInstanceStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2InternetGateway

Il seguente esempio di codice mostra come utilizzare. Get-EC2InternetGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il gateway Internet specificato.

```
Get-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

Output:

Attachments	InternetGatewayId	Tags
-----	-----	----
{vpc-1a2b3c4d}	igw-1a2b3c4d	{}

Esempio 2: questo esempio descrive tutti i gateway Internet specificati.

```
Get-EC2InternetGateway
```

Output:

Attachments	InternetGatewayId	Tags
-----	-----	----
{vpc-1a2b3c4d}	igw-1a2b3c4d	{}
{}	igw-2a3b4c5d	{}

- Per i dettagli sull'API, vedere [DescribeInternetGateways](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2KeyPair

Il seguente esempio di codice mostra come utilizzare. Get-EC2KeyPair

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive la coppia di chiavi specificata.

```
Get-EC2KeyPair -KeyName my-key-pair
```

Output:

KeyFingerprint	KeyName
-----	-----
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f	my-key-pair

Esempio 2: questo esempio descrive tutte le coppie di chiavi.

Get-EC2KeyPair

- Per i dettagli sull'API, vedere [DescribeKeyPairs](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2NetworkAcl

Il seguente esempio di codice mostra come utilizzare. Get-EC2NetworkAcl

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive la lista di controllo degli accessi (ACL) di rete specificata.

```
Get-EC2NetworkAcl -NetworkAclId acl-12345678
```

Output:

```
Associations : {aclassoc-1a2b3c4d}
Entries      : {Amazon.EC2.Model.NetworkAclEntry, Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {Name}
VpcId        : vpc-12345678
```

Esempio 2: questo esempio descrive le regole per lista di controllo degli accessi alla rete (ACL) specificata.

```
(Get-EC2NetworkAcl -NetworkAclId acl-12345678).Entries
```

Output:

```
CidrBlock    : 0.0.0.0/0
Egress       : True
IcmpTypeCode :
PortRange    :
Protocol     : -1
RuleAction   : deny
RuleNumber   : 32767

CidrBlock    : 0.0.0.0/0
```

```
Egress      : False
IcmpTypeCode :
PortRange   :
Protocol    : -1
RuleAction   : deny
RuleNumber   : 32767
```

Esempio 3: Questo esempio descrive tutta la rete ACLs.

```
Get-EC2NetworkAcl
```

- Per i dettagli sull'API, vedere [DescribeNetworkAcls](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2NetworkInterface

Il seguente esempio di codice mostra come utilizzare. Get-EC2NetworkInterface

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'interfaccia di rete specificata.

```
Get-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

Output:

```
Association      :
Attachment       : Amazon.EC2.Model.NetworkInterfaceAttachment
AvailabilityZone  : us-west-2c
Description      :
Groups           : {my-security-group}
MacAddress        : 0a:e9:a6:19:4c:7f
NetworkInterfaceId : eni-12345678
OwnerId          : 123456789012
PrivateDnsName    : ip-10-0-0-107.us-west-2.compute.internal
PrivateIpAddress  : 10.0.0.107
PrivateIpAddresses : {ip-10-0-0-107.us-west-2.compute.internal}
RequesterId      :
RequesterManaged : False
SourceDestCheck   : True
Status           : in-use
SubnetId         : subnet-1a2b3c4d
```



```
TagSet          : {}  
VpcId           : vpc-12345678
```

Esempio 2: questo esempio descrive tutte le interfacce di rete.

```
Get-EC2NetworkInterface
```

- Per i dettagli sull'API, vedere [DescribeNetworkInterfaces](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2NetworkInterfaceAttribute

Il seguente esempio di codice mostra come utilizzare. Get-EC2NetworkInterfaceAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'interfaccia di rete specificata.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute  
Attachment
```

Output:

```
Attachment      : Amazon.EC2.Model.NetworkInterfaceAttachment
```

Esempio 2: questo esempio descrive l'interfaccia di rete specificata.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute  
Description
```

Output:

```
Description     : My description
```

Esempio 3: questo esempio descrive l'interfaccia di rete specificata.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute  
GroupSet
```

Output:

```
Groups           : {my-security-group}
```

Esempio 4: questo esempio descrive l'interfaccia di rete specificata.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute  
SourceDestCheck
```

Output:

```
SourceDestCheck   : True
```

- Per i dettagli sull'API, vedere [DescribeNetworkInterfaceAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2PasswordData

Il seguente esempio di codice mostra come utilizzare. Get-EC2PasswordData

Strumenti per PowerShell V4

Esempio 1: questo esempio decrittografa la password che Amazon ha EC2 assegnato all'account Administrator per l'istanza Windows specificata. Poiché è stato specificato un file pem, viene automaticamente applicata l'impostazione del parametro -Decrypt.

```
Get-EC2PasswordData -InstanceId i-12345678 -PemFile C:\path\my-key-pair.pem
```

Output:

```
mYZ(PA9?C)Q
```

Esempio 2: (PowerShell solo Windows) Ispeziona l'istanza per determinare il nome della coppia di chiavi utilizzata per avviare l'istanza, quindi tenta di trovare i dati della coppia di chiavi corrispondente nell'archivio di configurazione di AWS Toolkit for Visual Studio. Se vengono individuati i dati della coppia di chiavi, la password viene decrittografata.

```
Get-EC2PasswordData -InstanceId i-12345678 -Decrypt
```

Output:

```
mYZ(PA9?C)Q
```

Esempio 3: restituisce i dati della password crittografata per l'istanza.

```
Get-EC2PasswordData -InstanceId i-12345678
```

Output:

```
iVz3BAK/WAXV.....dqt8WeMA==
```

- Per i dettagli sull'API, vedere [GetPasswordData](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2PlacementGroup

Il seguente esempio di codice mostra come utilizzare. Get-EC2PlacementGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il gruppo di posizionamento specificato.

```
Get-EC2PlacementGroup -GroupName my-placement-group
```

Output:

GroupName	State	Strategy
-----	-----	-----
my-placement-group	available	cluster

- Per i dettagli sull'API, vedere [DescribePlacementGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2PrefixList

Il seguente esempio di codice mostra come utilizzare. Get-EC2PrefixList

Strumenti per PowerShell V4

Esempio 1: Questo esempio recupera il formato di elenco Servizi AWS di prefissi disponibile per la regione

Get-EC2PrefixList

Output:

Cidrs	PrefixListId	PrefixListName
-----	-----	-----
{52.94.5.0/24, 52.119.240.0/21, 52.94.24.0/23}	pl-6fa54006	com.amazonaws.eu-west-1.dynamodb
{52.218.0.0/17, 54.231.128.0/19}	pl-6da54004	com.amazonaws.eu-west-1.s3

- Per i dettagli sull'API, vedere [DescribePrefixLists](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2Region

Il seguente esempio di codice mostra come utilizzare. Get-EC2Region

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le Regioni disponibili.

Get-EC2Region

Output:

Endpoint	RegionName
-----	-----
ec2.eu-west-1.amazonaws.com	eu-west-1
ec2.ap-southeast-1.amazonaws.com	ap-southeast-1
ec2.ap-southeast-2.amazonaws.com	ap-southeast-2
ec2.eu-central-1.amazonaws.com	eu-central-1
ec2.ap-northeast-1.amazonaws.com	ap-northeast-1
ec2.us-east-1.amazonaws.com	us-east-1
ec2.sa-east-1.amazonaws.com	sa-east-1
ec2.us-west-1.amazonaws.com	us-west-1
ec2.us-west-2.amazonaws.com	us-west-2

- Per i dettagli sull'API, vedere [DescribeRegions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2RouteTable

Il seguente esempio di codice mostra come utilizzare `Get-EC2RouteTable`

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive tutte le tabelle di routing disponibili.

```
Get-EC2RouteTable
```

Output:

```
DestinationCidrBlock    : 10.0.0.0/16
DestinationPrefixListId :
GatewayId               : local
InstanceId               :
InstanceOwnerId          :
NetworkInterfaceId      :
Origin                  : CreateRouteTable
State                   : active
VpcPeeringConnectionId  :

DestinationCidrBlock    : 0.0.0.0/0
DestinationPrefixListId :
GatewayId               : igw-1a2b3c4d
InstanceId               :
InstanceOwnerId          :
NetworkInterfaceId      :
Origin                  : CreateRoute
State                   : active
VpcPeeringConnectionId  :
```

Esempio 2: questo esempio restituisce i dettagli della tabella di routing specificata.

```
Get-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

Esempio 3: questo esempio descrive le tabelle di routing per il VPC specificato.

```
Get-EC2RouteTable -Filter @{ Name="vpc-id"; Values="vpc-1a2b3c4d" }
```

Output:

```
Associations      : {rtbassoc-12345678}
PropagatingVgws  : {}
Routes           : {, }
RouteTableId     : rtb-1a2b3c4d
Tags             : {}
VpcId            : vpc-1a2b3c4d
```

- Per i dettagli sull'API, vedere [DescribeRouteTables](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2ScheduledInstance

Il seguente esempio di codice mostra come utilizzare. Get-EC2ScheduledInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'istanza pianificata specificata.

```
Get-EC2ScheduledInstance -ScheduledInstanceId sci-1234-1234-1234-1234-123456789012
```

Output:

```
AvailabilityZone      : us-west-2b
CreateDate            : 1/25/2016 1:43:38 PM
HourlyPrice           : 0.095
InstanceCount        : 1
InstanceType          : c4.large
NetworkPlatform      : EC2-VPC
NextSlotStartTime     : 1/31/2016 1:00:00 AM
Platform             : Linux/UNIX
PreviousSlotEndTime   :
Recurrence            : Amazon.EC2.Model.ScheduledInstanceRecurrence
ScheduledInstanceId   : sci-1234-1234-1234-1234-123456789012
SlotDurationInHours   : 32
TermEndDate          : 1/31/2017 1:00:00 AM
TermStartDate         : 1/31/2016 1:00:00 AM
TotalScheduledInstanceHours : 1696
```

Esempio 2: questo esempio descrive tutte le istanze pianificate.

```
Get-EC2ScheduledInstance
```

- Per i dettagli sull'API, vedere [DescribeScheduledInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2ScheduledInstanceAvailability

Il seguente esempio di codice mostra come utilizzare. Get-EC2ScheduledInstanceAvailability

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive una pianificazione che si verifica ogni settimana di domenica, a partire dalla data specificata.

```
Get-EC2ScheduledInstanceAvailability -Recurrence_Frequency
Weekly -Recurrence_Interval 1 -Recurrence_OccurrenceDay 1 -
FirstSlotStartTimeRange_EarliestTime 2016-01-31T00:00:00Z -
FirstSlotStartTimeRange_LatestTime 2016-01-31T04:00:00Z
```

Output:

```
AvailabilityZone           : us-west-2b
AvailableInstanceCount     : 20
FirstSlotStartTime         : 1/31/2016 8:00:00 AM
HourlyPrice                : 0.095
InstanceType               : c4.large
MaxTermDurationInDays      : 366
MinTermDurationInDays      : 366
NetworkPlatform            : EC2-VPC
Platform                   : Linux/UNIX
PurchaseToken              : eyJ2IjoiMSIsInMiOiEsImMiOi...
Recurrence                 : Amazon.EC2.Model.ScheduledInstanceRecurrence
SlotDurationInHours        : 23
TotalScheduledInstanceHours : 1219
...
```

Esempio 2: come limitare i risultati, è possibile aggiungere filtri come criteri, ad esempio sistema operativo, rete e tipo di istanza.

```
-Filter @{ Name="platform";Values="Linux/UNIX" },@{ Name="network-
platform";Values="EC2-VPC" },@{ Name="instance-type";Values="c4.large" }
```

- Per i dettagli sull'API, vedere [DescribeScheduledInstanceAvailability](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2SecurityGroup

Il seguente esempio di codice mostra come utilizzare. Get-EC2SecurityGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il gruppo di sicurezza specificato per un VPC. Quando si lavora con gruppi di sicurezza appartenenti a un VPC, è necessario utilizzare l'ID del gruppo di sicurezza (GroupId parametro -), non il nome (GroupName parametro -), per fare riferimento al gruppo.

```
Get-EC2SecurityGroup -GroupId sg-12345678
```

Output:

```
Description      : default VPC security group
GroupId          : sg-12345678
GroupName        : default
IpPermissions    : {Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {Amazon.EC2.Model.IpPermission}
OwnerId          : 123456789012
Tags             : {}
VpcId            : vpc-12345678
```

Esempio 2: Questo esempio descrive il gruppo di sicurezza specificato per EC2 -Classic. Quando si lavora con i gruppi di sicurezza per EC2 -Classic, è possibile utilizzare il nome del gruppo (GroupName parametro -) o l'ID del gruppo (GroupId parametro -) per fare riferimento al gruppo di sicurezza.

```
Get-EC2SecurityGroup -GroupName my-security-group
```

Output:

```
Description      : my security group
GroupId          : sg-45678901
GroupName        : my-security-group
IpPermissions    : {Amazon.EC2.Model.IpPermission, Amazon.EC2.Model.IpPermission}
```



```

IpPermissionsEgress : {}
OwnerId              : 123456789012
Tags                 : {}
VpcId                :

```

Esempio 3: questo esempio recupera tutti i gruppi di sicurezza per vpc-0fc1ff23456b789eb

```
Get-EC2SecurityGroup -Filter @{Name="vpc-id";Values="vpc-0fc1ff23456b789eb"}
```

- Per i dettagli sull'API, vedere [DescribeSecurityGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2Snapshot

Il seguente esempio di codice mostra come utilizzare. Get-EC2Snapshot

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive lo snapshot specificato.

```
Get-EC2Snapshot -SnapshotId snap-12345678
```

Output:

```

DataEncryptionKeyId :
Description           : Created by CreateImage(i-1a2b3c4d) for ami-12345678 from
                        vol-12345678
Encrypted              : False
KmsKeyId              :
OwnerAlias            :
OwnerId               : 123456789012
Progress              : 100%
SnapshotId            : snap-12345678
StartTime             : 10/23/2014 6:01:28 AM
State                 : completed
StateMessage          :
Tags                  : {}
VolumeId              : vol-12345678
VolumeSize            : 8

```

Esempio 2: questo esempio descrive gli snapshot contenenti un tag "Name".

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" }
```

Esempio 3: Questo esempio descrive le istantanee che hanno un tag 'Nome' con il valore ".
TestValue

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" -and  
$_.Tags.Value -eq "TestValue" }
```

Esempio 4: questo esempio descrive tutti gli snapshot.

```
Get-EC2Snapshot -Owner self
```

- Per i dettagli sull'API, vedere [DescribeSnapshots](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2SnapshotAttribute

Il seguente esempio di codice mostra come utilizzare. Get-EC2SnapshotAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'attributo specificato dello snapshot specificato.

```
Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute ProductCodes
```

Output:

CreateVolumePermissions	ProductCodes	SnapshotId
-----	-----	-----
{}	{}	snap-12345678

Esempio 2: questo esempio descrive l'attributo specificato dello snapshot specificato.

```
(Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute  
CreateVolumePermission).CreateVolumePermissions
```

Output:

Group	UserId
----	-----

```
all
```

- Per i dettagli sull'API, vedere [DescribeSnapshotAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2SpotDatafeedSubscription

Il seguente esempio di codice mostra come utilizzare. Get-EC2SpotDatafeedSubscription
Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il feed di dati dell'istanza spot.

```
Get-EC2SpotDatafeedSubscription
```

Output:

```
Bucket   : amzn-s3-demo-bucket
Fault    :
OwnerId  : 123456789012
Prefix   : spotdata
State    : Active
```

- Per i dettagli sull'API, vedere [DescribeSpotDatafeedSubscription](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2SpotFleetInstance

Il seguente esempio di codice mostra come utilizzare. Get-EC2SpotFleetInstance
Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le istanze associate alla richiesta del parco istanze spot specificata.

```
Get-EC2SpotFleetInstance -SpotFleetRequestId sfr-73fbd2ce-  
aa30-494c-8788-1cee4EXAMPLE
```

Output:

InstanceId	InstanceType	SpotInstanceRequestId

-----	-----	-----
i-f089262a	c3.large	sir-12345678
i-7e8b24a4	c3.large	sir-87654321

- Per i dettagli sull'API, vedere [DescribeSpotFleetInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2SpotFleetRequest

Il seguente esempio di codice mostra come utilizzare. Get-EC2SpotFleetRequest

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive la richiesta del parco istanze spot specificata.

```
Get-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
| format-list
```

Output:

```
ConfigData           : Amazon.EC2.Model.SpotFleetRequestConfigData
CreateTime           : 12/26/2015 8:23:33 AM
SpotFleetRequestId   : sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
SpotFleetRequestState : active
```

Esempio 2: questo esempio descrive tutte le richieste del parco istanze spot.

```
Get-EC2SpotFleetRequest
```

- Per i dettagli sull'API, vedere [DescribeSpotFleetRequests](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2SpotFleetRequestHistory

Il seguente esempio di codice mostra come utilizzare. Get-EC2SpotFleetRequestHistory

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive la cronologia della richiesta del parco istanze spot specificata.

```
Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z
```

Output:

```
HistoryRecords      : {Amazon.EC2.Model.HistoryRecord,
  Amazon.EC2.Model.HistoryRecord...}
LastEvaluatedTime   : 12/26/2015 8:29:11 AM
NextToken           :
SpotFleetRequestId  : sfr-088bc5f1-7e7b-451a-bd13-757f10672b93
StartTime           : 12/25/2015 8:00:00 AM
```

```
(Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z).HistoryRecords
```

Output:

EventInformation	EventType	Timestamp
-----	-----	-----
Amazon.EC2.Model.EventInformation	fleetRequestChange	12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation	fleetRequestChange	12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation	fleetRequestChange	12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation	launched	12/26/2015 8:25:34 AM
Amazon.EC2.Model.EventInformation	launched	12/26/2015 8:25:05 AM

- Per i dettagli sull'API, vedere [DescribeSpotFleetRequestHistory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2SpotInstanceRequest

Il seguente esempio di codice mostra come utilizzare. Get-EC2SpotInstanceRequest

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive la richiesta di istanze spot specificata.

```
Get-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

Output:

```

ActualBlockHourlyPrice      :
AvailabilityZoneGroup       :
BlockDurationMinutes        : 0
CreateTime                  : 4/8/2015 2:51:33 PM
Fault                       :
InstanceId                   : i-12345678
LaunchedAvailabilityZone    : us-west-2b
LaunchGroup                 :
LaunchSpecification         : Amazon.EC2.Model.LaunchSpecification
ProductDescription          : Linux/UNIX
SpotInstanceRequestId       : sir-12345678
SpotPrice                   : 0.020000
State                       : active
Status                      : Amazon.EC2.Model.SpotInstanceStatus
Tags                        : {Name}
Type                        : one-time

```

Esempio 2: questo esempio descrive tutte le richieste di istanze spot.

```
Get-EC2SpotInstanceRequest
```

- Per i dettagli sull'API, vedere [DescribeSpotInstanceRequests](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2SpotPriceHistory

Il seguente esempio di codice mostra come utilizzare. Get-EC2SpotPriceHistory

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene le ultime 10 voci della cronologia del prezzo spot per il tipo di istanza e la zona di disponibilità specificati. Si noti che il valore specificato per il AvailabilityZone parametro - deve essere valido per il valore della regione fornito al parametro -Region del cmdlet (non mostrato nell'esempio) o impostato come predefinito nella shell. Questo comando di esempio presuppone che nell'ambiente sia stata impostata una Regione predefinita "us-west-2".

```
Get-EC2SpotPriceHistory -InstanceType c3.large -AvailabilityZone us-west-2a -
MaxResult 10
```

Output:

```

AvailabilityZone : us-west-2a
InstanceType     : c3.large
Price            : 0.017300
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp        : 12/25/2015 7:39:49 AM

AvailabilityZone : us-west-2a
InstanceType     : c3.large
Price            : 0.017200
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp        : 12/25/2015 7:38:29 AM

AvailabilityZone : us-west-2a
InstanceType     : c3.large
Price            : 0.017300
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp        : 12/25/2015 6:57:13 AM
...

```

- Per i dettagli sull'API, vedere [DescribeSpotPriceHistory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2Subnet

Il seguente esempio di codice mostra come utilizzare. Get-EC2Subnet

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive la sottorete specificata.

```
Get-EC2Subnet -SubnetId subnet-1a2b3c4d
```

Output:

```

AvailabilityZone      : us-west-2c
AvailableIpAddressCount : 251
CidrBlock             : 10.0.0.0/24
DefaultForAz          : False
MapPublicIpOnLaunch   : False
State                 : available
SubnetId              : subnet-1a2b3c4d
Tags                  : {}

```

```
VpcId : vpc-12345678
```

Esempio 2: questo esempio descrive tutte le sottoreti.

```
Get-EC2Subnet
```

- Per i dettagli sull'API, vedere [DescribeSubnets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2Tag

Il seguente esempio di codice mostra come utilizzare. Get-EC2Tag

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera i tag per il tipo di risorsa "image"

```
Get-EC2Tag -Filter @{Name="resource-type";Values="image"}
```

Output:

Key	ResourceId	ResourceType	Value
---	-----	-----	----
Name	ami-0a123b4ccb567a8ea	image	Win7-Imported
auto-delete	ami-0a123b4ccb567a8ea	image	never

Esempio 2: questo esempio recupera tutti i tag per tutte le risorse e li raggruppa per tipo di risorsa

```
Get-EC2Tag | Group-Object resourcetype
```

Output:

Count	Name	Group
-----	----	-----
9	subnet	{Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription...}
53	instance	{Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription...}


```

3 route-table           {Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
5 security-group        {Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription...}
30 volume               {Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription...}
1 internet-gateway      {Amazon.EC2.Model.TagDescription}
3 network-interface     {Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
4 elastic-ip            {Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription}
1 dhcp-options          {Amazon.EC2.Model.TagDescription}
2 image                 {Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription}
3 vpc                   {Amazon.EC2.Model.TagDescription,
Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}

```

Esempio 3: questo esempio mostra tutte le risorse con il tag “auto-delete” con il valore “no” per la Regione specificata

```
Get-EC2Tag -Region eu-west-1 -Filter @{Name="tag:auto-delete";Values="no"}
```

Output:

Key	ResourceId	ResourceType	Value
---	-----	-----	----
auto-delete	i-0f1bce234d5dd678b	instance	no
auto-delete	vol-01d234aa5678901a2	volume	no
auto-delete	vol-01234bfb5def6f7b8	volume	no
auto-delete	vol-01ccb23f4c5e67890	volume	no

Esempio 4: questo esempio ottiene tutte le risorse con il tag «auto-delete» con valore «no» e ulteriori filtri nella pipe successiva per analizzare solo i tipi di risorse «istanza» e infine crea il tag «» per ogni risorsa di istanza il cui valore è l'ID dell'istanza stessa ThisInstance

```

Get-EC2Tag -Region eu-west-1 -Filter @{Name="tag:auto-delete";Values="no"} |
Where-Object ResourceType -eq "instance" | ForEach-Object {New-EC2Tag -ResourceId
$_.ResourceId -Tag @{Key="ThisInstance";Value=$_.ResourceId}}

```

Esempio 5: questo esempio recupera i tag per tutte le risorse dell'istanza e le chiavi "Name" e li visualizza in un formato di tabella

```
Get-EC2Tag -Filter @{{Name="resource-
type";Values="instance"},@{{Name="key";Values="Name"}} | Select-Object ResourceId,
@{{Name="Name-Tag";Expression={$PSItem.Value}} | Format-Table -AutoSize
```

Output:

ResourceId	Name-Tag
-----	-----
i-012e3cb4df567e1aa	jump1
i-01c23a45d6fc7a89f	repro-3

- Per i dettagli sull'API, vedere [DescribeTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2Volume

Il seguente esempio di codice mostra come utilizzare. Get-EC2Volume

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il volume EBS specificato.

```
Get-EC2Volume -VolumeId vol-12345678
```

Output:

```
Attachments      : {}
AvailabilityZone  : us-west-2c
CreateTime       : 7/17/2015 4:35:19 PM
Encrypted        : False
Iops              : 90
KmsKeyId         :
Size             : 30
SnapshotId       : snap-12345678
State            : in-use
Tags             : {}
VolumeId         : vol-12345678
```

```
VolumeType      : standard
```

Esempio 2: questo esempio descrive i volumi EBS il cui stato è “available” (Disponibile).

```
Get-EC2Volume -Filter @{ Name="status"; Values="available" }
```

Output:

```
Attachments      : {}
AvailabilityZone  : us-west-2c
CreateTime       : 12/21/2015 2:31:29 PM
Encrypted        : False
Iops             : 60
KmsKeyId         :
Size             : 20
SnapshotId       : snap-12345678
State            : available
Tags            : {}
VolumeId         : vol-12345678
VolumeType       : gp2
...
```

Esempio 3: questo esempio descrive tutti i volumi EBS.

```
Get-EC2Volume
```

- Per i dettagli sull'API, vedere [DescribeVolumes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2VolumeAttribute

Il seguente esempio di codice mostra come utilizzare. Get-EC2VolumeAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'attributo specificato del volume specificato.

```
Get-EC2VolumeAttribute -VolumeId vol-12345678 -Attribute AutoEnableIO
```

Output:

AutoEnableIO	ProductCodes	VolumeId
-----	-----	-----
False	{}	vol-12345678

- Per i dettagli sull'API, vedere [DescribeVolumeAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2VolumeStatus

Il seguente esempio di codice mostra come utilizzare. Get-EC2VolumeStatus

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive lo stato del volume specificato.

```
Get-EC2VolumeStatus -VolumeId vol-12345678
```

Output:

```
Actions           : {}
AvailabilityZone   : us-west-2a
Events            : {}
VolumeId          : vol-12345678
VolumeStatus      : Amazon.EC2.Model.VolumeStatusInfo
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus
```

Output:

Details	Status
-----	-----
{io-enabled, io-performance}	ok

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus.Details
```

Output:

Name	Status

----	-----
io-enabled	passed
io-performance	not-applicable

- Per i dettagli sull'API, vedere [DescribeVolumeStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2Vpc

Il seguente esempio di codice mostra come utilizzare. Get-EC2Vpc

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il VPC specificato.

```
Get-EC2Vpc -VpcId vpc-12345678
```

Output:

```
CidrBlock      : 10.0.0.0/16
DhcpOptionsId  : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault      : False
State          : available
Tags           : {Name}
VpcId          : vpc-12345678
```

Esempio 2: questo esempio descrive il VPC predefinito (può essercene solo uno per Regione). Se il tuo account supporta EC2 -Classic in questa regione, non esiste un VPC predefinito.

```
Get-EC2Vpc -Filter @{"Name"="isDefault"; Values="true"}
```

Output:

```
CidrBlock      : 172.31.0.0/16
DhcpOptionsId  : dopt-12345678
InstanceTenancy : default
IsDefault      : True
State          : available
Tags           : {}
```

```
VpcId : vpc-45678901
```

Esempio 3: questo esempio descrive quelli VPCs che corrispondono al filtro specificato (ovvero, hanno un CIDR che corrisponde al valore '10.0.0.0/16' e sono nello stato 'disponibile').

```
Get-EC2Vpc -Filter @{Name="cidr";  
Values="10.0.0.0/16"},@{Name="state";Values="available"}
```

Esempio 4: Questo esempio descrive tutti i tuoi VPCs

```
Get-EC2Vpc
```

- Per i dettagli sull'API, vedere [DescribeVpcs](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2VpcAttribute

Il seguente esempio di codice mostra come utilizzare Get-EC2VpcAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'attributo 'enableDnsSupport'.

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsSupport
```

Output:

```
EnableDnsSupport  
-----  
True
```

Esempio 2: questo esempio descrive l'attributo enableDnsHostnames ".

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsHostnames
```

Output:

```
EnableDnsHostnames  
-----
```

```
True
```

- Per i dettagli sull'API, vedere [DescribeVpcAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2VpcClassicLink

Il seguente esempio di codice mostra come utilizzare. Get-EC2VpcClassicLink

Strumenti per PowerShell V4

Esempio 1: L'esempio precedente restituisce tutti VPCs con il relativo ClassicLinkEnabled stato per la regione

```
Get-EC2VpcClassicLink -Region eu-west-1
```

Output:

ClassicLinkEnabled	Tags	VpcId
-----	----	-----
False	{Name}	vpc-0fc1ff23f45b678eb
False	{}	vpc-01e23c4a5d6db78e9
False	{Name}	vpc-0123456b078b9d01f
False	{}	vpc-12cf3b4f
False	{Name}	vpc-0b12d3456a7e8901d

- Per i dettagli sull'API, vedere [DescribeVpcClassicLink](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2VpcClassicLinkDnsSupport

Il seguente esempio di codice mostra come utilizzare. Get-EC2VpcClassicLinkDnsSupport

Strumenti per PowerShell V4

Esempio 1: Questo esempio descrive lo stato del supporto ClassicLink DNS di VPCs per la regione eu-west-1

```
Get-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

Output:

```
ClassicLinkDnsSupported VpcId
-----
False                  vpc-0b12d3456a7e8910d
False                  vpc-12cf3b4f
```

- Per i dettagli sull'API, vedere [DescribeVpcClassicLinkDnsSupport](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2VpcEndpoint

Il seguente esempio di codice mostra come utilizzare. Get-EC2VpcEndpoint

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive uno o più endpoint VPC per la Regione eu-west-1. Quindi reindirizza l'output al comando successivo, che seleziona la VpcEndpointId proprietà e restituisce l'ID VPC dell'array come array di stringhe

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object -ExpandProperty VpcEndpointId
```

Output:

```
vpce-01a2ab3f4f5cc6f7d
vpce-01d2b345a6787890b
vpce-0012e34d567890e12
vpce-0c123db4567890123
```

Esempio 2: Questo esempio descrive tutti gli endpoint vpc per la regione eu-west-1 e seleziona VpcEndpointId ServiceName e PrivateDnsEnabled proprietà per presentarla in un VpcId formato tabulare

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object VpcEndpointId, VpcId,
ServiceName, PrivateDnsEnabled | Format-Table -AutoSize
```

Output:

```
VpcEndpointId      VpcId      ServiceName
PrivateDnsEnabled
```



```

-----
-----
-----
vpce-02a2ab2f2f2cc2f2d vpce-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ssm
    True
vpce-01d1b111a1114561b vpce-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ec2
    True
vpce-0011e23d45167e838 vpce-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ec2messages
    True
vpce-0c123db4567890123 vpce-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ssmmessages
    True

```

Esempio 3: questo esempio esporta il documento di policy per l'endpoint VPC
vpce-01a2ab3f4f5cc6f7d into a json file

```
Get-EC2VpcEndpoint -Region eu-west-1 -VpcEndpointId vpce-01a2ab3f4f5cc6f7d | Select-Object -expand PolicyDocument | Out-File vpce_policyDocument.json
```

- Per i dettagli sull'API, vedere [DescribeVpcEndpoints](#) in Cmdlet Reference (V4).AWS Strumenti per PowerShell

Get-EC2VpcEndpointService

Il seguente esempio di codice mostra come utilizzare. Get-EC2VpcEndpointService

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il servizio endpoint EC2 VPC con il filtro specificato, in questo caso com.amazonaws.eu-west-1.ecs. Inoltre, ServiceDetails espande la proprietà e visualizza i dettagli

```
Get-EC2VpcEndpointService -Region eu-west-1 -MaxResult 5 -Filter @{Name="service-name";Values="com.amazonaws.eu-west-1.ecs"} | Select-Object -ExpandProperty ServiceDetails
```

Output:

```

AcceptanceRequired      : False
AvailabilityZones       : {eu-west-1a, eu-west-1b, eu-west-1c}
BaseEndpointDnsNames    : {ecs.eu-west-1.vpce.amazonaws.com}
Owner                   : amazon
PrivateDnsName          : ecs.eu-west-1.amazonaws.com

```

```

ServiceName           : com.amazonaws.eu-west-1.ecs
ServiceType           : {Amazon.EC2.Model.ServiceTypeDetail}
VpcEndpointPolicySupported : False

```

Esempio 2: questo esempio recupera tutti i servizi EC2 VPC Endpoint e restituisce ServiceNames il corrispondente «ssm»

```

Get-EC2VpcEndpointService -Region eu-west-1 | Select-Object -ExpandProperty
  Servicenames | Where-Object { -match "ssm"}

```

Output:

```

com.amazonaws.eu-west-1.ssm
com.amazonaws.eu-west-1.ssmmessages

```

- Per i dettagli sull'API, vedere [DescribeVpcEndpointServices](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2VpnConnection

Il seguente esempio di codice mostra come utilizzare. Get-EC2VpnConnection

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive la connessione VPN specificata.

```

Get-EC2VpnConnection -VpnConnectionId vpn-12345678

```

Output:

```

CustomerGatewayConfiguration : [XML document]
CustomerGatewayId           : cgw-1a2b3c4d
Options                     : Amazon.EC2.Model.VpnConnectionOptions
Routes                     : {Amazon.EC2.Model.VpnStaticRoute}
State                      : available
Tags                      : {}
Type                      : ipsec.1
VgwTelemetry               : {Amazon.EC2.Model.VgwTelemetry,
  Amazon.EC2.Model.VgwTelemetry}
VpnConnectionId           : vpn-12345678

```

```
VpnGatewayId : vgw-1a2b3c4d
```

Esempio 2: questo esempio descrive qualsiasi connessione VPN il cui stato è “pending” (In sospeso) o “available” (Disponibile).

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnConnection -Filter $filter
```

Esempio 3: questo esempio descrive tutte le connessioni VPN.

```
Get-EC2VpnConnection
```

- Per i dettagli sull'API, vedere [DescribeVpnConnections](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EC2VpnGateway

Il seguente esempio di codice mostra come utilizzare. Get-EC2VpnGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il gateway privato virtuale specificato.

```
Get-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

Output:

```
AvailabilityZone :
State           : available
Tags            : {}
Type            : ipsec.1
VpcAttachments  : {vpc-12345678}
VpnGatewayId    : vgw-1a2b3c4d
```

Esempio 2: questo esempio descrive qualsiasi gateway privato virtuale il cui stato è “pending” (In sospeso) o “available” (Disponibile).

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnGateway -Filter $filter
```

Esempio 3: questo esempio descrive tutti i gateway privati virtuali.

```
Get-EC2VpnGateway
```

- Per i dettagli sull'API, vedere [DescribeVpnGateways](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Grant-EC2SecurityGroupEgress

Il seguente esempio di codice mostra come utilizzare. Grant-EC2SecurityGroupEgress

Strumenti per PowerShell V4

Esempio 1: questo esempio definisce una regola di uscita per il gruppo di sicurezza specificato per EC2 -VPC. La regola concede l'accesso agli intervalli di indirizzi IP specificati sulla porta TCP 80. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
$ip = @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; IpRanges="203.0.113.0/24" }
Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

Esempio 2: con PowerShell la versione 2, è necessario utilizzare New-Object per creare l'oggetto. IpPermission

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 80
$ip.ToPort = 80
$ip.IpRanges.Add("203.0.113.0/24")

Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

Esempio 3: questo esempio concede l'accesso al gruppo di sicurezza di origine specificato sulla porta TCP 80.

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"

Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission
@ ( @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; UserIdGroupPairs=$ug } )
```

- Per i dettagli sull'API, vedere [AuthorizeSecurityGroupEgress](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Grant-EC2SecurityGroupIngress

Il seguente esempio di codice mostra come utilizzare `Grant-EC2SecurityGroupIngress`

Strumenti per PowerShell V4

Esempio 1: questo esempio definisce le regole di ingresso per un gruppo di sicurezza per EC2 - VPC. Queste regole garantiscono l'accesso a un indirizzo IP specifico per SSH (porta 22) e RDC (porta 3389). Si noti che è necessario identificare i gruppi di sicurezza per EC2 -VPC utilizzando l'ID del gruppo di sicurezza e non il nome del gruppo di sicurezza. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
$ip1 = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22"; IpRanges="203.0.113.25/32" }
$ip2 = @{ IpProtocol="tcp"; FromPort="3389"; ToPort="3389";
  IpRanges="203.0.113.25/32" }

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission @( $ip1, $ip2 )
```

Esempio 2: con PowerShell la versione 2, è necessario utilizzare `New-Object` per creare gli oggetti. `IpPermission`

```
$ip1 = New-Object Amazon.EC2.Model.IpPermission
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")

$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
```

```
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission @( $ip1, $ip2 )
```

Esempio 3: questo esempio definisce le regole di ingresso per un gruppo di sicurezza per - Classic. EC2 Queste regole garantiscono l'accesso a un indirizzo IP specifico per SSH (porta 22) e RDC (porta 3389). La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
$ip1 = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22"; IpRanges="203.0.113.25/32" }
$ip2 = @{ IpProtocol="tcp"; FromPort="3389"; ToPort="3389";
  IpRanges="203.0.113.25/32" }

Grant-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission @( $ip1,
  $ip2 )
```

Esempio 4: con PowerShell la versione 2, è necessario utilizzare New-Object per creare gli oggetti. IpPermission

```
$ip1 = New-Object Amazon.EC2.Model.IpPermission
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")

$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")

Grant-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission @( $ip1,
  $ip2 )
```

Esempio 5: questo esempio concede l'accesso alla porta TCP 8081 dal gruppo di sicurezza di origine specificato (sg-1a2b3c4d) al gruppo di sicurezza specificato (sg-12345678).

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"
```

```
Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission
@( @{ IpProtocol="tcp"; FromPort="8081"; ToPort="8081"; UserIdGroupPairs=$ug } )
```

Esempio 6: questo esempio aggiunge il CIDR 5.5.5.5/32 alle regole di ingresso del gruppo di sicurezza sg-1234abcd per il traffico della porta TCP 22 con una descrizione.

```
$IpRange = New-Object -TypeName Amazon.EC2.Model.IpRange
$IpRange.CidrIp = "5.5.5.5/32"
$IpRange.Description = "SSH from Office"
$IpPermission = New-Object Amazon.EC2.Model.IpPermission
$IpPermission.IpProtocol = "tcp"
$IpPermission.ToPort = 22
$IpPermission.FromPort = 22
$IpPermission.Ipv4Ranges = $IpRange
Grant-EC2SecurityGroupIngress -GroupId sg-1234abcd -IpPermission $IpPermission
```

- Per i dettagli sull'API, vedere [AuthorizeSecurityGroupIngress](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Import-EC2Image

Il seguente esempio di codice mostra come utilizzare. Import-EC2Image

Strumenti per PowerShell V4

Esempio 1: questo esempio importa un'immagine di macchina virtuale a disco singolo dal bucket Amazon S3 specificato su Amazon con un token di EC2 idempotenza. L'esempio richiede che esista un ruolo VM Import Service con il nome predefinito «vmimport», con una policy che consenta ad EC2 Amazon l'accesso al bucket specificato, come spiegato nell'argomento VM Import Prerequisites. Per utilizzare un ruolo personalizzato, specifica il nome del ruolo utilizzando il parametro **-RoleName**.

```
$container = New-Object Amazon.EC2.Model.ImageDiskContainer
$container.Format="VMDK"
$container.UserBucket = New-Object Amazon.EC2.Model.UserBucket
$container.UserBucket.S3Bucket = "amzn-s3-demo-bucket"
$container.UserBucket.S3Key = "Win_2008_Server_Standard_SP2_64-bit-disk1.vmdk"

$params = @{
```

```
"ClientToken"="idempotencyToken"
"Description"="Windows 2008 Standard Image Import"
"Platform"="Windows"
"LicenseType"="AWS"
}
```

```
Import-EC2Image -DiskContainer $container @parms
```

Output:

```
Architecture      :
Description       : Windows 2008 Standard Image
Hypervisor        :
ImageId           :
ImportTaskId      : import-ami-abcdefgh
LicenseType       : AWS
Platform          : Windows
Progress          : 2
SnapshotDetails   : {}
Status            : active
StatusMessage     : pending
```

- Per i dettagli sull'API, vedere [ImportImage](#) in Cmdlet Reference (V4).AWS Strumenti per PowerShell

Import-EC2KeyPair

Il seguente esempio di codice mostra come utilizzare. `Import-EC2KeyPair`

Strumenti per PowerShell V4

Esempio 1: Questo esempio importa una chiave pubblica in EC2. La prima riga archivia il contenuto del file della chiave pubblica (*.pub) nella variabile **\$publickey**. Successivamente, l'esempio converte il UTF8 formato del file della chiave pubblica in una stringa con codifica Base64 e memorizza la stringa convertita nella variabile. **\$pkbase64** Nell'ultima riga, la chiave pubblica convertita viene importata in. EC2 Il cmdlet restituisce l'impronta digitale e il nome della chiave come risultati.

```
$publickey=[Io.File]::ReadAllText("C:\Users\TestUser\.ssh\id_rsa.pub")
$pkbase64 =
[System.Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes($publickey))
```



```
Import-EC2KeyPair -KeyName Example-user-key -PublicKey $pkbase64
```

Output:

KeyFingerprint	KeyName
-----	-----
do:d0:15:8f:79:97:12:be:00:fd:df:31:z3:b1:42:z1	Example-user-key

- Per i dettagli sull'API, vedere [ImportKeyPair](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Import-EC2Snapshot

Il seguente esempio di codice mostra come utilizzare. Import-EC2Snapshot

Strumenti per PowerShell V4

Esempio 1: questo esempio importa un'immagine del disco di una macchina virtuale in formato "VMDK" in uno snapshot Amazon EBS. L'esempio richiede un ruolo VM Import Service con il nome predefinito «vmimport», con una politica che consenta ad EC2 Amazon l'accesso al bucket specificato, come spiegato nell'argomento in **VM Import Prerequisites** <http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/VMImportPrerequisites.html>. Per utilizzare un ruolo personalizzato, specifica il nome del ruolo utilizzando il parametro **-RoleName**.

```
$parms = @{
    "ClientToken"="idempotencyToken"
    "Description"="Disk Image Import"
    "DiskContainer_Description" = "Data disk"
    "DiskContainer_Format" = "VMDK"
    "DiskContainer_S3Bucket" = "amzn-s3-demo-bucket"
    "DiskContainer_S3Key" = "datadiskimage.vmdk"
}

Import-EC2Snapshot @parms
```

Output:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----

```
Disk Image Import      import-snap-abcdefgh  
Amazon.EC2.Model.SnapshotTaskDetail
```

- Per i dettagli sull'API, vedere [ImportSnapshot](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Move-EC2AddressToVpc

Il seguente esempio di codice mostra come utilizzare. Move-EC2AddressToVpc

Strumenti per PowerShell V4

Esempio 1: questo esempio sposta un' EC2 istanza con un indirizzo IP pubblico 12.345.67.89 sulla piattaforma EC2 -VPC nella regione Stati Uniti orientali (Virginia del Nord).

```
Move-EC2AddressToVpc -PublicIp 12.345.67.89 -Region us-east-1
```

Esempio 2: questo esempio reindirizza i risultati di un comando al cmdlet. Get-EC2Instance Move-EC2AddressToVpc Il Get-EC2Instance comando ottiene un'istanza specificata dall'ID dell'istanza, quindi restituisce la proprietà dell'indirizzo IP pubblico dell'istanza.

```
(Get-EC2Instance -Instance i-12345678).Instances.PublicIpAddress | Move-  
EC2AddressToVpc
```

- Per i dettagli sull'API, vedere [MoveAddressToVpc](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2Address

Il seguente esempio di codice mostra come utilizzare. New-EC2Address

Strumenti per PowerShell V4

Esempio 1: questo esempio alloca un indirizzo IP elastico da utilizzare con un'istanza in un VPC.

```
New-EC2Address -Domain Vpc
```

Output:

AllocationId	Domain	PublicIp
-----	-----	-----
eipalloc-12345678	vpc	198.51.100.2

Esempio 2: Questo esempio alloca un indirizzo IP elastico da utilizzare con un'istanza in EC2 - Classic.

```
New-EC2Address
```

Output:

AllocationId	Domain	PublicIp
-----	-----	-----
	standard	203.0.113.17

- Per i dettagli sull'API, vedere [AllocateAddress](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2CustomerGateway

Il seguente esempio di codice mostra come utilizzare `New-EC2CustomerGateway`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea il gateway del cliente specificato.

```
New-EC2CustomerGateway -Type ipsec.1 -PublicIp 203.0.113.12 -BgpAsn 65534
```

Output:

```
BgpAsn           : 65534
CustomerGatewayId : cgw-1a2b3c4d
IpAddress        : 203.0.113.12
State            : available
Tags             : {}
Type             : ipsec.1
```

- Per i dettagli sull'API, vedere [CreateCustomerGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2DhcpOption

Il seguente esempio di codice mostra come utilizzare `New-EC2DhcpOption`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea il set specificato di opzioni DHCP. La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
$options = @( @{Key="domain-name";Values=@("abc.local")}, @{Key="domain-name-servers";Values=@("10.0.0.101","10.0.0.102")})
New-EC2DhcpOption -DhcpConfiguration $options
```

Output:

DhcpConfigurations	DhcpOptionsId	Tags
-----	-----	----
{domain-name, domain-name-servers}	dopt-1a2b3c4d	{}

Esempio 2: con la PowerShell versione 2, è necessario utilizzare `New-Object` per creare ogni opzione DHCP.

```
$option1 = New-Object Amazon.EC2.Model.DhcpConfiguration
$option1.Key = "domain-name"
$option1.Values = "abc.local"

$option2 = New-Object Amazon.EC2.Model.DhcpConfiguration
$option2.Key = "domain-name-servers"
$option2.Values = @("10.0.0.101","10.0.0.102")

New-EC2DhcpOption -DhcpConfiguration @($option1, $option2)
```

Output:

DhcpConfigurations	DhcpOptionsId	Tags
-----	-----	----
{domain-name, domain-name-servers}	dopt-2a3b4c5d	{}

- Per i dettagli sull'API, vedere [CreateDhcpOptions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2FlowLog

Il seguente esempio di codice mostra come utilizzare. New-EC2FlowLog

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un EC2 flowlog per la sottorete subnet-1d234567 e il file cloud-watch-log denominato «subnet1-log» per tutto il traffico «REJECT» utilizzando le autorizzazioni del ruolo «Amministratore»

```
New-EC2FlowLog -ResourceId "subnet-1d234567" -LogDestinationType cloud-watch-logs -LogGroupName subnet1-log -TrafficType "REJECT" -ResourceType Subnet -DeliverLogsPermissionArn "arn:aws:iam::98765432109:role/Admin"
```

Output:

ClientToken	FlowLogIds	Unsuccessful
-----	-----	-----
m1VN2cxP3iB4qo//VUK15EU6cF7gQL0xcqNefvjeTGw=	{f1-012fc34eed5678c9d}	{}

- Per i [CreateFlowLogs](#) AWS Strumenti per PowerShell dettagli sull'API, vedere in Cmdlet Reference (V4).

New-EC2Host

Il seguente esempio di codice mostra come utilizzare. New-EC2Host

Strumenti per PowerShell V4

Esempio 1: questo esempio alloca un host dedicato all'account per il tipo di istanza e la zona di disponibilità specificati

```
New-EC2Host -AutoPlacement on -AvailabilityZone eu-west-1b -InstanceType m4.xlarge -Quantity 1
```

Output:

```
h-01e23f4cd567890f3
```

- Per i dettagli sull'API, vedere [AllocateHosts](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2HostReservation

Il seguente esempio di codice mostra come utilizzare. New-EC2HostReservation

Strumenti per PowerShell V4

Esempio 1: questo esempio acquista l'offerta di prenotazione hro-0c1f23456789d0ab con configurazioni che corrispondono a quelle dell'host dedicato h-01e23f4cd567890f1

```
New-EC2HostReservation -OfferingId hro-0c1f23456789d0ab HostIdSet  
h-01e23f4cd567890f1
```

Output:

```
ClientToken      :  
CurrencyCode     :  
Purchase         : {hr-0123f4b5d67bedc89}  
TotalHourlyPrice : 1.307  
TotalUpfrontPrice : 0.000
```

- Per i dettagli sull'API, vedere [PurchaseHostReservation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2Image

Il seguente esempio di codice mostra come utilizzare. New-EC2Image

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un'AMI senza il nome e la descrizione specificati dall'istanza specificata. Amazon EC2 tenta di chiudere definitivamente l'istanza prima di creare l'immagine e riavvia l'istanza al termine.

```
New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description "My web  
server AMI"
```

Esempio 2: questo esempio crea un'AMI senza il nome e la descrizione specificati dall'istanza specificata. Amazon EC2 crea l'immagine senza chiudere e riavviare l'istanza; pertanto, l'integrità del file system sull'immagine creata non può essere garantita.

```
New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description "My web
server AMI" -NoReboot $true
```

Esempio 3: questo esempio crea un'AMI con tre volumi. Il primo volume è basato su uno snapshot Amazon EBS. Il secondo volume è un volume Amazon EBS vuoto da 100 GiB. Il terzo volume è un volume di archivio dell'istanza. La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
$ebsBlock1 = @{SnapshotId="snap-1a2b3c4d"}
$ebsBlock2 = @{VolumeSize=100}

New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description
"My web server AMI" -BlockDeviceMapping @( @{DeviceName="/dev/sdf";Ebs=
$ebsBlock1}, @{DeviceName="/dev/sdg";Ebs=$ebsBlock2}, @{DeviceName="/dev/
sdc";VirtualName="ephemeral0"})
```

- Per i dettagli sull'API, vedere [CreateImage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2Instance

Il seguente esempio di codice mostra come utilizzare. New-EC2Instance

Strumenti per PowerShell V4

Esempio 1: questo esempio avvia una singola istanza dell'AMI specificato in EC2 -Classic o un VPC predefinito.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -InstanceType
m3.medium -KeyName my-key-pair -SecurityGroup my-security-group
```

Esempio 2: questo esempio avvia un'unica istanza dell'AMI specificata in un VPC.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -SubnetId
subnet-12345678 -InstanceType t2.micro -KeyName my-key-pair -SecurityGroupId
sg-12345678
```

Esempio 3: come aggiungere un volume EBS o un volume dell'archivio dell'istanza, definire una mappatura dei dispositivi a blocchi e aggiungerla al comando. In questo esempio viene aggiunto un volume dell'archivio dell'istanza.

```
$bdm = New-Object Amazon.EC2.Model.BlockDeviceMapping
$bdm.VirtualName = "ephemeral0"
$bdm.DeviceName = "/dev/sdf"

New-EC2Instance -ImageId ami-12345678 -BlockDeviceMapping $bdm ...
```

Esempio 4: per specificare uno dei Windows correnti AMIs, ottieni il relativo ID AMI utilizzando `Get-EC2ImageByName`. Questo esempio avvia un'istanza dall'AMI di base corrente per Windows Server 2016.

```
$ami = Get-EC2ImageByName WINDOWS_2016_BASE

New-EC2Instance -ImageId $ami.ImageId ...
```

Esempio 5: avvia un'istanza nell'ambiente dell'host dedicato specificato.

```
New-EC2Instance -ImageId ami-1a2b3c4d -InstanceType m4.large -KeyName my-key-pair
-SecurityGroupId sg-1a2b3c4d -AvailabilityZone us-west-1a -Tenancy host -HostID
h-1a2b3c4d5e6f1a2b3
```

Esempio 6: questa richiesta avvia due istanze e applica alle istanze un tag con la chiave "webserver" e il valore "production". La richiesta applica inoltre un tag con la chiave cost-center e il valore cc123 ai volumi creati (in questo caso il volume root per ogni istanza).

```
$tag1 = @{ Key="webserver"; Value="production" }
$tag2 = @{ Key="cost-center"; Value="cc123" }

$tagspec1 = new-object Amazon.EC2.Model.TagSpecification
$tagspec1.ResourceType = "instance"
$tagspec1.Tags.Add($tag1)

$tagspec2 = new-object Amazon.EC2.Model.TagSpecification
$tagspec2.ResourceType = "volume"
$tagspec2.Tags.Add($tag2)

New-EC2Instance -ImageId "ami-1a2b3c4d" -KeyName "my-key-pair" -MaxCount 2 -
InstanceType "t2.large" -SubnetId "subnet-1a2b3c4d" -TagSpecification $tagspec1,
$tagspec2
```

- Per i dettagli sull'API, vedere [RunInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2InstanceExportTask

Il seguente esempio di codice mostra come utilizzare. New-EC2InstanceExportTask

Strumenti per PowerShell V4

Esempio 1: questo esempio esporta l'istanza arrestata **i-0800b00a00EXAMPLE** come disco rigido virtuale (VHD) nel bucket S3 **testbucket-export-instances-2019**. L'ambiente di destinazione è **Microsoft**, e il parametro region viene aggiunto perché l'istanza si trova nella **us-east-1** regione, mentre la AWS regione predefinita dell'utente non è us-east-1. Per ottenere lo stato dell'attività di esportazione, copia il valore **ExportTaskId** dai risultati di questo comando, quindi esegui **Get-EC2ExportTask -ExportTaskId export_task_ID_from_results..**

```
New-EC2InstanceExportTask -InstanceId i-0800b00a00EXAMPLE -  
ExportToS3Task_DiskImageFormat VHD -ExportToS3Task_S3Bucket "amzn-s3-demo-bucket" -  
TargetEnvironment Microsoft -Region us-east-1
```

Output:

```
Description          :  
ExportTaskId         : export-i-077c73108aEXAMPLE  
ExportToS3Task       : Amazon.EC2.Model.ExportToS3Task  
InstanceExportDetails : Amazon.EC2.Model.InstanceExportDetails  
State                : active  
StatusMessage        :
```

- Per i dettagli sull'API, vedere [CreateInstanceExportTask](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2InternetGateway

Il seguente esempio di codice mostra come utilizzare. New-EC2InternetGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un gateway Internet.

```
New-EC2InternetGateway
```

Output:

Attachments	InternetGatewayId	Tags
-----	-----	----
{}	igw-1a2b3c4d	{}

- Per i dettagli sull'API, vedere [CreateInternetGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2KeyPair

Il seguente esempio di codice mostra come utilizzare. New-EC2KeyPair

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una coppia di chiavi e acquisisce la chiave privata RSA con codifica PEM in un file con il nome specificato. Quando si utilizza PowerShell, la codifica deve essere impostata su ascii per generare una chiave valida. Per ulteriori informazioni, consulta Create, Display and Delete Amazon EC2 Key Pairs (<https://docs.aws.amazon.com/cli/latest/userguide/cli-services-ec2-keypairs.html>) nella AWS Command Line Interface User Guide.

```
(New-EC2KeyPair -KeyName "my-key-pair").KeyMaterial | Out-File -Encoding ascii -
FilePath C:\path\my-key-pair.pem
```

- Per i dettagli sull'API, vedere [CreateKeyPair](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2NetworkAcl

Il seguente esempio di codice mostra come utilizzare. New-EC2NetworkAcl

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una lista di controllo degli accessi alla rete (ACL) per il VPC specificato.

```
New-EC2NetworkAcl -VpcId vpc-12345678
```

Output:

```
Associations : {}
```

```

Entries      : {Amazon.EC2.Model.NetworkAclEntry, Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {}
VpcId        : vpc-12345678

```

- Per i dettagli sull'API, vedere [CreateNetworkAcl](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2NetworkAclEntry

Il seguente esempio di codice mostra come utilizzare. New-EC2NetworkAclEntry

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una voce per la lista di controllo degli accessi alla rete (ACL) specificata. La regola consente il traffico in entrata da qualsiasi posizione (0.0.0.0/0) sulla porta UDP 53 (DNS) in qualsiasi sottorete associata.

```

New-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100
-Protocol 17 -PortRange_From 53 -PortRange_To 53 -CidrBlock 0.0.0.0/0 -RuleAction
allow

```

- Per i dettagli sull'API, vedere [CreateNetworkAclEntry](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2NetworkInterface

Il seguente esempio di codice mostra come utilizzare. New-EC2NetworkInterface

Strumenti per PowerShell V4

Esempio 1: questo esempio crea l'interfaccia di rete specificata.

```

New-EC2NetworkInterface -SubnetId subnet-1a2b3c4d -Description "my network
interface" -Group sg-12345678 -PrivateIpAddress 10.0.0.17

```

Output:

```

Association      :
Attachment       :

```

```
AvailabilityZone : us-west-2c
Description      : my network interface
Groups           : {my-security-group}
MacAddress       : 0a:72:bc:1a:cd:7f
NetworkInterfaceId : eni-12345678
OwnerId          : 123456789012
PrivateDnsName   : ip-10-0-0-17.us-west-2.compute.internal
PrivateIpAddress : 10.0.0.17
PrivateIpAddresses : {}
RequesterId      :
RequesterManaged : False
SourceDestCheck  : True
Status           : pending
SubnetId         : subnet-1a2b3c4d
TagSet           : {}
VpcId            : vpc-12345678
```

- Per i dettagli sull'API, vedere [CreateNetworkInterface](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2PlacementGroup

Il seguente esempio di codice mostra come utilizzare. New-EC2PlacementGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un gruppo di posizionamento con il nome specificato.

```
New-EC2PlacementGroup -GroupName my-placement-group -Strategy cluster
```

- Per i dettagli sull'API, vedere [CreatePlacementGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2Route

Il seguente esempio di codice mostra come utilizzare. New-EC2Route

Strumenti per PowerShell V4

Esempio 1: questo esempio crea la route specificata per la tabella di routing specificata. La route acquisisce tutto il traffico e lo indirizza al gateway Internet specificato.

```
New-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0 -GatewayId igw-1a2b3c4d
```

Output:

```
True
```

- Per i dettagli sull'API, vedere [CreateRoute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2RouteTable

Il seguente esempio di codice mostra come utilizzare. New-EC2RouteTable

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una tabella di routing per il VPC specificato.

```
New-EC2RouteTable -VpcId vpc-12345678
```

Output:

```
Associations      : {}  
PropagatingVgws   : {}  
Routes            : {}  
RouteTableId      : rtb-1a2b3c4d  
Tags              : {}  
VpcId             : vpc-12345678
```

- Per i dettagli sull'API, vedere [CreateRouteTable](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2ScheduledInstance

Il seguente esempio di codice mostra come utilizzare. New-EC2ScheduledInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio avvia l'istanza pianificata specificata.

```
New-EC2ScheduledInstance -ScheduledInstanceId sci-1234-1234-1234-1234-123456789012 -
InstanceCount 1 `
-IamInstanceProfile_Name my-iam-role `
-LaunchSpecification_ImageId ami-12345678 `
-LaunchSpecification_InstanceType c4.large `
-LaunchSpecification_SubnetId subnet-12345678`
-LaunchSpecification_SecurityGroupId sg-12345678
```

- Per i dettagli sull'API, vedere [RunScheduledInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2ScheduledInstancePurchase

Il seguente esempio di codice mostra come utilizzare. New-EC2ScheduledInstancePurchase
Strumenti per PowerShell V4

Esempio 1: questo esempio acquista un'istanza pianificata.

```
$request = New-Object Amazon.EC2.Model.PurchaseRequest
$request.InstanceCount = 1
$request.PurchaseToken = "eyJ2IjoiMSIsInMiOiEsImMiOi..."
New-EC2ScheduledInstancePurchase -PurchaseRequest $request
```

Output:

```
AvailabilityZone      : us-west-2b
CreateDate            : 1/25/2016 1:43:38 PM
HourlyPrice           : 0.095
InstanceCount        : 1
InstanceType         : c4.large
NetworkPlatform      : EC2-VPC
NextSlotStartTime     : 1/31/2016 1:00:00 AM
Platform             : Linux/UNIX
PreviousSlotEndTime   :
Recurrence            : Amazon.EC2.Model.ScheduledInstanceRecurrence
ScheduledInstanceId   : sci-1234-1234-1234-1234-123456789012
SlotDurationInHours   : 32
TermEndDate          : 1/31/2017 1:00:00 AM
TermStartDate        : 1/31/2016 1:00:00 AM
TotalScheduledInstanceHours : 1696
```

- Per i dettagli sull'API, vedere [PurchaseScheduledInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2SecurityGroup

Il seguente esempio di codice mostra come utilizzare. New-EC2SecurityGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un gruppo di sicurezza per il VPC specificato.

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security group" -VpcId vpc-12345678
```

Output:

```
sg-12345678
```

Esempio 2: Questo esempio crea un gruppo di sicurezza per EC2 -Classic.

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security group"
```

Output:

```
sg-45678901
```

- Per i dettagli sull'API, vedere [CreateSecurityGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2Snapshot

Il seguente esempio di codice mostra come utilizzare. New-EC2Snapshot

Strumenti per PowerShell V4

Esempio 1: questo esempio crea uno snapshot del volume specificato.

```
New-EC2Snapshot -VolumeId vol-12345678 -Description "This is a test"
```

Output:

```
DataEncryptionKeyId :  
Description         : This is a test  
Encrypted           : False  
KmsKeyId            :  
OwnerAlias          :  
OwnerId             : 123456789012  
Progress            :  
SnapshotId          : snap-12345678  
StartTime           : 12/22/2015 1:28:42 AM  
State               : pending  
StateMessage        :  
Tags                : {}  
VolumeId            : vol-12345678  
VolumeSize          : 20
```

- Per i dettagli sull'API, vedere [CreateSnapshot](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2SpotDatafeedSubscription

Il seguente esempio di codice mostra come utilizzare `New-EC2SpotDatafeedSubscription`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un feed di dati dell'istanza spot.

```
New-EC2SpotDatafeedSubscription -Bucket amzn-s3-demo-bucket -Prefix spotdata
```

Output:

```
Bucket   : amzn-s3-demo-bucket  
Fault    :  
OwnerId  : 123456789012  
Prefix   : spotdata  
State    : Active
```

- Per i dettagli sull'API, vedere [CreateSpotDatafeedSubscription](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2Subnet

Il seguente esempio di codice mostra come utilizzare. New-EC2Subnet

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una sottorete con il CIDR specificato.

```
New-EC2Subnet -VpcId vpc-12345678 -CidrBlock 10.0.0.0/24
```

Output:

```
AvailabilityZone      : us-west-2c
AvailableIpAddressCount : 251
CidrBlock             : 10.0.0.0/24
DefaultForAz          : False
MapPublicIpOnLaunch   : False
State                 : pending
SubnetId              : subnet-1a2b3c4d
Tag                   : {}
VpcId                 : vpc-12345678
```

- Per i dettagli sull'API, vedere [CreateSubnet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2Tag

Il seguente esempio di codice mostra come utilizzare. New-EC2Tag

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un unico tag alla risorsa specificata. La chiave del tag è 'myTag' e il valore del tag è ". myTagValue La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
New-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag"; Value="myTagValue" }
```

Esempio 2: questo esempio aggiorna o aggiunge i tag specificati alla risorsa specificata. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
New-EC2Tag -Resource i-12345678 -Tag @( @{ Key="myTag"; Value="newTagValue" },
@{ Key="test"; Value="anotherTagValue" } )
```

Esempio 3: con PowerShell la versione 2, è necessario utilizzare New-Object per creare il tag per il parametro Tag.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"
$tag.Value = "myTagValue"

New-EC2Tag -Resource i-12345678 -Tag $tag
```

- Per i dettagli sull'API, vedere [CreateTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2Volume

Il seguente esempio di codice mostra come utilizzare New-EC2Volume

Strumenti per PowerShell V4

Esempio 1: questo esempio crea il volume specificato.

```
New-EC2Volume -Size 50 -AvailabilityZone us-west-2a -VolumeType gp2
```

Output:

```
Attachments      : {}
AvailabilityZone  : us-west-2a
CreateTime       : 12/22/2015 1:42:07 AM
Encrypted        : False
Iops             : 150
KmsKeyId         :
Size             : 50
SnapshotId       :
State            : creating
Tags             : {}
VolumeId         : vol-12345678
VolumeType       : gp2
```

Esempio 2: questa richiesta di esempio crea un volume e applica un tag con la chiave “stack” e il valore “production”.

```
$tag = @{ Key="stack"; Value="production" }

$tagspec = new-object Amazon.EC2.Model.TagSpecification
$tagspec.ResourceType = "volume"
$tagspec.Tags.Add($tag)

New-EC2Volume -Size 80 -AvailabilityZone "us-west-2a" -TagSpecification $tagspec
```

- Per i dettagli sull'API, vedere [CreateVolume](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2Vpc

Il seguente esempio di codice mostra come utilizzare. New-EC2Vpc

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un VPC con il CIDR specificato. Amazon VPC crea anche un set di opzioni DHCP predefinito, una tabella di routing principale e una lista di controllo degli accessi alla rete (ACL) predefinita.

```
New-EC2VPC -CidrBlock 10.0.0.0/16
```

Output:

```
CidrBlock      : 10.0.0.0/16
DhcpOptionsId  : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault      : False
State          : pending
Tags           : {}
VpcId          : vpc-12345678
```

- Per i dettagli sull'API, vedere [CreateVpc](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2VpcEndpoint

Il seguente esempio di codice mostra come utilizzare. New-EC2VpcEndpoint

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo endpoint VPC per il servizio com.amazonaws.eu-west-1.s3 nel VPC vpc-0fc1ff23f45b678eb

```
New-EC2VpcEndpoint -ServiceName com.amazonaws.eu-west-1.s3 -VpcId
vpc-0fc1ff23f45b678eb
```

Output:

```
ClientToken VpcEndpoint
-----
Amazon.EC2.Model.VpcEndpoint
```

- Per i dettagli sull'API, vedere [CreateVpcEndpoint](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2VpnConnection

Il seguente esempio di codice mostra come utilizzare. New-EC2VpnConnection

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una connessione VPN tra il gateway privato virtuale specificato e il gateway del cliente specificato. L'output include le informazioni di configurazione necessarie all'amministratore di rete, in formato XML.

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
vgw-1a2b3c4d
```

Output:

```
CustomerGatewayConfiguration : [XML document]
CustomerGatewayId            : cgw-1a2b3c4d
Options                      :
Routes                      : {}
```

```

State                : pending
Tags                 : {}
Type                 :
VgwTelemetry         : {}
VpnConnectionId      : vpn-12345678
VpnGatewayId         : vgw-1a2b3c4d

```

Esempio 2: questo esempio crea la connessione VPN e acquisisce la configurazione in un file con il nome specificato.

```

(New-EC2VpnConnection -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
vgw-1a2b3c4d).CustomerGatewayConfiguration | Out-File C:\path\vpn-configuration.xml

```

Esempio 3: questo esempio crea una connessione VPN con routing statico tra il gateway privato virtuale specificato e il gateway del cliente specificato.

```

New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
vgw-1a2b3c4d -Options_StaticRoutesOnly $true

```

- Per i dettagli sull'API, vedere [CreateVpnConnection](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2VpnConnectionRoute

Il seguente esempio di codice mostra come utilizzare New-EC2VpnConnectionRoute

Strumenti per PowerShell V4

Esempio 1: questo esempio crea la route statica specificata per la connessione VPN specificata.

```

New-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
11.12.0.0/16

```

- Per i dettagli sull'API, vedere [CreateVpnConnectionRoute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EC2VpnGateway

Il seguente esempio di codice mostra come utilizzare New-EC2VpnGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio crea il gateway privato virtuale specificato.

```
New-EC2VpnGateway -Type ipsec.1
```

Output:

```
AvailabilityZone :  
State           : available  
Tags            : {}  
Type            : ipsec.1  
VpcAttachments  : {}  
VpnGatewayId    : vgw-1a2b3c4d
```

- Per i dettagli sull'API, vedere [CreateVpnGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-EC2Address

Il seguente esempio di codice mostra come utilizzare. Register-EC2Address

Strumenti per PowerShell V4

Esempio 1: questo esempio associa l'indirizzo IP elastico specificato all'istanza specificata in un VPC.

```
C:\> Register-EC2Address -InstanceId i-12345678 -AllocationId eipalloc-12345678
```

Output:

```
eipassoc-12345678
```

Esempio 2: Questo esempio associa l'indirizzo IP elastico specificato all'istanza specificata in EC2 -Classic.

```
C:\> Register-EC2Address -InstanceId i-12345678 -PublicIp 203.0.113.17
```

- Per i dettagli sull'API, vedere [AssociateAddress](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-EC2DhcpOption

Il seguente esempio di codice mostra come utilizzare. Register-EC2DhcpOption

Strumenti per PowerShell V4

Esempio 1: questo esempio associa il set di opzioni DHCP specificato al VPC specificato.

```
Register-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d -VpcId vpc-12345678
```

Esempio 2: questo esempio associa il set di opzioni DHCP predefinite al VPC specificato.

```
Register-EC2DhcpOption -DhcpOptionsId default -VpcId vpc-12345678
```

- Per i dettagli sull'API, vedere [AssociateDhcpOptions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-EC2Image

Il seguente esempio di codice mostra come utilizzare. Register-EC2Image

Strumenti per PowerShell V4

Esempio 1: questo esempio registra un'AMI utilizzando il file manifesto specificato in Amazon S3.

```
Register-EC2Image -ImageLocation amzn-s3-demo-bucket/my-web-server-ami/  
image.manifest.xml -Name my-web-server-ami
```

- Per i dettagli sull'API, vedere [RegisterImage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-EC2PrivateIpAddress

Il seguente esempio di codice mostra come utilizzare. Register-EC2PrivateIpAddress

Strumenti per PowerShell V4

Esempio 1: questo esempio assegna l'indirizzo IP privato secondario specificato all'interfaccia di rete specificata.

```
Register-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -PrivateIpAddress  
10.0.0.82
```

Esempio 2: questo esempio crea due indirizzi IP privati secondari e li assegna all'interfaccia di rete specificata.

```
Register-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -  
SecondaryPrivateIpAddressCount 2
```

- Per i dettagli sull'API, vedere [AssignPrivateIpAddresses](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-EC2RouteTable

Il seguente esempio di codice mostra come utilizzare. Register-EC2RouteTable

Strumenti per PowerShell V4

Esempio 1: questo esempio associa la tabella di routing specificata alla sottorete specificata.

```
Register-EC2RouteTable -RouteTableId rtb-1a2b3c4d -SubnetId subnet-1a2b3c4d
```

Output:

```
rtbassoc-12345678
```

- Per i dettagli sull'API, vedere [AssociateRouteTable](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2Address

Il seguente esempio di codice mostra come utilizzare. Remove-EC2Address

Strumenti per PowerShell V4

Esempio 1: questo esempio rilascia l'indirizzo IP elastico specificato per le istanze in un VPC.

```
Remove-EC2Address -AllocationId eipalloc-12345678 -Force
```


Esempio 2: questo esempio rilascia l'indirizzo IP elastico specificato per le istanze in EC2 - Classic.

```
Remove-EC2Address -PublicIp 198.51.100.2 -Force
```

- Per i dettagli sull'API, vedere [ReleaseAddress](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2CapacityReservation

Il seguente esempio di codice mostra come utilizzare. Remove-EC2CapacityReservation Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la prenotazione della capacità cr-0c1f2345db6f7cdba

```
Remove-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2CapacityReservation (CancelCapacityReservation)"
on target "cr-0c1f2345db6f7cdba".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): y
True
```

- Per i dettagli sull'API, vedere [CancelCapacityReservation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2CustomerGateway

Il seguente esempio di codice mostra come utilizzare. Remove-EC2CustomerGateway Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il gateway del cliente specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2CustomerGateway (DeleteCustomerGateway)" on Target
"cgw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteCustomerGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2DhcpOption

Il seguente esempio di codice mostra come utilizzare. Remove-EC2DhcpOption

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il set di opzioni DHCP specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2DhcpOption (DeleteDhcpOptions)" on Target
"dopt-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteDhcpOptions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2FlowLog

Il seguente esempio di codice mostra come utilizzare. Remove-EC2FlowLog

Strumenti per PowerShell V4

Esempio 1: Questo esempio rimuove il dato FlowLogId fl-01a2b3456a789c01

```
Remove-EC2FlowLog -FlowLogId fl-01a2b3456a789c01
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2FlowLog (DeleteFlowLogs)" on target
"fl-01a2b3456a789c01".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4). [DeleteFlowLogs](#) AWS Strumenti per PowerShell

Remove-EC2Host

Il seguente esempio di codice mostra come utilizzare. Remove-EC2Host

Strumenti per PowerShell V4

Esempio 1: questo esempio rilascia l'ID host specificato h-0badafd1dcb2f3456

```
Remove-EC2Host -HostId h-0badafd1dcb2f3456
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Host (ReleaseHosts)" on target
"h-0badafd1dcb2f3456".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

Successful                Unsuccessful
-----
{h-0badafd1dcb2f3456} {}
```

- Per i dettagli sull'API, vedere [ReleaseHosts](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2Instance

Il seguente esempio di codice mostra come utilizzare. Remove-EC2Instance

Strumenti per PowerShell V4

Esempio 1: questo esempio termina l'istanza specificata (l'istanza può essere in esecuzione o con stato "Arrestato). Il cmdlet richiederà una conferma prima di procedere; utilizza l'opzione -Force per disabilitare la conferma.

```
Remove-EC2Instance -InstanceId i-12345678
```

Output:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

- Per i dettagli sull'API, vedere [TerminateInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2InternetGateway

Il seguente esempio di codice mostra come utilizzare. Remove-EC2InternetGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il gateway Internet specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

Output:

```
Confirm
Are you sure you want to perform this action?
```

```
Performing operation "Remove-EC2InternetGateway (DeleteInternetGateway)" on Target
"igw-1a2b3c4d".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteInternetGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2KeyPair

Il seguente esempio di codice mostra come utilizzare. Remove-EC2KeyPair

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la coppia di chiavi specificata. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2KeyPair -KeyName my-key-pair
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2KeyPair (DeleteKeyPair)" on Target "my-key-pair".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteKeyPair](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2NetworkAcl

Il seguente esempio di codice mostra come utilizzare. Remove-EC2NetworkAcl

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la lista di controllo degli accessi alla rete (ACL) specificata. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2NetworkAcl -NetworkAclId acl-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkAcl (DeleteNetworkAcl)" on Target
"acl-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteNetworkAcl](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2NetworkAclEntry

Il seguente esempio di codice mostra come utilizzare. Remove-EC2NetworkAclEntry

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la regola specificata dalla lista di controllo degli accessi alla rete (ACL) specificata. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkAclEntry (DeleteNetworkAclEntry)" on Target
"acl-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteNetworkAclEntry](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2NetworkInterface

Il seguente esempio di codice mostra come utilizzare. Remove-EC2NetworkInterface

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'interfaccia di rete specificata. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkInterface (DeleteNetworkInterface)" on Target
"eni-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteNetworkInterface](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2PlacementGroup

Il seguente esempio di codice mostra come utilizzare. Remove-EC2PlacementGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il gruppo di posizionamento specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2PlacementGroup -GroupName my-placement-group
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2PlacementGroup (DeletePlacementGroup)" on Target
"my-placement-group".
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):
```

- Per i dettagli sull'API, vedere [DeletePlacementGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2Route

Il seguente esempio di codice mostra come utilizzare. Remove-EC2Route

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la route specificata dalla tabella di routing specificata. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Route (DeleteRoute)" on Target "rtb-1a2b3c4d".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):
```

- Per i dettagli sull'API, vedere [DeleteRoute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2RouteTable

Il seguente esempio di codice mostra come utilizzare. Remove-EC2RouteTable

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la tabella di routing specificata. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```


Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2RouteTable (DeleteRouteTable)" on Target
"rtb-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteRouteTable](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2SecurityGroup

Il seguente esempio di codice mostra come utilizzare. Remove-EC2SecurityGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il gruppo di sicurezza specificato per EC2 -VPC. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2SecurityGroup -GroupId sg-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2SecurityGroup (DeleteSecurityGroup)" on Target
"sg-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Esempio 2: questo esempio elimina il gruppo di sicurezza specificato per -Classic. EC2

```
Remove-EC2SecurityGroup -GroupName my-security-group -Force
```

- Per i dettagli sull'API, vedere [DeleteSecurityGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2Snapshot

Il seguente esempio di codice mostra come utilizzare. Remove-EC2Snapshot

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina lo snapshot specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2Snapshot -SnapshotId snap-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Snapshot (DeleteSnapshot)" on target
"snap-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteSnapshot](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2SpotDatafeedSubscription

Il seguente esempio di codice mostra come utilizzare. Remove-EC2SpotDatafeedSubscription

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il feed di dati dell'istanza spot. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2SpotDatafeedSubscription
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2SpotDatafeedSubscription
(DeleteSpotDatafeedSubscription)" on Target "".
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):
```

- Per i dettagli sull'API, vedere [DeleteSpotDatafeedSubscription](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2Subnet

Il seguente esempio di codice mostra come utilizzare. Remove-EC2Subnet

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la sottorete specificata. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2Subnet -SubnetId subnet-1a2b3c4d
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Subnet (DeleteSubnet)" on Target "subnet-1a2b3c4d".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):
```

- Per i dettagli sull'API, vedere [DeleteSubnet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2Tag

Il seguente esempio di codice mostra come utilizzare. Remove-EC2Tag

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il tag specificato dalla risorsa specificata, indipendentemente dal valore del tag. La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag" } -Force
```

Esempio 2: questo esempio elimina il tag specificato dalla risorsa specificata, ma solo se il valore del tag corrisponde. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag";Value="myTagValue" } -Force
```

Esempio 3: questo esempio elimina il tag specificato dalla risorsa specificata, indipendentemente dal valore del tag.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

Esempio 4: questo esempio elimina il tag specificato dalla risorsa specificata, ma solo se il valore del tag corrisponde.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"
$tag.Value = "myTagValue"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

- Per i dettagli sull'API, vedere [DeleteTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2Volume

Il seguente esempio di codice mostra come utilizzare. Remove-EC2Volume

Strumenti per PowerShell V4

Esempio 1: questo esempio scollega il volume specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2Volume -VolumeId vol-12345678
```

Output:

```
Confirm
```

```
Are you sure you want to perform this action?  
Performing the operation "Remove-EC2Volume (DeleteVolume)" on target "vol-12345678".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteVolume](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2Vpc

Il seguente esempio di codice mostra come utilizzare. Remove-EC2Vpc

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il VPC specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2Vpc -VpcId vpc-12345678
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing operation "Remove-EC2Vpc (DeleteVpc)" on Target "vpc-12345678".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteVpc](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2VpnConnection

Il seguente esempio di codice mostra come utilizzare. Remove-EC2VpnConnection

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la connessione VPN specificata. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2VpnConnection -VpnConnectionId vpn-12345678
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnConnection (DeleteVpnConnection)" on Target
"vpn-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteVpnConnection](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2VpnConnectionRoute

Il seguente esempio di codice mostra come utilizzare. Remove-EC2VpnConnectionRoute

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la route statica specificata dalla connessione VPN specificata. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
11.12.0.0/16
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnConnectionRoute (DeleteVpnConnectionRoute)" on
Target "vpn-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteVpnConnectionRoute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EC2VpnGateway

Il seguente esempio di codice mostra come utilizzare. Remove-EC2VpnGateway

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il gateway privato virtuale specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnGateway (DeleteVpnGateway)" on Target
"vgw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteVpnGateway](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Request-EC2SpotFleet

Il seguente esempio di codice mostra come utilizzare Request-EC2SpotFleet

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una richiesta del parco istanze spot nella zona di disponibilità con il prezzo più basso per il tipo di istanza specificato. Se il tuo account supporta solo EC2 -VPC, il parco istanze Spot avvia le istanze nella zona di disponibilità più economica con una sottorete predefinita. Se il tuo account supporta EC2 -Classic, la flotta Spot avvia le istanze in -Classic nella zona di disponibilità più economica. EC2 Nota che il prezzo da pagare non supererà il prezzo spot specificato per la richiesta.

```
$sg = New-Object Amazon.EC2.Model.GroupIdentifier
$sg.GroupId = "sg-12345678"
$lc = New-Object Amazon.EC2.Model.SpotFleetLaunchSpecification
$lc.ImageId = "ami-12345678"
$lc.InstanceType = "m3.medium"
$lc.SecurityGroups.Add($sg)
Request-EC2SpotFleet -SpotFleetRequestConfig_SpotPrice 0.04 `
-SpotFleetRequestConfig_TargetCapacity 2 `
```

```
-SpotFleetRequestConfig_IamFleetRole arn:aws:iam::123456789012:role/my-spot-fleet-
role `
-SpotFleetRequestConfig_LaunchSpecification $lc
```

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4). [RequestSpotFleet](#) AWS Strumenti per PowerShell

Request-EC2SpotInstance

Il seguente esempio di codice mostra come utilizzare. Request-EC2SpotInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio richiede un'istanza spot una tantum nella sottorete specificata. Nota che il gruppo di sicurezza deve essere creato per il VPC contenente la sottorete specificata e deve essere specificato tramite ID utilizzando l'interfaccia di rete. Quando si specifica un'interfaccia di rete, bisogna includere l'ID della sottorete utilizzando l'interfaccia di rete.

```
$n = New-Object Amazon.EC2.Model.InstanceNetworkInterfaceSpecification
$n.DeviceIndex = 0
$n.SubnetId = "subnet-12345678"
$n.Groups.Add("sg-12345678")
Request-EC2SpotInstance -InstanceCount 1 -SpotPrice 0.050 -Type one-time `
-IamInstanceProfile_Arn arn:aws:iam::123456789012:instance-profile/my-iam-role `
-LaunchSpecification_ImageId ami-12345678 `
-LaunchSpecification_InstanceType m3.medium `
-LaunchSpecification_NetworkInterface $n
```

Output:

```
ActualBlockHourlyPrice      :
AvailabilityZoneGroup       :
BlockDurationMinutes        : 0
CreateTime                  : 12/26/2015 7:44:10 AM
Fault                       :
InstanceId                  :
LaunchedAvailabilityZone    :
LaunchGroup                 :
LaunchSpecification         : Amazon.EC2.Model.LaunchSpecification
ProductDescription          : Linux/UNIX
SpotInstanceRequestId      : sir-12345678
```



```
SpotPrice      : 0.050000
State          : open
Status         : Amazon.EC2.Model.SpotInstanceStatus
Tags           : {}
Type           : one-time
```

- Per i dettagli sull'API, vedere [RequestSpotInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Reset-EC2ImageAttribute

Il seguente esempio di codice mostra come utilizzare. Reset-EC2ImageAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio reimposta l'attributo "launchPermission" sul relativo valore predefinito. Per impostazione predefinita, AMLs sono privati.

```
Reset-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

- Per i dettagli sull'API, vedere [ResetImageAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Reset-EC2InstanceAttribute

Il seguente esempio di codice mostra come utilizzare. Reset-EC2InstanceAttribute

Strumenti per PowerShell V4

Esempio 1: Questo esempio reimposta l'attributo 'sriovNetSupport' per l'istanza specificata.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

Esempio 2: questo esempio reimposta l'attributo "ebsOptimized" per l'istanza specificata.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

Esempio 3: Questo esempio reimposta l'attributo 'sourceDestCheck' per l'istanza specificata.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sourceDestCheck
```

Esempio 4: Questo esempio reimposta l'attributo 'disableApiTermination' per l'istanza specificata.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute disableApiTermination
```

Esempio 5: questo esempio reimposta l'attributo 'instanceInitiatedShutdownBehavior' per l'istanza specificata.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute  
instanceInitiatedShutdownBehavior
```

- Per i dettagli sull'API, vedere [ResetInstanceAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Reset-EC2NetworkInterfaceAttribute

Il seguente esempio di codice mostra come utilizzare. Reset-EC2NetworkInterfaceAttribute

Strumenti per PowerShell V4

Esempio 1: Questo esempio ripristina il source/destination controllo dell'interfaccia di rete specificata.

```
Reset-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -SourceDestCheck
```

- Per i dettagli sull'API, vedere [ResetNetworkInterfaceAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Reset-EC2SnapshotAttribute

Il seguente esempio di codice mostra come utilizzare. Reset-EC2SnapshotAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio reimposta l'attributo specificato dello snapshot specificato.

```
Reset-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute  
CreateVolumePermission
```

- Per i dettagli sull'API, vedere [ResetSnapshotAttribute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Restart-EC2Instance

Il seguente esempio di codice mostra come utilizzare. Restart-EC2Instance

Strumenti per PowerShell V4

Esempio 1: questo esempio riavvia l'istanza specificata.

```
Restart-EC2Instance -InstanceId i-12345678
```

- Per i dettagli sull'API, vedere [RebootInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Revoke-EC2SecurityGroupEgress

Il seguente esempio di codice mostra come utilizzare. Revoke-EC2SecurityGroupEgress

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la regola per il gruppo di sicurezza specificato per EC2 - VPC. Questa operazione revoca l'accesso agli intervalli di indirizzi IP specificati sulla porta TCP 80. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
$ip = @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; IpRanges="203.0.113.0/24" }  
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

Esempio 2: con PowerShell la versione 2, è necessario utilizzare New-Object per creare l'oggetto. IpPermission

```
$ip = New-Object Amazon.EC2.Model.IpPermission  
$ip.IpProtocol = "tcp"  
$ip.FromPort = 80  
$ip.ToPort = 80  
$ip.IpRanges.Add("203.0.113.0/24")  
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

Esempio 3: questo esempio revoca l'accesso al gruppo di sicurezza di origine specificato sulla porta TCP 80.

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
```

```
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission
@ ( @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; UserIdGroupPairs=$ug } )
```

- Per i dettagli sull'API, vedere [RevokeSecurityGroupEgress](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Revoke-EC2SecurityGroupIngress

Il seguente esempio di codice mostra come utilizzare `Revoke-EC2SecurityGroupIngress`

Strumenti per PowerShell V4

Esempio 1: Questo esempio revoca l'accesso alla porta TCP 22 dall'intervallo di indirizzi specificato per il gruppo di sicurezza specificato per `-VPC`. EC2 Si noti che è necessario identificare i gruppi di sicurezza per EC2 `-VPC` utilizzando l'ID del gruppo di sicurezza e non il nome del gruppo di sicurezza. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22"; IpRanges="203.0.113.0/24" }
Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

Esempio 2: con PowerShell la versione 2, è necessario utilizzare `New-Object` per creare l'oggetto `IpPermission`

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 22
$ip.ToPort = 22
$ip.IpRanges.Add("203.0.113.0/24")

Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

Esempio 3: Questo esempio revoca l'accesso alla porta TCP 22 dall'intervallo di indirizzi specificato per il gruppo di sicurezza specificato per `-Classic`. EC2 La sintassi utilizzata da questo esempio richiede PowerShell la versione 3 o successiva.

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22"; IpRanges="203.0.113.0/24" }
```

```
Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

Esempio 4: con PowerShell la versione 2, è necessario utilizzare New-Object per creare l'oggetto. IpPermission

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 22
$ip.ToPort = 22
$ip.IpRanges.Add("203.0.113.0/24")

Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

- Per i dettagli sull'API, vedere [RevokeSecurityGroupIngress](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Send-EC2InstanceStatus

Il seguente esempio di codice mostra come utilizzare. Send-EC2InstanceStatus

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il feedback sullo stato per l'istanza specificata.

```
Send-EC2InstanceStatus -Instance i-12345678 -Status impaired -ReasonCode
unresponsive
```

- Per i dettagli sull'API, vedere [ReportInstanceStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-EC2NetworkAclAssociation

Il seguente esempio di codice mostra come utilizzare. Set-EC2NetworkAclAssociation

Strumenti per PowerShell V4

Esempio 1: questo esempio associa la lista di controllo degli accessi alla rete (ACL) specificata alla sottorete dell'associazione lista di controllo degli accessi alla rete (ACL) specificata.

```
Set-EC2NetworkAclAssociation -NetworkAclId acl-12345678 -AssociationId  
aclassoc-1a2b3c4d
```

Output:

```
aclassoc-87654321
```

- Per i dettagli sull'API, vedere [ReplaceNetworkAclAssociation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-EC2NetworkAclEntry

Il seguente esempio di codice mostra come utilizzare. Set-EC2NetworkAclEntry

Strumenti per PowerShell V4

Esempio 1: questo esempio sostituisce la voce specificata per lista di controllo degli accessi alla rete (ACL) specificata. La nuova regola consente il traffico in entrata dall'indirizzo specificato verso qualsiasi sottorete associata.

```
Set-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100  
-Protocol 17 -PortRange_From 53 -PortRange_To 53 -CidrBlock 203.0.113.12/24 -  
RuleAction allow
```

- Per i dettagli sull'API, vedere [ReplaceNetworkAclEntry](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-EC2Route

Il seguente esempio di codice mostra come utilizzare. Set-EC2Route

Strumenti per PowerShell V4

Esempio 1: questo esempio sostituisce la route specificata per la tabella di routing specificata. La nuova route invia il traffico specificato al gateway privato virtuale specificato.

```
Set-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 10.0.0.0/24 -GatewayId  
vgw-1a2b3c4d
```

- Per i dettagli sull'API, vedere [ReplaceRoute](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-EC2RouteTableAssociation

Il seguente esempio di codice mostra come utilizzare. Set-EC2RouteTableAssociation

Strumenti per PowerShell V4

Esempio 1: questo esempio associa la tabella di routing specificata alla sottorete per l'associazione di tabelle di routing specificata.

```
Set-EC2RouteTableAssociation -RouteTableId rtb-1a2b3c4d -AssociationId
rtbassoc-12345678
```

Output:

```
rtbassoc-87654321
```

- Per i dettagli sull'API, vedere [ReplaceRouteTableAssociation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-EC2Instance

Il seguente esempio di codice mostra come utilizzare. Start-EC2Instance

Strumenti per PowerShell V4

Esempio 1: questo esempio avvia l'istanza specificata.

```
Start-EC2Instance -InstanceId i-12345678
```

Output:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

Esempio 2: questo esempio avvia le istanze specificate.

```
@("i-12345678", "i-76543210") | Start-EC2Instance
```

Esempio 3: questo esempio avvia il set di istanze attualmente arrestate. Gli oggetti Instance restituiti da Get-EC2Instance vengono reindirizzati a Start-EC2Instance. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
(Get-EC2Instance -Filter @{ Name="instance-state-name"; Values="stopped"}).Instances  
| Start-EC2Instance
```

Esempio 4: con la PowerShell versione 2, è necessario utilizzare New-Object per creare il filtro per il parametro Filter.

```
$filter = New-Object Amazon.EC2.Model.Filter  
$filter.Name = "instance-state-name"  
$filter.Values = "stopped"  
  
(Get-EC2Instance -Filter $filter).Instances | Start-EC2Instance
```

- Per i dettagli sull'API, vedere [StartInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-EC2InstanceMonitoring

Il seguente esempio di codice mostra come utilizzare Start-EC2InstanceMonitoring Strumenti per PowerShell V4

Esempio 1: questo esempio abilita il monitoraggio dettagliato per l'istanza specificata.

```
Start-EC2InstanceMonitoring -InstanceId i-12345678
```

Output:

InstanceId	Monitoring
-----	-----
i-12345678	Amazon.EC2.Model.Monitoring

- Per i dettagli sull'API, vedere [MonitorInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-EC2ImportTask

Il seguente esempio di codice mostra come utilizzare. Stop-EC2ImportTask

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla l'attività di importazione specificata (importazione di snapshot o immagini). Se necessario, è possibile specificare un motivo utilizzando il parametro - **CancelReason**.

```
Stop-EC2ImportTask -ImportTaskId import-ami-abcdefgh
```

- Per i dettagli sull'API, vedere [CancelImportTask](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-EC2Instance

Il seguente esempio di codice mostra come utilizzare. Stop-EC2Instance

Strumenti per PowerShell V4

Esempio 1: questo esempio arresta l'istanza specificata.

```
Stop-EC2Instance -InstanceId i-12345678
```

Output:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

- Per i dettagli sull'API, vedere [StopInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-EC2InstanceMonitoring

Il seguente esempio di codice mostra come utilizzare. Stop-EC2InstanceMonitoring

Strumenti per PowerShell V4

Esempio 1: questo esempio disabilita il monitoraggio dettagliato per l'istanza specificata.

```
Stop-EC2InstanceMonitoring -InstanceId i-12345678
```

Output:

InstanceId	Monitoring
i-12345678	Amazon.EC2.Model.Monitoring

- Per i dettagli sull'API, vedere [UnmonitorInstances](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-EC2SpotFleetRequest

Il seguente esempio di codice mostra come utilizzare. Stop-EC2SpotFleetRequest

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la richiesta del parco istanze spot specificata e termina le istanze associate.

```
Stop-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-  
aa30-494c-8788-1cee4EXAMPLE -TerminateInstance $true
```

Esempio 2: questo esempio annulla la richiesta del parco istanze spot specificata senza terminare le istanze associate.

```
Stop-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-  
aa30-494c-8788-1cee4EXAMPLE -TerminateInstance $false
```

- Per i dettagli sull'API, vedere [CancelSpotFleetRequests](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-EC2SpotInstanceRequest

Il seguente esempio di codice mostra come utilizzare. Stop-EC2SpotInstanceRequest

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la richiesta di istanze Spot specificata.

```
Stop-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

Output:

SpotInstanceRequestId	State
-----	-----
sir-12345678	cancelled

- Per i dettagli sull'API, vedere [CancelSpotInstanceRequests](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-EC2Address

Il seguente esempio di codice mostra come utilizzare. Unregister-EC2Address

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove l'associazione dell'indirizzo IP elastico specificato all'istanza specificata in un VPC.

```
Unregister-EC2Address -AssociationId eipassoc-12345678
```

Esempio 2: Questo esempio dissocia l'indirizzo IP elastico specificato dall'istanza specificata in - Classic. EC2

```
Unregister-EC2Address -PublicIp 203.0.113.17
```

- Per i dettagli sull'API, vedere [DisassociateAddress](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-EC2Image

Il seguente esempio di codice mostra come utilizzare. Unregister-EC2Image

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la registrazione dell'AMI specificata.

```
Unregister-EC2Image -ImageId ami-12345678
```

- Per i dettagli sull'API, vedere [DeregisterImage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-EC2PrivateIpAddress

Il seguente esempio di codice mostra come utilizzare. Unregister-EC2PrivateIpAddress

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla l'assegnazione dell'indirizzo IP privato specificato all'interfaccia di rete specificata.

```
Unregister-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -PrivateIpAddress 10.0.0.82
```

- Per i dettagli sull'API, vedere [UnassignPrivateIpAddresses](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-EC2RouteTable

Il seguente esempio di codice mostra come utilizzare. Unregister-EC2RouteTable

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove l'associazione specificata tra una tabella di routing e una sottorete.

```
Unregister-EC2RouteTable -AssociationId rtbassoc-1a2b3c4d
```

- Per i dettagli sull'API, vedere [DisassociateRouteTable](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-EC2SecurityGroupRuleIngressDescription

Il seguente esempio di codice mostra come utilizzare. Update-EC2SecurityGroupRuleIngressDescription

Strumenti per PowerShell V4

Esempio 1: aggiorna la descrizione di una regola del gruppo di sicurezza in ingresso.

```
$existingInboundRule = Get-EC2SecurityGroupRule -SecurityGroupRuleId
"sgr-1234567890"
$ruleWithUpdatedDescription = [Amazon.EC2.Model.SecurityGroupRuleDescription]@{
    "SecurityGroupRuleId" = $existingInboundRule.SecurityGroupRuleId
    "Description" = "Updated rule description"
}

Update-EC2SecurityGroupRuleIngressDescription -GroupId $existingInboundRule.GroupId
-SecurityGroupRuleDescription $ruleWithUpdatedDescription
```

Esempio 2: rimuove la descrizione di una regola del gruppo di sicurezza in ingresso esistente (omettendo il parametro nella richiesta).

```
$existingInboundRule = Get-EC2SecurityGroupRule -SecurityGroupRuleId
"sgr-1234567890"
$ruleWithoutDescription = [Amazon.EC2.Model.SecurityGroupRuleDescription]@{
    "SecurityGroupRuleId" = $existingInboundRule.SecurityGroupRuleId
}

Update-EC2SecurityGroupRuleIngressDescription -GroupId $existingInboundRule.GroupId
-SecurityGroupRuleDescription $ruleWithoutDescription
```

- Per i dettagli sull'API, vedere [UpdateSecurityGroupRuleDescriptionsIngress](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon ECR con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Amazon ECR.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-ECRLoginCommand

Il seguente esempio di codice mostra come usare `Get-ECRLoginCommand`

Strumenti per PowerShell V4

Esempio 1: restituisce informazioni di accesso `PSObject` contenenti che possono essere utilizzate per l'autenticazione in qualsiasi registro Amazon ECR a cui il principale IAM ha accesso. Le credenziali e l'endpoint della Regione necessari per la chiamata per ottenere il token di autorizzazione vengono ottenuti dai valori predefiniti della shell (impostati dai cmdlet **`Set-AWSCredential/Set-DefaultAWSRegion`** o **`Initialize-AWSDefaultConfiguration`**). È possibile utilizzare la proprietà `Command` con `Invoke-Expression` per accedere al registro specificato o utilizzare le credenziali restituite in altri strumenti che richiedono l'accesso.

```
Get-ECRLoginCommand
```

Output:

```
Username       : AWS
Password       : eyJwYXlsb2Fk...kRBVEFFS0VZIn0=
ProxyEndpoint  : https://123456789012.dkr.ecr.us-west-2.amazonaws.com
Endpoint       : https://123456789012.dkr.ecr.us-west-2.amazonaws.com
ExpiresAt      : 9/26/2017 6:08:23 AM
Command        : docker login --username AWS --password
eyJwYXlsb2Fk...kRBVEFFS0VZIn0= https://123456789012.dkr.ecr.us-west-2.amazonaws.com
```

Esempio 2: recupera le informazioni di accesso `PSObject` contenenti che utilizzi come input per un comando `docker login`. È possibile specificare qualsiasi URI di registro Amazon ECR a cui eseguire l'autenticazione, a condizione che l'entità principale IAM abbia accesso a tale registro.

```
(Get-ECRLoginCommand).Password | docker login --username AWS --password-stdin
012345678910.dkr.ecr.us-east-1.amazonaws.com
```

- Per i dettagli sull'API, vedere [Get- ECRLogin Command](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon ECS con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon ECS.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-ECSClusterDetail

Il seguente esempio di codice mostra come usare. Get-ECSClusterDetail

Strumenti per PowerShell V4

Esempio 1: questo cmdlet descrive uno o più cluster ECS.

```
Get-ECSClusterDetail -Cluster "LAB-ECS-CL" -Include SETTINGS | Select-Object *
```

Output:

```
LoggedAt       : 12/27/2019 9:27:41 PM
Clusters       : {LAB-ECS-CL}
Failures       : {}
ResponseMetadata : Amazon.Runtime.ResponseMetadata
ContentLength   : 396
HttpStatusCode  : OK
```

- Per i dettagli sull'API, vedere [DescribeClusters](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ECSClusterList

Il seguente esempio di codice mostra come utilizzare. Get-ECSClusterList

Strumenti per PowerShell V4

Esempio 1: questo cmdlet restituisce un elenco di cluster ECS esistenti.

```
Get-ECSClusterList
```

Output:

```
arn:aws:ecs:us-west-2:012345678912:cluster/LAB-ECS-CL  
arn:aws:ecs:us-west-2:012345678912:cluster/LAB-ECS
```

- Per i dettagli sull'API, vedere [ListClusters](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ECSClusterService

Il seguente esempio di codice mostra come utilizzare. Get-ECSClusterService

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutti i servizi in esecuzione nel cluster predefinito.

```
Get-ECSClusterService
```

Esempio 2: questo esempio elenca tutti i servizi in esecuzione nel cluster specificato.

```
Get-ECSClusterService -Cluster myCluster
```

- Per i dettagli sull'API, vedere [ListServices](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ECSService

Il seguente esempio di codice mostra come utilizzare. Get-ECSService

Strumenti per PowerShell V4

Esempio 1: questo esempio seguente mostra come recuperare i dettagli di un servizio specifico dal cluster predefinito.

```
Get-ECSService -Service my-http-service
```

Esempio 2: questo esempio mostra come recuperare i dettagli di un servizio specifico in esecuzione nel cluster denominato.

```
Get-ECSService -Cluster myCluster -Service my-http-service
```

- Per i dettagli sull'API, vedere [DescribeServices](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ECSCluster

Il seguente esempio di codice mostra come utilizzare. New-ECSCluster

Strumenti per PowerShell V4

Esempio 1: questo cmdlet crea un nuovo cluster Amazon ECS.

```
New-ECSCluster -ClusterName "LAB-ECS-CL" -Setting @{Name="containerInsights";  
Value="enabled"}
```

Output:

```
ActiveServicesCount      : 0  
Attachments              : {}  
AttachmentsStatus       :  
CapacityProviders       : {}  
ClusterArn               : arn:aws:ecs:us-west-2:012345678912:cluster/LAB-  
ECS-CL  
ClusterName              : LAB-ECS-CL  
DefaultCapacityProviderStrategy : {}  
PendingTasksCount       : 0  
RegisteredContainerInstancesCount : 0  
RunningTasksCount       : 0
```

```
Settings           : {containerInsights}
Statistics         : {}
Status            : ACTIVE
Tags              : {}
```

- Per i dettagli sull'API, vedere [CreateCluster](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ECSService

Il seguente esempio di codice mostra come utilizzare. New-ECSService

Strumenti per PowerShell V4

Esempio 1: Questo comando di esempio crea un servizio nel cluster predefinito chiamato `ecs-simple-service`. Il servizio utilizza la definizione di attività "ecs-demo" e conserva 10 istanze di tale attività.

```
New-ECSService -ServiceName ecs-simple-service -TaskDefinition ecs-demo -
DesiredCount 10
```

Esempio 2: Questo comando di esempio crea un servizio basato su un sistema di bilanciamento del carico nel cluster predefinito denominato `ecs-simple-service`. Il servizio utilizza la definizione di attività "ecs-demo" e conserva 10 istanze di tale attività.

```
$lb = @{
    LoadBalancerName = "EC2Contai-EcsElast-S06278JGSJCM"
    ContainerName = "simple-demo"
    ContainerPort = 80
}
New-ECSService -ServiceName ecs-simple-service -TaskDefinition ecs-demo -
DesiredCount 10 -LoadBalancer $lb
```

- Per i dettagli sull'API, vedere [CreateService](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ECSCluster

Il seguente esempio di codice mostra come utilizzare. Remove-ECSCluster

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elimina il cluster ECS specificato. È necessario annullare la registrazione di tutte le istanze di container da questo cluster prima di poterlo eliminare.

```
Remove-ECSCluster -Cluster "LAB-ECS"
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ECSCluster (DeleteCluster)" on target "LAB-ECS".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Per i dettagli sull'API, vedere [DeleteCluster](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ECSService

Il seguente esempio di codice mostra come utilizzare. Remove-ECSService

Strumenti per PowerShell V4

Esempio 1: elimina il servizio denominato 'my-http-service' nel cluster predefinito. Il servizio deve includere un valore numerico desiderato e un numero corrente pari a 0 prima di poterlo eliminare. Viene richiesta una conferma prima di procedere con l'esecuzione del comando. Per disabilitare la richiesta di conferma, aggiungi il parametro -Force.

```
Remove-ECSService -Service my-http-service
```

Esempio 2: elimina il servizio denominato 'my-http-service' nel cluster denominato.

```
Remove-ECSService -Cluster myCluster -Service my-http-service
```

- Per i dettagli sull'API, vedere [DeleteService](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-ECSClusterSetting

Il seguente esempio di codice mostra come utilizzare. Update-ECSClusterSetting

Strumenti per PowerShell V4

Esempio 1: questo cmdlet modifica le impostazioni da utilizzare per un cluster ECS.

```
Update-ECSClusterSetting -Cluster "LAB-ECS-CL" -Setting @{Name="containerInsights";
Value="disabled"}
```

Output:

```
ActiveServicesCount      : 0
Attachments              : {}
AttachmentsStatus       :
CapacityProviders       : {}
ClusterArn               : arn:aws:ecs:us-west-2:012345678912:cluster/LAB-
ECS-CL
ClusterName              : LAB-ECS-CL
DefaultCapacityProviderStrategy : {}
PendingTasksCount       : 0
RegisteredContainerInstancesCount : 0
RunningTasksCount       : 0
Settings                 : {containerInsights}
Statistics               : {}
Status                   : ACTIVE
Tags                     : {}
```

- Per i dettagli sull'API, vedere [UpdateClusterSettings](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-ECSService

Il seguente esempio di codice mostra come utilizzare. Update-ECSService

Strumenti per PowerShell V4

Esempio 1: Questo comando di esempio aggiorna il servizio my-http-service `` per utilizzare la definizione del task amazon-ecs-sample ``.

```
Update-ECSService -Service my-http-service -TaskDefinition amazon-ecs-sample
```

Esempio 2: questo comando di esempio aggiorna il conteggio desiderato del servizio my-http-service `` a 10.

```
Update-ECSService -Service my-http-service -DesiredCount 10
```

- Per i dettagli sull'API, vedere [UpdateService](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon EFS con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la versione AWS Strumenti per PowerShell V4 con Amazon EFS.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Edit-EFSMountTargetSecurityGroup

Il seguente esempio di codice mostra come usare Edit-EFSMountTargetSecurityGroup.

Strumenti per PowerShell V4

Esempio 1: aggiorna i gruppi di sicurezza in vigore per la destinazione di montaggio specificata. È possibile specificarne fino a 5, nel formato "sg-xxxxxxx".

```
Edit-EFSMountTargetSecurityGroup -MountTargetId fsmt-1a2b3c4d -SecurityGroup sg-group1,sg-group3
```

- Per i dettagli sull'API, vedere [ModifyMountTargetSecurityGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EFSFileSystem

Il seguente esempio di codice mostra come utilizzare `Get-EFSFileSystem`

Strumenti per PowerShell V4

Esempio 1: restituisce la raccolta di tutti i file system appartenenti all'account del chiamante nell'area.

```
Get-EFSFileSystem
```

Output:

```
CreationTime      : 5/26/2015 4:02:38 PM
CreationToken     : 1a2bff54-85e0-4747-bd95-7bc172c4f555
FileSystemId      : fs-1a2b3c4d
LifeCycleState    : available
Name              :
NumberOfMountTargets : 0
OwnerId           : 123456789012
SizeInBytes       : Amazon.ElasticFileSystem.Model.FileSystemSize

CreationTime      : 5/26/2015 4:06:23 PM
CreationToken     : 2b4daa14-85e0-4747-bd95-7bc172c4f555
FileSystemId      : fs-4d3c2b1a
...
```

Esempio 2: restituisce i dettagli del file system specificato.

```
Get-EFSFileSystem -FileSystemId fs-1a2b3c4d
```

Esempio 3: restituisce i dettagli di un file system utilizzando il token di creazione basata sull'idempotenza specificato durante la creazione del file system.

```
Get-EFSFileSystem -CreationToken 1a2bff54-85e0-4747-bd95-7bc172c4f555
```

- Per i dettagli sull'API, vedere [DescribeFileSystems](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EFSMountTarget

Il seguente esempio di codice mostra come utilizzare. Get-EFSMountTarget

Strumenti per PowerShell V4

Esempio 1: restituisce la raccolta delle destinazioni di montaggio associate al file system specificato.

```
Get-EFSMountTarget -FileSystemId fs-1a2b3c4d
```

Output:

```
FileSystemId      : fs-1a2b3c4d
IpAddress         : 10.0.0.131
LifeCycleState    : available
MountTargetId     : fsmt-1a2b3c4d
NetworkInterfaceId : eni-1a2b3c4d
OwnerId           : 123456789012
SubnetId          : subnet-1a2b3c4d
```

- Per i dettagli sull'API, vedere [DescribeMountTargets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EFSMountTargetSecurityGroup

Il seguente esempio di codice mostra come utilizzare. Get-EFSMountTargetSecurityGroup

Strumenti per PowerShell V4

Esempio 1: restituisce gli ID dei gruppi di sicurezza attualmente assegnati all'interfaccia di rete associata alla destinazione di montaggio.

```
Get-EFSMountTargetSecurityGroup -MountTargetId fsmt-1a2b3c4d
```

Output:

```
sg-1a2b3c4d
```

- Per i dettagli sull'API, vedere [DescribeMountTargetSecurityGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EFSTag

Il seguente esempio di codice mostra come utilizzare. Get-EFSTag

Strumenti per PowerShell V4

Esempio 1: restituisce la raccolta di tag attualmente associati al file system specificato.

```
Get-EFSTag -FileSystemId fs-1a2b3c4d
```

Output:

Key	Value
---	----
Name	My File System
tagkey1	tagvalue1
tagkey2	tagvalue2

- Per i dettagli sull'API, vedere [DescribeTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EFSFileSystem

Il seguente esempio di codice mostra come utilizzare. New-EFSFileSystem

Strumenti per PowerShell V4

Esempio 1: crea un nuovo file system vuoto. Il token utilizzato per garantire la creazione dell'idempotenza verrà generato automaticamente e sarà accessibile dal membro **CreationToken** dell'oggetto restituito.

```
New-EFSFileSystem
```

Output:

CreationTime	: 5/26/2015 4:02:38 PM
CreationToken	: 1a2bff54-85e0-4747-bd95-7bc172c4f555
FileSystemId	: fs-1a2b3c4d
LifeCycleState	: creating
Name	:


```
NumberOfMountTargets : 0
OwnerId               : 123456789012
SizeInBytes           : Amazon.ElasticFileSystem.Model.FileSystemSize
```

Esempio 2: crea un nuovo file system vuoto utilizzando un token personalizzato per garantire una creazione basata sull'idempotenza.

```
New-EFSFileSystem -CreationToken "MyUniqueToken"
```

- Per i dettagli sull'API, vedere [CreateFileSystem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EFSMountTarget

Il seguente esempio di codice mostra come utilizzare. New-EFSMountTarget

Strumenti per PowerShell V4

Esempio 1: crea una destinazione di montaggio per un file system. La sottorete specificata verrà utilizzata per determinare il cloud privato virtuale (VPC) in cui verrà creata la destinazione di montaggio e l'indirizzo IP che verrà assegnato automaticamente (dall'intervallo di indirizzi della sottorete). L'indirizzo IP assegnato può essere utilizzato per poi montare questo file system su un' EC2 istanza Amazon. Poiché non è stato specificato un gruppo di sicurezza, l'interfaccia di rete creata per la destinazione è associata al gruppo di sicurezza predefinito per il VPC della sottorete.

```
New-EFSMountTarget -FileSystemId fs-1a2b3c4d -SubnetId subnet-1a2b3c4d
```

Output:

```
FileSystemId      : fs-1a2b3c4d
IpAddress         : 10.0.0.131
LifecycleState    : creating
MountTargetId     : fsmt-1a2b3c4d
NetworkInterfaceId : eni-1a2b3c4d
OwnerId           : 123456789012
SubnetId          : subnet-1a2b3c4d
```

Esempio 2: crea una nuova destinazione di montaggio per il file system specificato con un indirizzo IP assegnato automaticamente. L'interfaccia di rete creata per la destinazione di

montaggio è associata ai gruppi di sicurezza specificati (un massimo di 5, nel formato “sg-xxxxxxx”).

```
New-EFSMountTarget -FileSystemId fs-1a2b3c4d -SubnetId subnet-1a2b3c4d -
SecurityGroup sg-group1,sg-group2,sg-group3
```

Esempio 3: crea una nuova destinazione di montaggio per il file system specificato con l'indirizzo IP specificato.

```
New-EFSMountTarget -FileSystemId fs-1a2b3c4d -SubnetId subnet-1a2b3c4d -IpAddress
10.0.0.131
```

- Per i dettagli sull'API, vedere [CreateMountTarget](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EFSTag

Il seguente esempio di codice mostra come utilizzare. New-EFSTag

Strumenti per PowerShell V4

Esempio 1: applica la raccolta di tag al file system specificato. Se un tag con la chiave specificata esiste già nel file system, il valore del tag viene aggiornato.

```
New-EFSTag -FileSystemId fs-1a2b3c4d -Tag
@{Key="tagkey1";Value="tagvalue1"},@{Key="tagkey2";Value="tagvalue2"}
```

Esempio 2: imposta il name tag per il file system specificato. Questo valore viene restituito insieme ad altri dettagli del file system quando viene utilizzato il Get-EFSFileSystem cmdlet.

```
New-EFSTag -FileSystemId fs-1a2b3c4d -Tag @{Key="Name";Value="My File System"}
```

- Per i dettagli sull'API, vedere [CreateTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EFSFileSystem

Il seguente esempio di codice mostra come utilizzare. Remove-EFSFileSystem

Strumenti per PowerShell V4

Esempio 1: elimina il file system non più in uso specificato (se il file system ha destinazioni di montaggio, tali destinazioni devono prima essere rimosse). Prima che il cmdlet proceda con l'esecuzione dell'operazione, viene richiesta una conferma. Per disabilitare la conferma, utilizza il parametro **-Force**.

```
Remove-EFSFileSystem -FileSystemId fs-1a2b3c4d
```

- Per i dettagli sull'API, vedere [DeleteFileSystem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EFSMountTarget

Il seguente esempio di codice mostra come utilizzare. Remove-EFSMountTarget

Strumenti per PowerShell V4

Esempio 1: elimina la destinazione di montaggio specificata. Prima di procedere con l'operazione, viene richiesta una conferma. Per disabilitare la richiesta, utilizza il parametro **-Force**. Nota che questa operazione forza l'interruzione di qualsiasi montaggio del file system tramite la destinazione. Se possibile, puoi valutare la possibilità di smontare il file system prima di eseguire questo comando.

```
Remove-EFSMountTarget -MountTargetId fsmt-1a2b3c4d
```

- Per i dettagli sull'API, vedere [DeleteMountTarget](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EFSTag

Il seguente esempio di codice mostra come utilizzare. Remove-EFSTag

Strumenti per PowerShell V4

Esempio 1: elimina la raccolta di uno o più tag da un file system. Prima che il cmdlet proceda con l'esecuzione dell'operazione, viene richiesta una conferma. Per disabilitare la conferma, utilizza il parametro **-Force**.

```
Remove-EFSTag -FileSystemId fs-1a2b3c4d -TagKey "tagkey1","tagkey2"
```

- Per i dettagli sull'API, vedere [DeleteTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon EKS con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon EKS.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-EKSResourceTag

Il seguente esempio di codice mostra come usare Add-EKSResourceTag.

Strumenti per PowerShell V4

Esempio 1: questo cmdlet associa i tag specificati a una risorsa con l'ARN di risorsa specificato.

```
Add-EKSResourceTag -ResourceArn "arn:aws:eks:us-west-2:012345678912:cluster/PROD" -  
Tag @{Name = "EKSPRODCLUSTER"}
```

- Per i dettagli sull'API, vedere [TagResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EKSCluster

Il seguente esempio di codice mostra come utilizzare. Get-EKSCluster

Strumenti per PowerShell V4

Esempio 1: questo cmdlet restituisce informazioni descrittive su un cluster Amazon EKS.

```
Get-EKSCluster -Name "PROD"
```

Output:

```
Arn                : arn:aws:eks:us-west-2:012345678912:cluster/PROD
CertificateAuthority : Amazon.EKS.Model.Certificate
ClientRequestToken  :
CreatedAt           : 12/25/2019 6:46:17 AM
Endpoint            : https://669608765450FBBE54D1D78A3D71B72C.gr8.us-
west-2.eks.amazonaws.com
Identity            : Amazon.EKS.Model.Identity
Logging             : Amazon.EKS.Model.Logging
Name                : PROD
PlatformVersion     : eks.7
ResourcesVpcConfig  : Amazon.EKS.Model.VpcConfigResponse
RoleArn             : arn:aws:iam::012345678912:role/eks-iam-role
Status              : ACTIVE
Tags                : {}
Version             : 1.14
```

- Per i dettagli sull'API, vedere [DescribeCluster](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EKSClusterList

Il seguente esempio di codice mostra come utilizzare. Get-EKSClusterList

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elenca i cluster Amazon EKS presenti Account AWS nella regione specificata.

```
Get-EKSClusterList
```

Output:

```
PROD
```

- Per i dettagli sull'API, vedere [ListClusters](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EKSFargateProfile

Il seguente esempio di codice mostra come utilizzare. Get-EKSFargateProfile

Strumenti per PowerShell V4

Esempio 1: questo cmdlet restituisce informazioni descrittive su un profilo AWS Fargate.

```
Get-EKSFargateProfile -FargateProfileName "EKSFargate" -ClusterName "TEST"
```

Output:

```
ClusterName      : TEST
CreatedAt        : 12/26/2019 12:34:47 PM
FargateProfileArn : arn:aws:eks:us-east-2:012345678912:fargateprofile/TEST/
                  EKSFargate/42b7a119-e16b-a279-ce97-bdf303adec92
FargateProfileName : EKSFargate
PodExecutionRoleArn : arn:aws:iam::012345678912:role/
                  AmazonEKSFargatePodExecutionRole
Selectors        : {Amazon.EKS.Model.FargateProfileSelector}
Status           : ACTIVE
Subnets         : {subnet-0cd976f08d5fbfaae, subnet-02f6ff500ff2067a0}
Tags             : {}
```

- Per i dettagli sull'API, vedere [DescribeFargateProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EKSFargateProfileList

Il seguente esempio di codice mostra come utilizzare. Get-EKSFargateProfileList

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elenca i profili AWS Fargate associati al cluster specificato Account AWS nella regione specificata.

```
Get-EKSFargateProfileList -ClusterName "TEST"
```

Output:

```
EKSFargate
EKSFargateProfile
```

- Per i dettagli sull'API, vedere [ListFargateProfiles](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EKSNodegroup

Il seguente esempio di codice mostra come utilizzare. Get-EKSNodegroup

Strumenti per PowerShell V4

Esempio 1: questo cmdlet restituisce informazioni descrittive su un gruppo di nodi Amazon EKS.

```
Get-EKSNodegroup -NodegroupName "ProdEKSNodeGroup" -ClusterName "PROD"
```

Output:

```
AmiType       : AL2_x86_64
ClusterName   : PROD
CreatedAt     : 12/25/2019 10:16:45 AM
DiskSize      : 40
Health        : Amazon.EKS.Model.NodegroupHealth
InstanceTypes : {t3.large}
Labels        : {}
ModifiedAt    : 12/25/2019 10:16:45 AM
NodegroupArn  : arn:aws:eks:us-west-2:012345678912:nodegroup/PROD/
ProdEKSNodeGroup/7eb79e47-82b6-04d9-e984-95110db6fa85
NodegroupName : ProdEKSNodeGroup
NodeRole       : arn:aws:iam::012345678912:role/NodeInstanceRole
ReleaseVersion : 1.14.7-20190927
RemoteAccess   :
Resources      :
ScalingConfig  : Amazon.EKS.Model.NodegroupScalingConfig
Status        : CREATING
Subnets       : {subnet-0d1a9fff35efa7691, subnet-0a3f4928edbc224d4}
Tags           : {}
Version        : 1.14
```

- Per i dettagli sull'API, vedere [DescribeNodegroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EKSNodegroupList

Il seguente esempio di codice mostra come utilizzare. Get-EKSNodegroupList

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elenca i gruppi di nodi Amazon EKS associati al cluster specificato Account AWS nella regione specificata.

```
Get-EKSNodegroupList -ClusterName PROD
```

Output:

```
ProdEKSNodeGroup
```

- Per i dettagli sull'API, vedere [ListNodegroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EKSResourceTag

Il seguente esempio di codice mostra come utilizzare. Get-EKSResourceTag

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elenca i tag di una risorsa Amazon EKS.

```
Get-EKSResourceTag -ResourceArn "arn:aws:eks:us-west-2:012345678912:cluster/PROD"
```

Output:

```
Key  Value
---  -
Name EKSPRODCLUSTER
```

- Per i dettagli sull'API, vedere [ListTagsForResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EKSUpdate

Il seguente esempio di codice mostra come utilizzare. Get-EKSUpdate

Strumenti per PowerShell V4

Esempio 1: questo cmdlet restituisce informazioni descrittive su un aggiornamento del cluster Amazon EKS o del gruppo di nodi gestiti associato.

```
Get-EKSUpdate -Name "PROD" -UpdateId "ee708232-7d2e-4ed7-9270-d0b5176f0726"
```

Output:

```
CreatedAt : 12/25/2019 5:03:07 PM
Errors    : {}
Id        : ee708232-7d2e-4ed7-9270-d0b5176f0726
Params    : {Amazon.EKS.Model.UpdateParam}
Status    : Successful
Type      : LoggingUpdate
```

- Per i dettagli sull'API, vedere [DescribeUpdate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-EKSUpdateList

Il seguente esempio di codice mostra come utilizzare. Get-EKSUpdateList

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elenca gli aggiornamenti associati a un cluster Amazon EKS o a un gruppo di nodi gestiti nella tua regione Account AWS, nella regione specificata.

```
Get-EKSUpdateList -Name "PROD"
```

Output:

```
ee708232-7d2e-4ed7-9270-d0b5176f0726
```

- Per i dettagli sull'API, vedere [ListUpdates](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EKSCluster

Il seguente esempio di codice mostra come utilizzare. New-EKSCluster

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo cluster denominato "prod".

```
New-EKSCluster -Name prod -ResourcesVpcConfig
@{SubnetIds=@("subnet-0a1b2c3d","subnet-3a2b1c0d");SecurityGroupIds="sg-6979fe18"}
-RoleArn "arn:aws:iam::012345678901:role/eks-service-role"
```

Output:

```
Arn                : arn:aws:eks:us-west-2:012345678901:cluster/prod
CertificateAuthority : Amazon.EKS.Model.Certificate
ClientRequestToken  :
CreatedAt           : 12/10/2018 9:25:31 PM
Endpoint            :
Name                : prod
PlatformVersion     : eks.3
ResourcesVpcConfig  : Amazon.EKS.Model.VpcConfigResponse
RoleArn             : arn:aws:iam::012345678901:role/eks-service-role
Status              : CREATING
Version             : 1.10
```

- Per i dettagli sull'API, vedere [CreateCluster](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EKSFargateProfile

Il seguente esempio di codice mostra come utilizzare. New-EKSFargateProfile

Strumenti per PowerShell V4

Esempio 1: questo cmdlet crea un profilo AWS Fargate per il tuo cluster Amazon EKS. Devi disporre almeno di un profilo Fargate in un cluster per poter pianificare i pod nell'infrastruttura Fargate.

```
New-EKSFargateProfile -FargateProfileName EKSFargateProfile -ClusterName TEST -
Subnet "subnet-02f6ff500ff2067a0", "subnet-0cd976f08d5fbfaae" -PodExecutionRoleArn
```

```
arn:aws:iam::012345678912:role/AmazonEKSFargatePodExecutionRole -Selector
@{Namespace="default"}
```

Output:

```
ClusterName      : TEST
CreatedAt        : 12/26/2019 12:38:21 PM
FargateProfileArn : arn:aws:eks:us-east-2:012345678912:fargateprofile/TEST/
EKSFargateProfile/20b7a11b-8292-41c1-bc56-ffa5e60f6224
FargateProfileName : EKSFargateProfile
PodExecutionRoleArn : arn:aws:iam::012345678912:role/
AmazonEKSFargatePodExecutionRole
Selectors        : {Amazon.EKS.Model.FargateProfileSelector}
Status           : CREATING
Subnets         : {subnet-0cd976f08d5fbfaae, subnet-02f6ff500ff2067a0}
Tags             : {}
```

- Per i dettagli sull'API, vedere [CreateFargateProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-EKSNodeGroup

Il seguente esempio di codice mostra come utilizzare. New-EKSNodeGroup

Strumenti per PowerShell V4

Esempio 1: questo cmdlet crea un gruppo di nodi worker gestito per un cluster Amazon EKS. È possibile creare solo un gruppo di nodi per il cluster che sia uguale alla versione corrente di Kubernetes per il cluster. Tutti i gruppi di nodi vengono creati con l'ultima versione AMI per la rispettiva versione minore di Kubernetes del cluster.

```
New-EKSNodeGroup -NodeGroupName "ProdEKSNodeGroup" -AmiType "AL2_x86_64"
-DiskSize 40 -ClusterName "PROD" -ScalingConfig_DesiredSize 2 -
ScalingConfig_MinSize 2 -ScalingConfig_MaxSize 5 -InstanceType t3.large
-NodeRole "arn:aws:iam::012345678912:role/NodeInstanceRole" -Subnet
"subnet-0d1a9fff35efa7691", "subnet-0a3f4928edbc224d4"
```

Output:

```
AmiType          : AL2_x86_64
```

```

ClusterName      : PROD
CreatedAt        : 12/25/2019 10:16:45 AM
DiskSize         : 40
Health           : Amazon.EKS.Model.NodegroupHealth
InstanceTypes    : {t3.large}
Labels           : {}
ModifiedAt       : 12/25/2019 10:16:45 AM
NodegroupArn      : arn:aws:eks:us-west-2:012345678912:nodegroup/PROD/
ProdEKSNodeGroup/7eb79e47-82b6-04d9-e984-95110db6fa85
NodegroupName    : ProdEKSNodeGroup
NodeRole          : arn:aws:iam::012345678912:role/NodeInstanceRole
ReleaseVersion    : 1.14.7-20190927
RemoteAccess      :
Resources         :
ScalingConfig     : Amazon.EKS.Model.NodegroupScalingConfig
Status           : CREATING
Subnets          : {subnet-0d1a9fff35efa7691, subnet-0a3f4928edbc224d4}
Tags              : {}
Version           : 1.14

```

- Per i dettagli sull'API, vedere [CreateNodegroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EKSCluster

Il seguente esempio di codice mostra come utilizzare. Remove-EKSCluster

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elimina il piano di controllo (control-plane) del cluster Amazon EKS.

```
Remove-EKSCluster -Name "DEV-KUBE-CL"
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EKSCluster (DeleteCluster)" on target "DEV-KUBE-CL".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"): Y

```

```

Arn                : arn:aws:eks:us-west-2:012345678912:cluster/DEV-KUBE-CL
CertificateAuthority : Amazon.EKS.Model.Certificate
ClientRequestToken  :
CreatedAt           : 12/25/2019 9:33:25 AM
Endpoint            : https://02E6D31E3E4F8C15D7BE7F58D527776A.y14.us-
west-2.eks.amazonaws.com
Identity            : Amazon.EKS.Model.Identity
Logging             : Amazon.EKS.Model.Logging
Name                : DEV-KUBE-CL
PlatformVersion     : eks.7
ResourcesVpcConfig  : Amazon.EKS.Model.VpcConfigResponse
RoleArn             : arn:aws:iam::012345678912:role/eks-iam-role
Status              : DELETING
Tags                : {}
Version             : 1.14

```

- Per i dettagli sull'API, vedere [DeleteCluster](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EKSFargateProfile

Il seguente esempio di codice mostra come utilizzare `Remove-EKSFargateProfile`

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elimina un profilo AWS Fargate. Quando elimini un profilo Fargate, vengono eliminati anche tutti i pod in esecuzione in Fargate creati con tale profilo.

```
Remove-EKSFargateProfile -FargateProfileName "EKSFargate" -ClusterName "TEST"
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EKSFargateProfile (DeleteFargateProfile)" on target
"EKSFargate".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

ClusterName      : TEST
CreatedAt         : 12/26/2019 12:34:47 PM

```

```
FargateProfileArn    : arn:aws:eks:us-east-2:012345678912:fargateprofile/TEST/
EKSFargate/42b7a119-e16b-a279-ce97-bdf303adec92
FargateProfileName   : EKSFargate
PodExecutionRoleArn  : arn:aws:iam::012345678912:role/
AmazonEKSFargatePodExecutionRole
Selectors            : {Amazon.EKS.Model.FargateProfileSelector}
Status               : DELETING
Subnets              : {subnet-0cd976f08d5fbfaae, subnet-02f6ff500ff2067a0}
Tags                  : {}
```

- Per i dettagli sull'API, vedere [DeleteFargateProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EKSNodegroup

Il seguente esempio di codice mostra come utilizzare. Remove-EKSNodegroup

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elimina un gruppo di nodi Amazon EKS per un cluster.

```
Remove-EKSNodegroup -NodegroupName "ProdEKSNodeGroup" -ClusterName "PROD"
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EKSNodegroup (DeleteNodegroup)" on target
"ProdEKSNodeGroup".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

AmiType           : AL2_x86_64
ClusterName       : PROD
CreatedAt         : 12/25/2019 10:16:45 AM
DiskSize          : 40
Health            : Amazon.EKS.Model.NodegroupHealth
InstanceTypes     : {t3.large}
Labels            : {}
ModifiedAt        : 12/25/2019 11:01:16 AM
NodegroupArn      : arn:aws:eks:us-west-2:012345678912:nodegroup/PROD/
ProdEKSNodeGroup/7eb79e47-82b6-04d9-e984-95110db6fa85
```

```

NodegroupName : ProdEKSClusterNodeGroup
NodeRole       : arn:aws:iam::012345678912:role/NodeInstanceRole
ReleaseVersion : 1.14.7-20190927
RemoteAccess   :
Resources      : Amazon.EKS.Model.NodegroupResources
ScalingConfig  : Amazon.EKS.Model.NodegroupScalingConfig
Status         : DELETING
Subnets       : {subnet-0d1a9fff35efa7691, subnet-0a3f4928edbc224d4}
Tags           : {}
Version        : 1.14

```

- Per i dettagli sull'API, vedere [DeleteNodegroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-EKSResourceTag

Il seguente esempio di codice mostra come utilizzare. Remove-EKSResourceTag

Strumenti per PowerShell V4

Esempio 1: questo cmdlet elimina i tag specificati da una risorsa EKS.

```

Remove-EKSResourceTag -ResourceArn "arn:aws:eks:us-west-2:012345678912:cluster/PROD"
-TagKey "Name"

```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EKSResourceTag (UntagResource)" on target
"arn:aws:eks:us-west-2:012345678912:cluster/PROD".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y

```

- Per i dettagli sull'API, vedere [UntagResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-EKSClusterConfig

Il seguente esempio di codice mostra come utilizzare. Update-EKSClusterConfig

Strumenti per PowerShell V4

Esempio 1: aggiorna una configurazione di cluster Amazon EKS. Il cluster continua a funzionare durante l'aggiornamento.

```
Update-EKSClusterConfig -Name "PROD" -Logging_ClusterLogging
@{Types="api","audit","authenticator","controllerManager","scheduler",Enabled="True"}
```

Output:

```
CreatedAt : 12/25/2019 5:03:07 PM
Errors    : {}
Id        : ee708232-7d2e-4ed7-9270-d0b5176f0726
Params    : {Amazon.EKS.Model.UpdateParam}
Status    : InProgress
Type      : LoggingUpdate
```

- Per i dettagli sull'API, vedere [UpdateClusterConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-EKSClusterVersion

Il seguente esempio di codice mostra come utilizzare. Update-EKSClusterVersion

Strumenti per PowerShell V4

Esempio 1: questo cmdlet aggiorna un cluster Amazon EKS alla versione Kubernetes specificata. Il cluster continua a funzionare durante l'aggiornamento.

```
Update-EKSClusterVersion -Name "PROD-KUBE-CL" -Version 1.14
```

Output:

```
CreatedAt : 12/26/2019 9:50:37 AM
Errors    : {}
Id        : ef186eff-3b3a-4c25-bcfc-3dcdf9e898a8
Params    : {Amazon.EKS.Model.UpdateParam, Amazon.EKS.Model.UpdateParam}
Status    : InProgress
Type      : VersionUpdate
```


- Per i dettagli sull'API, vedere [UpdateClusterVersion](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

ELB - Esempi della versione 1 che utilizzano Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con ELB - Versione 1.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-ELBLoadBalancerToSubnet

Il seguente esempio di codice mostra come utilizzare. Add-ELBLoadBalancerToSubnet

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge la sottorete specificata al set di sottoreti configurate per il bilanciatore del carico specificato. L'output include l'elenco completo delle sottoreti.

```
Add-ELBLoadBalancerToSubnet -LoadBalancerName my-load-balancer -Subnet  
subnet-12345678
```

Output:

```
subnet-12345678  
subnet-87654321
```

- Per i dettagli sull'API, vedere [AttachLoadBalancerToSubnets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-ELBResourceTag

Il seguente esempio di codice mostra come utilizzare. Add-ELBResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge i tag specificati al bilanciatore del carico specificato. La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
Add-ELBResourceTag -LoadBalancerName my-load-balancer -Tag
@{ Key="project";Value="lima" },@{ Key="department";Value="digital-media" }
```

Esempio 2: con PowerShell la versione 2, è necessario utilizzare New-Object per creare un tag per il parametro Tag.

```
$tag = New-Object Amazon.ElasticLoadBalancing.Model.Tag
$tag.Key = "project"
$tag.Value = "lima"
Add-ELBResourceTag -LoadBalancerName my-load-balancer -Tag $tag
```

- Per i dettagli sull'API, vedere [AddTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Disable-ELBAvailabilityZoneForLoadBalancer

Il seguente esempio di codice mostra come utilizzare. Disable-ELBAvailabilityZoneForLoadBalancer

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la zona di disponibilità specificata dal bilanciatore del carico specificato. L'output include le zone di disponibilità rimanenti.

```
Disable-ELBAvailabilityZoneForLoadBalancer -LoadBalancerName my-load-balancer -
AvailabilityZone us-west-2a
```

Output:

```
us-west-2b
```

- Per i dettagli sull'API, vedere [DisableAvailabilityZonesForLoadBalancer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Dismount-ELBLoadBalancerFromSubnet

Il seguente esempio di codice mostra come utilizzare `Dismount-ELBLoadBalancerFromSubnet` Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la sottorete specificata dal set di sottoreti configurate per il bilanciatore del carico specificato. L'output include le sottoreti rimanenti.

```
Dismount-ELBLoadBalancerFromSubnet -LoadBalancerName my-load-balancer -Subnet  
subnet-12345678
```

Output:

```
subnet-87654321
```

- Per i dettagli sull'API, vedere [DetachLoadBalancerFromSubnets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-ELBLoadBalancerAttribute

Il seguente esempio di codice mostra come utilizzare `Edit-ELBLoadBalancerAttribute` Strumenti per PowerShell V4

Esempio 1: questo esempio abilita il bilanciamento del carico tra zone per il bilanciatore del carico specificato.

```
Edit-ELBLoadBalancerAttribute -LoadBalancerName my-load-balancer -  
CrossZoneLoadBalancing_Enabled $true
```

Esempio 2: questo esempio disabilita lo svuotamento della connessione per il bilanciatore del carico specificato.

```
Edit-ELBLoadBalancerAttribute -LoadBalancerName my-load-balancer -  
ConnectionDraining_Enabled $false
```

Esempio 3: questo esempio abilita la registrazione dei log degli accessi per il bilanciatore del carico specificato.

```
Edit-ELBLoadBalancerAttribute -LoadBalancerName my-load-balancer `  
>> -AccessLog_Enabled $true `  
>> -AccessLog_S3BucketName amzn-s3-demo-logging-bucket `  
>> -AccessLog_S3BucketPrefix my-app/prod `  
>> -AccessLog_EmitInterval 60
```

- Per i dettagli sull'API, vedere [ModifyLoadBalancerAttributes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-ELBAvailabilityZoneForLoadBalancer

Il seguente esempio di codice mostra come utilizzare. `Enable-ELBAvailabilityZoneForLoadBalancer`

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge la zona di disponibilità specificata al bilanciatore del carico specificato. L'output include l'elenco completo delle zone di disponibilità.

```
Enable-ELBAvailabilityZoneForLoadBalancer -LoadBalancerName my-load-balancer -  
AvailabilityZone us-west-2a
```

Output:

```
us-west-2a  
us-west-2b
```

- Per i dettagli sull'API, vedere [EnableAvailabilityZonesForLoadBalancer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELBInstanceHealth

Il seguente esempio di codice mostra come utilizzare. `Get-ELBInstanceHealth`

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive lo stato delle istanze registrate con il bilanciatore del carico specificato.

```
Get-ELBInstanceHealth -LoadBalancerName my-load-balancer
```

Output:

Description State	InstanceId	ReasonCode
-----	-----	-----
N/A	i-87654321	N/A
InService		
Instance has failed at lea... i-12345678		Instance
OutOfService		

Esempio 2: questo esempio descrive lo stato dell'istanza registrata con il bilanciatore del carico specificato.

```
Get-ELBInstanceHealth -LoadBalancerName my-load-balancer -Instance i-12345678
```

Esempio 3: questo esempio visualizza la descrizione completa dello stato dell'istanza specificata.

```
(Get-ELBInstanceHealth -LoadBalancerName my-load-balancer -Instance  
i-12345678).Description
```

Output:

```
Instance has failed at least the UnhealthyThreshold number of health checks  
consecutively.
```

- Per i dettagli sull'API, vedere [DescribeInstanceHealth](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELBLoadBalancer

Il seguente esempio di codice mostra come utilizzare. Get-ELBLoadBalancer

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i nomi dei bilanciatori del carico.

```
Get-ELBLoadBalancer | format-table -property LoadBalancerName
```

Output:

```
LoadBalancerName
-----
my-load-balancer
my-other-load-balancer
my-internal-load-balancer
```

Esempio 2: questo esempio descrive il bilanciatore del carico specificato.

```
Get-ELBLoadBalancer -LoadBalancerName my-load-balancer
```

Output:

```
AvailabilityZones      : {us-west-2a, us-west-2b}
BackendServerDescriptions :
  {Amazon.ElasticLoadBalancing.Model.BackendServerDescription}
CanonicalHostedZoneName : my-load-balancer-1234567890.us-west-2.elb.amazonaws.com
CanonicalHostedZoneNameID : Z3DZXE0EXAMPLE
CreatedTime            : 4/11/2015 12:12:45 PM
DNSName                 : my-load-balancer-1234567890.us-west-2.elb.amazonaws.com
HealthCheck             : Amazon.ElasticLoadBalancing.Model.HealthCheck
Instances               : {i-207d9717, i-afefb49b}
ListenerDescriptions    : {Amazon.ElasticLoadBalancing.Model.ListenerDescription}
LoadBalancerName        : my-load-balancer
Policies                : Amazon.ElasticLoadBalancing.Model.Policies
Scheme                 : internet-facing
SecurityGroups          : {sg-a61988c3}
SourceSecurityGroup      : Amazon.ElasticLoadBalancing.Model.SourceSecurityGroup
Subnets                : {subnet-15aaab61}
VPCId                   : vpc-a01106c2
```

Esempio 3: Questo esempio descrive tutti i sistemi di bilanciamento del carico nella regione corrente. AWS

Get-ELBLoadBalancer

Esempio 4: questo esempio descrive tutti i bilanciatori del carico in tutte le Regioni AWS disponibili.

```
Get-AWSRegion | % { Get-ELBLoadBalancer -Region $_ }
```

- Per i dettagli sull'API, vedere [DescribeLoadBalancers](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELBLoadBalancerAttribute

Il seguente esempio di codice mostra come utilizzare. Get-ELBLoadBalancerAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive gli attributi del bilanciatore del carico specificato.

```
Get-ELBLoadBalancerAttribute -LoadBalancerName my-load-balancer
```

Output:

```
AccessLog           : Amazon.ElasticLoadBalancing.Model.AccessLog
AdditionalAttributes : {}
ConnectionDraining  : Amazon.ElasticLoadBalancing.Model.ConnectionDraining
ConnectionSettings  : Amazon.ElasticLoadBalancing.Model.ConnectionSettings
CrossZoneLoadBalancing : Amazon.ElasticLoadBalancing.Model.CrossZoneLoadBalancing
```

- Per i dettagli sull'API, vedere [DescribeLoadBalancerAttributes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELBLoadBalancerPolicy

Il seguente esempio di codice mostra come utilizzare. Get-ELBLoadBalancerPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le policy associate al bilanciatore del carico specificato.

```
Get-ELBLoadBalancerPolicy -LoadBalancerName my-load-balancer
```

Output:

PolicyAttributeDescriptions PolicyTypeName	PolicyName
-----	-----
{ProxyProtocol}	my-ProxyProtocol-policy
ProxyProtocolPolicyType	
{CookieName}	my-app-cookie-policy
AppCookieStickinessPolicyType	

Esempio 2: questo esempio descrive gli attributi del bilanciatore del carico specificato.

```
(Get-ELBLoadBalancerPolicy -LoadBalancerName my-load-balancer -PolicyName my-ProxyProtocol-policy).PolicyAttributeDescriptions
```

Output:

AttributeName	AttributeValue
-----	-----
ProxyProtocol	true

Esempio 3: questo esempio descrive le policy predefinite, incluse le policy di esempio. I nomi delle politiche di esempio hanno il prefisso ELBSample -.

```
Get-ELBLoadBalancerPolicy
```

Output:

PolicyAttributeDescriptions PolicyTypeName	PolicyName
-----	-----
{Protocol-SSLv2, Protocol-TLSv1, Pro... SSLNegotiationPolicyType	ELBSecurityPolicy-2015-05
{Protocol-SSLv2, Protocol-TLSv1, Pro... SSLNegotiationPolicyType	ELBSecurityPolicy-2015-03
{Protocol-SSLv2, Protocol-TLSv1, Pro... SSLNegotiationPolicyType	ELBSecurityPolicy-2015-02


```
{Protocol-SSLv2, Protocol-TLSv1, Pro... ELBSecurityPolicy-2014-10
  SSLNegotiationPolicyType
{Protocol-SSLv2, Protocol-TLSv1, Pro... ELBSecurityPolicy-2014-01
  SSLNegotiationPolicyType
{Protocol-SSLv2, Protocol-TLSv1, Pro... ELBSecurityPolicy-2011-08
  SSLNegotiationPolicyType
{Protocol-SSLv2, Protocol-TLSv1, Pro... ELBSample-ELBDefaultCipherPolicy
  SSLNegotiationPolicyType
{Protocol-SSLv2, Protocol-TLSv1, Pro... ELBSample-OpenSSLDefaultCipherPolicy
  SSLNegotiationPolicyType
```

- Per i dettagli sull'API, vedere [DescribeLoadBalancerPolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELBLoadBalancerPolicyType

Il seguente esempio di codice mostra come utilizzare `Get-ELBLoadBalancerPolicyType`

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene i tipi di policy supportati da Elastic Load Balancing.

```
Get-ELBLoadBalancerPolicyType
```

Output:

Description PolicyTypeName ----- -----	PolicyAttributeTypeDescriptions
Stickiness policy with session lifet... LBCookieStickinessPolicyType	{CookieExpirationPeriod}
Policy that controls authentication ... BackendServerAuthenticationPolicyType	{PublicKeyPolicyName}
Listener policy that defines the cip... SSLNegotiationPolicyType	{Protocol-SSLv2, Protocol-TLSv1, Pro...
Policy containing a list of public k... PublicKeyPolicyType	{PublicKey}
Stickiness policy with session lifet... AppCookieStickinessPolicyType	{CookieName}
Policy that controls whether to incl... ProxyProtocolPolicyType	{ProxyProtocol}

Esempio 2: questo esempio descrive il tipo di policy specificato.

```
Get-ELBLoadBalancerPolicyType -PolicyTypeName ProxyProtocolPolicyType
```

Output:

Description	PolicyAttributeTypeDescriptions
PolicyTypeName	
-----	-----

Policy that controls whether to incl...	{ProxyProtocol}
ProxyProtocolPolicyType	

Esempio 3: questo esempio visualizza la descrizione completa del tipo di policy specificato.

```
(Get-ELBLoadBalancerPolicyType -PolicyTypeName).Description
```

Output:

```
Policy that controls whether to include the IP address and port of the originating
request for TCP messages.
This policy operates on TCP/SSL listeners only
```

- Per i dettagli sull'API, vedere [DescribeLoadBalancerPolicyTypes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELBResourceTag

Il seguente esempio di codice mostra come utilizzare. Get-ELBResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i tag dei bilanciatori del carico specificati.

```
Get-ELBResourceTag -LoadBalancerName @("my-load-balancer","my-internal-load-
balancer")
```

Output:

LoadBalancerName	Tags

```

-----
my-load-balancer      {project, department}
my-internal-load-balancer {project, department}

```

Esempio 2: questo esempio descrive i tag del bilanciatore del carico specificato.

```
(Get-ELBResourceTag -LoadBalancerName my-load-balancer).Tags
```

Output:

Key	Value
---	-----
project	lima
department	digital-media

- Per i dettagli sull'API, vedere [DescribeTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Join-ELBSecurityGroupToLoadBalancer

Il seguente esempio di codice mostra come utilizzare `Join-ELBSecurityGroupToLoadBalancer`

Strumenti per PowerShell V4

Esempio 1: questo esempio sostituisce il gruppo di sicurezza corrente per il bilanciatore del carico specificato con il gruppo di sicurezza specificato.

```
Join-ELBSecurityGroupToLoadBalancer -LoadBalancerName my-load-balancer -
SecurityGroup sg-87654321
```

Output:

```
sg-87654321
```

Esempio 2: come mantenere il gruppo di sicurezza corrente e specificare un gruppo di sicurezza aggiuntivo, specifica sia il gruppo di sicurezza esistente che quello nuovo.

```
Join-ELBSecurityGroupToLoadBalancer -LoadBalancerName my-load-balancer -
SecurityGroup @("sg-12345678", "sg-87654321")
```

Output:

```
sg-12345678  
sg-87654321
```

- Per i dettagli sull'API, vedere [ApplySecurityGroupsToLoadBalancer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ELBAppCookieStickinessPolicy

Il seguente esempio di codice mostra come utilizzare. New-ELBAppCookieStickinessPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una policy di persistenza conforme alla durata delle sessioni persistenti del cookie generato dall'applicazione specificato.

```
New-ELBAppCookieStickinessPolicy -LoadBalancerName my-load-balancer -PolicyName my-  
app-cookie-policy -CookieName my-app-cookie
```

- Per i dettagli sull'API, vedere [CreateAppCookieStickinessPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ELBLBCookieStickinessPolicy

Il seguente esempio di codice mostra come utilizzare. New-ELBLBCookieStickinessPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una policy di persistenza con durate delle sessioni persistenti controllate dal periodo di scadenza specificato (in secondi).

```
New-ELBLBCookieStickinessPolicy -LoadBalancerName my-load-balancer -PolicyName my-  
duration-cookie-policy -CookieExpirationPeriod 60
```

Esempio 2: questo esempio crea una policy di persistenza con cicli di vita delle sessioni persistenti controllati dal ciclo di vita del browser (user-agent).

```
New-ELBLBCookieStickinessPolicy -LoadBalancerName my-load-balancer -PolicyName my-  
duration-cookie-policy
```

- Per i dettagli sull'API, vedere [CreateLbCookieStickinessPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ELBLoadBalancer

Il seguente esempio di codice mostra come utilizzare New-ELBLoadBalancer

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un bilanciatore del carico con un listener HTTP in un VPC.

```
$httpListener = New-Object Amazon.ElasticLoadBalancing.Model.Listener
$httpListener.Protocol = "http"
$httpListener.LoadBalancerPort = 80
$httpListener.InstanceProtocol = "http"
$httpListener.InstancePort = 80
New-ELBLoadBalancer -LoadBalancerName my-vpc-load-balancer -SecurityGroup sg-
a61988c3 -Subnet subnet-15aaab61 -Listener $httpListener

my-vpc-load-balancer-1234567890.us-west-2.elb.amazonaws.com
```

Esempio 2: Questo esempio crea un load balancer con un listener HTTP in -Classic. EC2

```
New-ELBLoadBalancer -LoadBalancerName my-classic-load-balancer -AvailabilityZone us-
west-2a -Listener $httpListener
```

Output:

```
my-classic-load-balancer-123456789.us-west-2.elb.amazonaws.com
```

Esempio 3: questo esempio crea un bilanciatore del carico con un listener HTTP.

```
$httpsListener = New-Object Amazon.ElasticLoadBalancing.Model.Listener
$httpsListener.Protocol = "https"
$httpsListener.LoadBalancerPort = 443
$httpsListener.InstanceProtocol = "http"
$httpsListener.InstancePort = 80
$httpsListener.SSLCertificateId="arn:aws:iam::123456789012:server-certificate/my-
server-cert"
New-ELBLoadBalancer -LoadBalancerName my-load-balancer -AvailabilityZone us-west-2a
-Listener $httpsListener
```

```
my-load-balancer-123456789.us-west-2.elb.amazonaws.com
```

- Per i dettagli sull'API, vedere [CreateLoadBalancer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ELBLoadBalancerListener

Il seguente esempio di codice mostra come utilizzare `New-ELBLoadBalancerListener`

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un listener HTTPS al bilanciatore del carico specificato.

```
$httpsListener = New-Object Amazon.ElasticLoadBalancing.Model.Listener
$httpsListener.Protocol = "https"
$httpsListener.LoadBalancerPort = 443
$httpsListener.InstanceProtocol = "https"
$httpsListener.InstancePort = 443
$httpsListener.SSLCertificateId="arn:aws:iam::123456789012:server-certificate/my-
server-cert"
New-ELBLoadBalancerListener -LoadBalancerName my-load-balancer -Listener
$httpsListener
```

- Per i dettagli sull'API, vedere [CreateLoadBalancerListeners](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ELBLoadBalancerPolicy

Il seguente esempio di codice mostra come utilizzare `New-ELBLoadBalancerPolicy`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova policy del protocollo proxy per un bilanciatore del carico specificato.

```
$attribute = New-Object Amazon.ElasticLoadBalancing.Model.PolicyAttribute -Property
@{
    AttributeName="ProxyProtocol"
    AttributeValue="True"
}
```

```
New-ELBLoadBalancerPolicy -LoadBalancerName my-load-balancer -PolicyName my-ProxyProtocol-policy -PolicyTypeName ProxyProtocolPolicyType -PolicyAttribute $attribute
```

- Per i dettagli sull'API, vedere [CreateLoadBalancerPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-ELBInstanceWithLoadBalancer

Il seguente esempio di codice mostra come utilizzare. Register-ELBInstanceWithLoadBalancer

Strumenti per PowerShell V4

Esempio 1: Questo esempio registra l' EC2 istanza specificata con il sistema di bilanciamento del carico specificato.

```
Register-ELBInstanceWithLoadBalancer -LoadBalancerName my-load-balancer -Instance i-12345678
```

Output:

```
InstanceId
-----
i-12345678
i-87654321
```

- Per i dettagli sull'API, vedere [RegisterInstancesWithLoadBalancer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELBInstanceFromLoadBalancer

Il seguente esempio di codice mostra come utilizzare. Remove-ELBInstanceFromLoadBalancer

Strumenti per PowerShell V4

Esempio 1: Questo esempio rimuove l' EC2 istanza specificata dal sistema di bilanciamento del carico specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-ELBInstanceFromLoadBalancer -LoadBalancerName my-load-balancer -Instance  
i-12345678
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing operation "Remove-ELBInstanceFromLoadBalancer  
(DeregisterInstancesFromLoadBalancer)" on Target  
"Amazon.ElasticLoadBalancing.Model.Instance".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"):  
  
InstanceId  
-----  
i-87654321
```

- Per i dettagli sull'API, vedere [DeregisterInstancesFromLoadBalancer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELBLoadBalancer

Il seguente esempio di codice mostra come utilizzare `Remove-ELBLoadBalancer`

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il bilanciatore del carico specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro `Force`.

```
Remove-ELBLoadBalancer -LoadBalancerName my-load-balancer
```

Output:

```
Confirm  
Are you sure you want to perform this action?  
Performing operation "Remove-ELBLoadBalancer (DeleteLoadBalancer)" on Target "my-  
load-balancer".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"):
```


- Per i dettagli sull'API, vedere [DeleteLoadBalancer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELBLoadBalancerListener

Il seguente esempio di codice mostra come utilizzare. Remove-ELBLoadBalancerListener

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il listener sulla porta 80 per il bilanciatore del carico specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-ELBLoadBalancerListener -LoadBalancerName my-load-balancer -LoadBalancerPort 80
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ELBLoadBalancerListener (DeleteLoadBalancerListeners)"
on Target "80".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteLoadBalancerListeners](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELBLoadBalancerPolicy

Il seguente esempio di codice mostra come utilizzare. Remove-ELBLoadBalancerPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la policy specificata dal bilanciatore del carico specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force.

```
Remove-ELBLoadBalancerPolicy -LoadBalancerName my-load-balancer -PolicyName my-
duration-cookie-policy
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ELBLoadBalancerPolicy (DeleteLoadBalancerPolicy)" on
Target "my-duration-cookie-policy".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

- Per i dettagli sull'API, vedere [DeleteLoadBalancerPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELBResourceTag

Il seguente esempio di codice mostra come utilizzare. Remove-ELBResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il tag specificato dal bilanciatore del carico specificato. Prima di procedere con l'operazione, viene richiesta una conferma, a meno che non si specifichi anche il parametro Force. La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
Remove-ELBResourceTag -LoadBalancerName my-load-balancer -Tag @{ Key="project" }
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELBResourceTag (RemoveTags)" on target
"Amazon.ElasticLoadBalancing.Model.TagKeyOnly".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"):
```

Esempio 2: con PowerShell versione 2, è necessario utilizzare New-Object per creare il tag per il parametro Tag.

```
$tag = New-Object Amazon.ElasticLoadBalancing.Model.TagKeyOnly
$tag.Key = "project"
```

```
Remove-ELBResourceTag -Tag $tag -Force
```

- Per i dettagli sull'API, vedere [RemoveTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ELBHealthCheck

Il seguente esempio di codice mostra come utilizzare. Set-ELBHealthCheck

Strumenti per PowerShell V4

Esempio 1: questo esempio configura le impostazioni del controllo dell'integrità per il bilanciatore del carico specificato.

```
Set-ELBHealthCheck -LoadBalancerName my-load-balancer `
>> -HealthCheck_HealthyThreshold 2 `
>> -HealthCheck_UnhealthyThreshold 2 `
>> -HealthCheck_Target "HTTP:80/ping" `
>> -HealthCheck_Interval 30 `
>> -HealthCheck_Timeout 3
```

Output:

```
HealthyThreshold    : 2
Interval            : 30
Target              : HTTP:80/ping
Timeout             : 3
UnhealthyThreshold  : 2
```

- Per i dettagli sull'API, vedere [ConfigureHealthCheck](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ELBLoadBalancerListenerSSLCertificate

Il seguente esempio di codice mostra come utilizzare. Set-ELBLoadBalancerListenerSSLCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio sostituisce il certificato che termina le connessioni SSL per il listener specificato.

```
Set-ELBLoadBalancerListenerSSLCertificate -LoadBalancerName my-load-balancer `
>> -LoadBalancerPort 443 `
>> -SSLCertificateId "arn:aws:iam::123456789012:server-certificate/new-server-cert"
```

- Per i dettagli sull'API, vedere [SetLoadBalancerListenerSslCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ELBLoadBalancerPolicyForBackendServer

Il seguente esempio di codice mostra come utilizzare. Set-ELBLoadBalancerPolicyForBackendServer

Strumenti per PowerShell V4

Esempio 1: questo esempio sostituisce le policy per la porta specificata con la policy specificata.

```
Set-ELBLoadBalancerPolicyForBackendServer -LoadBalancerName my-load-balancer -
InstancePort 80 -PolicyName my-ProxyProtocol-policy
```

Esempio 2: questo esempio rimuove tutte le policy associate alla porta specificata.

```
Set-ELBLoadBalancerPolicyForBackendServer -LoadBalancerName my-load-balancer -
InstancePort 80
```

- Per i dettagli sull'API, vedere [SetLoadBalancerPoliciesForBackendServer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ELBLoadBalancerPolicyOfListener

Il seguente esempio di codice mostra come utilizzare. Set-ELBLoadBalancerPolicyOfListener

Strumenti per PowerShell V4

Esempio 1: questo esempio sostituisce le policy per il listener specificato con la policy specificata.

```
Set-ELBLoadBalancerPolicyOfListener -LoadBalancerName my-load-balancer -
LoadBalancerPort 443 -PolicyName my-SSLNegotiation-policy
```

Esempio 2: questo esempio rimuove tutte le policy associate al listener specificato.

```
Set-ELBLoadBalancerPolicyOfListener -LoadBalancerName my-load-balancer -  
LoadBalancerPort 443
```

- Per i dettagli sull'API, vedere [SetLoadBalancerPoliciesOfListener](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

ELB - Esempi della versione 2 che utilizzano Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con ELB - Versione 2.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-ELB2ListenerCertificate

Il seguente esempio di codice mostra come utilizzare. Add-ELB2ListenerCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un certificato aggiuntivo al listener specificato.

```
Add-ELB2ListenerCertificate -ListenerArn 'arn:aws:elasticloadbalancing:us-  
east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/3873f123b98f7618' -  
Certificate @{CertificateArn = 'arn:aws:acm:us-  
east-1:123456789012:certificate/19478bd5-491d-47d4-b1d7-5217feba1d97'}
```

Output:

```

CertificateArn
IsDefault
-----
-----
arn:aws:acm:us-east-1:123456789012:certificate/19478bd5-491d-47d4-b1d7-5217feba1d97
False

```

- Per i dettagli sull'API, vedere [AddListenerCertificates](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-ELB2Tag

Il seguente esempio di codice mostra come utilizzare. Add-ELB2Tag

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un nuovo tag alla risorsa **AWS.Tools.ElasticLoadBalancingV2** specificata.

```

Add-ELB2Tag -ResourceArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -Tag @{Key = 'productVersion'; Value = '1.0.0'}

```

- Per i dettagli sull'API, vedere [AddTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-ELB2Listener

Il seguente esempio di codice mostra come utilizzare. Edit-ELB2Listener

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica l'azione predefinita del listener specificato impostandola sulla risposta fissa.

```

$newDefaultAction = [Amazon.ElasticLoadBalancingV2.Model.Action]@{
    "FixedResponseConfig" = @{
        "ContentType" = "text/plain"
        "MessageBody" = "Hello World"
        "StatusCode" = "200"
    }
}

```

```
"Type" = [Amazon.ElasticLoadBalancingV2.ActionTypeEnum]::FixedResponse
}

Edit-ELB2Listener -ListenerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/testALB/3e2f03b558e19676/d19f2f14974db685' -Port 8080 -DefaultAction $newDefaultAction
```

Output:

```
Certificates      : {}
DefaultActions    : {Amazon.ElasticLoadBalancingV2.Model.Action}
ListenerArn       : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/testALB/3e2f03b558e19676/d19f2f14974db685
LoadBalancerArn   : arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/testALB/3e2f03b558e19676
Port              : 8080
Protocol          : HTTP
SslPolicy         :
```

- Per i dettagli sull'API, vedere [ModifyListener](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-ELB2LoadBalancerAttribute

Il seguente esempio di codice mostra come utilizzare `Edit-ELB2LoadBalancerAttribute`

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica gli attributi del bilanciatore del carico specificato.

```
Edit-ELB2LoadBalancerAttribute -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -Attribute @{Key = 'deletion_protection.enabled'; Value = 'true'}
```

Output:

Key	Value
---	----
deletion_protection.enabled	true
access_logs.s3.enabled	false
access_logs.s3.bucket	
access_logs.s3.prefix	

```
idle_timeout.timeout_seconds      60
routing.http2.enabled             true
routing.http.drop_invalid_header_fields.enabled false
```

- Per i dettagli sull'API, vedere [ModifyLoadBalancerAttributes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-ELB2Rule

Il seguente esempio di codice mostra come utilizzare. Edit-ELB2Rule

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica le configurazioni delle regole specificate per il listener.

```
$newRuleCondition = [Amazon.ElasticLoadBalancingV2.Model.RuleCondition]@{
    "PathPatternConfig" = @{
        "Values" = "/login1", "/login2", "/login3"
    }
    "Field" = "path-pattern"
}

Edit-ELB2Rule -RuleArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/testALB/3e2f03b558e19676/1c84f02aec143e80/f4f51dfaa033a8cc' -Condition $newRuleCondition
```

Output:

```
Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}
IsDefault    : False
Priority      : 10
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/testALB/3e2f03b558e19676/1c84f02aec143e80/f4f51dfaa033a8cc
```

- Per i dettagli sull'API, vedere [ModifyRule](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-ELB2TargetGroup

Il seguente esempio di codice mostra come utilizzare. Edit-ELB2TargetGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica le proprietà del gruppo di destinazione specificato.

```
Edit-ELB2TargetGroup -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970' -HealthCheckIntervalSecond 60 -HealthCheckPath '/index.html' -HealthCheckPort 8080
```

Output:

```
HealthCheckEnabled      : True
HealthCheckIntervalSeconds : 60
HealthCheckPath         : /index.html
HealthCheckPort         : 8080
HealthCheckProtocol     : HTTP
HealthCheckTimeoutSeconds : 5
HealthyThresholdCount   : 5
LoadBalancerArns       : {}
Matcher                 : Amazon.ElasticLoadBalancingV2.Model.Matcher
Port                    : 80
Protocol                : HTTP
TargetGroupArn          : arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970
TargetGroupName         : test-tg
TargetType              : instance
UnhealthyThresholdCount : 2
VpcId                   : vpc-2cfd7000
```

- Per i dettagli sull'API, vedere [ModifyTargetGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-ELB2TargetGroupAttribute

Il seguente esempio di codice mostra come utilizzare. Edit-ELB2TargetGroupAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica l'attributo deregistration_delay del gruppo di destinazione specificato.

```
Edit-ELB2TargetGroupAttribute -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970' -Attribute @{Key = 'deregistration_delay.timeout_seconds'; Value = 600}
```

Output:

Key	Value
---	----
stickiness.enabled	false
deregistration_delay.timeout_seconds	600
stickiness.type	lb_cookie
stickiness.lb_cookie.duration_seconds	86400
slow_start.duration_seconds	0
load_balancing.algorithm.type	round_robin

- Per i dettagli sull'API, vedere [ModifyTargetGroupAttributes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2AccountLimit

Il seguente esempio di codice mostra come utilizzare. Get-ELB2AccountLimit

Strumenti per PowerShell V4

Esempio 1: Questo comando elenca i limiti di ELB2 account per una determinata regione.

```
Get-ELB2AccountLimit
```

Output:

Max	Name
---	----
3000	target-groups
1000	targets-per-application-load-balancer
50	listeners-per-application-load-balancer
100	rules-per-application-load-balancer
50	network-load-balancers
3000	targets-per-network-load-balancer
500	targets-per-availability-zone-per-network-load-balancer
50	listeners-per-network-load-balancer
5	condition-values-per-alb-rule

```

5    condition-wildcards-per-alb-rule
100  target-groups-per-application-load-balancer
5    target-groups-per-action-on-application-load-balancer
1    target-groups-per-action-on-network-load-balancer
50   application-load-balancers

```

- Per i dettagli sull'API, vedere [DescribeAccountLimits](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2Listener

Il seguente esempio di codice mostra come utilizzare. Get-ELB2Listener

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive i listener del bilanciatore ALB/NLB specificato.

```

Get-ELB2Listener -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f'

```

Output:

```

Certificates      : {}
DefaultActions    : {Amazon.ElasticLoadBalancingV2.Model.Action}
ListenerArn       : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/
test-alb/3651b4394dd9a24f/1dac07c21187d41e
LoadBalancerArn   : arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/
app/test-alb/3651b4394dd9a24f
Port              : 80
Protocol          : HTTP
SslPolicy         :

Certificates      : {Amazon.ElasticLoadBalancingV2.Model.Certificate}
DefaultActions    : {Amazon.ElasticLoadBalancingV2.Model.Action}
ListenerArn       : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/
test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b
LoadBalancerArn   : arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/
app/test-alb/3651b4394dd9a24f
Port              : 443
Protocol          : HTTPS
SslPolicy         : ELBSecurityPolicy-2016-08

```

- Per i dettagli sull'API, vedere [DescribeListeners](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2ListenerCertificate

Il seguente esempio di codice mostra come utilizzare. Get-ELB2ListenerCertificate
Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il certificato per il listener specificato.

```
Get-ELB2ListenerCertificate -ListenerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b'
```

Output:

```
CertificateArn
IsDefault
-----
-----
arn:aws:acm:us-east-1:123456789012:certificate/5fc7c092-68bf-4862-969c-22fd48b6e17c
True
```

- Per i dettagli sull'API, vedere [DescribeListenerCertificates](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2LoadBalancer

Il seguente esempio di codice mostra come utilizzare. Get-ELB2LoadBalancer
Strumenti per PowerShell V4

Esempio 1: questo esempio mostra tutti i bilanciatori del carico per la Regione specificata.

```
Get-ELB2LoadBalancer
```

Output:

```
AvailabilityZones      : {us-east-1c}
```

```
CanonicalHostedZoneId : Z26RNL4JYFT0TI
CreatedTime            : 6/22/18 11:21:50 AM
DNSName                : test-elb1234567890-238d34ad8d94bc2e.elb.us-
east-1.amazonaws.com
IpAddressType          : ipv4
LoadBalancerArn        : arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/net/test-elb1234567890/238d34ad8d94bc2e
LoadBalancerName       : test-elb1234567890
Scheme                 : internet-facing
SecurityGroups         : {}
State                  : Amazon.ElasticLoadBalancingV2.Model.LoadBalancerState
Type                   : network
VpcId                  : vpc-2cf00000
```

- Per i dettagli sull'API, vedere [DescribeLoadBalancers](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2LoadBalancerAttribute

Il seguente esempio di codice mostra come utilizzare `Get-ELB2LoadBalancerAttribute` Strumenti per PowerShell V4

Esempio 1: questo comando descrive gli attributi del bilanciatore del carico specificato.

```
Get-ELB2LoadBalancerAttribute -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/net/test-elb/238d34ad8d94bc2e'
```

Output:

Key	Value
---	-----
access_logs.s3.enabled	false
load_balancing.cross_zone.enabled	true
access_logs.s3.prefix	
deletion_protection.enabled	false
access_logs.s3.bucket	

- Per i dettagli sull'API, vedere [DescribeLoadBalancerAttributes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2Rule

Il seguente esempio di codice mostra come utilizzare. Get-ELB2Rule

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le regole per l'ARN del listener specificato.

```
Get-ELB2Rule -ListenerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b'
```

Output:

```
Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}
IsDefault    : False
Priority      : 1
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b/2286fff5055e0f79

Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}
IsDefault    : False
Priority      : 2
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b/14e7b036567623ba

Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {}
IsDefault    : True
Priority      : default
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b/853948cf3aa9b2bf
```

- Per i dettagli sull'API, vedere [DescribeRules](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2SSLPolicy

Il seguente esempio di codice mostra come utilizzare. Get-ELB2SSLPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le politiche di ascolto disponibili per la ElasticLoadBalancing V2.

```
Get-ELB2SSLPolicy
```

Output:

Ciphers	Name	SslProtocols
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-2016-08	{TLSv1, TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-TLS-1-2-2017-01	{TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-TLS-1-1-2017-01	{TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-TLS-1-2-Ext-2018-06	{TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-FS-2018-06	{TLSv1, TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-2015-05	{TLSv1, TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-TLS-1-0-2015-04	{TLSv1, TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-FS-1-2-Res-2019-08	{TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-FS-1-1-2019-08	{TLSv1.1, TLSv1.2}
{ECDHE-ECDSA-AES128-GCM-SHA256, ECDHE-RSA-AES128-GCM-SHA256, ECDHE-ECDSA-AES128-SHA256, ECDHE-RSA-AES128-SHA256}	ELBSecurityPolicy-FS-1-2-2019-08	{TLSv1.2}

- Per i dettagli sull'API, vedere [DescribeSslPolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2Tag

Il seguente esempio di codice mostra come utilizzare. Get-ELB2Tag

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutti i tag per la risorsa specificata.

```
Get-ELB2Tag -ResourceArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f'
```

Output:

```
ResourceArn
            Tags
-----
-----
arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f {stage, internalName, version}
```

- Per i dettagli sull'API, vedere [DescribeTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2TargetGroup

Il seguente esempio di codice mostra come utilizzare. Get-ELB2TargetGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive il gruppo di destinazione specificato.

```
Get-ELB2TargetGroup -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970'
```

Output:

```
HealthCheckEnabled      : True
HealthCheckIntervalSeconds : 30
HealthCheckPath         : /
HealthCheckPort         : traffic-port
HealthCheckProtocol     : HTTP
```



```

HealthCheckTimeoutSeconds : 5
HealthyThresholdCount      : 5
LoadBalancerArns           : {arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f}
Matcher                    : Amazon.ElasticLoadBalancingV2.Model.Matcher
Port                        : 80
Protocol                   : HTTP
TargetGroupArn             : arn:aws:elasticloadbalancing:us-
east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970
TargetGroupName           : test-tg
TargetType                 : instance
UnhealthyThresholdCount    : 2
VpcId                     : vpc-2cfd7000

```

- Per i dettagli sull'API, vedere [DescribeTargetGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2TargetGroupAttribute

Il seguente esempio di codice mostra come utilizzare `Get-ELB2TargetGroupAttribute`

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive gli attributi del gruppo di destinazione specificato.

```

Get-ELB2TargetGroupAttribute -TargetGroupArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970'

```

Output:

Key	Value
---	----
stickiness.enabled	false
deregistration_delay.timeout_seconds	300
stickiness.type	lb_cookie
stickiness.lb_cookie.duration_seconds	86400
slow_start.duration_seconds	0
load_balancing.algorithm.type	round_robin

- Per i dettagli sull'API, vedere [DescribeTargetGroupAttributes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ELB2TargetHealth

Il seguente esempio di codice mostra come utilizzare. Get-ELB2TargetHealth

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce lo stato di integrità delle destinazioni presenti nel gruppo di destinazione specificato.

```
Get-ELB2TargetHealth -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970'
```

Output:

HealthCheckPort	Target	TargetHealth
80	Amazon.ElasticLoadBalancingV2.Model.TargetDescription	
	Amazon.ElasticLoadBalancingV2.Model.TargetHealth	

- Per i dettagli sull'API, vedere [DescribeTargetHealth](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ELB2Listener

Il seguente esempio di codice mostra come utilizzare. New-ELB2Listener

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo listener ALB con l'azione predefinita "Forward" per inviare il traffico al gruppo di destinazione specificato.

```
$defaultAction = [Amazon.ElasticLoadBalancingV2.Model.Action]@{
    ForwardConfig = @{
        TargetGroups = @(
            @{ TargetGroupArn = "arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/testAlbTG/3d61c2f20aa5bccb" }
        )
        TargetGroupStickinessConfig = @{
            DurationSeconds = 900
            Enabled = $true
        }
    }
}
```

```
Type = "Forward"
}

New-ELB2Listener -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/testALB/3e2f03b558e19676' -Port 8001 -Protocol "HTTP" -DefaultAction $defaultAction
```

Output:

```
Certificates      : {}
DefaultActions    : {Amazon.ElasticLoadBalancingV2.Model.Action}
ListenerArn       : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/testALB/3e2f03b558e19676/1c84f02aec143e80
LoadBalancerArn   : arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/testALB/3e2f03b558e19676
Port              : 8001
Protocol          : HTTP
SslPolicy         :
```

- Per i dettagli sull'API, vedere [CreateListener](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ELB2LoadBalancer

Il seguente esempio di codice mostra come utilizzare. New-ELB2LoadBalancer

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo Application Load Balancer con connessione Internet con due sottoreti.

```
New-ELB2LoadBalancer -Type application -Scheme internet-facing -IpAddressType ipv4 -Name 'New-Test-ALB' -SecurityGroup 'sg-07c3414abb8811cbd' -subnet 'subnet-c37a67a6', 'subnet-fc02eea0'
```

Output:

```
AvailabilityZones : {us-east-1b, us-east-1a}
CanonicalHostedZoneId : Z35SXD0TRQ7X7K
CreatedTime       : 12/28/19 2:58:03 PM
DNSName           : New-Test-ALB-1391502222.us-east-1.elb.amazonaws.com
```

```

IpAddressType      : ipv4
LoadBalancerArn    : arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/app/New-Test-ALB/dab2e4d90eb51493
LoadBalancerName   : New-Test-ALB
Scheme             : internet-facing
SecurityGroups     : {sg-07c3414abb8811cbd}
State              : Amazon.ElasticLoadBalancingV2.Model.LoadBalancerState
Type               : application
VpcId              : vpc-2cfd7000

```

- Per i dettagli sull'API, vedere [CreateLoadBalancer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ELB2Rule

Il seguente esempio di codice mostra come utilizzare. New-ELB2Rule

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova regola del listener con un'azione a risposta fissa basata sul valore dell'intestazione del cliente per il listener specificato.

```

$newRuleAction = [Amazon.ElasticLoadBalancingV2.Model.Action]@{
    "FixedResponseConfig" = @{
        "ContentType" = "text/plain"
        "MessageBody" = "Hello World"
        "StatusCode" = "200"
    }
    "Type" = [Amazon.ElasticLoadBalancingV2.ActionTypeEnum]::FixedResponse
}

$newRuleCondition = [Amazon.ElasticLoadBalancingV2.Model.RuleCondition]@{
    "httpHeaderConfig" = @{
        "HttpHeaderName" = "customHeader"
        "Values" = "header2","header1"
    }
    "Field" = "http-header"
}

New-ELB2Rule -ListenerArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:listener/app/testALB/3e2f03b558e19676/1c84f02aec143e80' -Action
$newRuleAction -Condition $newRuleCondition -Priority 10

```

Output:

```

Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}
IsDefault    : False
Priority      : 10
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/
testALB/3e2f03b558e19676/1c84f02aec143e80/f4f51dfaa033a8cc

```

- Per i dettagli sull'API, vedere [CreateRule](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ELB2TargetGroup

Il seguente esempio di codice mostra come utilizzare `New-ELB2TargetGroup`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo gruppo di destinazione con i parametri specificati.

```

New-ELB2TargetGroup -HealthCheckEnabled 1 -HealthCheckIntervalSeconds 30 -
HealthCheckPath '/index.html' -HealthCheckPort 80 -HealthCheckTimeoutSecond 5 -
HealthyThresholdCount 2 -UnhealthyThresholdCount 5 -Port 80 -Protocol 'HTTP' -
TargetType instance -VpcId 'vpc-2cfd7000' -Name 'NewTargetGroup'

```

Output:

```

HealthCheckEnabled      : True
HealthCheckIntervalSeconds : 30
HealthCheckPath         : /index.html
HealthCheckPort         : 80
HealthCheckProtocol     : HTTP
HealthCheckTimeoutSeconds : 5
HealthyThresholdCount   : 2
LoadBalancerArns       : {}
Matcher                 : Amazon.ElasticLoadBalancingV2.Model.Matcher
Port                    : 80
Protocol                : HTTP
TargetGroupArn          : arn:aws:elasticloadbalancing:us-
east-1:123456789012:targetgroup/NewTargetGroup/534e484681d801bf
TargetGroupName         : NewTargetGroup
TargetType              : instance

```

```
UnhealthyThresholdCount    : 5
VpcId                      : vpc-2cfd7000
```

- Per i dettagli sull'API, vedere [CreateTargetGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-ELB2Target

Il seguente esempio di codice mostra come utilizzare. Register-ELB2Target

Strumenti per PowerShell V4

Esempio 1: questo esempio registra l'istanza "i-0672a4c4cdeae3111" con il gruppo di destinazione specificato.

```
Register-ELB2Target -TargetGroupArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:targetgroup/test-tg/a4e04b3688be1970' -Target @{Port = 80; Id =
'i-0672a4c4cdeae3111'}
```

- Per i dettagli sull'API, vedere [RegisterTargets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELB2Listener

Il seguente esempio di codice mostra come utilizzare. Remove-ELB2Listener

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il listener specificato.

```
Remove-ELB2Listener -ListenerArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/66e10e3aaf5b6d9b'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2Listener (DeleteListener)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-
alb/3651b4394dd9a24f/66e10e3aaf5b6d9b".
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): y
```

Esempio 2: questo esempio rimuove il listener specificato dal bilanciatore del carico.

```
Remove-ELB2Listener -ListenerArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/3873f123b98f7618'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2Listener (DeleteListener)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-
alb/3651b4394dd9a24f/3873f123b98f7618".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): y
```

- Per i dettagli sull'API, vedere [DeleteListener](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELB2ListenerCertificate

Il seguente esempio di codice mostra come utilizzare Remove-ELB2ListenerCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il certificato specificato dal gruppo di destinazione specificato.

```
Remove-ELB2ListenerCertificate -Certificate @{CertificateArn = 'arn:aws:acm:us-
east-1:123456789012:certificate/19478bd5-491d-47d4-b1d7-5217feba1d97'} -ListenerArn
'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/test-
alb/3651b4394dd9a24f/3873f123b98f7618'
```

Output:

```
Confirm
Are you sure you want to perform this action?
```

```
Performing the operation "Remove-ELB2ListenerCertificate
(RemoveListenerCertificates)" on target "arn:aws:elasticloadbalancing:us-
east-1:123456789012:listener/app/test-alb/3651b4394dd9a24f/3873f123b98f7618".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): y
```

- Per i dettagli sull'API, vedere [RemoveListenerCertificates](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELB2LoadBalancer

Il seguente esempio di codice mostra come utilizzare. Remove-ELB2LoadBalancer

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il bilanciatore del carico specificato.

```
Remove-ELB2LoadBalancer -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2LoadBalancer (DeleteLoadBalancer)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-
alb/3651b4394dd9a24f".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): y
```

- Per i dettagli sull'API, vedere [DeleteLoadBalancer](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELB2Rule

Il seguente esempio di codice mostra come utilizzare. Remove-ELB2Rule

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la regola specificata dal listener


```
Remove-ELB2Rule -RuleArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-alb/3651b4394dd9a24f/3873f123b98f7618/4b25eb10a42e33ab'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2Rule (DeleteRule)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-alb/3651b4394dd9a24f/3873f123b98f7618/4b25eb10a42e33ab".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y
```

- Per i dettagli sull'API, vedere [DeleteRule](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELB2Tag

Il seguente esempio di codice mostra come utilizzare. Remove-ELB2Tag

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il tag per la chiave specificata.

```
Remove-ELB2Tag -ResourceArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -TagKey 'productVersion'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2Tag (RemoveTags)" on target
"arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y
```

- Per i dettagli sull'API, vedere [RemoveTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-ELB2TargetGroup

Il seguente esempio di codice mostra come utilizzare. Remove-ELB2TargetGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il gruppo di destinazione specificato.

```
Remove-ELB2TargetGroup -TargetGroupArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/testsssss/4e0b6076bc6483a7'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ELB2TargetGroup (DeleteTargetGroup)" on
target "arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/
testsssss/4e0b6076bc6483a7".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): y
```

- Per i dettagli sull'API, vedere [DeleteTargetGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ELB2IpAddressType

Il seguente esempio di codice mostra come utilizzare. Set-ELB2IpAddressType

Strumenti per PowerShell V4

Esempio 1: Questo esempio modifica il tipo di indirizzo IP del Load Balancer da " a IPv4 ".
DualStack

```
Set-ELB2IpAddressType -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -IpAddressType dualstack
```

Output:

```
Value
-----
```

```
dualstack
```

- Per i dettagli sull'API, vedere [SetIpAddressType](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ELB2RulePriority

Il seguente esempio di codice mostra come utilizzare Set-ELB2RulePriority

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica la priorità della regola del listener specificata.

```
Set-ELB2RulePriority -RulePriority -RulePriority @{Priority = 11; RuleArn =  
'arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/test-  
alb/3651b4394dd9a24f/a4eb199fa5046f80/dbf4c6dcef3ec6f8'}
```

Output:

```
Actions      : {Amazon.ElasticLoadBalancingV2.Model.Action}  
Conditions   : {Amazon.ElasticLoadBalancingV2.Model.RuleCondition}  
IsDefault    : False  
Priority      : 11  
RuleArn      : arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/app/  
test-alb/3651b4394dd9a24f/a4eb199fa5046f80/dbf4c6dcef3ec6f8
```

- Per i dettagli sull'API, vedere [SetRulePriorities](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ELB2SecurityGroup

Il seguente esempio di codice mostra come utilizzare Set-ELB2SecurityGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge il gruppo di sicurezza "sg-07c3414abb8811cbd" al bilanciatore del carico specificato.

```
Set-ELB2SecurityGroup -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-  
east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -SecurityGroup  
'sg-07c3414abb8811cbd'
```

Output:

```
sg-07c3414abb8811cbd
```

- Per i dettagli sull'API, vedere [SetSecurityGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-ELB2Subnet

Il seguente esempio di codice mostra come utilizzare. Set-ELB2Subnet

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica le sottoreti del bilanciatore del carico specificato.

```
Set-ELB2Subnet -LoadBalancerArn 'arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/app/test-alb/3651b4394dd9a24f' -Subnet 'subnet-7d8a0a51', 'subnet-c37a67a6'
```

Output:

LoadBalancerAddresses	SubnetId	ZoneName
-----	-----	-----
{}	subnet-7d8a0a51	us-east-1c
{}	subnet-c37a67a6	us-east-1b

- Per i dettagli sull'API, vedere [SetSubnets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-ELB2Target

Il seguente esempio di codice mostra come utilizzare. Unregister-ELB2Target

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la registrazione dell'istanza "i-0672a4c4cdeae3111" dal gruppo di destinazione specificato.

```
$targetDescription = New-Object Amazon.ElasticLoadBalancingV2.Model.TargetDescription
```

```
$targetDescription.Id = 'i-0672a4c4cdeae3111'  
Unregister-ELB2Target -Target $targetDescription -TargetGroupArn  
  'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/test-tg/  
a4e04b3688be1970'
```

- Per i dettagli sull'API, vedere [DeregisterTargets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

FSx Esempi di Amazon con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon FSx.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-FSXResourceTag

Il seguente esempio di codice mostra come usare Add-FSXResourceTag.

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge tag alla risorsa specificata.

```
Add-FSXResourceTag -ResourceARN "arn:aws:fsx:eu-west-1:123456789012:file-system/  
fs-01cd23bc4bdf5678a" -Tag @{Key="Users";Value="Test"} -PassThru
```

Output:

```
arn:aws:fsx:eu-west-1:123456789012:file-system/fs-01cd23bc4bdf5678a
```

- Per i dettagli sull'API, vedere [TagResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-FSXBackup

Il seguente esempio di codice mostra come utilizzare. Get-FSXBackup

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera i backup creati da ieri per il file system specificato con l'ID specificato.

```
Get-FSXBackup -Filter @{Name="file-system-id";Values=$fsx.FileSystemId} | Where-Object CreationTime -gt (Get-Date).AddDays(-1)
```

Output:

```
BackupId       : backup-01dac234e56782bcc
CreationTime   : 6/14/2019 3:35:14 AM
FailureDetails :
FileSystem     : Amazon.FSx.Model.FileSystem
KmsKeyId       : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-1b23-1bde-a1f1-e1234c5af123
Lifecycle     : AVAILABLE
ProgressPercent : 100
ResourceARN    : arn:aws:fsx:eu-west-1:123456789012:backup/backup-01dac234e56782bcc
Tags           : {}
Type           : AUTOMATIC
```

- Per i dettagli sull'API, vedere [DescribeBackups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-FSXFileSystem

Il seguente esempio di codice mostra come utilizzare. Get-FSXFileSystem

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce la descrizione dell'oggetto filesystemId specificato.

```
Get-FSXFileSystem -FileSystemId fs-01cd23bc4bdf5678a
```

Output:

```

CreationTime      : 1/17/2019 9:55:30 AM
DNSName           : fs-01cd23bc4bdf5678a.ktmsad.local
FailureDetails    :
FileSystemId       : fs-01cd23bc4bdf5678a
FileSystemType     : WINDOWS
KmsKeyId           : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-5b67-8bde-
a9f0-e1234c5af678
Lifecycle         : AVAILABLE
LustreConfiguration :
NetworkInterfaceIds : {eni-07d1dda1322b7e209}
OwnerId           : 123456789012
ResourceARN        : arn:aws:fsx:eu-west-1:123456789012:file-system/
fs-01cd23bc4bdf5678a
StorageCapacity    : 300
SubnetIds          : {subnet-7d123456}
Tags               : {FSx-Service}
VpcId              : vpc-41cf2b3f
WindowsConfiguration : Amazon.FSx.Model.WindowsFileSystemConfiguration

```

- Per i dettagli sull'API, vedere [DescribeFileSystems](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-FSXResourceTagList

Il seguente esempio di codice mostra come utilizzare. Get-FSXResourceTagList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i tag per l'ARN di risorsa specificato.

```
Get-FSXResourceTagList -ResourceARN $fsx.ResourceARN
```

Output:

Key	Value
---	----
FSx-Service	Windows
Users	Dev

- Per i dettagli sull'API, vedere [ListTagsForResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-FSXBackup

Il seguente esempio di codice mostra come utilizzare. New-FSXBackup

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un backup del file system specificato.

```
New-FSXBackup -FileSystemId fs-0b1fac2345623456ba
```

Output:

```
BackupId       : backup-0b1fac2345623456ba
CreationTime    : 6/14/2019 5:37:17 PM
FailureDetails  :
FileSystem      : Amazon.FSx.Model.FileSystem
KmsKeyId        : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-1b23-1bde-a1f3-
e1234c5af678
Lifecycle       : CREATING
ProgressPercent : 0
ResourceARN     : arn:aws:fsx:eu-west-1:123456789012:backup/
backup-0b1fac2345623456ba
Tags            : {}
Type            : USER_INITIATED
```

- Per i dettagli sull'API, vedere [CreateBackup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-FSXFileSystem

Il seguente esempio di codice mostra come utilizzare. New-FSXFileSystem

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo file system Windows da 300 GB, che consente l'accesso dalla sottorete specificata, che supporta un throughput fino a 8 megabyte al secondo. Il nuovo file system viene aggiunto automaticamente al servizio Microsoft Active Directory specificato.


```
New-FSXFileSystem -FileSystemType WINDOWS -StorageCapacity
300 -SubnetId subnet-1a2b3c4d5e6f -WindowsConfiguration
@{ThroughputCapacity=8;ActiveDirectoryId='d-1a2b3c4d'}
```

Output:

```
CreationTime      : 12/10/2018 6:06:59 PM
DNSName           : fs-abcdef01234567890.example.com
FailureDetails    :
FileSystemId       : fs-abcdef01234567890
FileSystemType     : WINDOWS
KmsKeyId          : arn:aws:kms:us-west-2:123456789012:key/a1234567-252c-45e9-
afaa-123456789abc
Lifecycle         : CREATING
LustreConfiguration :
NetworkInterfaceIds : {}
OwnerId           : 123456789012
ResourceARN       : arn:aws:fsx:us-west-2:123456789012:file-system/fs-
abcdef01234567890
StorageCapacity   : 300
SubnetIds         : {subnet-1a2b3c4d5e6f}
Tags              : {}
VpcId             : vpc-1a2b3c4d5e6f
WindowsConfiguration : Amazon.FSx.Model.WindowsFileSystemConfiguration
```

- Per i dettagli sull'API, vedere [CreateFileSystem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-FSXFileSystemFromBackup

Il seguente esempio di codice mostra come utilizzare `New-FSXFileSystemFromBackup`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo FSx file system Amazon da un backup esistente di Amazon FSx for Windows File Server.

```
New-FSXFileSystemFromBackup -BackupId $backupID -Tag @{Key="tag:Name";Value="from-
manual-backup"} -SubnetId $SubnetID -SecurityGroupId $SG_ID -WindowsConfiguration
@{ThroughputCapacity=8;ActiveDirectoryId=$DirectoryID}
```

Output:

```

CreationTime      : 8/8/2019 12:59:58 PM
DNSName           : fs-012ff34e56789120.ktmsad.local
FailureDetails    :
FileSystemId       : fs-012ff34e56789120
FileSystemType    : WINDOWS
KmsKeyId          : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-5b67-1bde-
a2f3-e4567c8a9321
Lifecycle         : CREATING
LustreConfiguration :
NetworkInterfaceIds : {}
OwnerId           : 933303704102
ResourceARN       : arn:aws:fsx:eu-west-1:123456789012:file-system/
fs-012ff34e56789120
StorageCapacity   : 300
SubnetIds         : {subnet-fa1ae23c}
Tags              : {tag:Name}
VpcId             : vpc-12cf3b4f
WindowsConfiguration : Amazon.FSx.Model.WindowsFileSystemConfiguration

```

- Per i dettagli sull'API, vedere [CreateFileSystemFromBackup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-FSXBackup

Il seguente esempio di codice mostra come utilizzare. Remove-FSXBackup

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove l'ID di backup specificato.

```
Remove-FSXBackup -BackupId $backupID
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-FSXBackup (DeleteBackup)" on target
"backup-0bbca1e2345678e12".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

```

BackupId	Lifecycle
-----	-----
backup-0bbca1e2345678e12	DELETED

- Per i dettagli sull'API, vedere [DeleteBackup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-FSXFileSystem

Il seguente esempio di codice mostra come utilizzare. Remove-FSXFileSystem

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove l'ID del file system FSX specificato.

```
Remove-FSXFileSystem -FileSystemId fs-012ff34e567890120
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-FSXFileSystem (DeleteFileSystem)" on target
"fs-012ff34e567890120".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

FileSystemId      Lifecycle WindowsResponse
-----
fs-012ff34e567890120 DELETING  Amazon.FSx.Model.DeleteFileSystemWindowsResponse
```

- Per i dettagli sull'API, vedere [DeleteFileSystem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-FSXResourceTag

Il seguente esempio di codice mostra come utilizzare. Remove-FSXResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il tag di risorsa per l'ARN di risorsa del file system FSX specificato.

```
Remove-FSXResourceTag -ResourceARN $FSX.ResourceARN -TagKey Users
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-FSXResourceTag (UntagResource)" on target
"arn:aws:fsx:eu-west-1:933303704102:file-system/fs-07cd45bc6bdf2674a".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y
```

- Per i dettagli sull'API, vedere [UntagResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-FSXFileSystem

Il seguente esempio di codice mostra come utilizzare. Update-FSXFileSystem

Strumenti per PowerShell V4

Esempio 1: Questo esempio aggiorna i giorni di conservazione automatica dei backup del file system FSX tramite. UpdateFileSystemWindowsConfiguration

```
$UpdateFSXWinConfig = [Amazon.FSx.Model.UpdateFileSystemWindowsConfiguration]::new()
$UpdateFSXWinConfig.AutomaticBackupRetentionDays = 35
Update-FSXFileSystem -FileSystemId $FSX.FileSystemId -WindowsConfiguration
$UpdateFSXWinConfig
```

Output:

```
CreationTime      : 1/17/2019 9:55:30 AM
DNSName           : fs-01cd23bc4bdf5678a.ktmsad.local
FailureDetails    :
FileSystemId      : fs-01cd23bc4bdf5678a
FileSystemType    : WINDOWS
```

```

KmsKeyId      : arn:aws:kms:eu-west-1:123456789012:key/f1af23c4-1b23-1bde-
a1f2-e1234c5af678
Lifecycle     : AVAILABLE
LustreConfiguration :
NetworkInterfaceIds : {eni-01cd23bc4bdf5678a}
OwnerId       : 933303704102
ResourceARN    : arn:aws:fsx:eu-west-1:933303704102:file-system/
fs-07cd45bc6bdf2674a
StorageCapacity : 300
SubnetIds      : {subnet-1d234567}
Tags           : {FSx-Service}
VpcId          : vpc-23cf4b5f
WindowsConfiguration : Amazon.FSx.Model.WindowsFileSystemConfiguration

```

- Per i dettagli sull'API, vedere [UpdateFileSystem](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon Glacier con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon Glacier.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-GLCJob

Il seguente esempio di codice mostra come usare. Get-GLCJob

Strumenti per PowerShell V4

Esempio 1: restituisce i dettagli del processo specificato. Quando il processo viene completato correttamente, è possibile utilizzare il cmdlet `GCJob Read-Output` per recuperare il contenuto del processo (un archivio o un elenco di inventario) nel file system locale.

```
Get-GLCJob -VaultName myvault -JobId "op1x...JSbthM"
```

Output:

```
Action                : ArchiveRetrieval
ArchiveId              : o909j...X-TpIhQJw
ArchiveSHA256TreeHash  : 79f3ea754c02f58...dc57bf4395b
ArchiveSizeInBytes     : 38034480
Completed              : False
CompletionDate         : 1/1/0001 12:00:00 AM
CreationDate           : 12/13/2018 11:00:14 AM
InventoryRetrievalParameters :
InventorySizeInBytes   : 0
JobDescription         :
JobId                  : op1x...JSbthM
JobOutputPath          :
OutputLocation         :
RetrievalByteRange     : 0-38034479
SelectParameters       :
SHA256TreeHash         : 79f3ea754c02f58...dc57bf4395b
SNSTopic               :
StatusCode             : InProgress
StatusMessage         :
Tier                   : Standard
VaultARN               : arn:aws:glacier:us-west-2:012345678912:vaults/test
```

- Per i dettagli sull'API, vedere [DescribeJob](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-GLCVault

Il seguente esempio di codice mostra come utilizzare `New-GLCVault`

Strumenti per PowerShell V4

Esempio 1: crea un nuovo vault per l'account dell'utente. Poiché non è stato fornito alcun valore al AccountId parametro -, i cmdlet utilizzano un valore predefinito di «-» che indica l'account corrente.

```
New-GLCVault -VaultName myvault
```

Output:

```
/01234567812/vaults/myvault
```

- Per i dettagli sull'API, vedere [CreateVault](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Read-GLCJobOutput

Il seguente esempio di codice mostra come utilizzare. Read-GLCJobOutput

Strumenti per PowerShell V4

Esempio 1: scarica il contenuto dell'archivio di cui è stato pianificato il recupero nel processo specificato e lo archivia in un file su disco. Il download convalida automaticamente il checksum, se disponibile. Se lo si desidera, è possibile restituire l'intera risposta, incluso il checksum, specificando **-Select '*'**.

```
Read-GLCJobOutput -VaultName myvault -JobId "HSWjArc...Zq2XLiW" -FilePath "c:\temp\blue.bin"
```

- Per i dettagli sull'API, vedere [GetJobOutput](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-GLCJob

Il seguente esempio di codice mostra come utilizzare. Start-GLCJob

Strumenti per PowerShell V4

Esempio 1: avvia un processo per recuperare un archivio dal vault specificato di proprietà dell'utente. Lo stato del processo può essere verificato utilizzando il GLCJob cmdlet Get-. Quando

il processo viene completato correttamente, è possibile utilizzare il cmdlet GCJob Read-Output per recuperare il contenuto dell'archivio nel file system locale.

```
Start-GLCJob -VaultName myvault -JobType "archive-retrieval" -JobDescription
"archive retrieval" -ArchiveId "o909j...TX-TpIhQJw"
```

Output:

JobId	JobOutputPath	Location
-----	-----	-----
op1x...JSbthM		/012345678912/vaults/test/jobs/op1xe...I4HqCHkSJSbthM

- Per i dettagli sull'API, vedere [InitiateJob](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-GLCArchive

Il seguente esempio di codice mostra come utilizzare. Write-GLCArchive

Strumenti per PowerShell V4

Esempio 1: carica un unico file nell'archivio specificato, restituendo l'ID dell'archivio e il checksum calcolato.

```
Write-GLCArchive -VaultName myvault -FilePath c:\temp\blue.bin
```

Output:

FilePath	ArchiveId	Checksum
-----	-----	-----
C:\temp\blue.bin	o909jUUs...TTX-TpIhQJw	79f3e...f4395b

Esempio 2: carica il contenuto di una gerarchia di cartelle nel vault specificato nell'account dell'utente. Per ogni file caricato, il cmdlet genera il nome di file, l'ID di archivio corrispondente e il checksum calcolato dell'archivio.

```
Write-GLCArchive -VaultName myvault -FolderPath . -Recurse
```

Output:

FilePath	ArchiveId	Checksum
-----	-----	-----
C:\temp\blue.bin	o909jUUs...TTX-TpIhQJw	79f3e...f4395b
C:\temp\green.bin	qXAf0dSG...czo729UHXrw	d50a1...9184b9
C:\temp\lum.bin	39aNifP3...q9nb8nZkFIg	28886...5c3e27
C:\temp\red.bin	vp7E6rU_...Ejk_HhjAxKA	e05f7...4e34f5
C:\temp\Folder1\file1.txt	_eRINlip...5Sxy7dD2BaA	d0d2a...c8a3ba
C:\temp\Folder2\file2.iso	-Ix3jlm...iXiDh-Xf0PA	7469e...3e86f1

- Per i dettagli sull'API, vedere [UploadArchive](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

AWS Glue esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con AWS Glue

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

New-GLUEJob

Il seguente esempio di codice mostra come utilizzare `New-GLUEJob`.

Strumenti per PowerShell V4

Esempio 1: Questo esempio crea un nuovo lavoro in AWS Glue. Il valore del nome del comando è sempre **glueetl**. AWS Glue supporta l'esecuzione di script di lavoro scritti in Python o Scala. In questo esempio, lo script di lavoro (MyTestGlueJob.py) è scritto in Python. I parametri Python vengono specificati nella **\$DefArgs** variabile e quindi passati al PowerShell

comando nel **DefaultArguments** parametro, che accetta una tabella hash. I parametri nella **\$JobParams** variabile provengono dall' CreateJob API, documentata nell'argomento Jobs (<https://docs.aws.amazon.com/glue/latest/dg/aws-glue-api-jobs-job.html>) del riferimento all'API AWS Glue.

```
$Command = New-Object Amazon.Glue.Model.JobCommand
$Command.Name = 'glueetl'
$Command.ScriptLocation = 's3://amzn-s3-demo-source-bucket/admin/MyTestGlueJob.py'
$Command

$Source = "source_test_table"
$Target = "target_test_table"
$Connections = $Source, $Target

$DefArgs = @{
    '--TempDir' = 's3://amzn-s3-demo-bucket/admin'
    '--job-bookmark-option' = 'job-bookmark-disable'
    '--job-language' = 'python'
}
$DefArgs

$ExecutionProp = New-Object Amazon.Glue.Model.ExecutionProperty
$ExecutionProp.MaxConcurrentRuns = 1
$ExecutionProp

$JobParams = @{
    "AllocatedCapacity"    = "5"
    "Command"              = $Command
    "Connections_Connection" = $Connections
    "DefaultArguments"     = $DefArgs
    "Description"          = "This is a test"
    "ExecutionProperty"    = $ExecutionProp
    "MaxRetries"           = "1"
    "Name"                 = "MyOregonTestGlueJob"
    "Role"                 = "Amazon-GlueServiceRoleForSSM"
    "Timeout"              = "20"
}

New-GlueJob @JobParams
```

- Per i dettagli sull'API, vedere [CreateJob](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

AWS Health esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. AWS Health

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-HLTHEvent

Il seguente esempio di codice mostra come utilizzare Get-HLTHEvent.

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce gli eventi da AWS Personal Health Dashboard. L'utente aggiunge il parametro -Region per visualizzare gli eventi disponibili per il servizio nella Regione Stati Uniti orientali (Virginia settentrionale), ma il parametro -Filter_Region filtra gli eventi registrati nelle Regioni Europa (Londra) e Stati Uniti occidentali (Oregon) (eu-west-2 e us-west-2). Il StartTime parametro -Filter_ filtra per un intervallo di volte in cui gli eventi possono iniziare, mentre il EndTime parametro -Filter_ filtra per un intervallo di volte in cui gli eventi possono terminare. Il risultato è un evento di manutenzione pianificato per RDS che inizia all'interno dell'intervallo - Filter_ specificato e termina nell'StartTime intervallo pianificato -Filter_. EndTime

```
Get-HLTHEvent -Region us-east-1 -Filter_Region "eu-west-2","us-west-2" -  
Filter_StartTime @{from="3/14/2019 6:30:00AM";to="3/15/2019 5:00:00PM"} -  
Filter_EndTime @{from="3/21/2019 7:00:00AM";to="3/21/2019 5:00:00PM"}
```

Output:

```
Arn                : arn:aws:health:us-west-2::event/RDS/
AWS_RDS_HARDWARE_MAINTENANCE_SCHEDULED/
AWS_RDS_HARDWARE_MAINTENANCE_SCHEDULED_USW2_20190314_20190321
AvailabilityZone   :
EndTime           : 3/21/2019 2:00:00 PM
EventTypeCategory  : scheduledChange
EventTypeCode      : AWS_RDS_HARDWARE_MAINTENANCE_SCHEDULED
LastUpdatedTime    : 2/28/2019 2:26:07 PM
Region            : us-west-2
Service           : RDS
StartTime         : 3/14/2019 2:00:00 PM
StatusCode        : open
```

- Per i dettagli sull'API, vedere [DescribeEvents](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi IAM che utilizzano Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con IAM.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-IAMClientIDToOpenIDConnectProvider

Il seguente esempio di codice mostra come utilizzare `Add-IAMClientIDToOpenIDConnectProvider`.

Strumenti per PowerShell V4

Esempio 1: questo comando aggiunge l'ID client (o il pubblico) **my-application-ID** al provider OIDC esistente denominato **server.example.com**.

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"  
-OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/  
server.example.com"
```

- Per i dettagli sull'API, vedere [AddClientIDToOpenIDConnectProvider](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-IAMRoleTag

Il seguente esempio di codice mostra come utilizzare. Add-IAMRoleTag

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un tag a un ruolo nel servizio di gestione identità

```
Add-IAMRoleTag -RoleName AdminRoleaccess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Per i dettagli sull'API, vedere [TagRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-IAMRoleToInstanceProfile

Il seguente esempio di codice mostra come utilizzare. Add-IAMRoleToInstanceProfile

Strumenti per PowerShell V4

Esempio 1: questo comando aggiunge il ruolo denominato **S3Access** a un profilo dell'istanza esistente denominato **webserver**. Per creare il profilo dell'istanza, utilizza il comando **New-IAMInstanceProfile**. Dopo aver creato il profilo dell'istanza e averlo associato a un ruolo utilizzando questo comando, potete collegarlo a un' EC2 istanza. A tale scopo, utilizza il cmdlet **New-EC2Instance** con il parametro **InstanceProfile_Arn** o con il parametro **InstanceProfile-Name** per avviare la nuova istanza.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName "webserver"
```

- Per i dettagli sull'API, vedere [AddRoleToInstanceProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-IAMUserTag

Il seguente esempio di codice mostra come utilizzare. Add-IAMUserTag

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un tag a un utente nel servizio di gestione identità

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Per i dettagli sull'API, vedere [TagUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Add-IAMUserToGroup

Il seguente esempio di codice mostra come utilizzare. Add-IAMUserToGroup

Strumenti per PowerShell V4

Esempio 1: questo comando aggiunge l'utente denominato **Bob** al gruppo denominato **Admins**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Per i dettagli sull'API, vedere [AddUserToGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Disable-IAMMFADevice

Il seguente esempio di codice mostra come utilizzare. Disable-IAMMFADevice

Strumenti per PowerShell V4

Esempio 1: questo comando disabilita il dispositivo hardware MFA associato all'utente **Bob** con il numero di serie **123456789012**.

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Esempio 2: questo comando disabilita il dispositivo MFA virtuale associato all'utente **David** con ARN **arn:aws:iam::210987654321:mfa/David**. Tieni presente che il dispositivo MFA virtuale non viene eliminato dall'account. Il dispositivo virtuale è ancora presente e appare nell'output del comando **Get-IAMVirtualMFADevice**. Prima di poter creare un nuovo dispositivo MFA virtuale per lo stesso utente, è necessario eliminare quello precedente utilizzando il comando **Remove-IAMVirtualMFADevice**.

```
Disable-IAMMFADevice -UserName "David" -SerialNumber "arn:aws:iam::210987654321:mfa/David"
```

- Per i dettagli sull'API, vedere [DeactivateMfaDevice](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-IAMPassword

Il seguente esempio di codice mostra come utilizzare. Edit-IAMPassword

Strumenti per PowerShell V4

Esempio 1: questo comando modifica la password dell'utente che esegue il comando. Questo comando può essere chiamato solo dagli utenti IAM. Se questo comando viene chiamato quando si accede con le credenziali AWS dell'account (root), restituisce un errore. **InvalidUserType**

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Per i dettagli sull'API, vedere [ChangePassword](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Enable-IAMMFADevice

Il seguente esempio di codice mostra come utilizzare. Enable-IAMMFADevice

Strumenti per PowerShell V4

Esempio 1: questo comando abilita il dispositivo hardware MFA con il numero di serie **987654321098** e associa il dispositivo all'utente **Bob**. Include i primi due codici in sequenza provenienti dal dispositivo.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Esempio 2: questo esempio crea e abilita un dispositivo MFA virtuale. Il primo comando crea il dispositivo virtuale e restituisce la rappresentazione dell'oggetto del dispositivo nella variabile **\$MFADevice**. È possibile utilizzare le proprietà **.Base32StringSeed** o **QRCodePng** per configurare l'applicazione software dell'utente. Il comando finale assegna il dispositivo all'utente **David**, identificandolo in base al numero di serie. Il comando sincronizza inoltre il dispositivo con AWS l'inclusione dei primi due codici in sequenza dal dispositivo MFA virtuale.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber $MFADevice.SerialNumber
-AuthenticationCode1 "24681357" -AuthenticationCode2
"13572468"
```

- Per i dettagli sull'API, vedere [EnableMfaDevice](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMAccessKey

Il seguente esempio di codice mostra come utilizzare. Get-IAMAccessKey

Strumenti per PowerShell V4

Esempio 1: questo comando elenca le chiavi di accesso per l'utente IAM denominato **Bob**. Tenere presente che non è possibile elencare le chiavi di accesso segrete per gli utenti IAM. Se le chiavi di accesso segrete vengono perse, sarà necessario creare nuove chiavi di accesso con il cmdlet **New-IAMAccessKey**.

```
Get-IAMAccessKey -UserName "Bob"
```

Output:

AccessKeyId	CreateDate	Status	UserName
-----	-----	-----	-----
AKIAIOSFODNN7EXAMPLE	12/3/2014 10:53:41 AM	Active	Bob

AKIAI44QH8DHBEXAMPLE	6/6/2013 8:42:26 PM	Inactive	Bob

- Per i dettagli sull'API, vedere [ListAccessKeys](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMAccessKeyLastUsed

Il seguente esempio di codice mostra come utilizzare. Get-IAMAccessKeyLastUsed

Strumenti per PowerShell V4

Esempio 1: restituisce il nome utente proprietario e le informazioni sull'ultimo utilizzo della chiave di accesso fornita.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Per i dettagli sull'API, vedere [GetAccessKeyLastUsed](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMAccountAlias

Il seguente esempio di codice mostra come utilizzare. Get-IAMAccountAlias

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce l'alias dell'account per l' Account AWS.

```
Get-IAMAccountAlias
```

Output:

```
ExampleCo
```

- Per i dettagli sull'API, vedere [ListAccountAliases](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMAccountAuthorizationDetail

Il seguente esempio di codice mostra come utilizzare. Get-IAMAccountAuthorizationDetail

Strumenti per PowerShell V4

Esempio 1: Questo esempio ottiene i dettagli di autorizzazione sulle identità nell' AWS account e visualizza l'elenco degli elementi dell'oggetto restituito, inclusi utenti, gruppi e ruoli. Ad esempio, la proprietà **UserDetailList** visualizza i dettagli degli utenti. Informazioni simili sono disponibili nelle proprietà **RoleDetailList** e **GroupDetailList**.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Output:

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker         :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/Administrator
CreateDate   : 10/16/2014 9:03:09 AM
GroupList    : {Administrators}
Path         : /
UserId       : AIDACKCEVSQ6CEXAMPLE1
UserName     : Administrator
UserPolicyList : {}

Arn          : arn:aws:iam::123456789012:user/Bob
CreateDate   : 4/6/2015 12:54:42 PM
GroupList    : {Developers}
Path         : /
UserId       : AIDACKCEVSQ6CEXAMPLE2
UserName     : bab
UserPolicyList : {}

Arn          : arn:aws:iam::123456789012:user/BackupToS3
CreateDate   : 1/27/2015 10:15:08 AM
GroupList    : {Backup}
```

```
Path           : /
UserId         : AIDACKCEVSQ6CEXAMPLE3
UserName       : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Per i dettagli sull'API, vedere [GetAccountAuthorizationDetails](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMAccountPasswordPolicy

Il seguente esempio di codice mostra come utilizzare. `Get-IAMAccountPasswordPolicy`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce dettagli della policy delle password per l'account corrente. Se non è definita alcuna policy delle password per l'account, il comando restituisce un errore **NoSuchEntity**.

```
Get-IAMAccountPasswordPolicy
```

Output:

```
AllowUsersToChangePassword : True
ExpirePasswords             : True
HardExpiry                  : False
MaxPasswordAge              : 90
MinimumPasswordLength       : 8
PasswordReusePrevention     : 20
RequireLowercaseCharacters  : True
RequireNumbers               : True
RequireSymbols               : False
RequireUppercaseCharacters  : True
```

- Per i dettagli sull'API, vedere [GetAccountPasswordPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMAccountSummary

Il seguente esempio di codice mostra come utilizzare. `Get-IAMAccountSummary`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce informazioni sull'utilizzo corrente e sulle quote correnti delle entità IAM nell' Account AWS.

```
Get-IAMAccountSummary
```

Output:

Key	Value
Users	7
GroupPolicySizeQuota	5120
PolicyVersionsInUseQuota	10000
ServerCertificatesQuota	20
AccountSigningCertificatesPresent	0
AccountAccessKeysPresent	0
Groups	3
UsersQuota	5000
RolePolicySizeQuota	10240
UserPolicySizeQuota	2048
GroupsPerUserQuota	10
AssumeRolePolicySizeQuota	2048
AttachedPoliciesPerGroupQuota	2
Roles	9
VersionsPerPolicyQuota	5
GroupsQuota	100
PolicySizeQuota	5120
Policies	5
RolesQuota	250
ServerCertificates	0
AttachedPoliciesPerRoleQuota	2
MFADevicesInUse	2
PoliciesQuota	1000
AccountMFAEnabled	1
Providers	2
InstanceProfilesQuota	100
MFADevices	4
AccessKeysPerUserQuota	2
AttachedPoliciesPerUserQuota	2
SigningCertificatesPerUserQuota	2
PolicyVersionsInUse	4
InstanceProfiles	1

...

- Per i dettagli sull'API, vedere [GetAccountSummary](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMAttachedGroupPolicyList

Il seguente esempio di codice mostra come utilizzare. Get-IAMAttachedGroupPolicyList

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i nomi e ARNs le politiche gestite allegate al gruppo IAM denominato **Admins** nell' AWS account. Per visualizzare l'elenco delle policy in linea incorporate nel gruppo, usa il comando **Get-IAMGroupPolicyList**.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Output:

PolicyArn	PolicyName
-----	-----
arn:aws:iam::aws:policy/SecurityAudit	SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess	AdministratorAccess

- Per i dettagli sull'API, vedere [ListAttachedGroupPolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMAttachedRolePolicyList

Il seguente esempio di codice mostra come utilizzare. Get-IAMAttachedRolePolicyList

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i nomi e ARNs le politiche gestite allegate al ruolo IAM denominato **SecurityAuditRole** nell' AWS account. Per visualizzare l'elenco delle policy in linea incorporate nel ruolo, usa il comando **Get-IAMRolePolicyList**.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Output:

PolicyArn	PolicyName
-----	-----
arn:aws:iam::aws:policy/SecurityAudit	SecurityAudit

- Per i dettagli sull'API, vedere [ListAttachedRolePolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMAttachedUserPolicyList

Il seguente esempio di codice mostra come utilizzare. Get-IAMAttachedUserPolicyList

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i nomi e ARNs le politiche gestite per l'utente IAM indicato **Bob** nell' AWS account. Per visualizzare l'elenco delle policy in linea incorporate nell'utente IAM, usa il comando **Get-IAMUserPolicyList**.

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Output:

PolicyArn	PolicyName
-----	-----
arn:aws:iam::aws:policy/TesterPolicy	TesterPolicy

- Per i dettagli sull'API, vedere [ListAttachedUserPolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMContextKeysForCustomPolicy

Il seguente esempio di codice mostra come utilizzare. Get-IAMContextKeysForCustomPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera tutte le chiavi di contesto presenti nella policy JSON fornita. Per fornire più policy puoi specificare un elenco di valori separati da virgole.

```
$policy1 = '{"Version":"2012-10-17", "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
```

```
west-2:123456789012:table/", "Condition": {"DateGreaterThan":
{"aws:CurrentTime": "2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version": "2012-10-17", "Statement":
{"Effect": "Allow", "Action": "dynamodb:*", "Resource": "arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Per i dettagli sull'API, vedere [GetContextKeysForCustomPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMContextKeysForPrincipalPolicy

Il seguente esempio di codice mostra come utilizzare. Get-IAMContextKeysForPrincipalPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera tutte le chiavi di contesto presenti nella policy json fornita e le policy collegate all'entità IAM (user/role ecc.). Per: PolicyInputList puoi fornire più elenchi di valori come valori separati da virgole.

```
$policy1 = '{"Version": "2012-10-17", "Statement":
{"Effect": "Allow", "Action": "dynamodb:*", "Resource": "arn:aws:dynamodb:us-
west-2:123456789012:table/", "Condition": {"DateGreaterThan":
{"aws:CurrentTime": "2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version": "2012-10-17", "Statement":
{"Effect": "Allow", "Action": "dynamodb:*", "Resource": "arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Per i dettagli sull'API, vedere [GetContextKeysForPrincipalPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMCredentialReport

Il seguente esempio di codice mostra come utilizzare. Get-IAMCredentialReport

Strumenti per PowerShell V4

Esempio 1: questo esempio apre il report restituito e lo invia alla pipeline come array di righe di testo. La prima riga è l'intestazione con i nomi delle colonne separati da virgole. Ogni riga successiva è la riga di dettaglio per un utente, con ogni campo separato da virgole. Prima di poter visualizzare il report, è necessario generarlo con il cmdlet **Request-IAMCredentialReport**. Per recuperare il report come singola stringa, utilizzare **-Raw** invece di **-AsTextArray**. L'alias **-SplitLines** è accettato anche per lo switch **-AsTextArray**. Per l'elenco completo delle colonne nell'output, consulta la documentazione di riferimento delle API del servizio. Tieni presente che se non utilizzi **-AsTextArray** o **-SplitLines**, devi estrarre il testo dalla proprietà **.Content** utilizzando la classe **StreamReader** .NET.

Request-IAMCredentialReport

Output:

Description	State
-----	-----
No report exists. Starting a new report generation task	STARTED

```
Get-IAMCredentialReport -AsTextArray
```

Output:

```

user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,password_expiration_time,mfa_enabled,root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015-04-20T16:31:25+00:00,A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:00+00:00,A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/A,A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/A,N/A,A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A

```

- Per i dettagli sull'API, vedere [GetCredentialReport](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMEntitiesForPolicy

Il seguente esempio di codice mostra come utilizzare. Get-IAMEntitiesForPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce un elenco di gruppi, ruoli e utenti IAM a cui è collegata la policy **arn:aws:iam::123456789012:policy/TestPolicy**.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Output:

```
IsTruncated   : False
Marker        :
PolicyGroups  : {}
PolicyRoles   : {testRole}
PolicyUsers   : {Bob, Theresa}
```

- Per i dettagli sull'API, vedere [ListEntitiesForPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMGroup

Il seguente esempio di codice mostra come utilizzare. Get-IAMGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce dettagli del gruppo IAM **Testers**, inclusa una raccolta di tutti gli utenti IAM che appartengono al gruppo.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Output:

Group	IsTruncated	Marker
Users		
----	-----	-----

```
Amazon.IdentityManagement.Model.Group      False  
    {Theresa, David}
```

```
$results.Group
```

Output:

```
Arn          : arn:aws:iam::123456789012:group/Testers  
CreateDate   : 12/10/2014 3:39:11 PM  
GroupId      : 3RHNZZGQJ7QHMAEXAMPLE1  
GroupName    : Testers  
Path         : /
```

```
$results.Users
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/Theresa  
CreateDate    : 12/10/2014 3:39:27 PM  
PasswordLastUsed : 1/1/0001 12:00:00 AM  
Path          : /  
UserId        : 40SVDDJJTF4XEEXAMPLE2  
UserName      : Theresa  
  
Arn          : arn:aws:iam::123456789012:user/David  
CreateDate    : 12/10/2014 3:39:27 PM  
PasswordLastUsed : 3/19/2015 8:44:04 AM  
Path          : /  
UserId        : Y4FKWQCXTA52QEXAMPLE3  
UserName      : David
```

- Per i dettagli sull'API, vedere [GetGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMGroupForUser

Il seguente esempio di codice mostra come utilizzare `Get-IAMGroupForUser`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce l'elenco dei gruppi IAM a cui appartiene l'utente IAM **David**.

```
Get-IAMGroupForUser -UserName David
```

Output:

```
Arn      : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId   : 6WCH4TRY3KIHEXAMPLE1
GroupName : Administrators
Path      : /

Arn      : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId   : RHNZZGQJ7QHMAEXAMPLE2
GroupName : Testers
Path      : /

Arn      : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId   : ZU2E0WMK6WBZ0EXAMPLE3
GroupName : Developers
Path      : /
```

- Per i dettagli sull'API, vedere [ListGroupsForUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMGroupList

Il seguente esempio di codice mostra come utilizzare. Get-IAMGroupList

Strumenti per PowerShell V4

Esempio 1: Questo esempio restituisce una raccolta di tutti i gruppi IAM definiti nella versione corrente Account AWS.

```
Get-IAMGroupList
```

Output:

```

Arn      : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId   : 6WCH4TRY3KIHIEEXAMPLE1
GroupName : Administrators
Path      : /

Arn      : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId   : ZU2E0WMK6WBZ0EXAMPLE2
GroupName : Developers
Path      : /

Arn      : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId   : RHNZZGQJ7QHMAEXAMPLE3
GroupName : Testers
Path      : /

```

- Per i dettagli sull'API, vedere [ListGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMGroupPolicy

Il seguente esempio di codice mostra come utilizzare `Get-IAMGroupPolicy`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce dettagli della policy in linea incorporata denominata **PowerUserAccess-Testers** per il gruppo **Testers**. La proprietà **PolicyDocument** è codificata tramite URL. In questo esempio viene decodificato con il metodo **UrlDecode** .NET.

```

$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-Testers
$results

```

Output:

GroupName	PolicyDocument	PolicyName
-----	-----	-----
Testers	%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...	PowerUserAccess-Testers

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
  ]
}
```

- Per i dettagli sull'API, vedere [GetGroupPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMGroupPolicyList

Il seguente esempio di codice mostra come utilizzare. Get-IAMGroupPolicyList

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce l'elenco dei nomi delle policy in linea incorporate nel gruppo **Testers**. Per ottenere le policy gestite collegate al gruppo, utilizzare il comando **Get-IAMAttachedGroupPolicyList**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Output:

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Per i dettagli sull'API, vedere [ListGroupPolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMInstanceProfile

Il seguente esempio di codice mostra come utilizzare. Get-IAMInstanceProfile

Strumenti per PowerShell V4

Esempio 1: Questo esempio restituisce i dettagli del profilo di istanza denominato **ec2instancerole** definito nell' AWS account corrente.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Output:

```
Arn                : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate         : 2/17/2015 2:49:04 PM
InstanceProfileId  : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path              : /
Roles             : {ec2instancerole}
```

- Per i dettagli sull'API, vedere [GetInstanceProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMInstanceProfileForRole

Il seguente esempio di codice mostra come utilizzare. Get-IAMInstanceProfileForRole

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce i dettagli del profilo dell'istanza associato al ruolo **ec2instancerole**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Output:

```
Arn                : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
CreateDate         : 2/17/2015 2:49:04 PM
InstanceProfileId  : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
```

```
Path          : /  
Roles         : {ec2instancerole}
```

- Per i dettagli sull'API, vedere [ListInstanceProfilesForRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMInstanceProfileList

Il seguente esempio di codice mostra come utilizzare. Get-IAMInstanceProfileList

Strumenti per PowerShell V4

Esempio 1: Questo esempio restituisce una raccolta dei profili di istanza definiti nella versione corrente Account AWS.

```
Get-IAMInstanceProfileList
```

Output:

```
Arn           : arn:aws:iam::123456789012:instance-profile/ec2instancerole  
CreateDate    : 2/17/2015 2:49:04 PM  
InstanceId    : HH36PTZQJUR32EXAMPLE1  
InstanceProfileName : ec2instancerole  
Path          : /  
Roles         : {ec2instancerole}
```

- Per i dettagli sull'API, vedere [ListInstanceProfiles](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMLoginProfile

Il seguente esempio di codice mostra come utilizzare. Get-IAMLoginProfile

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce la data di creazione della password e se è necessaria una reimpostazione della password per l'utente IAM **David**.

```
Get-IAMLoginProfile -UserName David
```

Output:

CreateDate	PasswordResetRequired	UserName
-----	-----	-----
12/10/2014 3:39:44 PM	False	David

- Per i dettagli sull'API, vedere [GetLoginProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMMFADevice

Il seguente esempio di codice mostra come utilizzare. Get-IAMMFADevice

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce i dettagli del dispositivo MFA assegnato all'utente IAM **David**. In questo esempio si può affermare che si tratta di un dispositivo virtuale perché **SerialNumber** è un ARN e non il numero di serie effettivo di un dispositivo fisico.

```
Get-IAMMFADevice -UserName David
```

Output:

EnableDate	SerialNumber	UserName
-----	-----	-----
4/8/2015 9:41:10 AM	arn:aws:iam::123456789012:mfa/David	David

- Per i dettagli sull'API, vedere [ListMfaDevices](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMOpenIDConnectProvider

Il seguente esempio di codice mostra come utilizzare. Get-IAMOpenIDConnectProvider

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce i dettagli del provider OpenID Connect il cui ARN è **arn:aws:iam::123456789012:oidc-provider/accounts.google.com**. La **ClientIDList** proprietà è una raccolta che contiene tutto il Client IDs definito per questo provider.


```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Output:

ClientIDList	CreateDate	ThumbprintList
Url		
-----	-----	-----

{MyOIDCApp}	2/3/2015 3:00:30 PM	
{12345abcdefghijk67890lmnopqrst98765uvwxyz}		oidc.example.com

- Per i dettagli sull'API, vedere [GetOpenIdConnectProvider](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMOpenIDConnectProviderList

Il seguente esempio di codice mostra come utilizzare `Get-IAMOpenIDConnectProviderList`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce un elenco di ARN di tutti i provider OpenID Connect definiti nell' Account AWS corrente.

Get-IAMOpenIDConnectProviderList

Output:

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Per i dettagli sull'API, vedere [ListOpenIdConnectProviders](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMPolicy

Il seguente esempio di codice mostra come utilizzare `Get-IAMPolicy`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce i dettagli sulla policy gestita il cui ARN è **arn:aws:iam::123456789012:policy/MySamplePolicy**.

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
Arn           : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount : 0
CreateDate    : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description   :
IsAttachable  : True
Path         : /
PolicyId      : Z27SI6FQMGNQ2EXAMPLE1
PolicyName    : MySamplePolicy
UpdateDate    : 2/6/2015 10:40:08 AM
```

- Per i dettagli sull'API, vedere [GetPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMPolicyList

Il seguente esempio di codice mostra come utilizzare. Get-IAMPolicyList

Strumenti per PowerShell V4

Esempio 1: Questo esempio restituisce una raccolta delle prime tre politiche gestite disponibili nell' AWS account corrente. Poiché non **-scope** è specificato, per **all** impostazione predefinita include sia le politiche AWS gestite che quelle gestite dai clienti.

```
Get-IAMPolicyList -MaxItem 3
```

Output:

```
Arn           : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount : 0
```

```

CreateDate      : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description     :
IsAttachable    : True
Path           : /
PolicyId        : Z27SI6FQMGNQ2EXAMPLE1
PolicyName      : AWSDirectConnectReadOnlyAccess
UpdateDate      : 2/6/2015 10:40:08 AM

Arn             : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount : 0
CreateDate      : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description     :
IsAttachable    : True
Path           : /
PolicyId        : NJKMU274MET4EEXAMPLE2
PolicyName      : AmazonGlacierReadOnlyAccess
UpdateDate      : 2/6/2015 10:40:27 AM

Arn             : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount : 0
CreateDate      : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description     :
IsAttachable    : True
Path           : /
PolicyId        : 5ULJS02FYVPYGEXAMPLE3
PolicyName      : AWSMarketplaceFullAccess
UpdateDate      : 2/11/2015 9:21:45 AM

```

Esempio 2: Questo esempio restituisce una raccolta delle prime due politiche gestite dai clienti disponibili nell'account corrente AWS . Utilizza **-Scope local** per limitare l'output alle sole policy gestite dal cliente.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Output:

```

Arn             : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount : 0
CreateDate      : 2/12/2015 9:39:09 AM

```

```

DefaultVersionId : v2
Description       :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VA0UCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description       :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2S0EXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM

```

- Per i dettagli sull'API, vedere [ListPolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMPolicyVersion

Il seguente esempio di codice mostra come utilizzare. Get-IAMPolicyVersion

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il documento della policy per la versione **v2** della policy il cui ARN è **arn:aws:iam::123456789012:policy/MyManagedPolicy**. Il documento di policy contenuto nella proprietà **Document** è codificato nell'URL e in questo esempio viene decodificato con il metodo .NET **UrlDecode**.

```

$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results

```

Output:

```

CreateDate      Document
IsDefaultVersion VersionId

```

```

-----
-----
2/12/2015 9:39:53 AM   %7B%0A%20%2022Version%22%3A%20%222012-10...   True
v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version": "2012-10-17",
  "Statement":
  {
    "Effect": "Allow",
    "Action": [
      "ec2:DescribeInstances"
    ],
    "Resource": [
      "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
    ]
  }
}

```

- Per i dettagli sull'API, vedere [GetPolicyVersion](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMPolicyVersionList

Il seguente esempio di codice mostra come utilizzare `Get-IAMPolicyVersionList`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce l'elenco delle versioni disponibili della policy il cui ARN è **arn:aws:iam::123456789012:policy/MyManagedPolicy**. Per ottenere il documento relativo alla policy per una versione specifica, utilizza il comando **Get-IAMPolicyVersion** e specifica l'**VersionId** della versione desiderata.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/MyManagedPolicy
```

Output:

CreateDate VersionId	Document	IsDefaultVersion

```

-----
-----
-----
2/12/2015 9:39:53 AM                True
v2
2/12/2015 9:39:09 AM                False
v1

```

- Per i dettagli sull'API, vedere [ListPolicyVersions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMRole

Il seguente esempio di codice mostra come utilizzare. Get-IAMRole

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce i dettagli di **lambda_exec_role**. Include il documento della policy di attendibilità, che specifica chi può assumere questo ruolo. Il documento di policy è codificato tramite URL e può essere decodificato utilizzando il metodo .NET **UrlDecode**. In questo esempio, la policy originale aveva rimosso tutti gli spazi bianchi prima di essere caricata nella policy. Per visualizzare i documenti relativi alle policy di autorizzazioni che determinano cosa può fare qualcuno che assume il ruolo, utilizza **Get-IAMRolePolicy** per le policy in linea e **Get-IAMPolicyVersion** per le policy gestite allegate.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Output:

```

Arn                : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%22012-10-17%22%2C%22Statement%22%3A%5B%7B%22Sid%22%3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal%22%3A%7B%22Service%22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A%22sts%3AAssumeRole%22%7D%5D%7D
CreateDate          : 4/2/2015 9:16:11 AM
Path                : /
RoleId              : 2YBIKAIBHNKB4EXAMPLE1

```

```
RoleName          : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Output:

```
{"Version":"2012-10-17",      "Statement":[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Per i dettagli sull'API, vedere [GetRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMRoleList

Il seguente esempio di codice mostra come utilizzare. Get-IAMRoleList

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera un elenco di tutti i ruoli IAM nell' Account AWS.

```
Get-IAMRoleList
```

- Per i dettagli sull'API, vedere [ListRoles](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMRolePolicy

Il seguente esempio di codice mostra come utilizzare. Get-IAMRolePolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il documento sulla policy di autorizzazioni per la policy **oneClick_lambda_exec_role_policy** denominata incorporata nel ruolo IAM **lamda_exec_role**. Il documento di policy risultante è codificato nell'URL. In questo esempio viene decodificato con il metodo **UrlDecode** .NET.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
oneClick_lambda_exec_role_policy
```

```
$results
```

Output:

PolicyDocument	PolicyName
<pre> UserName ----- ----- %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%... oneClick_lambda_exec_role_policy lambda_exec_role </pre>	

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Output:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:us-east-1:555555555555:log-group:/aws/lambda/aws-
example-function:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    }
  ]
}
```

- Per i dettagli sull'API, vedere [GetRolePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMRolePolicyList

Il seguente esempio di codice mostra come utilizzare. `Get-IAMRolePolicyList`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce l'elenco dei nomi delle policy in linea incorporate nel ruolo IAM **lambda_exec_role**. Per visualizzare i dettagli di una policy in linea, usa il comando **Get-IAMRolePolicy**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Output:

```
oneClick_lambda_exec_role_policy
```

- Per i dettagli sull'API, vedere [ListRolePolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMRoleTagList

Il seguente esempio di codice mostra come utilizzare. `Get-IAMRoleTagList`

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera il tag associato al ruolo.

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Per i dettagli sull'API, vedere [ListRoleTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMSAMLProvider

Il seguente esempio di codice mostra come utilizzare. `Get-IAMSAMLProvider`

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera i dettagli del provider SAML 2.0 il cui ARN è `arn:aws:iam::123456789012:saml-provider/SAMLADFS`. La risposta include il documento di

metadati che hai ricevuto dal provider di identità per creare l'entità del provider AWS SAML, nonché le date di creazione e scadenza.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Output:

CreateDate ValidUntil	SAMLMetadataDocument
----- -----	-----
12/23/2014 12:16:55 PM 12/23/2114 12:16:54 PM	<EntityDescriptor ID="_12345678-1234-5678-9012-example1...

- Per i dettagli sull'API, vedere [GetSamlProvider](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMSAMLProviderList

Il seguente esempio di codice mostra come utilizzare. Get-IAMSAMLProviderList

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera l'elenco dei provider SAML 2.0 creati nell' Account AWS corrente. Restituisce l'ARN, la data di creazione e la data di scadenza per ogni provider SAML.

```
Get-IAMSAMLProviderList
```

Output:

Arn ValidUntil	CreateDate
--- -----	-----
arn:aws:iam::123456789012:saml-provider/SAMLADFS 12/23/2114 12:16:54 PM	12/23/2014 12:16:55 PM

- Per i dettagli sull'API, vedere [List SAMLProviders](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMServerCertificate

Il seguente esempio di codice mostra come utilizzare `Get-IAMServerCertificate`

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera i dettagli del certificato server denominato **MyServerCertificate**. È possibile trovare i dettagli del certificato nelle proprietà **CertificateBody** e **ServerCertificateMetadata**.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Output:

```
CertificateBody      : -----BEGIN CERTIFICATE-----

MIICiTCCAfICCD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMakGA1UEBhMC

VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6

b24xFDASBgNVBA5TC0lBTsBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAd

BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

MTIwNDI0MjA0NTIxWjCBiDELMakGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD

VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC0lBTsBDb25z

b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

YXpvbi5jb20wZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMak0dn

+a4GmWIWJ

21uUSfwfEvySWtC2XADZ4nB+BLygVIk60CpiwsZ3G93vUEI03IyNoH/

f0wYK8m9T

rDHudUZg3qX4waLG5M43q7Wgc/

MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE

Ibb30hjZnzcVQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4

nUhVVxYUntneD9+h8Mg9q6q

+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb

NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
```

```

-----END CERTIFICATE-----
CertificateChain      :
ServerCertificateMetadata :
  Amazon.IdentityManagement.Model.ServerCertificateMetadata

```

```
$result.ServerCertificateMetadata
```

Output:

```

Arn                : arn:aws:iam::123456789012:server-certificate/0rg1/0rg2/
MyServerCertificate
Expiration          : 1/14/2018 9:52:36 AM
Path                : /0rg1/0rg2/
ServerCertificateId : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate          : 4/21/2015 11:14:16 AM

```

- Per i dettagli sull'API, vedere [GetServerCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMServerCertificateList

Il seguente esempio di codice mostra come utilizzare `Get-IAMServerCertificateList`

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera l'elenco dei certificati del server che sono stati caricati nell'Account AWS corrente.

```
Get-IAMServerCertificateList
```

Output:

```

Arn                : arn:aws:iam::123456789012:server-certificate/0rg1/0rg2/
MyServerCertificate
Expiration          : 1/14/2018 9:52:36 AM
Path                : /0rg1/0rg2/
ServerCertificateId : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate          : 4/21/2015 11:14:16 AM

```

- Per i dettagli sull'API, vedere [ListServerCertificates](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get- IAMServiceLastAccessedDetail

Il seguente esempio di codice mostra come utilizzare. Get- IAMServiceLastAccessedDetail

Strumenti per PowerShell V4

Esempio 1: questo esempio fornisce i dettagli dell'ultimo accesso al servizio da parte dell'entità IAM (utente, gruppo, ruolo o policy) associata alla chiamata Request.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Output:

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Per i dettagli sull'API, vedere [GetServiceLastAccessedDetails](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get- IAMServiceLastAccessedDetailWithEntity

Il seguente esempio di codice mostra come utilizzare. Get- IAMServiceLastAccessedDetailWithEntity

Strumenti per PowerShell V4

Esempio 1: questo esempio fornisce il timestamp dell'ultimo accesso per il servizio contenuto nella richiesta della rispettiva entità IAM.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f -ServiceNamespace ec2  
$results
```

Output:

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error              :
IsTruncated        : False
JobCompletionDate  : 12/29/19 11:19:31 AM
JobCreationDate    : 12/29/19 11:19:31 AM
JobStatus          : COMPLETED
Marker             :
```

```
$results.EntityDetailsList
```

Output:

```
EntityInfo                                LastAuthenticated
-----                                -
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Output:

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- Per i dettagli sull'API, vedere [GetServiceLastAccessedDetailsWithEntities](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMSigningCertificate

Il seguente esempio di codice mostra come utilizzare `Get-IAMSigningCertificate`

Strumenti per PowerShell V4

Esempio 1: Questo esempio recupera i dettagli del certificato di firma associato all'utente denominato **Bob**.

```
Get-IAMSigningCertificate -UserName Bob
```

Output:

```

CertificateBody : -----BEGIN CERTIFICATE-----
                MIICiTCCAfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMakGA1UEBhMC
                VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
                b24xFDASBgNVBA5TC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxH3Ad
                BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
                MTIwNDI0MjA0NTIxWjCBiDELMakGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
                VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC0lBTSBDb25z
                b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxH3AdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                YXpvi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                21uUSfwfEvySWtC2XADZ4nB+BLygVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
                rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
                Ibb30hjZnzcVQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
                FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
                NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob

```

- Per i dettagli sull'API, vedere [ListSigningCertificates](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMUser

Il seguente esempio di codice mostra come utilizzare. Get-IAMUser

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera i dettagli dell'utente denominato **David**.

```
Get-IAMUser -UserName David
```

Output:

```

Arn           : arn:aws:iam::123456789012:user/David
CreateDate    : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path          : /

```

```
UserId      : Y4FKWQCXTA52QEXAMPLE1
UserName    : David
```

Esempio 2: questo esempio recupera i dettagli dell'utente IAM correntemente connesso.

```
Get-IAMUser
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/Bob
CreateDate   : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path         : /
UserId       : 7K3GJEANSKZF2EXAMPLE2
UserName     : Bob
```

- Per i dettagli sull'API, vedere [GetUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMUserList

Il seguente esempio di codice mostra come utilizzare. Get-IAMUserList

Strumenti per PowerShell V4

Esempio 1: Questo esempio recupera una raccolta di utenti nella cartella corrente. Account AWS

```
Get-IAMUserList
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/Administrator
CreateDate   : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path         : /
UserId       : 7K3GJEANSKZF2EXAMPLE1
UserName     : Administrator

Arn          : arn:aws:iam::123456789012:user/Bob
CreateDate   : 4/6/2015 12:54:42 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path         : /
UserId       : L3EWNONDOM3YUEXAMPLE2
```



```

UserName      : bab

Arn           : arn:aws:iam::123456789012:user/David
CreateDate    : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path          : /
UserId        : Y4FKWQCXTA52QEXAMPLE3
UserName      : David

```

- Per i dettagli sull'API, vedere [ListUsers](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMUserPolicy

Il seguente esempio di codice mostra come utilizzare Get-IAMUserPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera i dettagli della policy in linea denominata **Davids_IAM_Admin_Policy** che è incorporata nell'utente IAM denominato **David**. Il documento di policy è codificato nell'URL.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Output:

```

PolicyDocument                                     PolicyName
-----
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%... Davids_IAM_Admin_Policy
David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [

```

```
        "iam:GetUser",
        "iam:ListUsers"
    ],
    "Resource": [
        "arn:aws:iam::111122223333:user/*"
    ]
  }
]
```

- Per i dettagli sull'API, vedere [GetUserPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMUserPolicyList

Il seguente esempio di codice mostra come utilizzare. Get-IAMUserPolicyList

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera l'elenco dei nomi delle policy in linea incorporate nell'utente IAM denominato **David**.

```
Get-IAMUserPolicyList -UserName David
```

Output:

```
Davids_IAM_Admin_Policy
```

- Per i dettagli sull'API, vedere [ListUserPolicies](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMUserTagList

Il seguente esempio di codice mostra come utilizzare. Get-IAMUserTagList

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera il tag associato all'utente.

```
Get-IAMUserTagList -UserName joe
```

- Per i dettagli sull'API, vedere [ListUserTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-IAMVirtualMFADevice

Il seguente esempio di codice mostra come utilizzare. Get-IAMVirtualMFADevice

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera una raccolta di dispositivi MFA virtuali assegnati agli utenti dell'account. AWS La proprietà **User** di ciascuno è un oggetto con i dettagli dell'utente IAM a cui è assegnato il dispositivo.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Output:

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Per i dettagli sull'API, vedere [ListVirtualMfaDevices](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMAccessKey

Il seguente esempio di codice mostra come utilizzare. New-IAMAccessKey

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova chiave di accesso e una coppia di chiavi di accesso segrete e le assegna all'utente **David**. Assicurati di salvare i valori **AccessKeyId** e **SecretAccessKey** in un file perché questa è l'unica volta in cui puoi ottenere il

SecretAccessKey. Non puoi recuperarla in un secondo momento. Se perdi la chiave segreta, dovrai creare una nuova coppia di chiavi di accesso.

```
New-IAMAccessKey -UserName David
```

Output:

```
AccessKeyId      : AKIAIOSFODNN7EXAMPLE
CreateDate       : 4/13/2015 1:00:42 PM
SecretAccessKey  : wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Per i dettagli sull'API, vedere [CreateAccessKey](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMAccountAlias

Il seguente esempio di codice mostra come utilizzare `New-IAMAccountAlias`

Strumenti per PowerShell V4

Esempio 1: Questo esempio modifica l'alias dell' AWS account in **mycompanyaws**. L'indirizzo della pagina di accesso dell'utente cambia in `panyaws.signin.aws.amazon.com/console`. `https://mycom` L'URL originale che utilizza il numero ID dell'account anziché l'alias (`https://<accountidnumber>.signin.aws.amazon.com/console`) continua a funzionare. Tuttavia, qualsiasi URLs alias precedentemente definito smette di funzionare.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Per i dettagli sull'API, vedere [CreateAccountAlias](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMGroup

Il seguente esempio di codice mostra come utilizzare `New-IAMGroup`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo gruppo IAM denominato **Developers**.

```
New-IAMGroup -GroupName Developers
```

Output:

```
Arn      : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId   : QNEJ5PM4NFSQCEXAMPLE1
GroupName : Developers
Path      : /
```

- Per i dettagli sull'API, vedere [CreateGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMInstanceProfile

Il seguente esempio di codice mostra come utilizzare `New-IAMInstanceProfile`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo profilo dell'istanza IAM denominato **ProfileForDevEC2Instance**. È necessario eseguire il comando **Add-IAMRoleToInstanceProfile** separatamente per associare il profilo dell'istanza a un ruolo IAM esistente che fornisce le autorizzazioni all'istanza. Infine, collega il profilo dell'istanza a un'EC2 istanza quando la avvii. A tale scopo, utilizza il cmdlet **New-EC2Instance** con il parametro **InstanceProfile_Arn** o **InstanceProfile_Name**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Output:

```
Arn      : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate : 4/14/2015 11:31:39 AM
InstanceProfileId : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path      : /
Roles     : {}
```

- Per i dettagli sull'API, vedere [CreateInstanceProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMLoginProfile

Il seguente esempio di codice mostra come utilizzare New-IAMLoginProfile

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una password (temporanea) per l'utente IAM Bob e imposta l'indicatore che richiede all'utente di modificare la password al successivo accesso di **Bob**.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Output:

CreateDate	PasswordResetRequired	UserName
-----	-----	-----
4/14/2015 12:26:30 PM	True	Bob

- Per i dettagli sull'API, vedere [CreateLoginProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMOpenIDConnectProvider

Il seguente esempio di codice mostra come utilizzare New-IAMOpenIDConnectProvider

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un provider OIDC IAM associato al servizio del provider compatibile con OIDC che si trova nell'URL **https://example.oidcprovider.com** e nell'ID client **my-testapp-1**. Il provider OIDC fornisce l'impronta digitale. Per autenticare l'impronta digitale, segui i passaggi su <http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint>

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Output:

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Per i dettagli sull'API, vedere [CreateOpenIdConnectProvider](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMPolicy

Il seguente esempio di codice mostra come utilizzare. New-IAMPolicy

Strumenti per PowerShell V4

Esempio 1: Questo esempio crea una nuova policy IAM nell' AWS account corrente denominato **MySamplePolicy** Il file **MySamplePolicy.json** fornisce il contenuto della policy. Tenere presente che per elaborare correttamente il file della policy JSON è necessario utilizzare il parametro di cambio **-Raw**.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
MySamplePolicy.json)
```

Output:

```
Arn           : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount : 0
CreateDate     : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description    :
IsAttachable   : True
Path           : /
PolicyId       : LD4KP6HVFE7WGEXAMPLE1
PolicyName     : MySamplePolicy
UpdateDate     : 4/14/2015 2:45:59 PM
```

- Per i dettagli sull'API, vedere [CreatePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMPolicyVersion

Il seguente esempio di codice mostra come utilizzare. New-IAMPolicyVersion

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova versione "v2" della policy IAM il cui ARN è **arn:aws:iam::123456789012:policy/MyPolicy** e la rende la versione predefinita. Il file

NewPolicyVersion.json fornisce il contenuto della policy. Tenere presente che per elaborare correttamente il file della policy JSON è necessario utilizzare il parametro di cambio **-Raw**.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Output:

CreateDate VersionId	Document	IsDefaultVersion
----- -----	-----	-----
4/15/2015 10:54:54 AM v2		True

- Per i dettagli sull'API, vedere [CreatePolicyVersion](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMRole

Il seguente esempio di codice mostra come utilizzare. New-IAMRole

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo ruolo denominato **MyNewRole** e collega la policy trovata nel file **NewRoleTrustPolicy.json**. Tenere presente che per elaborare correttamente il file della policy JSON è necessario utilizzare il parametro di cambio **-Raw**. Il documento di policy visualizzato nell'output è codificato nell'URL. In questo esempio viene decodificato con il metodo **UrlDecode** .NET.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Output:

```
Arn : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0D
%0A%20%20%22Statement%22
```



```

%3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%2C
%0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow%22%2C
%0D%0A%20%20%20%20%20%20%22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
%3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
%20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D%0A
%20%20%20%20%7D%0D%0A%20
%20%5D%0D%0A%7D
CreateDate          : 4/15/2015 11:04:23 AM
Path                : /
RoleId              : V5PAJI2KPN4EAEXAMPLE1
RoleName            : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

- Per i dettagli sull'API, vedere [CreateRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMSAMLProvider

Il seguente esempio di codice mostra come utilizzare. New-IAMSAMLProvider

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova entità per il provider SAML in IAM. È denominato **MySAMLProvider** ed è descritto dal documento di metadati SAML contenuto nel file

SAMLMetaData.xml, che è stato scaricato separatamente dal sito web del provider di servizi SAML.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

Output:

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Per i dettagli sull'API, vedere [Create SAMLProvider](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMServiceLinkedRole

Il seguente esempio di codice mostra come utilizzare. New-IAMServiceLinkedRole

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un ruolo collegato ai servizi per il servizio di scalabilità automatica.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix RoleNameEndsWithThis -Description "My service-linked role to support autoscaling"
```

- Per i dettagli sull'API, vedere [CreateServiceLinkedRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMUser

Il seguente esempio di codice mostra come utilizzare. New-IAMUser

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un utente IAM denominato **Bob**. Se Bob deve accedere alla AWS console, è necessario eseguire separatamente il comando **New-IAMLoginProfile** per creare un profilo di accesso con una password. Se Bob deve eseguire AWS PowerShell comandi CLI multiplatforma o AWS effettuare chiamate API, è necessario eseguire separatamente **New-IAMAccessKey** il comando per creare le chiavi di accesso.

```
New-IAMUser -UserName Bob
```

Output:

```
Arn          : arn:aws:iam::123456789012:user/Bob
CreateDate   : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path         : /
UserId       : AIDAJWGEFDMEMEXAMPLE1
UserName     : Bob
```

- Per i dettagli sull'API, vedere [CreateUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-IAMVirtualMFADevice

Il seguente esempio di codice mostra come utilizzare `New-IAMVirtualMFADevice`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo dispositivo MFA virtuale. Le righe 2 e 3 estraggono il valore **Base32StringSeed** necessario al programma software MFA virtuale per creare un account (in alternativa al codice QR). Dopo aver configurato il programma con il valore, ottieni due codici di autenticazione sequenziali dal programma. Infine, utilizza l'ultimo comando per collegare il dispositivo MFA virtuale all'utente IAM **Bob** e sincronizzare l'account con i due codici di autenticazione.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Output:

```
-- Pause here to enter base-32 string seed code into virtual MFA program to register
account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Esempio 2: questo esempio crea un nuovo dispositivo MFA virtuale. Le righe 2 e 3 estraggono il valore **QRCodePNG** e lo scrivono in un file. Questa immagine può essere scansionata dal programma software MFA virtuale per creare un account (in alternativa all'immissione manuale del valore StringSeed Base32). Dopo aver creato l'account nel tuo programma MFA virtuale, ottieni due codici di autenticazione sequenziali e inseriscili negli ultimi comandi per collegare il dispositivo MFA virtuale all'utente IAM **Bob** e sincronizzare l'account.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path QRCode.png
```

Output:

```
-- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Per i dettagli sull'API, vedere [CreateVirtualMfaDevice](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Publish-IAMServerCertificate

Il seguente esempio di codice mostra come utilizzare `Publish-IAMServerCertificate`

Strumenti per PowerShell V4

Esempio 1: questo esempio carica un nuovo certificato server sull'account IAM. I file contenenti il corpo del certificato, la chiave privata e (facoltativamente) la catena di certificati devono tutti essere codificati con PEM. Tieni presente che i parametri richiedono il contenuto effettivo dei file e non i nomi dei file. Per elaborare correttamente il contenuto del file JSON è necessario utilizzare il parametro di cambio **-Raw**.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
(Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Output:

```
Arn                : arn:aws:iam::123456789012:server-certificate/MyTestCert
```

```
Expiration      : 1/14/2018 9:52:36 AM
Path            : /
ServerCertificateId : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate      : 4/21/2015 11:14:16 AM
```

- Per i dettagli sull'API, vedere [UploadServerCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Publish-IAMSigningCertificate

Il seguente esempio di codice mostra come utilizzare `Publish-IAMSigningCertificate`

Strumenti per PowerShell V4

Esempio 1: questo esempio carica un nuovo certificato di firma X.509 e lo associa all'utente IAM denominato **Bob**. Il file contenente il corpo del certificato è codificato in PEM. Il parametro **CertificateBody** richiede il contenuto effettivo del file e non il nome del file. Per elaborare correttamente il file è necessario utilizzare il parametro di cambio **-Raw**.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
SampleSigningCert.pem)
```

Output:

```
CertificateBody : -----BEGIN CERTIFICATE-----
MIICiTCCAFICCD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xZDASBgNVBASTC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHjAd
BgkqhkiG9w0BCQEWEG5vb25lQGFTYXpvi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI1MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xZDASBgNVBASTC0lBTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHjAdBgkqhkiG9w0BCQEWEG5vb25lQGFT
YXpvi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnzcVQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----
```

```
CertificateId    : Y3EK7RMEXAMPLESV33FCEXAMPLEHMLU
Status           : Active
UploadDate       : 4/20/2015 1:26:01 PM
UserName         : Bob
```

- Per i dettagli sull'API, vedere [UploadSigningCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-IAMGroupPolicy

Il seguente esempio di codice mostra come utilizzare. Register-IAMGroupPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio collega la policy gestita dal cliente denominata **TesterPolicy** al gruppo IAM **Testers**. Gli utenti di quel gruppo sono immediatamente interessati dalle autorizzazioni definite nella versione predefinita di tale policy.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
arn:aws:iam::123456789012:policy/TesterPolicy
```

Esempio 2: questo esempio allega la policy AWS gestita denominata **AdministratorAccess** al gruppo IAM. **Admins** Gli utenti di quel gruppo sono immediatamente interessati dalle autorizzazioni definite nella versione predefinita di tale policy.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Per i dettagli sull'API, vedere [AttachGroupPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-IAMRolePolicy

Il seguente esempio di codice mostra come utilizzare. Register-IAMRolePolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio allega la policy AWS gestita denominata **SecurityAudit** al ruolo IAM. **CoSecurityAuditors** Gli utenti che assumono quel ruolo sono immediatamente interessati dalle autorizzazioni definite nella versione predefinita di tale policy.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn  
arn:aws:iam::aws:policy/SecurityAudit
```

- Per i dettagli sull'API, vedere [AttachRolePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-IAMUserPolicy

Il seguente esempio di codice mostra come utilizzare. Register-IAMUserPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio allega la policy AWS gestita denominata **AmazonCognitoPowerUser** all'utente IAM. **Bob** L'utente è immediatamente interessato dalle autorizzazioni definite nella versione più recente di tale policy.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/  
AmazonCognitoPowerUser
```

- Per i dettagli sull'API, vedere [AttachUserPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMAccessKey

Il seguente esempio di codice mostra come utilizzare. Remove-IAMAccessKey

Strumenti per PowerShell V4

Esempio 1: Questo esempio elimina la coppia di chiavi di AWS accesso con l'ID della chiave **AKIAIOSFODNN7EXAMPLE** dall'utente denominato **Bob**.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Per i dettagli sull'API, vedere [DeleteAccessKey](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMAccountAlias

Il seguente esempio di codice mostra come utilizzare. Remove-IAMAccountAlias

Strumenti per PowerShell V4

Esempio 1: Questo esempio rimuove l'alias dell'account dal tuo Account AWS. La pagina di accesso utente con l'alias all'indirizzo <https://mycompanyaws.signin.aws.amazon.com/console> non funziona più. Devi invece utilizzare l'URL originale con il tuo numero ID su <https://signin.aws.amazon.com/console>. Account AWS <accountidnumber>

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4). [DeleteAccountAlias](#) AWS Strumenti per PowerShell

Remove-IAMAccountPasswordPolicy

Il seguente esempio di codice mostra come utilizzare Remove-IAMAccountPasswordPolicy

Strumenti per PowerShell V4

Esempio 1: Questo esempio elimina la politica delle password per Account AWS e ripristina tutti i valori ai valori predefiniti originali. Se attualmente non esiste una politica in materia di password, viene visualizzato il seguente messaggio di errore: Impossibile trovare la politica dell'account con il nome PasswordPolicy .

```
Remove-IAMAccountPasswordPolicy
```

- Per i dettagli sull'API, vedere [DeleteAccountPasswordPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMClientIDFromOpenIDConnectProvider

Il seguente esempio di codice mostra come utilizzare Remove-IAMClientIDFromOpenIDConnectProvider

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove l'ID client **My-TestApp-3** dall'elenco dei client IDs associati al provider IAM OIDC il cui ARN è **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com**


```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3  
-OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/  
example.oidcprovider.com
```

- Per i dettagli sull'API, vedere [RemoveClientIDFromOpenIDConnectProvider](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMGroup

Il seguente esempio di codice mostra come utilizzare. Remove-IAMGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il gruppo IAM denominato **MyTestGroup**. Il primo comando rimuove tutti gli utenti IAM che sono membri del gruppo e il secondo comando elimina il gruppo IAM. Entrambi i comandi funzionano senza alcuna richiesta di conferma.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName  
MyTestGroup -Force  
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Per i dettagli sull'API, vedere [DeleteGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMGroupPolicy

Il seguente esempio di codice mostra come utilizzare. Remove-IAMGroupPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la policy in linea denominata **TesterPolicy** dal gruppo IAM **Testers**. Gli utenti di quel gruppo perdono immediatamente le autorizzazioni definite in tale policy.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Per i dettagli sull'API, vedere [DeleteGroupPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMInstanceProfile

Il seguente esempio di codice mostra come utilizzare. Remove-IAMInstanceProfile

Strumenti per PowerShell V4

Esempio 1: Questo esempio elimina il profilo di EC2 istanza denominato.

MyAppInstanceProfile Il primo comando scollega tutti i ruoli dal profilo dell'istanza, quindi il secondo comando elimina il profilo dell'istanza.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles | Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile  
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Per i dettagli sull'API, vedere [DeleteInstanceProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMLoginProfile

Il seguente esempio di codice mostra come utilizzare. Remove-IAMLoginProfile

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il profilo di accesso dall'utente IAM denominato **Bob**. Ciò impedisce all'utente di accedere alla console. AWS Non impedisce all'utente di eseguire chiamate AWS CLI o API utilizzando chiavi di AWS accesso che potrebbero essere ancora collegate all'account utente. PowerShell

```
Remove-IAMLoginProfile -UserName Bob
```

- Per i dettagli sull'API, vedere [DeleteLoginProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMOpenIDConnectProvider

Il seguente esempio di codice mostra come utilizzare. Remove-IAMOpenIDConnectProvider

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il provider OIDC IAM che si connette al provider **example.oidcprovider.com**. Assicurati di aggiornare o eliminare tutti i ruoli che fanno riferimento a questo provider nell'elemento **Principal** della policy di attendibilità del ruolo.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn  
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Per i dettagli sull'API, vedere [DeleteOpenIdConnectProvider](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMPolicy

Il seguente esempio di codice mostra come utilizzare. Remove-IAMPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la policy il cui ARN è **arn:aws:iam::123456789012:policy/MySamplePolicy**. Prima di poter eliminare la policy, devi prima eliminare tutte le versioni tranne quella predefinita eseguendo **Remove-IAMPolicyVersion**. È inoltre necessario scollegare la policy da qualsiasi utente, gruppo o ruolo IAM.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Esempio 2: questo esempio elimina una policy eliminando prima tutte le versioni non predefinite della policy, scollegandola da tutte le entità IAM collegate ed eliminando quindi la policy stessa. La prima riga recupera l'oggetto della policy. La seconda riga recupera tutte le versioni della policy che non sono contrassegnate come versione predefinita in una raccolta e quindi elimina ogni policy nella raccolta. La terza riga recupera tutti gli utenti, i gruppi e i ruoli IAM a cui è collegata la policy. Le righe da quattro a sei scollegano la policy da ogni entità collegata. L'ultima riga utilizza questo comando per rimuovere la policy gestita e la versione predefinita rimanente. L'esempio include il parametro di cambio **-Force** su qualsiasi riga che lo richieda per sopprimere le richieste di conferma.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy  
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |  
Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
```

```
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Per i dettagli sull'API, vedere [DeletePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMPolicyVersion

Il seguente esempio di codice mostra come utilizzare `Remove-IAMPolicyVersion`

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la versione identificata come **v2** dalla policy il cui ARN è **arn:aws:iam::123456789012:policy/MySamplePolicy**.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy -
VersionID v2
```

Esempio 2: questo esempio elimina una policy eliminando prima tutte le versioni non predefinite della policy e quindi eliminando la policy stessa. La prima riga recupera l'oggetto della policy. La seconda riga recupera tutte le versioni della policy che non sono contrassegnate come predefinite in una raccolta e quindi utilizza questo comando per eliminare ogni policy nella raccolta. L'ultima riga rimuove la policy stessa e la versione predefinita rimanente. Tieni presente che per eliminare correttamente una policy gestita, devi anche scollegare la policy da qualsiasi utente, gruppo o ruolo utilizzando i comandi **Unregister-IAMUserPolicy**, **Unregister-IAMGroupPolicy** e **Unregister-IAMRolePolicy**. Vedere l'esempio per il cmdlet **Remove-IAMPolicy**.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
  Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Per i dettagli sull'API, vedere [DeletePolicyVersion](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMRole

Il seguente esempio di codice mostra come utilizzare. Remove-IAMRole

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il ruolo denominato **MyNewRole** dall'account IAM corrente. Prima di poter eliminare il ruolo, devi utilizzare il comando **Unregister-IAMRolePolicy** per scollegare eventuali policy gestite. Le policy in linea vengono eliminate con il ruolo.

```
Remove-IAMRole -RoleName MyNewRole
```

Esempio 2: questo esempio rimuove tutte le policy gestite dal ruolo denominato **MyNewRole** e quindi elimina il ruolo. La prima riga recupera tutte le policy gestite collegate al ruolo come raccolta e quindi le scollega dal ruolo. La seconda riga elimina il ruolo stesso. Le policy in linea vengono eliminate insieme al ruolo.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -  
RoleName MyNewRole  
Remove-IAMRole -RoleName MyNewRole
```

- Per i dettagli sull'API, vedere [DeleteRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMRoleFromInstanceProfile

Il seguente esempio di codice mostra come utilizzare. Remove-IAMRoleFromInstanceProfile

Strumenti per PowerShell V4

Esempio 1: Questo esempio elimina il ruolo denominato **MyNewRole** dal profilo di EC2 istanza denominato. **MyNewRole** Un profilo dell'istanza creato nella console IAM ha sempre lo stesso nome del ruolo, come in questo esempio. Se li crei nell'API o nella CLI, possono avere nomi diversi.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName MyNewRole  
-Force
```

- Per i dettagli sull'API, vedere [RemoveRoleFromInstanceProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMRolePermissionsBoundary

Il seguente esempio di codice mostra come utilizzare. Remove-IAMRolePermissionsBoundary

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra come rimuovere il limite delle autorizzazioni associato a un ruolo IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Per i dettagli sull'API, vedere [DeleteRolePermissionsBoundary](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMRolePolicy

Il seguente esempio di codice mostra come utilizzare. Remove-IAMRolePolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la policy in linea **S3AccessPolicy** che è incorporata nel ruolo IAM **S3BackupRole**.

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Per i dettagli sull'API, vedere [DeleteRolePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMRoleTag

Il seguente esempio di codice mostra come utilizzare. Remove-IAMRoleTag

Strumenti per PowerShell V4

Esempio 1: Questo esempio rimuove il tag dal ruolo denominato "MyRoleName" con la chiave del tag come «abac». Per rimuovere più tag, fornisci un elenco di chiavi di tag separate da virgole.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Per i dettagli sull'API, vedere [UntagRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMSAMLProvider

Il seguente esempio di codice mostra come utilizzare. Remove-IAMSAMLProvider

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il provider SAML 2.0 IAM il cui ARN è **arn:aws:iam::123456789012:saml-provider/SAMLADFSPProvider**.

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/SAMLADFSPProvider
```

- Per i dettagli sull'API, vedere [Delete SAMLProvider](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMServerCertificate

Il seguente esempio di codice mostra come utilizzare. Remove-IAMServerCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il certificato server denominato **MyServerCert**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Per i dettagli sull'API, vedere [DeleteServerCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMServiceLinkedRole

Il seguente esempio di codice mostra come utilizzare. Remove-IAMServiceLinkedRole

Strumenti per PowerShell V4

Esempio 1: questo esempio ha eliminato il ruolo collegato al servizio. Tieni presente che se il servizio utilizza ancora questo ruolo, allora questo comando genererà un errore.

```
Remove-IAMServiceLinkedRole -RoleName  
AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Per i dettagli sull'API, vedere [DeleteServiceLinkedRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMSigningCertificate

Il seguente esempio di codice mostra come utilizzare. Remove-IAMSigningCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il certificato di firma con l'ID **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** dell'utente IAM denominato **Bob**.

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId  
Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Per i dettagli sull'API, vedere [DeleteSigningCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMUser

Il seguente esempio di codice mostra come utilizzare. Remove-IAMUser

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'utente IAM denominato **Bob**.

```
Remove-IAMUser -UserName Bob
```

Esempio 2: questo esempio elimina l'utente IAM denominato **Theresa** insieme a tutti gli elementi che devono essere eliminati per primi.

```
$name = "Theresa"  
  
# find any groups and remove user from them  
$groups = Get-IAMGroupForUser -UserName $name  
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName -  
UserName $name -Force }
```



```
# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
    $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn $pol.PolicyArn
    -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
    $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
    UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
# suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -SerialNumber
        $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Per i dettagli sull'API, vedere [DeleteUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMUserFromGroup

Il seguente esempio di codice mostra come utilizzare. Remove-IAMUserFromGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove l'utente IAM **Bob** dal gruppo **Testers**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Esempio 2: questo esempio trova tutti i gruppi di cui l'utente IAM **Theresa** è membro e quindi rimuove **Theresa** da tali gruppi.

```
$groups = Get-IAMGroupForUser -UserName Theresa  
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName -  
  UserName Theresa -Force }
```

Esempio 3: questo esempio mostra un modo alternativo per rimuovere l'utente IAM **Bob** dal gruppo **Testers**.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -GroupName  
  Testers -Force
```

- Per i dettagli sull'API, vedere [RemoveUserFromGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMUserPermissionsBoundary

Il seguente esempio di codice mostra come utilizzare. Remove-IAMUserPermissionsBoundary

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra come rimuovere il limite delle autorizzazioni associato a un utente IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Per i dettagli sull'API, vedere [DeleteUserPermissionsBoundary](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMUserPolicy

Il seguente esempio di codice mostra come utilizzare. Remove-IAMUserPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la policy in linea denominata **AccessToEC2Policy** che è incorporata nell'utente IAM denominato **Bob**.

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Esempio 2: questo esempio trova tutte le policy in linea incorporate nell'utente IAM denominato **Theresa** e quindi le elimina.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa  
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName  
    Theresa -Force }
```

- Per i dettagli sull'API, vedere [DeleteUserPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMUserTag

Il seguente esempio di codice mostra come utilizzare. Remove-IAMUserTag

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il tag dall'utente denominato "joe" con la chiave di tag "abac" e "xyzw". Per rimuovere più tag, fornisci un elenco di chiavi di tag separate da virgole.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Per i dettagli sull'API, vedere [UntagUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-IAMVirtualMFADevice

Il seguente esempio di codice mostra come utilizzare. Remove-IAMVirtualMFADevice

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il dispositivo MFA virtuale IAM il cui ARN è.
arn:aws:iam::123456789012:mfa/bob.

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Esempio 2: questo esempio verifica se all'utente IAM Theresa è assegnato un dispositivo MFA. Se ne viene trovato uno, il dispositivo viene disabilitato per l'utente IAM. Se il dispositivo è virtuale, anch'esso viene eliminato.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -SerialNumber
    $mfa.SerialNumber }
}
```

- Per i dettagli sull'API, vedere [DeleteVirtualMfaDevice](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Request-IAMCredentialReport

Il seguente esempio di codice mostra come utilizzare. Request-IAMCredentialReport

Strumenti per PowerShell V4

Esempio 1: questo esempio richiede la generazione di un nuovo report, operazione che può essere eseguita ogni quattro ore. Se l'ultimo report è ancora recente, il campo Stato riporta **COMPLETE**. Utilizzare **Get-IAMCredentialReport** per visualizzare il report completato.

```
Request-IAMCredentialReport
```

Output:

Description	State
-----	-----
No report exists. Starting a new report generation task	STARTED

- Per i dettagli sull'API, vedere [GenerateCredentialReport](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Request-IAServiceLastAccessedDetail

Il seguente esempio di codice mostra come utilizzare. Request-IAServiceLastAccessedDetail

Strumenti per PowerShell V4

Esempio 1: questo esempio è un cmdlet equivalente all'API.

GenerateServiceLastAccessedDetails Ciò fornisce un job id che può essere utilizzato in Get-IAServiceLastAccessedDetail e Get-IAServiceLastAccessedDetailWithEntity

```
Request-IAServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Per i dettagli sull'API, vedere [GenerateServiceLastAccessedDetails](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-IAMDefaultPolicyVersion

Il seguente esempio di codice mostra come utilizzare. Set-IAMDefaultPolicyVersion

Strumenti per PowerShell V4

Esempio 1: questo esempio imposta la versione **v2** della policy il cui ARN è

arn:aws:iam::123456789012:policy/MyPolicy come versione attiva predefinita.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -  
VersionId v2
```

- Per i dettagli sull'API, vedere [SetDefaultPolicyVersion](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-IAMRolePermissionsBoundary

Il seguente esempio di codice mostra come utilizzare. Set-IAMRolePermissionsBoundary

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra come impostare il limite delle autorizzazioni per un ruolo IAM. È possibile impostare politiche AWS gestite o politiche personalizzate come limite di autorizzazione.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary  
arn:aws:iam::123456789012:policy/intern-boundary
```

- Per i dettagli sull'API, vedere [PutRolePermissionsBoundary](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-IAMUserPermissionsBoundary

Il seguente esempio di codice mostra come utilizzare. Set-IAMUserPermissionsBoundary

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra come impostare il limite delle autorizzazioni per l'utente. È possibile impostare politiche AWS gestite o politiche personalizzate come limite di autorizzazione.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary  
arn:aws:iam::123456789012:policy/intern-boundary
```

- Per i dettagli sull'API, vedere [PutUserPermissionsBoundary](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Sync-IAMMFADevice

Il seguente esempio di codice mostra come utilizzare. Sync-IAMMFADevice

Strumenti per PowerShell V4

Esempio 1: questo esempio sincronizza il dispositivo MFA associato all'utente IAM **Bob** e il cui ARN è **arn:aws:iam::123456789012:mfa/bob** con un programma di autenticazione che ha fornito i due codici di autenticazione.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -  
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Esempio 2: questo esempio sincronizza il dispositivo MFA IAM associato all'utente IAM **Theresa** con un dispositivo fisico che ha il numero di serie **ABCD12345678** e che ha fornito i due codici di autenticazione.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -  
AuthenticationCode2 987654 -UserName Theresa
```

- Per i dettagli sull'API, vedere [ResyncMfaDevice](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-IAMGroupPolicy

Il seguente esempio di codice mostra come utilizzare. Unregister-IAMGroupPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio scollega la policy del gruppo gestita il cui ARN è **arn:aws:iam::123456789012:policy/TesterAccessPolicy** dal gruppo denominato **Testers**.

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn  
arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Esempio 2: questo esempio trova tutte le policy gestite collegate al gruppo denominato **Testers** e le scollega dal gruppo.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -  
Groupname Testers
```

- Per i dettagli sull'API, vedere [DetachGroupPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-IAMRolePolicy

Il seguente esempio di codice mostra come utilizzare. Unregister-IAMRolePolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio scollega la policy del gruppo gestita il cui ARN è **arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy** dal ruolo denominato **FedTesterRole**.

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn  
arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Esempio 2: questo esempio trova tutte le policy gestite collegate al ruolo denominato **FedTesterRole** e le scollega dal ruolo.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy -  
Rolenam FedTesterRole
```

- Per i dettagli sull'API, vedere [DetachRolePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-IAMUserPolicy

Il seguente esempio di codice mostra come utilizzare. Unregister-IAMUserPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio scollega la policy gestita il cui ARN è **arn:aws:iam::123456789012:policy/TesterPolicy** dall'utente IAM denominato **Bob**.

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::123456789012:policy/  
TesterPolicy
```

Esempio 2: questo esempio trova tutte le policy gestite collegate all'utente IAM denominato **Theresa** e le scollega dall'utente.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -Username  
Theresa
```

- Per i dettagli sull'API, vedere [DetachUserPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMAccessKey

Il seguente esempio di codice mostra come utilizzare. Update-IAMAccessKey

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica lo stato della chiave di accesso **AKIAIOSFODNN7EXAMPLE** per l'utente IAM denominato **Bob** in **Inactive**.

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status Inactive
```

- Per i dettagli sull'API, vedere [UpdateAccessKey](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMAccountPasswordPolicy

Il seguente esempio di codice mostra come utilizzare. Update-IAMAccountPasswordPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la policy delle password per l'account con le impostazioni specificate. Tenere presente che tutti i parametri non inclusi nel comando non vengono lasciati invariati. Vengono invece ripristinati ai valori predefiniti.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry  
$false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20  
-RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -  
RequireUppercaseCharacters $true
```

- Per i dettagli sull'API, vedere [UpdateAccountPasswordPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMAssumeRolePolicy

Il seguente esempio di codice mostra come utilizzare. Update-IAMAssumeRolePolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna il ruolo IAM denominato **ClientRole** con una nuova policy di attendibilità, il cui contenuto proviene dal file **ClientRolePolicy.json**. Tenere presente che per elaborare correttamente il contenuto del file JSON è necessario utilizzare il parametro di cambio **-Raw**.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw ClientRolePolicy.json)
```

- Per i dettagli sull'API, vedere [UpdateAssumeRolePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMGroup

Il seguente esempio di codice mostra come utilizzare. Update-IAMGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio rinomina il gruppo IAM **Testers** in **AppTesters**.

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Esempio 2: questo esempio modifica il percorso del gruppo IAM **AppTesters** in **/Org1/Org2/**. Questo modifica l'ARN del gruppo in **arn:aws:iam::123456789012:group/Org1/Org2/AppTesters**.

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Per i dettagli sull'API, vedere [UpdateGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMLoginProfile

Il seguente esempio di codice mostra come utilizzare. Update-IAMLoginProfile

Strumenti per PowerShell V4

Esempio 1: questo esempio imposta una nuova password temporanea per l'utente IAM **Bob** e richiede all'utente di modificare la password al successivo accesso.

```
Update-IAMLoginProfile -Username Bob -Password "P@ssw0rd1234" -PasswordResetRequired $true
```

- Per i dettagli sull'API, vedere [UpdateLoginProfile](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMOpenIDConnectProviderThumbprint

Il seguente esempio di codice mostra come utilizzare. Update-IAMOpenIDConnectProviderThumbprint

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna l'elenco delle impronte digitali dei certificati per il provider OIDC il cui ARN è **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com** per utilizzare una nuova impronta digitale. Il provider OIDC condivide il nuovo valore quando il certificato associato al provider cambia.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
7359755EXAMPLEEabc3060bce3EXAMPLEEc4542a3
```

- Per i dettagli sull'API, vedere [UpdateOpenIdConnectProviderThumbprint](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMRole

Il seguente esempio di codice mostra come utilizzare. Update-IAMRole

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la descrizione del ruolo e il valore della durata massima della sessione (in secondi) per cui è possibile richiedere la sessione di un ruolo.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Per i dettagli sull'API, vedere [UpdateRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMRoleDescription

Il seguente esempio di codice mostra come utilizzare. Update-IAMRoleDescription

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la descrizione di un ruolo IAM nel tuo account.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Per i dettagli sull'API, vedere [UpdateRoleDescription](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMSAMLProvider

Il seguente esempio di codice mostra come utilizzare. Update-IAMSAMLProvider

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna il provider SAML in IAM il cui ARN è **arn:aws:iam::123456789012:saml-provider/SAMLADFS** con un nuovo documento di metadati SAML dal file **SAMLMetaData.xml**. Tenere presente che per elaborare correttamente il contenuto del file JSON è necessario utilizzare il parametro di cambio **-Raw**.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Per i dettagli sull'API, vedere [UpdateSamlProvider](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMServerCertificate

Il seguente esempio di codice mostra come utilizzare. Update-IAMServerCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio rinomina il certificato denominato **MyServerCertificate** in **MyRenamedServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewServerCertificateName MyRenamedServerCertificate
```

Esempio 2: questo esempio sposta il certificato denominato **MyServerCertificate** nel percorso **/Org1/Org2/**. Questo modifica l'ARN della risorsa in **arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /  
Org1/Org2/
```

- Per i dettagli sull'API, vedere [UpdateServerCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMSigningCertificate

Il seguente esempio di codice mostra come utilizzare. Update-IAMSigningCertificate

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna il certificato associato all'utente IAM denominato **Bob** e il cui ID certificato per contrassegnarlo come inattivo è **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU**.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -  
UserName Bob -Status Inactive
```

- Per i dettagli sull'API, vedere [UpdateSigningCertificate](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-IAMUser

Il seguente esempio di codice mostra come utilizzare. Update-IAMUser

Strumenti per PowerShell V4

Esempio 1: questo esempio rinomina l'utente IAM **Bob** in **Robert**.

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Esempio 2: questo esempio modifica il percorso dell'utente IAM **Bob** in **/Org1/Org2/**, il che modifica effettivamente l'ARN dell'utente in **arn:aws:iam::123456789012:user/Org1/Org2/bob**.

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Per i dettagli sull'API, vedere [UpdateUser](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-IAMGroupPolicy

Il seguente esempio di codice mostra come utilizzare `Write-IAMGroupPolicy`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una policy in linea denominata **AppTesterPolicy** e la incorpora nel gruppo IAM **AppTesters**. Se esiste già una policy in linea con lo stesso nome, quest'ultima sarà sovrascritta. Il contenuto della policy JSON viene fornito nel file **apptesterpolicy.json**. Si noti che per elaborare correttamente il contenuto del file JSON è necessario utilizzare il parametro **-Raw**.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -  
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Per i dettagli sull'API, vedere [PutGroupPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-IAMRolePolicy

Il seguente esempio di codice mostra come utilizzare `Write-IAMRolePolicy`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una policy in linea denominata **FedTesterRolePolicy** e la incorpora nel ruolo IAM **FedTesterRole**. Se esiste già una policy in linea con lo stesso nome, quest'ultima sarà sovrascritta. Il contenuto della policy JSON proviene dal file **FedTesterPolicy.json**. Si noti che per elaborare correttamente il contenuto del file JSON è necessario utilizzare il parametro **-Raw**.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -  
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Per i dettagli sull'API, vedere [PutRolePolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-IAMUserPolicy

Il seguente esempio di codice mostra come utilizzare `Write-IAMUserPolicy`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una policy in linea denominata **EC2AccessPolicy** e la incorpora nell'utente IAM **Bob**. Se esiste già una policy in linea con lo stesso nome, quest'ultima sarà sovrascritta. Il contenuto della policy JSON proviene dal file **EC2AccessPolicy.json**. Si noti che per elaborare correttamente il contenuto del file JSON è necessario utilizzare il parametro **-Raw**.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument (Get-Content -Raw EC2AccessPolicy.json)
```

- Per i dettagli sull'API, vedere [PutUserPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Kinesis con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Kinesis.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-KINRecord

Il seguente esempio di codice mostra come utilizzare. Get-KINRecord

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra come restituire ed estrarre dati da una serie di uno o più record. L'iteratore supplied to Get-KINRecord determina la posizione iniziale dei record da

restituire, che in questo esempio vengono acquisiti in una variabile, `$records`. È quindi possibile accedere a ogni record indicizzando la raccolta `$records`. Supponendo che i dati nel record siano testo con codifica UTF-8, il comando finale mostra come estrarre i dati dall'oggetto e restituirli `MemoryStream` come testo alla console.

```
$records
$records = Get-KINRecord -ShardIterator "AAAAAAAAAAGIc....9VnbiRNaP"
```

Output:

MillisBehindLatest	NextShardIterator	Records
0	AAAAAAAAAAERNIq...uDn11HuUs	{Key1, Key2}

```
$records.Records[0]
```

Output:

ApproximateArrivalTimestamp	Data	PartitionKey	SequenceNumber
3/7/2016 5:14:33 PM	System.IO.MemoryStream	Key1	
4955986459776...931586			

```
[Text.Encoding]::UTF8.GetString($records.Records[0].Data.ToArray())
```

Output:

```
test data from string
```

- Per i dettagli sull'API, vedere [GetRecords](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-KINShardIterator

Il seguente esempio di codice mostra come utilizzare `Get-KINShardIterator`

Strumenti per PowerShell V4

Esempio 1: restituisce un'iterazione di shard per lo shard e la posizione iniziale specificati. I dettagli degli identificatori degli shard e dei numeri di sequenza possono essere ottenuti dall'output del Get-KINStream cmdlet, facendo riferimento all'insieme Shards dell'oggetto stream restituito. L'iteratore restituito può essere utilizzato con il Get-KINRecord cmdlet per estrarre i record di dati nello shard.

```
Get-KINShardIterator -StreamName "mystream" -ShardId "shardId-000000000000" -  
ShardIteratorType AT_SEQUENCE_NUMBER -StartingSequenceNumber "495598645..."
```

Output:

```
AAAAAAAAAAGIc....9VnbiRNaP
```

- Per i dettagli sull'API, vedere [GetShardIterator](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-KINStream

Il seguente esempio di codice mostra come utilizzare. Get-KINStream

Strumenti per PowerShell V4

Esempio 1: restituisce i dettagli del flusso specificato.

```
Get-KINStream -StreamName "mystream"
```

Output:

```
HasMoreShards      : False  
RetentionPeriodHours : 24  
Shards             : {}  
StreamARN          : arn:aws:kinesis:us-west-2:123456789012:stream/mystream  
StreamName         : mystream  
StreamStatus       : ACTIVE
```

- Per i dettagli sull'API, vedere [DescribeStream](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-KINStream

Il seguente esempio di codice mostra come utilizzare. New-KINStream

Strumenti per PowerShell V4

Esempio 1: crea un nuovo flusso. Per impostazione predefinita, questo cmdlet non restituisce alcun output, pertanto viene aggiunto lo PassThru switch - per restituire il valore fornito al StreamName parametro - per un uso successivo.

```
$streamName = New-KINStream -StreamName "mystream" -ShardCount 1 -PassThru
```

- Per i dettagli sull'API, vedere [CreateStream](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-KINStream

Il seguente esempio di codice mostra come utilizzare. Remove-KINStream

Strumenti per PowerShell V4

Esempio 1: elimina il flusso specificato. Viene richiesta una conferma prima dell'esecuzione del comando. Per disabilitare la richiesta di conferma, utilizza il parametro -Force.

```
Remove-KINStream -StreamName "mystream"
```

- Per i dettagli sull'API, vedere [DeleteStream](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-KINRecord

Il seguente esempio di codice mostra come utilizzare. Write-KINRecord

Strumenti per PowerShell V4

Esempio 1: scrive un record contenente la stringa fornita al parametro -Text.

```
Write-KINRecord -Text "test data from string" -StreamName "mystream" -PartitionKey "Key1"
```

Esempio 2: scrive un record contenente i dati contenuti nel file specificato. Il file viene trattato come una sequenza di byte. Pertanto, se contiene testo, deve essere scritto con la codifica necessaria prima di utilizzarlo con questo cmdlet.

```
Write-KINRecord -FilePath "C:\TestData.txt" -StreamName "mystream" -PartitionKey  
"Key2"
```

- Per i dettagli sull'API, vedere [PutRecord](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Lambda con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Lambda.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-LMResourceTag

Il seguente esempio di codice mostra come utilizzare. Add-LMResourceTag

Strumenti per PowerShell V4

Esempio 1: aggiunge i tre tag (Washington, Oregon e California) e i relativi valori associati alla funzione specificata identificata dal relativo ARN.

```
Add-LMResourceTag -Resource "arn:aws:lambda:us-  
west-2:123456789012:function:MyFunction" -Tag @{ "Washington" = "Olympia"; "Oregon"  
= "Salem"; "California" = "Sacramento" }
```

- Per i dettagli sull'API, vedere [TagResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMAccountSetting

Il seguente esempio di codice mostra come utilizzare. Get-LMAccountSetting

Strumenti per PowerShell V4

Esempio 1: questo esempio viene visualizzato per confrontare il limite dell'account e l'utilizzo dell'account

```
Get-LMAccountSetting | Select-Object
@{Name="TotalCodeSizeLimit";Expression={$_.AccountLimit.TotalCodeSize}},
@{Name="TotalCodeSizeUsed";Expression={$_.AccountUsage.TotalCodeSize}}
```

Output:

TotalCodeSizeLimit	TotalCodeSizeUsed
80530636800	15078795

- Per i dettagli sull'API, vedere [GetAccountSettings](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMAlias

Il seguente esempio di codice mostra come utilizzare. Get-LMAlias

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera i pesi di configurazione del routing per uno specifico alias della funzione Lambda.

```
Get-LMAlias -FunctionName "MylambdaFunction123" -Name "newlabel1" -Select
RoutingConfig
```

Output:

AdditionalVersionWeights

```
-----  
{[1, 0.6]}
```

- Per i dettagli sull'API, vedere [GetAlias](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMFunctionConcurrency

Il seguente esempio di codice mostra come utilizzare. Get-LMFunctionConcurrency

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene la simultaneità riservata per la funzione Lambda

```
Get-LMFunctionConcurrency -FunctionName "MylambdaFunction123" -Select *
```

Output:

```
ReservedConcurrentExecutions  
-----  
100
```

- Per i dettagli sull'API, vedere [GetFunctionConcurrency](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMFunctionConfiguration

Il seguente esempio di codice mostra come utilizzare. Get-LMFunctionConfiguration

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera la configurazione specifica della versione di una funzione Lambda.

```
Get-LMFunctionConfiguration -FunctionName "MylambdaFunction123" -Qualifier  
"PowershellAlias"
```

Output:

```

CodeSha256           : uW0W0R7z+f0VyLuUg7+/D08hkMFsq0SF4seuyUZJ/R8=
CodeSize             : 1426
DeadLetterConfig      : Amazon.Lambda.Model.DeadLetterConfig
Description           : Version 3 to test Aliases
Environment           : Amazon.Lambda.Model.EnvironmentResponse
FunctionArn           : arn:aws:lambda:us-
east-1:123456789012:function:MylambdaFunction123
                     : PowershellAlias
FunctionName          : MylambdaFunction123
Handler               : lambda_function.launch_instance
KMSKeyArn             :
LastModified          : 2019-12-25T09:52:59.872+0000
LastUpdateStatus      : Successful
LastUpdateStatusReason :
LastUpdateStatusReasonCode :
Layers                : {}
MasterArn             :
MemorySize            : 128
RevisionId            : 5d7de38b-87f2-4260-8f8a-e87280e10c33
Role                  : arn:aws:iam::123456789012:role/service-role/lambda
Runtime               : python3.8
State                 : Active
StateReason           :
StateReasonCode       :
Timeout               : 600
TracingConfig         : Amazon.Lambda.Model.TracingConfigResponse
Version               : 4
VpcConfig             : Amazon.Lambda.Model.VpcConfigDetail

```

- Per i dettagli sull'API, vedere [GetFunctionConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMFunctionList

Il seguente esempio di codice mostra come utilizzare. Get-LMFunctionList

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra tutte le funzioni Lambda con dimensioni di codice ordinate

```

Get-LMFunctionList | Sort-Object -Property CodeSize | Select-Object FunctionName,
RunTime, Timeout, CodeSize

```

Output:

FunctionName	Runtime	Timeout
CodeSize		
-----	-----	-----
test	python2.7	3
243		
MylambdaFunction123	python3.8	600
659		
myfuncpython1	python3.8	303
675		

- Per i dettagli sull'API, vedere [ListFunctions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMPolicy

Il seguente esempio di codice mostra come utilizzare. Get-LMPolicy

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra la policy delle funzioni della funzione Lambda

```
Get-LMPolicy -FunctionName test -Select Policy
```

Output:

```
{"Version":"2012-10-17",      "Id":"default","Statement":
[{"Sid":"xxxx","Effect":"Allow","Principal":
{"Service":"sns.amazonaws.com"},"Action":"lambda:InvokeFunction","Resource":"arn:aws:lambda:
east-1:123456789102:function:test"]}]}
```

- Per i dettagli sull'API, vedere [GetPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMProvisionedConcurrencyConfig

Il seguente esempio di codice mostra come utilizzare. Get-LMProvisionedConcurrencyConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene la configurazione di simultaneità fornita per l'alias specificato della funzione Lambda.

```
C:\>Get-LMProvisionedConcurrencyConfig -FunctionName "MyLambdaFunction123" -
Qualifier "NewAlias1"
```

Output:

```
AllocatedProvisionedConcurrentExecutions : 0
AvailableProvisionedConcurrentExecutions : 0
LastModified                             : 2020-01-15T03:21:26+0000
RequestedProvisionedConcurrentExecutions : 70
Status                                    : IN_PROGRESS
StatusReason                             :
```

- Per i dettagli sull'API, vedere [GetProvisionedConcurrencyConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMProvisionedConcurrencyConfigList

Il seguente esempio di codice mostra come utilizzare. Get-LMProvisionedConcurrencyConfigList

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera l'elenco di configurazioni di simultaneità fornita per una funzione Lambda.

```
Get-LMProvisionedConcurrencyConfigList -FunctionName "MyLambdaFunction123"
```

- Per i dettagli sull'API, vedere [ListProvisionedConcurrencyConfigs](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMResourceTag

Il seguente esempio di codice mostra come utilizzare. Get-LMResourceTag

Strumenti per PowerShell V4

Esempio 1: recupera i tag e i relativi valori attualmente impostati sulla funzione specificata.

```
Get-LMResourceTag -Resource "arn:aws:lambda:us-west-2:123456789012:function:MyFunction"
```

Output:

Key	Value
---	----
California	Sacramento
Oregon	Salem
Washington	Olympia

- Per i dettagli sull'API, vedere [ListTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-LMVersionsByFunction

Il seguente esempio di codice mostra come utilizzare. Get-LMVersionsByFunction

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce l'elenco di configurazioni specifiche della versione per ogni versione della funzione Lambda.

```
Get-LMVersionsByFunction -FunctionName "MylambdaFunction123"
```

Output:

FunctionName	Runtime	MemorySize	Timeout	CodeSize	LastModified
RoleName					
-----	-----	-----	-----	-----	-----

MylambdaFunction123	python3.8	128	600	659	
2020-01-10T03:20:56.390+0000	lambda				
MylambdaFunction123	python3.8	128	5	1426	
2019-12-25T09:19:02.238+0000	lambda				
MylambdaFunction123	python3.8	128	5	1426	
2019-12-25T09:39:36.779+0000	lambda				

```
MyLambdaFunction123 python3.8          128      600      1426
2019-12-25T09:52:59.872+0000 lambda
```

- Per i dettagli sull'API, vedere [ListVersionsByFunction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-LMAlias

Il seguente esempio di codice mostra come utilizzare. New-LMAlias

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un nuovo alias Lambda per la versione e la configurazione di routing specificate per specificare la percentuale di richieste di invocazione ricevute.

```
New-LMAlias -FunctionName "MyLambdaFunction123" -
RoutingConfig_AdditionalVersionWeight @{Name="1";Value="0.6"} -Description "Alias for
version 4" -FunctionVersion 4 -Name "PowershellAlias"
```

- Per i dettagli sull'API, vedere [CreateAlias](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Publish-LMFunction

Il seguente esempio di codice mostra come utilizzare. Publish-LMFunction

Strumenti per PowerShell V4

Esempio 1: Questo esempio crea una nuova funzione C# (dotnetcore1.0 runtime) denominata in MyFunction AWS Lambda, che fornisce i file binari compilati per la funzione da un file zip sul file system locale (è possibile utilizzare percorsi relativi o assoluti). Le funzioni C# Lambda specificano il gestore per la funzione utilizzando la designazione: `:Namespace.AssemblyName.ClassName::MethodName`. È necessario sostituire in modo appropriato le parti relative al nome dell'assembly (senza il suffisso .dll), allo spazio dei nomi, al nome della classe e al nome del metodo delle specifiche dell'handler. La nuova funzione avrà le variabili di ambiente 'envvar1' e 'envvar2' impostate in base ai valori forniti.

```
Publish-LMFunction -Description "My C# Lambda Function" `
-FunctionName MyFunction `
-ZipFilename .\MyFunctionBinaries.zip `
```

```
-Handler "AssemblyName::Namespace.ClassName::MethodName" `
-Role "arn:aws:iam::123456789012:role/LambdaFullExecRole" `
-Runtime dotnetcore1.0 `
-Environment_Variable @{ "envvar1"="value";"envvar2"="value" }
```

Output:

```
CodeSha256      : /NgBMd...gq71I=
CodeSize       : 214784
DeadLetterConfig :
Description     : My C# Lambda Function
Environment     : Amazon.Lambda.Model.EnvironmentResponse
FunctionArn     : arn:aws:lambda:us-west-2:123456789012:function:ToUpper
FunctionName    : MyFunction
Handler        : AssemblyName::Namespace.ClassName::MethodName
KMSKeyArn       :
LastModified    : 2016-12-29T23:50:14.207+0000
MemorySize     : 128
Role           : arn:aws:iam::123456789012:role/LambdaFullExecRole
Runtime        : dotnetcore1.0
Timeout        : 3
Version        : $LATEST
VpcConfig      :
```

Esempio 2: questo esempio è simile a quello precedente, tranne per il fatto che i binari delle funzioni vengono prima caricati in un bucket Amazon S3 (che deve trovarsi nella stessa regione della funzione Lambda prevista) e l'oggetto S3 risultante viene quindi referenziato durante la creazione della funzione.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key MyFunctionBinaries.zip -File .
\MyFunctionBinaries.zip
Publish-LMFunction -Description "My C# Lambda Function" `
-FunctionName MyFunction `
-BucketName amzn-s3-demo-bucket `
-Key MyFunctionBinaries.zip `
-Handler "AssemblyName::Namespace.ClassName::MethodName" `
-Role "arn:aws:iam::123456789012:role/LambdaFullExecRole" `
-Runtime dotnetcore1.0 `
-Environment_Variable @{ "envvar1"="value";"envvar2"="value" }
```

- Per i dettagli sull'API, vedere [CreateFunction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Publish-LMVersion

Il seguente esempio di codice mostra come utilizzare. Publish-LMVersion

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una versione per lo snapshot esistente del codice funzione Lambda

```
Publish-LMVersion -FunctionName "MylambdaFunction123" -Description "Publishing Existing Snapshot of function code as a new version through Powershell"
```

- Per i dettagli sull'API, vedere [PublishVersion](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-LMAlias

Il seguente esempio di codice mostra come utilizzare. Remove-LMAlias

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'alias della funzione Lambda menzionato nel comando.

```
Remove-LMAlias -FunctionName "MylambdaFunction123" -Name "NewAlias"
```

- Per i dettagli sull'API, vedere [DeleteAlias](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-LMFunction

Il seguente esempio di codice mostra come utilizzare. Remove-LMFunction

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina una versione specifica di una funzione Lambda

```
Remove-LMFunction -FunctionName "MylambdaFunction123" -Qualifier '3'
```

- Per i dettagli sull'API, vedere [DeleteFunction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-LMFunctionConcurrency

Il seguente esempio di codice mostra come utilizzare. Remove-LMFunctionConcurrency

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la simultaneità della funzione dalla funzione Lambda.

```
Remove-LMFunctionConcurrency -FunctionName "MylambdaFunction123"
```

- Per i dettagli sull'API, vedere [DeleteFunctionConcurrency](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-LMPermission

Il seguente esempio di codice mostra come utilizzare. Remove-LMPermission

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la politica StatementId della funzione per la funzione Lambda specificata.

```
$policy = Get-LMPolicy -FunctionName "MylambdaFunction123" -Select Policy |  
ConvertFrom-Json | Select-Object -ExpandProperty Statement  
Remove-LMPermission -FunctionName "MylambdaFunction123" -StatementId $policy[0].Sid
```

- Per i dettagli sull'API, vedere [RemovePermission](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-LMProvisionedConcurrencyConfig

Il seguente esempio di codice mostra come utilizzare. Remove-LMProvisionedConcurrencyConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la configurazione della simultaneità fornita per un alias specifico.

```
Remove-LMProvisionedConcurrencyConfig -FunctionName "MylambdaFunction123" -Qualifier  
"NewAlias1"
```

- Per i dettagli sull'API, vedere [DeleteProvisionedConcurrencyConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-LMResourceTag

Il seguente esempio di codice mostra come utilizzare. Remove-LMResourceTag

Strumenti per PowerShell V4

Esempio 1: rimuove i tag forniti da una funzione. Il cmdlet richiederà una conferma prima di procedere, a meno che non venga specificato lo switch -Force. Viene effettuata una sola chiamata al servizio per rimuovere i tag.

```
Remove-LMResourceTag -Resource "arn:aws:lambda:us-west-2:123456789012:function:MyFunction" -TagKey "Washington","Oregon","California"
```

Esempio 2: rimuove i tag forniti da una funzione. Il cmdlet richiederà una conferma prima di procedere, a meno che non venga specificato lo switch -Force. Una volta effettuata la chiamata al servizio per tag fornito.

```
"Washington","Oregon","California" | Remove-LMResourceTag -Resource "arn:aws:lambda:us-west-2:123456789012:function:MyFunction"
```

- Per i dettagli sull'API, vedere [UntagResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-LMAlias

Il seguente esempio di codice mostra come utilizzare. Update-LMAlias

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la configurazione di un alias di funzione Lambda esistente. Aggiorna il RoutingConfiguration valore per spostare il 60% (0,6) del traffico alla versione 1

```
Update-LMAlias -FunctionName "MylambdaFunction123" -Description " Alias for version 2" -FunctionVersion 2 -Name "newlabel1" -RoutingConfig_AdditionalVersionWeight @{Name="1";Value="0.6"}
```

- Per i dettagli sull'API, vedere [UpdateAlias](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-LMFunctionCode

Il seguente esempio di codice mostra come utilizzare. Update-LMFunctionCode

Strumenti per PowerShell V4

Esempio 1: aggiorna la funzione denominata MyFunction " con il nuovo contenuto contenuto nel file zip specificato. Per una funzione Lambda C# .NET Core, il file zip deve contenere l'assembly compilato.

```
Update-LMFunctionCode -FunctionName MyFunction -ZipFilename .\UpdatedCode.zip
```

Esempio 2: questo esempio è simile a quello precedente ma utilizza un oggetto Amazon S3 contenente il codice aggiornato per aggiornare la funzione.

```
Update-LMFunctionCode -FunctionName MyFunction -BucketName amzn-s3-demo-bucket -Key UpdatedCode.zip
```

- Per i dettagli sull'API, vedere [UpdateFunctionCode](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-LMFunctionConfiguration

Il seguente esempio di codice mostra come utilizzare. Update-LMFunctionConfiguration

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la configurazione della funzione Lambda esistente

```
Update-LMFunctionConfiguration -FunctionName "MylambdaFunction123" -Handler "lambda_function.launch_instance" -Timeout 600 -Environment_Variable @{ "envvar1"="value";"envvar2"="value" } -Role arn:aws:iam::123456789101:role/service-role/lambda -DeadLetterConfig_TargetArn arn:aws:sns:us-east-1:123456789101:MyfirstTopic
```

- Per i dettagli sull'API, vedere [UpdateFunctionConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-LMFunctionConcurrency

Il seguente esempio di codice mostra come utilizzare Write-LMFunctionConcurrency

Strumenti per PowerShell V4

Esempio 1: questo esempio applica le impostazioni di simultaneità per l'intera funzione.

```
Write-LMFunctionConcurrency -FunctionName "MylambdaFunction123" -  
ReservedConcurrentExecution 100
```

- Per i dettagli sull'API, vedere [PutFunctionConcurrency](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-LMProvisionedConcurrencyConfig

Il seguente esempio di codice mostra come utilizzare Write-LMProvisionedConcurrencyConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge una configurazione di simultaneità fornita all'alias di una funzione

```
Write-LMProvisionedConcurrencyConfig -FunctionName "MylambdaFunction123" -  
ProvisionedConcurrentExecution 20 -Qualifier "NewAlias1"
```

- Per i dettagli sull'API, vedere [PutProvisionedConcurrencyConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon ML con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon ML.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-MLBatchPrediction

Il seguente esempio di codice mostra come usare `Get-MLBatchPrediction`.

Strumenti per PowerShell V4

Esempio 1: restituisce i metadati dettagliati per una previsione batch con ID id.

```
Get-MLBatchPrediction -BatchPredictionId ID
```

- Per i dettagli sull'API, vedere [GetBatchPrediction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-MLBatchPredictionList

Il seguente esempio di codice mostra come utilizzare `Get-MLBatchPredictionList`.

Strumenti per PowerShell V4

Esempio 1: restituisce un elenco di tutti BatchPredictions i record di dati associati che corrispondono al criterio di ricerca fornito nella richiesta.

```
Get-MLBatchPredictionList
```

Esempio 2: restituisce un elenco di tutti BatchPredictions con lo stato COMPLETATO.

```
Get-MLBatchPredictionList -FilterVariable Status -EQ COMPLETED
```

- Per i dettagli sull'API, vedere [DescribeBatchPredictions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-MLDataSource

Il seguente esempio di codice mostra come utilizzare. Get-MLDataSource

Strumenti per PowerShell V4

Esempio 1: restituisce i metadati, lo stato e le informazioni sul file di dati per a DataSource con l'ID id

```
Get-MLDataSource -DataSourceId ID
```

- Per i dettagli sull'API, vedere [GetDataSource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-MLDataSourceList

Il seguente esempio di codice mostra come utilizzare. Get-MLDataSourceList

Strumenti per PowerShell V4

Esempio 1: restituisce un elenco di tutti DataSources i record di dati associati.

```
Get-MLDataSourceList
```

Esempio 2: restituisce un elenco di tutti DataSources con lo stato COMPLETATO.

```
Get-MLDataDourceList -FilterVariable Status -EQ COMPLETED
```

- Per i dettagli sull'API, vedere [DescribeDataSources](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-MLEvaluation

Il seguente esempio di codice mostra come utilizzare. Get-MLEvaluation

Strumenti per PowerShell V4

Esempio 1: restituisce i metadati e lo stato di una valutazione con ID id.

```
Get-MLEvaluation -EvaluationId ID
```

- Per i dettagli sull'API, vedere [GetEvaluation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-MLEvaluationList

Il seguente esempio di codice mostra come utilizzare. Get-MLEvaluationList

Strumenti per PowerShell V4

Esempio 1: restituisce un elenco di tutte le risorse di valutazione

```
Get-MLEvaluationList
```

Esempio 2: restituisce un elenco di tutte le risorse di valutazione il cui stato è COMPLETED.

```
Get-MLEvaluationList -FilterVariable Status -EQ COMPLETED
```

- Per i dettagli sull'API, vedere [DescribeEvaluations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-MLModel

Il seguente esempio di codice mostra come utilizzare. Get-MLModel

Strumenti per PowerShell V4

Esempio 1: restituisce i metadati dettagliati, lo stato, lo schema e le informazioni sul file di dati per un MLModel with id ID.

```
Get-MLModel -ModelId ID
```

- Per i dettagli sull'API, vedere [Get MLModel](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-MLModelList

Il seguente esempio di codice mostra come utilizzare. Get-MLModelList

Strumenti per PowerShell V4

Esempio 1: restituisce un elenco di tutti i modelli e i relativi record di dati associati.

```
Get-MLModelList
```

Esempio 2: restituisce un elenco di tutti modelli il cui stato è COMPLETED.

```
Get-MLModelList -FilterVariable Status -EQ COMPLETED
```

- Per i dettagli sull'API, vedere [Descrivi MLModels](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-MLPrediction

Il seguente esempio di codice mostra come utilizzare. Get-MLPrediction

Strumenti per PowerShell V4

Esempio 1: invia un record all'URL dell'endpoint di previsione in tempo reale per il modello con ID id.

```
Get-MLPrediction -ModelId ID -PredictEndpoint URL -Record @{"A" = "B"; "C" = "D";}
```

- Per informazioni dettagliate sull'API, consulta [Predict](#) nella documentazione di riferimento dei cmdlet di AWS Strumenti per PowerShell (V4).

New-MLBatchPrediction

Il seguente esempio di codice mostra come usareNew-MLBatchPrediction.

Strumenti per PowerShell V4

Esempio 1: crea una nuova richiesta di previsione batch per il modello con ID id e inserisce l'output nella posizione S3 specificata.

```
New-MLBatchPrediction -ModelId ID -Name NAME -OutputURI s3://...
```

- Per i dettagli sull'API, vedere [CreateBatchPrediction](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-MLDataSourceFromS3

Il seguente esempio di codice mostra come utilizzare. New-MLDataSourceFromS3

Strumenti per PowerShell V4

Esempio 1: crea un'origine dati con i dati per una posizione S3, con il nome NAME e lo schema SCHEMA.

```
New-MLDataSourceFromS3 -Name NAME -ComputeStatistics $true -DataSpec_DataLocationS3  
"s3://BUCKET/KEY" -DataSchema SCHEMA
```

- Per i dettagli sull'API, vedere [CreateDataSourceFromS3](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-MLEvaluation

Il seguente esempio di codice mostra come utilizzare. New-MLEvaluation

Strumenti per PowerShell V4

Esempio 1: crea una valutazione per un determinato ID di origine dati e ID del modello

```
New-MLEvaluation -Name NAME -DataSourceId DSID -ModelId MID
```

- Per i dettagli sull'API, vedere [CreateEvaluation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-MLModel

Il seguente esempio di codice mostra come utilizzare. New-MLModel

Strumenti per PowerShell V4

Esempio 1: crea un nuovo modello con dati di addestramento.

```
New-MLModel -Name NAME -ModelType BINARY -Parameter @{...} -TrainingDataSourceId ID
```

- Per i dettagli sull'API, vedere [Create MLModel](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-MLRealtimeEndpoint

Il seguente esempio di codice mostra come utilizzare. New-MLRealtimeEndpoint

Strumenti per PowerShell V4

Esempio 1: crea un nuovo endpoint di previsione in tempo reale per l'ID modello specificato.

```
New-MLRealtimeEndpoint -ModelId ID
```

- Per i dettagli sull'API, vedere [CreateRealtimeEndpoint](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Macie che utilizzano Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Macie.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-MAC2FindingList

Il seguente esempio di codice mostra come utilizzare. Get-MAC2FindingList

Strumenti per PowerShell V4

Esempio 1: restituisce un elenco di risultati contenenti un rilevamento FindingIds di dati sensibili di tipo «CREDIT_CARD_NUMBER» o «US_SOCIAL_SECURITY_NUMBER»

```
$criterionAddProperties = New-Object
    Amazon.Macie2.Model.CriterionAdditionalProperties

$criterionAddProperties.Eq = @(
    "CREDIT_CARD_NUMBER"
    "US_SOCIAL_SECURITY_NUMBER"
)

$FindingCriterion = @{
    'classificationDetails.result.sensitiveData.detections.type' =
        [Amazon.Macie2.Model.CriterionAdditionalProperties]$criterionAddProperties
}

Get-MAC2FindingList -FindingCriteria_Criterion $FindingCriterion -MaxResult 5
```

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4). [ListFindings](#) AWS Strumenti per PowerShell

Listino prezzi AWS esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. Listino prezzi AWS

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-PLSAttributeValue

Il seguente esempio di codice mostra come utilizzare `Get-PLSAttributeValue`.

Strumenti per PowerShell V4

Esempio 1: restituisce i valori per l'attributo 'VolumeType' per Amazon EC2 nella regione us-east-1.

```
Get-PLSAttributeValue -ServiceCode AmazonEC2 -AttributeName "volumeType" -region us-east-1
```

Output:

```
Value
-----
Cold HDD
General Purpose
Magnetic
Provisioned IOPS
Throughput Optimized HDD
```

- Per i dettagli sull'API, vedere [GetAttributeValues](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-PLSProduct

Il seguente esempio di codice mostra come utilizzare. Get-PLSProduct

Strumenti per PowerShell V4

Esempio 1: dettagli dei resi di tutti i prodotti per Amazon EC2.

```
Get-PLSProduct -ServiceCode AmazonEC2 -Region us-east-1
```

Output:

```
{"product":{"productFamily":"Compute Instance","attributes":
{"enhancedNetworkingSupported":"Yes","memory":"30.5
GiB","dedicatedEbsThroughput":"800 Mbps","vcpu":"4","locationType":"AWS
Region","storage":"EBS only","instanceFamily":"Memory
optimized","operatingSystem":"SUSE","physicalProcessor":"Intel Xeon E5-2686 v4
(Broadwell)","clockSpeed":"2.3 GHz","ecu":"Variable","networkPerformance":"Up
to 10 Gigabit","servicename":"Amazon Elastic Compute
```



```
Cloud", "instanceType": "r4.xlarge", "tenancy": "Shared", "usagetype": "USW2-BoxUsage:r4.xlarge", "normalizationSizeFactor": "8", "processorFeatures": "Intel AVX, Intel AVX2, Intel Turbo", "servicecode": "AmazonEC2", "licenseModel": "No License required", "currentGeneration": "Yes", "preInstalledSw": "NA", "location": "US West (Oregon)", "processorArchitecture": "64-bit", "operation": "RunInstances:000g"}, ...
```

Esempio 2: restituisce i dati per Amazon EC2 nella regione us-east-1 filtrati per tipi di volume «General Purpose» con supporto SSD.

```
Get-PLSProduct -ServiceCode AmazonEC2 -Filter
@{Type="TERM_MATCH";Field="volumeType";Value="General
Purpose"},@{Type="TERM_MATCH";Field="storageMedia";Value="SSD-backed"} -Region us-
east-1
```

Output:

```
{"product":{"productFamily":"Storage","attributes":{"storageMedia":"SSD-
backed","maxThroughputvolume":"160 MB/sec","volumeType":"General
Purpose","maxIopsvolume":"10000"},...
```

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4). [GetProducts](#) AWS Strumenti per PowerShell

Get-PLSService

Il seguente esempio di codice mostra come utilizzare. Get-PLSService

Strumenti per PowerShell V4

Esempio 1: restituisce i metadati per tutti i codici disponibili del servizio nella Regione us-east.

```
Get-PLSService -Region us-east-1
```

Output:

AttributeNames	ServiceCode
-----	-----
{productFamily, servicecode, groupDescription, termType...}	AWSBudgets
{productFamily, servicecode, termType, usagetype...}	AWSCloudTrail
{productFamily, servicecode, termType, usagetype...}	AWSCodeCommit

{productFamily, servicecode, termType, usagetype...}	AWSCodeDeploy
{productFamily, servicecode, termType, usagetype...}	AWSCodePipeline
{productFamily, servicecode, termType, usagetype...}	AWSConfig
...	

Esempio 2: restituisce i metadati per il EC2 servizio Amazon nella regione us-east-1.

```
Get-PLSService -ServiceCode AmazonEC2 -Region us-east-1
```

Output:

AttributeNames	ServiceCode
-----	-----
{volumeType, maxIopsvolume, instanceCapacity10xlarge, locationType...}	AmazonEC2

- Per i dettagli sull'API, vedere [DescribeServices](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Resource Groups con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Resource Groups.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-RGResourceTag

Il seguente esempio di codice mostra come utilizzare Add-RGResourceTag.

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge la chiave di tag “Instances” con il valore “workboxes” all’ARN del gruppo di risorse specificato

```
Add-RGResourceTag -Tag @{Instances="workboxes"} -Arn arn:aws:resource-groups:eu-west-1:123456789012:group/workboxes
```

Output:

Arn	Tags
---	----
arn:aws:resource-groups:eu-west-1:123456789012:group/workboxes	{{Instances, workboxes}}

- Per informazioni dettagliate sull’API, consulta [Tag](#) nella documentazione di riferimento dei cmdlet di AWS Strumenti per PowerShell (V4).

Find-RGResource

Il seguente esempio di codice mostra come usare Find-RGResource.

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un tipo di risorsa ad ResourceQuery esempio con filtri di tag e trova risorse.

```
$query = [Amazon.ResourceGroups.Model.ResourceQuery]::new()
$query.Type = [Amazon.ResourceGroups.QueryType]::TAG_FILTERS_1_0
$query.Query = ConvertTo-Json -Compress -Depth 4 -InputObject @{
    ResourceTypeFilters = @('AWS::EC2::Instance')
    TagFilters = @(
        @{
            Key = 'auto'
            Values = @('no')
        }
    )
}

Find-RGResource -ResourceQuery $query | Select-Object -ExpandProperty ResourceIdentifiers
```

Output:

ResourceArn	ResourceType
-----	-----
arn:aws:ec2:eu-west-1:123456789012:instance/i-0123445b6cb7bd67b	AWS::EC2::Instance

- Per i dettagli sull'API, vedere [SearchResources](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-RGGroup

Il seguente esempio di codice mostra come utilizzare. Get-RGGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera il gruppo di risorse in base al nome del gruppo

```
Get-RGGroup -GroupName auto-no
```

Output:

Description	GroupArn	Name
-----	-----	----
	arn:aws:resource-groups:eu-west-1:123456789012:group/auto-no	auto-no

- Per i dettagli sull'API, vedere [GetGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-RGGroupList

Il seguente esempio di codice mostra come utilizzare. Get-RGGroupList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i gruppi di risorse già creati.

```
Get-RGGroupList
```

Output:

GroupArn	GroupName

-----	-----
arn:aws:resource-groups:eu-west-1:123456789012:group/auto-no	auto-no
arn:aws:resource-groups:eu-west-1:123456789012:group/auto-yes	auto-yes
arn:aws:resource-groups:eu-west-1:123456789012:group/build600	build600

- Per i dettagli sull'API, vedere [ListGroupsin](#) AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-RGGroupQuery

Il seguente esempio di codice mostra come utilizzare. Get-RGGroupQuery

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera la query di risorse per il gruppo di risorse specificato

```
Get-RGGroupQuery -GroupName auto-no | Select-Object -ExpandProperty ResourceQuery
```

Output:

```
Query
      Type
-----
      ----
{"ResourceTypeFilters":["AWS::EC2::Instance"],"TagFilters":[{"Key":"auto","Values":["no"]}]} TAG_FILTERS_1_0
```

- Per i dettagli sull'API, vedere [GetGroupQuery](#)in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-RGGroupResourceList

Il seguente esempio di codice mostra come utilizzare. Get-RGGroupResourceList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le risorse del gruppo filtrate in base al tipo di risorsa

```
Get-RGGroupResourceList -Filter @{Name="resource-type";Values="AWS::EC2::Instance"}
-GroupName auto-yes | Select-Object -ExpandProperty ResourceIdentifiers
```

Output:

ResourceArn	ResourceType
-----	-----
arn:aws:ec2:eu-west-1:123456789012:instance/i-0123bc45b567890e1	AWS::EC2::Instance
arn:aws:ec2:eu-west-1:123456789012:instance/i-0a1caf2345f67d8dc	AWS::EC2::Instance
arn:aws:ec2:eu-west-1:123456789012:instance/i-012e3cb4df567e8aa	AWS::EC2::Instance
arn:aws:ec2:eu-west-1:123456789012:instance/i-0fd12dd3456789012	AWS::EC2::Instance

- Per i dettagli sull'API, vedere [ListGroupResources](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-RGResourceTag

Il seguente esempio di codice mostra come utilizzare. Get-RGResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i tag per l'ARN del gruppo di risorse specificato

```
Get-RGResourceTag -Arn arn:aws:resource-groups:eu-west-1:123456789012:group/workboxes
```

Output:

Key	Value
---	----
Instances	workboxes

- Per i dettagli sull'API, vedere [GetTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-RGGroup

Il seguente esempio di codice mostra come utilizzare. New-RGGroup

Strumenti per PowerShell V4

Esempio 1: Questo esempio crea un nuovo gruppo di AWS risorse Resource Groups basato su tag denominato TestPowerShellGroup. Il gruppo include EC2 istanze Amazon nella regione

corrente contrassegnate con la chiave di tag «Name» e il valore del tag «test2». Il comando restituisce la query e il tipo di gruppo e i risultati dell'operazione.

```
$ResourceQuery = New-Object -TypeName Amazon.ResourceGroups.Model.ResourceQuery
$ResourceQuery.Type = "TAG_FILTERS_1_0"
$ResourceQuery.Query = '{"ResourceTypeFilters":["AWS::EC2::Instance"],"TagFilters":
[{"Key":"Name","Values":["test2"]}]} '
$ResourceQuery

New-RGGroup -Name TestPowerShellGroup -ResourceQuery $ResourceQuery -Description
"Test resource group."
```

Output:

```
Query
-----
Type
-----
{"ResourceTypeFilters":["AWS::EC2::Instance"],"TagFilters":[{"Key":"Name","Values":
["test2"]}]} TAG_FILTERS_1_0

LoggedAt      : 11/20/2018 2:40:59 PM
Group         : Amazon.ResourceGroups.Model.Group
ResourceQuery : Amazon.ResourceGroups.Model.ResourceQuery
Tags          : {}
ResponseMetadata : Amazon.Runtime.ResponseMetadata
ContentLength  : 338
HttpStatusCode : OK
```

- Per i dettagli sull'API, consulta AWS Strumenti per PowerShell Cmdlet [CreateGroup](#)Reference (V4).

Remove-RGGroup

Il seguente esempio di codice mostra come utilizzare. Remove-RGGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il gruppo di risorse denominato

```
Remove-RGGroup -GroupName non-tag-cfn-elbv2
```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-RGGroup (DeleteGroup)" on target "non-tag-cfn-
elbv2".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

Description GroupArn
Name
-----
----
                                arn:aws:resource-groups:eu-west-1:123456789012:group/non-tag-cfn-elbv2
non-tag-cfn-elbv2

```

- Per i dettagli sull'API, vedere [DeleteGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-RGResourceTag

Il seguente esempio di codice mostra come utilizzare. Remove-RGResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il tag menzionato dal gruppo di risorse

```

Remove-RGResourceTag -Arn arn:aws:resource-groups:eu-west-1:123456789012:group/
workboxes -Key Instances

```

Output:

```

Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-RGResourceTag (Untag)" on target "arn:aws:resource-
groups:eu-west-1:933303704102:group/workboxes".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y

Arn                                                                    Keys
---                                                                    ----

```



```
arn:aws:resource-groups:eu-west-1:123456789012:group/workboxes {Instances}
```

- Per informazioni dettagliate sull'API, consulta [Untag](#) nella documentazione di riferimento dei cmdlet di AWS Strumenti per PowerShell (V4).

Update-RGGroup

Il seguente esempio di codice mostra come usare `Update-RGGroup`.

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la descrizione del gruppo

```
Update-RGGroup -GroupName auto-yes -Description "Instances auto-remove"
```

Output:

Description Name	GroupArn
-----	-----

Instances to be cleaned	arn:aws:resource-groups:eu-west-1:123456789012:group/auto-yes auto-yes

- Per i dettagli sull'API, vedere [UpdateGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-RGGroupQuery

Il seguente esempio di codice mostra come utilizzare `Update-RGGroupQuery`.

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un oggetto query e aggiorna la query per il gruppo.

```
$query = [Amazon.ResourceGroups.Model.ResourceQuery]::new()
$query.Type = [Amazon.ResourceGroups.QueryType]::TAG_FILTERS_1_0
$query.Query = @{
    ResourceTypeFilters = @('AWS::EC2::Instance')
```

```
TagFilters = @( @{
    Key='Environment'
    Values='Build600.11'
})
} | ConvertTo-Json -Compress -Depth 4

Update-RGGroupQuery -GroupName build600 -ResourceQuery $query
```

Output:

```
GroupName ResourceQuery
-----
build600  Amazon.ResourceGroups.Model.ResourceQuery
```

- Per i dettagli sull'API, vedere [UpdateGroupQuery](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di API Resource Groups Tagging con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con l'API Resource Groups Tagging.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-RGTResourceTag

Il seguente esempio di codice mostra come utilizzare. Add-RGTResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge le chiavi di tag “stage” e “version” con i valori “beta” e “preprod_test” a un bucket Amazon S3 e una tabella Amazon DynamoDB. Viene effettuata una sola chiamata al servizio per applicare i tag.

```
$arn1 = "arn:aws:s3:::amzn-s3-demo-bucket"
$arn2 = "arn:aws:dynamodb:us-west-2:123456789012:table/mytable"

Add-RGTResourceTag -ResourceARNList $arn1,$arn2 -Tag @{ "stage"="beta";
"version"="preprod_test" }
```

Esempio 2: questo esempio aggiunge i tag e i valori specificati a un bucket Amazon S3 e a una tabella Amazon DynamoDB. Vengono effettuate due chiamate al servizio, una per ogni ARN di risorsa reindirizzato al cmdlet.

```
$arn1 = "arn:aws:s3:::amzn-s3-demo-bucket"
$arn2 = "arn:aws:dynamodb:us-west-2:123456789012:table/mytable"

$arn1,$arn2 | Add-RGTResourceTag -Tag @{ "stage"="beta"; "version"="preprod_test" }
```

- Per i dettagli sull'API, vedere [TagResources](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-RGTResource

Il seguente esempio di codice mostra come utilizzare. Get-RGTResource

Strumenti per PowerShell V4

Esempio 1: restituisce tutte le risorse con tag in una Regione e le chiavi di tag associate alla risorsa. Se non viene fornito alcun parametro -Region al cmdlet, tenterà di dedurre la regione dai metadati della shell o dell'istanza. EC2

```
Get-RGTResource
```

Output:

ResourceARN	Tags
-----	----
arn:aws:dynamodb:us-west-2:123456789012:table/mytable	{stage, version}

```
arn:aws:s3:::amzn-s3-demo-bucket                                {stage,
version, othertag}
```

Esempio 2: restituisce tutte le risorse con tag del tipo specificato in una Regione. La stringa per ogni nome di servizio e tipo di risorsa è la stessa incorporata nel nome della risorsa Amazon (ARN) di una risorsa.

```
Get-RGTRResource -ResourceType "s3"
```

Output:

ResourceARN	Tags
-----	----
arn:aws:s3:::amzn-s3-demo-bucket	{stage,
version, othertag}	

Esempio 3: restituisce tutte le risorse con tag del tipo specificato in una Regione. Nota che quando i tipi di risorse vengono reindirizzati al cmdlet, viene effettuata una chiamata al servizio per ogni tipo di risorsa fornito.

```
"dynamodb","s3" | Get-RGTRResource
```

Output:

ResourceARN	Tags
-----	----
arn:aws:dynamodb:us-west-2:123456789012:table/mytable	{stage, version}
arn:aws:s3:::amzn-s3-demo-bucket	{stage,
version, othertag}	

Esempio 4: restituisce tutte le risorse con tag che corrispondono al filtro specificato.

```
Get-RGTRResource -TagFilter @{ Key="stage" }
```

Output:

ResourceARN	Tags
-----	----
arn:aws:s3:::amzn-s3-demo-bucket	{stage,
version, othertag}	

Esempio 5: restituisce tutte le risorse con tag che corrispondono al filtro e al tipo di risorsa specificati.

```
Get-RGTResource -TagFilter @{ Key="stage" } -ResourceType "dynamodb"
```

Output:

ResourceARN	Tags
-----	----
arn:aws:dynamodb:us-west-2:123456789012:table/mytable	{stage, version}

Esempio 6: restituisce tutte le risorse con tag che corrispondono al filtro specificato.

```
Get-RGTResource -TagFilter @{ Key="stage"; Values=@("beta","gamma") }
```

Output:

ResourceARN	Tags
-----	----
arn:aws:dynamodb:us-west-2:123456789012:table/mytable	{stage, version}

- Per i dettagli sull'API, vedere [GetResources](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-RGTTagKey

Il seguente esempio di codice mostra come utilizzare. Get-RGTTagKey

Strumenti per PowerShell V4

Esempio 1: restituisce tutte le chiavi di tag nella Regione specificata. Se il parametro -Region non è specificato, il cmdlet tenterà di dedurre la regione dalla regione della shell o dai metadati dell'istanza predefiniti. EC2 Nota che le chiavi di tag non vengono restituite in base a un ordine specifico.

```
Get-RGTTagKey -region us-west-2
```

Output:

```
version
```

```
stage
```

- Per i dettagli sull'API, vedere [GetTagKeys](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-RGTTagValue

Il seguente esempio di codice mostra come utilizzare. Get-RGTTagValue

Strumenti per PowerShell V4

Esempio 1: restituisce il valore per il tag specificato in una Regione. Se il parametro -Region non è specificato, il cmdlet tenterà di dedurre la regione dalla regione della shell o dai metadati dell'istanza predefiniti. EC2

```
Get-RGTTagValue -Key "stage" -Region us-west-2
```

Output:

```
beta
```

- Per i dettagli sull'API, vedere [GetTagValues](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-RGTResourceTag

Il seguente esempio di codice mostra come utilizzare. Remove-RGTResourceTag

Strumenti per PowerShell V4

Esempio 1: rimuove le chiavi di tag "stage" e "version" e i valori associati da un bucket Amazon S3 e da una tabella Amazon DynamoDB. Viene effettuata una sola chiamata al servizio per rimuovere i tag. Prima di rimuovere i tag, il cmdlet richiederà una conferma. Per disabilitare la conferma, aggiungi il parametro -Force.

```
$arn1 = "arn:aws:s3:::amzn-s3-demo-bucket"
$arn2 = "arn:aws:dynamodb:us-west-2:123456789012:table/mytable"

Remove-RGTResourceTag -ResourceARNList $arn1,$arn2 -TagKey "stage","version"
```

Esempio 2: rimuove le chiavi di tag “stage” e “version” e i valori associati da un bucket Amazon S3 e da una tabella Amazon DynamoDB. Vengono effettuate due chiamate al servizio, una per ogni ARN di risorsa reindirizzato al cmdlet. Prima di ogni chiamata, il cmdlet richiederà una conferma. Per disabilitare la conferma, aggiungi il parametro -Force.

```
$arn1 = "arn:aws:s3:::amzn-s3-demo-bucket"
$arn2 = "arn:aws:dynamodb:us-west-2:123456789012:table/mytable"

$arn1,$arn2 | Remove-RGTResourceTag -TagKey "stage","version"
```

- Per i dettagli sull'API, vedere [UntagResources](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Route 53 utilizzando Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Route 53.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Edit-R53ResourceRecordSet

Il seguente esempio di codice mostra come utilizzare `Edit-R53ResourceRecordSet`.

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un record A per `www.example.com` e modifica il record A per `test.example.com` da `192.0.2.3` a `192.0.2.1`. Nota che i valori per le modifiche dei record di tipo TXT devono essere racchiusi tra virgolette doppie. Per ulteriori informazioni, consulta

la documentazione di Amazon Route 53. È possibile utilizzare il Get-R53Change cmdlet per eseguire un sondaggio per determinare quando le modifiche sono state completate.

```
$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "www.example.com"
$change1.ResourceRecordSet.Type = "TXT"
$change1.ResourceRecordSet.TTL = 600
$change1.ResourceRecordSet.ResourceRecords.Add(@{Value="item 1 item 2 item 3"})

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "DELETE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change2.ResourceRecordSet.Name = "test.example.com"
$change2.ResourceRecordSet.Type = "A"
$change2.ResourceRecordSet.TTL = 600
$change2.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.3"})

$change3 = New-Object Amazon.Route53.Model.Change
$change3.Action = "CREATE"
$change3.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change3.ResourceRecordSet.Name = "test.example.com"
$change3.ResourceRecordSet.Type = "A"
$change3.ResourceRecordSet.TTL = 600
$change3.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.1"})

$params = @{
    HostedZoneId="Z1PA6795UKMFR9"
    ChangeBatch_Comment="This change batch creates a TXT record for www.example.com.
    and changes the A record for test.example.com. from 192.0.2.3 to 192.0.2.1."
    ChangeBatch_Change=$change1,$change2,$change3
}

Edit-R53ResourceRecordSet @params
```

Esempio 2: questo esempio mostra come creare set di record di risorse alias. “Z222222222” è l’ID della zona ospitata di Amazon Route 53 in cui si sta creando il set di record di risorse alias. “example.com” è l’apex di zona per cui si desidera creare un alias e “www.example.com” è un sottodominio per il quale si desidera creare anche un alias. 'Z11111' è un esempio di ID di zona ospitata per il sistema di bilanciamento del carico e 'example-load-balancer-11.us-east-1.elb.amazonaws.com' è un esempio di nome di dominio del sistema di bilanciamento

del carico con cui Amazon Route 53 risponde alle domande relative a `example.com` e `www.example.com`. Per ulteriori informazioni, consulta la documentazione di Amazon Route 53. È possibile utilizzare il cmdlet per eseguire un sondaggio per determinare quando le modifiche sono state completate. `Get-R53Change`

```
$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change1.ResourceRecordSet.AliasTarget.HostedZoneId = "Z11111111111111"
$change1.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-1111111111.us-east-1.elb.amazonaws.com."
$change1.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $true

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "CREATE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "www.example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change1.ResourceRecordSet.AliasTarget.HostedZoneId = "Z11111111111111"
$change1.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-1111111111.us-east-1.elb.amazonaws.com."
$change1.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $false

$params = @{
    HostedZoneId="Z2222222222"
    ChangeBatch_Comment="This change batch creates two alias resource record sets, one
    for the zone apex, example.com, and one for www.example.com, that both point to
    example-load-balancer-1111111111.us-east-1.elb.amazonaws.com."
    ChangeBatch_Change=$change1,$change2
}

Edit-R53ResourceRecordSet @params
```

Esempio 3: questo esempio crea due record A per `www.example.com`. Un quarto delle volte (1/(1+3)) Amazon Route 53 risponde alle query relative a `www.example.com` con i due valori del primo set di record di risorse (192.0.2.9 e 192.0.2.10). Tre quarti delle volte (3/(1+3)) Amazon Route 53 risponde alle query relative a `www.example.com` con i due valori per il secondo set di record di risorse (192.0.2.11 e 192.0.2.12). Per ulteriori informazioni, consulta la documentazione

di Amazon Route 53. È possibile utilizzare il `Get-R53Change` cmdlet per eseguire un sondaggio per determinare quando le modifiche sono state completate.

```
$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "www.example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.SetIdentifier = "Rack 2, Positions 4 and 5"
$change1.ResourceRecordSet.Weight = 1
$change1.ResourceRecordSet.TTL = 600
$change1.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.9"})
$change1.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.10"})

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "CREATE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change2.ResourceRecordSet.Name = "www.example.com"
$change2.ResourceRecordSet.Type = "A"
$change2.ResourceRecordSet.SetIdentifier = "Rack 5, Positions 1 and 2"
$change2.ResourceRecordSet.Weight = 3
$change2.ResourceRecordSet.TTL = 600
$change2.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.11"})
$change2.ResourceRecordSet.ResourceRecords.Add(@{Value="192.0.2.12"})

$params = @{
    HostedZoneId="Z1PA6795UKMFR9"
    ChangeBatch_Comment="This change creates two weighted resource record sets, each
of which has two values."
    ChangeBatch_Change=$change1,$change2
}

Edit-R53ResourceRecordSet @params
```

Esempio 4: Questo esempio mostra come creare set di record di risorse alias ponderati presupponendo che `example.com` sia il dominio per il quale si desidera creare set di record di risorse alias ponderati. `SetIdentifier` differenzia i due set di record di risorse alias ponderati l'uno dall'altro. Questo elemento è obbligatorio perché gli elementi `Name` e `Type` hanno gli stessi valori per entrambi i set di record delle risorse. `Z11111` e `Z33333` sono esempi di zona ospitata IDs per il sistema di bilanciamento del carico ELB specificato dal valore di. `DNSName` `example-load-balancer-2222222222.us-east-1.elb.amazonaws.com` e `example-load-balancer-4444444444.us-east-1.elb.amazonaws.com` sono esempi di domini Elastic Load Balancing da cui

Amazon Route 53 risponde alle domande per example.com. Per ulteriori informazioni, consulta la documentazione di Amazon Route 53. È possibile utilizzare il cmdlet per eseguire un sondaggio per determinare quando le Get-R53Change modifiche sono state completate.

```
$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.SetIdentifier = "1"
$change1.ResourceRecordSet.Weight = 3
$change1.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change1.ResourceRecordSet.AliasTarget.HostedZoneId = "Z11111111111111"
$change1.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-2222222222.us-east-1.elb.amazonaws.com."
$change1.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $true

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "CREATE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change2.ResourceRecordSet.Name = "example.com"
$change2.ResourceRecordSet.Type = "A"
$change2.ResourceRecordSet.SetIdentifier = "2"
$change2.ResourceRecordSet.Weight = 1
$change2.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change2.ResourceRecordSet.AliasTarget.HostedZoneId = "Z33333333333333"
$change2.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-4444444444.us-east-1.elb.amazonaws.com."
$change2.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $false

$params = @{
    HostedZoneId="Z5555555555"
    ChangeBatch_Comment="This change batch creates two weighted alias resource
record sets. Amazon Route 53 responds to queries for example.com with the first ELB
domain 3/4ths of the times and the second one 1/4th of the time."
    ChangeBatch_Change=$change1,$change2
}

Edit-R53ResourceRecordSet @params
```

Esempio 5: questo esempio crea due set di record di risorse con alias con latenza, uno per un bilanciatore del carico ELB nella Regione Stati Uniti occidentali (Oregon) (us-west-2) e un altro

per un bilanciatore del carico nella Regione Asia Pacifico (Singapore) (ap-southeast-1). Per ulteriori informazioni, consulta la documentazione di Amazon Route 53. È possibile utilizzare il Get-R53Change cmdlet per eseguire un sondaggio per determinare quando le modifiche sono state completate.

```
$change1 = New-Object Amazon.Route53.Model.Change
$change1.Action = "CREATE"
$change1.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change1.ResourceRecordSet.Name = "example.com"
$change1.ResourceRecordSet.Type = "A"
$change1.ResourceRecordSet.SetIdentifier = "Oregon load balancer 1"
$change1.ResourceRecordSet.Region = us-west-2
$change1.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change1.ResourceRecordSet.AliasTarget.HostedZoneId = "Z11111111111111"
$change1.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-2222222222.us-west-2.elb.amazonaws.com"
$change1.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $true

$change2 = New-Object Amazon.Route53.Model.Change
$change2.Action = "CREATE"
$change2.ResourceRecordSet = New-Object Amazon.Route53.Model.ResourceRecordSet
$change2.ResourceRecordSet.Name = "example.com"
$change2.ResourceRecordSet.Type = "A"
$change2.ResourceRecordSet.SetIdentifier = "Singapore load balancer 1"
$change2.ResourceRecordSet.Region = ap-southeast-1
$change2.ResourceRecordSet.AliasTarget = New-Object Amazon.Route53.Model.AliasTarget
$change2.ResourceRecordSet.AliasTarget.HostedZoneId = "Z22222222222222"
$change2.ResourceRecordSet.AliasTarget.DNSName = "example-load-
balancer-1111111111.ap-southeast-1.elb.amazonaws.com"
$change2.ResourceRecordSet.AliasTarget.EvaluateTargetHealth = $true

$params = @{
    HostedZoneId="Z555555555555"
    ChangeBatch_Comment="This change batch creates two latency resource record
sets, one for the US West (Oregon) region and one for the Asia Pacific (Singapore)
region."
    ChangeBatch_Change=$change1,$change2
}

Edit-R53ResourceRecordSet @params
```

- Per i dettagli sull'API, vedere [ChangeResourceRecordSets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-R53AccountLimit

Il seguente esempio di codice mostra come utilizzare. Get-R53AccountLimit

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il numero massimo di zone ospitate che possono essere create utilizzando l'account corrente.

```
Get-R53AccountLimit -Type MAX_HOSTED_ZONES_BY_OWNER
```

Output:

```
15
```

- Per i dettagli sull'API, vedere [GetAccountLimit](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-R53CheckerIpRanges

Il seguente esempio di codice mostra come utilizzare. Get-R53CheckerIpRanges

Strumenti per PowerShell V4

Esempio 1: Questo esempio restituisce il valore CIDRs per i controllori sanitari di Route53

```
Get-R53CheckerIpRanges
```

Output:

```
15.177.2.0/23  
15.177.6.0/23  
15.177.10.0/23  
15.177.14.0/23  
15.177.18.0/23  
15.177.22.0/23
```

```
15.177.26.0/23
15.177.30.0/23
15.177.34.0/23
15.177.38.0/23
15.177.42.0/23
15.177.46.0/23
15.177.50.0/23
15.177.54.0/23
15.177.58.0/23
15.177.62.0/23
54.183.255.128/26
54.228.16.0/26
54.232.40.64/26
54.241.32.64/26
54.243.31.192/26
54.244.52.192/26
54.245.168.0/26
54.248.220.0/26
54.250.253.192/26
54.251.31.128/26
54.252.79.128/26
54.252.254.192/26
54.255.254.192/26
107.23.255.0/26
176.34.159.192/26
177.71.207.128/26
```

- Per i dettagli sull'API, vedere [GetCheckerIpRanges](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-R53HostedZone

Il seguente esempio di codice mostra come utilizzare. Get-R53HostedZone

Strumenti per PowerShell V4

Esempio 1: restituisce i dettagli della zona ospitata con ID PJN98 FT9 Z1D633.

```
Get-R53HostedZone -Id Z1D633PJN98FT9
```

- Per i dettagli sull'API, vedere [GetHostedZone](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-R53HostedZoneCount

Il seguente esempio di codice mostra come utilizzare. Get-R53HostedZoneCount

Strumenti per PowerShell V4

Esempio 1: restituisce il numero totale di zone ospitate pubbliche e private per la versione corrente Account AWS.

```
Get-R53HostedZoneCount
```

- Per i dettagli sull'API, vedere [GetHostedZoneCount](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-R53HostedZoneLimit

Il seguente esempio di codice mostra come utilizzare. Get-R53HostedZoneLimit

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il limite del numero massimo di record che è possibile creare nella zona ospitata specificata.

```
Get-R53HostedZoneLimit -HostedZoneId Z3MEQ8T7HAAAAF -Type MAX_RRSETS_BY_ZONE
```

Output:

```
5
```

- Per i dettagli sull'API, vedere [GetHostedZoneLimit](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-R53HostedZoneList

Il seguente esempio di codice mostra come utilizzare. Get-R53HostedZoneList

Strumenti per PowerShell V4

Esempio 1: restituisce tutte le zone ospitate pubbliche e private.

```
Get-R53HostedZoneList
```

Esempio 2: restituisce tutte le zone ospitate associate al set di deleghe riutilizzabile con l'ID X2CISAMPLE NZ8

```
Get-R53HostedZoneList -DelegationSetId NZ8X2CISAMPLE
```

- Per i dettagli sull'API, vedere [ListHostedZones](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-R53HostedZonesByName

Il seguente esempio di codice mostra come utilizzare. Get-R53HostedZonesByName

Strumenti per PowerShell V4

Esempio 1: restituisce tutte le zone ospitate pubbliche e private in ordine ASCII per nome di dominio.

```
Get-R53HostedZonesByName
```

Esempio 2: restituisce le zone ospitate pubbliche e private in ordine ASCII per nome di dominio a partire dal nome DNS specificato.

```
Get-R53HostedZonesByName -DnsName example2.com
```

- Per i dettagli sull'API, vedere [ListHostedZonesByName](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-R53QueryLoggingConfigList

Il seguente esempio di codice mostra come utilizzare. Get-R53QueryLoggingConfigList

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce tutte le configurazioni per la registrazione di query DNS associate all' Account AWS corrente.


```
Get-R53QueryLoggingConfigList
```

Output:

Id	HostedZoneId	CloudWatchLogsLogGroupArn
59b0fa33-4fea-4471-a88c-926476aaa40d	Z385PDS6EAAAZR	arn:aws:logs:us-east-1:111111111112:log-group:/aws/route53/example1.com:*
ee528e95-4e03-4fdc-9d28-9e24ddaaa063	Z94SJHBV1AAAAZ	arn:aws:logs:us-east-1:111111111112:log-group:/aws/route53/example2.com:*
e38dddda-ceb6-45c1-8cb7-f0ae56aaaa2b	Z3MEQ8T7AAA1BF	arn:aws:logs:us-east-1:111111111112:log-group:/aws/route53/example3.com:*

- Per i dettagli sull'API, vedere [ListQueryLoggingConfigs](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-R53ReusableDelegationSet

Il seguente esempio di codice mostra come utilizzare. `Get-R53ReusableDelegationSet`

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera le informazioni sul set di deleghe specificato, inclusi i quattro server dei nomi assegnati al set di deleghe.

```
Get-R53ReusableDelegationSet -Id N23DS9X4AYEAAA
```

Output:

Id	CallerReference	NameServers
/delegationset/N23DS9X4AYEAAA	testcaller	{ns-545.awsdns-04.net, ns-1264.awsdns-30.org, ns-2004.awsdns-58.co.uk, ns-240.awsdns-30.com}

- Per i dettagli sull'API, vedere [GetReusableDelegationSet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-R53HostedZone

Il seguente esempio di codice mostra come utilizzare. `New-R53HostedZone`

Strumenti per PowerShell V4

Esempio 1: crea una nuova zona ospitata denominata “example.com”, associata a un set di deleghe riutilizzabile. Si noti che è necessario fornire un valore per il CallerReference parametro in modo che le richieste debbano essere ritentate, se necessario, senza il rischio di eseguire l'operazione due volte. Poiché la zona ospitata viene creata in un VPC, è automaticamente privata e non è necessario impostare il parametro - HostedZoneConfig_PrivateZone .

```
$params = @{
    Name="example.com"
    CallerReference="myUniqueIdentifier"
    HostedZoneConfig_Comment="This is my first hosted zone"
    DelegationSetId="NZ8X2CISAMPLE"
    VPC_VPCId="vpc-1a2b3c4d"
    VPC_VPCRegion="us-east-1"
}

New-R53HostedZone @params
```

- Per i dettagli sull'API, vedere [CreateHostedZone](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-R53QueryLoggingConfig

Il seguente esempio di codice mostra come utilizzare. New-R53QueryLoggingConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova configurazione di registrazione dei log delle query DNS di Route 53 per la zona ospitata specificata. Amazon Route53 pubblicherà i log delle query DNS nel gruppo di log Cloudwatch specificato.

```
New-R53QueryLoggingConfig -HostedZoneId Z3MEQ8T7HAAAAF -CloudWatchLogsLogGroupArn
arn:aws:logs:us-east-1:111111111111:log-group:/aws/route53/example.com:*
```

Output:

QueryLoggingConfig	Location
-----	-----

```
Amazon.Route53.Model.QueryLoggingConfig https://route53.amazonaws.com/2013-04-01/
queryloggingconfig/ee5aaa95-4e03-4fdc-9d28-9e24ddaaaaa3
```

- Per i dettagli sull'API, vedere [CreateQueryLoggingConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-R53ReusableDelegationSet

Il seguente esempio di codice mostra come utilizzare. New-R53ReusableDelegationSet

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un set di deleghe riutilizzabile di 4 server dei nomi che possono essere riutilizzati da più zone ospitate.

```
New-R53ReusableDelegationSet -CallerReference testcallerreference
```

Output:

DelegationSet	Location
-----	-----
Amazon.Route53.Model.DelegationSet	https://route53.amazonaws.com/2013-04-01/
delegationset/N23DS9XAAAAAXM	

- Per i dettagli sull'API, vedere [CreateReusableDelegationSet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-R53VPCWithHostedZone

Il seguente esempio di codice mostra come utilizzare. Register-R53VPCWithHostedZone

Strumenti per PowerShell V4

Esempio 1: questo esempio associa il VPC specificato alla zona ospitata privata.

```
Register-R53VPCWithHostedZone -HostedZoneId Z3MEQ8T7HAAAAF -VPC_VPCId vpc-f1b9aaaa -
VPC_VPCRegion us-east-1
```

Output:

Id	Status	SubmittedAt	Comment
--	-----	-----	-----
/change/C3SCAAA633Z6DX	PENDING	01/28/2020 19:32:02	

- Per i dettagli sull'API, vedere [Associate VPCWith HostedZone](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-R53HostedZone

Il seguente esempio di codice mostra come utilizzare. Remove-R53HostedZone

Strumenti per PowerShell V4

Esempio 1: elimina la zona ospitata con l'ID specificato. Verrà richiesta una conferma prima di procedere con il comando, a meno che non sia stato aggiunto il parametro -Force.

```
Remove-R53HostedZone -Id Z1PA6795UKMFR9
```

- Per i dettagli sull'API, vedere [DeleteHostedZone](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-R53QueryLoggingConfig

Il seguente esempio di codice mostra come utilizzare. Remove-R53QueryLoggingConfig

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove la configurazione specificata per la registrazione dei log delle query DNS.

```
Remove-R53QueryLoggingConfig -Id ee528e95-4e03-4fdc-9d28-9e24daaa20063
```

- Per i dettagli sull'API, vedere [DeleteQueryLoggingConfig](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-R53ReusableDelegationSet

Il seguente esempio di codice mostra come utilizzare. Remove-R53ReusableDelegationSet

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il set di deleghe riutilizzabile specificato.

```
Remove-R53ReusableDelegationSet -Id N23DS9X4AYAAAM
```

- Per i dettagli sull'API, vedere [DeleteReusableDelegationSet](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-R53VPCFromHostedZone

Il seguente esempio di codice mostra come utilizzare. Unregister-R53VPCFromHostedZone

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla l'associazione tra il VPC specificato e la zona ospitata privata.

```
Unregister-R53VPCFromHostedZone -HostedZoneId Z3MEQ8T7HAAAAF -VPC_VPCId vpc-f1b9aaaa  
-VPC_VPCRegion us-east-1
```

Output:

Id	Status	SubmittedAt	Comment
--	-----	-----	-----
/change/C2XFCAAAA9HKZG	PENDING	01/28/2020 10:35:55	

- Per i dettagli sull'API, vedere [Disassociate VPCFrom HostedZone](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-R53HostedZoneComment

Il seguente esempio di codice mostra come utilizzare. Update-R53HostedZoneComment

Strumenti per PowerShell V4

Esempio 1: questo comando aggiorna il commento della zona ospitata specificata.

```
Update-R53HostedZoneComment -Id Z385PDS6AAAAAR -Comment "This is my first hosted  
zone"
```

Output:

```
Id                : /hostedzone/Z385PDS6AAAAAR
Name              : example.com.
CallerReference   : C5B55555-7147-EF04-8341-69131E805C89
Config           : Amazon.Route53.Model.HostedZoneConfig
ResourceRecordSetCount : 9
LinkedService     :
```

- Per i dettagli sull'API, vedere [UpdateHostedZoneComment](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon S3 con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon S3.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Copy-S3Object

Il seguente esempio di codice mostra come usare. Copy-S3Object

Strumenti per PowerShell V4

Esempio 1: questo comando copia l'oggetto "sample.txt" dal bucket "test-files" nello stesso bucket ma con una nuova chiave "sample-copy.txt".

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -DestinationKey
sample-copy.txt
```

Esempio 2: questo comando copia l'oggetto "sample.txt" dal bucket "test-files" nel bucket "backup-files" con una chiave "sample-copy.txt".

```
Copy-S3Object -BucketName amzn-s3-demo-source-bucket -Key sample.txt -DestinationKey sample-copy.txt -DestinationBucket amzn-s3-demo-destination-bucket
```

Esempio 3: questo comando scarica l'oggetto "sample.txt" dal bucket "test-files" in un file locale denominato "local-sample.txt".

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -LocalFile local-sample.txt
```

Esempio 4: scarica un unico oggetto nel file specificato. Il file scaricato si trova in c:\downloads\data\archive.zip

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key data/archive.zip -LocalFolder c:\downloads
```

Esempio 5: scarica tutti gli oggetti che corrispondono al prefisso di chiave specificato nella cartella locale. La corrispondente gerarchia di chiavi verrà conservata come sottocartelle nella posizione generale di scaricamento.

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix data -LocalFolder c:\downloads
```

- Per i dettagli sull'API, vedere [CopyObject](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3ACL

Il seguente esempio di codice mostra come utilizzare Get-S3ACL

Strumenti per PowerShell V4

Esempio 1: questo comando ottiene i dettagli del proprietario dell'oggetto S3.

```
Get-S3ACL -BucketName 'amzn-s3-demo-bucket' -key 'initialize.ps1' -Select AccessControlList.Owner
```

Output:

```
DisplayName Id
----- --
testusername 9988776a6554433d22f1100112e334acb45566778899009e9887bd7f66c5f544
```

- Per informazioni dettagliate sull'API, consulta [GetACL](#) nella documentazione di riferimento dei cmdlet di AWS Strumenti per PowerShell (V4).

Get-S3Bucket

Il seguente esempio di codice mostra come usare `Get-S3Bucket`.

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce tutti i bucket S3.

```
Get-S3Bucket
```

Esempio 2: questo comando restituisce un bucket denominato "test-files"

```
Get-S3Bucket -BucketName amzn-s3-demo-bucket
```

- Per i dettagli sull'API, vedere [ListBuckets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketAccelerateConfiguration

Il seguente esempio di codice mostra come utilizzare `Get-S3BucketAccelerateConfiguration`

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce il valore `Enabled` se le impostazioni di accelerazione del trasferimento sono abilitate per il bucket specificato.

```
Get-S3BucketAccelerateConfiguration -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Value
```



```
-----  
Enabled
```

- Per i dettagli sull'API, vedere [GetBucketAccelerateConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketAnalyticsConfiguration

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketAnalyticsConfiguration
Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i dettagli del filtro di analisi denominato “testfilter” nel bucket S3 specificato.

```
Get-S3BucketAnalyticsConfiguration -BucketName 'amzn-s3-demo-bucket' -AnalyticsId  
'testfilter'
```

- Per i dettagli sull'API, vedere [GetBucketAnalyticsConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketAnalyticsConfigurationList

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketAnalyticsConfigurationList

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i dettagli delle prime 100 configurazioni di analisi del bucket S3 specificato.

```
Get-S3BucketAnalyticsConfigurationList -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [ListBucketAnalyticsConfigurations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketEncryption

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketEncryption

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce tutte le regole di crittografia lato server associate al bucket specificato.

```
Get-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [GetBucketEncryption](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketInventoryConfiguration

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketInventoryConfiguration

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i dettagli dell'inventario denominato "testinventory" per il bucket S3 specificato.

```
Get-S3BucketInventoryConfiguration -BucketName 'amzn-s3-demo-bucket' -InventoryId  
'testinventory'
```

- Per i dettagli sull'API, vedere [GetBucketInventoryConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketInventoryConfigurationList

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketInventoryConfigurationList

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce le prime 100 configurazioni dell'inventario del bucket S3 specificato.

```
Get-S3BucketInventoryConfigurationList -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [ListBucketInventoryConfigurations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketLocation

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketLocation

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce il vincolo di posizione eventualmente presente per il bucket "amzn-s3-demo-bucket".

```
Get-S3BucketLocation -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Value
-----
ap-south-1
```

- Per i dettagli sull'API, vedere [GetBucketLocation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketLogging

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketLogging

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce lo stato di registrazione per il bucket specificato.

```
Get-S3BucketLogging -BucketName 'amzn-s3-demo-bucket'
```

Output:

TargetBucketName	Grants	TargetPrefix
-----	-----	-----
testbucket1	{}	testprefix

- Per i dettagli sull'API, vedere [GetBucketLogging](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketMetricsConfiguration

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketMetricsConfiguration
Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i dettagli sul filtro delle metriche denominato “testfilter” per il bucket S3 specificato.

```
Get-S3BucketMetricsConfiguration -BucketName 'amzn-s3-demo-bucket' -MetricsId  
'testfilter'
```

- Per i dettagli sull'API, vedere [GetBucketMetricsConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketNotification

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketNotification
Strumenti per PowerShell V4

Esempio 1: questo esempio recupera la configurazione di notifica del bucket specificato

```
Get-S3BucketNotification -BucketName amzn-s3-demo-bucket | select -ExpandProperty  
TopicConfigurations
```

Output:

Id	Topic
--	-----
mimo	arn:aws:sns:eu-west-1:123456789012:topic-1

- Per i dettagli sull'API, vedere [GetBucketNotification](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketPolicy

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketPolicy

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce la policy di bucket associata al bucket S3 specificato.

```
Get-S3BucketPolicy -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [GetBucketPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketPolicyStatus

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketPolicyStatus

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce lo stato delle policy per il bucket S3 specificato, ovvero indica se il bucket è pubblico.

```
Get-S3BucketPolicyStatus -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [GetBucketPolicyStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketReplication

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketReplication

Strumenti per PowerShell V4

Esempio 1: restituisce le informazioni della configurazione di replica impostate nel bucket denominato "amzn-s3-demo-bucket".

```
Get-S3BucketReplication -BucketName amzn-s3-demo-bucket
```

- Per i dettagli sull'API, vedere [GetBucketReplication](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketRequestPayment

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketRequestPayment

Strumenti per PowerShell V4

Esempio 1: restituisce la configurazione di pagamento della richiesta per il bucket denominato "amzn-s3-demo-bucket". Per impostazione predefinita, il proprietario del bucket paga gli scaricamenti dal bucket.

```
Get-S3BucketRequestPayment -BucketName amzn-s3-demo-bucket
```

- Per i dettagli sull'API, vedere [GetBucketRequestPayment](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketTagging

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketTagging

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce tutti i tag associati al bucket specificato.

```
Get-S3BucketTagging -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [GetBucketTagging](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketVersioning

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketVersioning

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce lo stato del controllo delle versioni in relazione al bucket specificato.

```
Get-S3BucketVersioning -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [GetBucketVersioning](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3BucketWebsite

Il seguente esempio di codice mostra come utilizzare. Get-S3BucketWebsite

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i dettagli delle configurazioni dei siti web statici del bucket S3 specificato.

```
Get-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [GetBucketWebsite](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3CORSConfiguration

Il seguente esempio di codice mostra come utilizzare. Get-S3CORSConfiguration

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce un oggetto contenente tutte le regole di configurazione CORS corrispondenti al bucket S3 specificato.

```
Get-S3CORSConfiguration -BucketName 'amzn-s3-demo-bucket' -Select  
Configuration.Rules
```

Output:

```
AllowedMethods : {PUT, POST, DELETE}  
AllowedOrigins : {http://www.example1.com}  
Id             :  
ExposeHeaders  : {}  
MaxAgeSeconds  : 0  
AllowedHeaders : {*}  
  
AllowedMethods : {PUT, POST, DELETE}  
AllowedOrigins : {http://www.example2.com}  
Id             :  
ExposeHeaders  : {}  
MaxAgeSeconds  : 0  
AllowedHeaders : {*}
```

```
AllowedMethods : {GET}
AllowedOrigins  : {*}
Id             :
ExposeHeaders  : {}
MaxAgeSeconds  : 0
AllowedHeaders : {}
```

- Per i dettagli sull'API, vedere [Get CORSConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3LifecycleConfiguration

Il seguente esempio di codice mostra come utilizzare. Get-S3LifecycleConfiguration

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera la configurazione del ciclo di vita per il bucket.

```
Get-S3LifecycleConfiguration -BucketName amzn-s3-demo-bucket
```

Output:

```
Rules
-----
{Remove-in-150-days, Archive-to-Glacier-in-30-days}
```

- Per i dettagli sull'API, vedere [GetLifecycleConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3Object

Il seguente esempio di codice mostra come utilizzare. Get-S3Object

Strumenti per PowerShell V4

Esempio 1: questo comando recupera le informazioni su tutti gli elementi presenti nel bucket "test-files".

```
Get-S3Object -BucketName amzn-s3-demo-bucket
```


Esempio 2: questo comando recupera le informazioni sull'elemento "sample.txt" dal bucket "test-files".

```
Get-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt
```

Esempio 3: questo comando recupera le informazioni su tutti gli elementi con il prefisso "sample" dal bucket "test-files".

```
Get-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix sample
```

- Per i dettagli sull'API, vedere [ListObjects](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3ObjectLockConfiguration

Il seguente esempio di codice mostra come utilizzare. Get-S3ObjectLockConfiguration

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce il valore "Enabled" se la configurazione Object lock è abilitata per il bucket S3 specificato.

```
Get-S3ObjectLockConfiguration -BucketName 'amzn-s3-demo-bucket' -Select  
ObjectLockConfiguration.ObjectLockEnabled
```

Output:

```
Value  
-----  
Enabled
```

- Per i dettagli sull'API, vedere [GetObjectLockConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3ObjectMetadata

Il seguente esempio di codice mostra come utilizzare. Get-S3ObjectMetadata

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i metadati dell'oggetto con la chiave 'ListTrusts.txt' nel bucket S3 specificato.

```
Get-S3ObjectMetadata -BucketName 'amzn-s3-demo-bucket' -Key 'ListTrusts.txt'
```

Output:

```
Headers                : Amazon.S3.Model.HeadersCollection
Metadata               : Amazon.S3.Model.MetadataCollection
DeleteMarker           :
AcceptRanges           : bytes
ContentRange           :
Expiration              :
RestoreExpiration       :
RestoreInProgress      : False
LastModified           : 01/01/2020 08:02:05
ETag                   : "d000011112a222e333e3bb4ee5d43d21"
MissingMeta            : 0
VersionId              : null
Expires                : 01/01/0001 00:00:00
WebsiteRedirectLocation :
ServerSideEncryptionMethod : AES256
ServerSideEncryptionCustomerMethod :
ServerSideEncryptionKeyManagementServiceKeyId :
ReplicationStatus      :
PartsCount             :
ObjectLockLegalHoldStatus :
ObjectLockMode         :
ObjectLockRetainUntilDate : 01/01/0001 00:00:00
StorageClass           :
RequestCharged         :
```

- Per i dettagli sull'API, vedere [GetObjectMetadata](#) in Cmdlet Reference (V4).AWS Strumenti per PowerShell

Get-S3ObjectRetention

Il seguente esempio di codice mostra come utilizzare. Get-S3ObjectRetention

Strumenti per PowerShell V4

Esempio 1: il comando restituisce la modalità e la data relative alla conservazione dell'oggetto.

```
Get-S3ObjectRetention -BucketName 'amzn-s3-demo-bucket' -Key 'testfile.txt'
```

- Per i dettagli sull'API, vedere [GetObjectRetention](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3ObjectTagSet

Il seguente esempio di codice mostra come utilizzare. Get-S3ObjectTagSet

Strumenti per PowerShell V4

Esempio 1: l'esempio restituisce i tag associati all'oggetto presente nel bucket S3 specificato.

```
Get-S3ObjectTagSet -Key 'testfile.txt' -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Key  Value
---  -
test value
```

- Per i dettagli sull'API, vedere [GetObjectTagging](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3PreSignedURL

Il seguente esempio di codice mostra come utilizzare. Get-S3PreSignedURL

Strumenti per PowerShell V4

Esempio 1: il comando restituisce un URL prefirmato per una chiave e una data di scadenza specificate.

```
Get-S3PreSignedURL -BucketName 'amzn-s3-demo-bucket' -Key 'testkey' -Expires  
'2023-11-16'
```

Esempio 2: il comando restituisce un URL prefirmato per un bucket di directory con una chiave e una data di scadenza specificate.

```
[Amazon.AWSConfigsS3]::UseSignatureVersion4 = $true
Get-S3PreSignedURL -BucketName amzn-s3-demo-bucket--usw2-az1--x-s3 -Key
'testkey' -Expire '2023-11-17'
```

- Per i dettagli sull'API, vedere [GetPreSignedURL](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3PublicAccessBlock

Il seguente esempio di codice mostra come utilizzare. Get-S3PublicAccessBlock

Strumenti per PowerShell V4

Esempio 1: il comando restituisce la configurazione del blocco dell'accesso pubblico del bucket S3 specificato.

```
Get-S3PublicAccessBlock -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [GetPublicAccessBlock](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-S3Version

Il seguente esempio di codice mostra come utilizzare. Get-S3Version

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i metadati relativi a tutte le versioni degli oggetti nel bucket S3 specificato.

```
Get-S3Version -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
IsTruncated      : False
KeyMarker        :
VersionIdMarker  :
```

```

NextKeyMarker      :
NextVersionIdMarker :
Versions           : {EC2.txt, EC2MicrosoftWindowsGuide.txt, ListDirectories.json,
  ListTrusts.json}
Name               : amzn-s3-demo-bucket
Prefix            :
MaxKeys           : 1000
CommonPrefixes    : {}
Delimiter         :

```

- Per i dettagli sull'API, vedere [ListVersions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-S3Bucket

Il seguente esempio di codice mostra come utilizzare `New-S3Bucket`

Strumenti per PowerShell V4

Esempio 1: questo comando crea un nuovo bucket privato denominato “sample-bucket”.

```
New-S3Bucket -BucketName amzn-s3-demo-bucket
```

Esempio 2: questo comando crea un nuovo bucket denominato “sample-bucket” con autorizzazioni di lettura-scrittura.

```
New-S3Bucket -BucketName amzn-s3-demo-bucket -PublicReadWrite
```

Esempio 3: questo comando crea un nuovo bucket denominato “sample-bucket” con autorizzazioni di sola lettura.

```
New-S3Bucket -BucketName amzn-s3-demo-bucket -PublicReadOnly
```

Esempio 4: Questo comando crea un nuovo bucket di directory denominato «amzn-s3-demo-bucket--use1-az5--x-s3” con `PutBucketConfiguration`

```

$bucketConfiguration = @{
    BucketInfo = @{
        DataRedundancy = 'SingleAvailabilityZone'
        Type = 'Directory'
    }
}

```

```

    }
    Location = @{
        Name = 'usw2-az1'
        Type = 'AvailabilityZone'
    }
}
New-S3Bucket -BucketName amzn-s3-demo-bucket--usw2-az1--x-s3 -BucketConfiguration
$bucketConfiguration -Region us-west-2

```

- Per i dettagli [PutBucket](#) sull'AWS Strumenti per PowerShell API, vedere in Cmdlet Reference (V4).

Read-S3Object

Il seguente esempio di codice mostra come utilizzare. Read-S3Object

Strumenti per PowerShell V4

Esempio 1: questo comando recupera l'elemento "sample.txt" dal bucket "test-files" e lo salva in un file denominato "local-sample.txt" nella posizione corrente. Il file "local-sample.txt" non deve esistere prima di chiamare questo comando.

```
Read-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -File local-sample.txt
```

Esempio 2: questo comando recupera la directory virtuale "DIR" dal bucket "test-files" e la salva in una cartella denominata "Local-DIR" nella posizione corrente. La cartella "Local-DIR" non deve esistere prima di chiamare questo comando.

```
Read-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix DIR -Folder Local-DIR
```

Esempio 3: scarica tutti gli oggetti con chiavi che terminano con ".json" dai bucket contenenti "config" nel relativo nome nei file nella cartella specificata. Le chiavi degli oggetti vengono utilizzate per impostare i nomi dei file.

```
Get-S3Bucket | ? { $_.BucketName -like '*config*' } | Get-S3Object | ? { $_.Key -
like '*.json' } | Read-S3Object -Folder C:\ConfigObjects
```

- Per i dettagli sull'API, vedere [GetObject](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3Bucket

Il seguente esempio di codice mostra come utilizzare. Remove-S3Bucket

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove tutti gli oggetti e le relative versioni dal bucket 'test-files', quindi elimina il bucket. Il comando richiederà una conferma prima di procedere. Aggiungere il parametro -Force per disabilitare le richieste di conferma. Nota che i bucket non vuoti non possono essere eliminati.

```
Remove-S3Bucket -BucketName amzn-s3-demo-bucket -DeleteBucketContent
```

- Per i dettagli sull'API, vedere [DeleteBucket](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3BucketAnalyticsConfiguration

Il seguente esempio di codice mostra come utilizzare. Remove-S3BucketAnalyticsConfiguration

Strumenti per PowerShell V4

Esempio 1: il comando rimuove il filtro di analisi denominato "testfilter" nel bucket S3 specificato.

```
Remove-S3BucketAnalyticsConfiguration -BucketName 'amzn-s3-demo-bucket' -AnalyticsId 'testfilter'
```

- Per i dettagli sull'API, vedere [DeleteBucketAnalyticsConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3BucketEncryption

Il seguente esempio di codice mostra come utilizzare. Remove-S3BucketEncryption

Strumenti per PowerShell V4

Esempio 1: questo comando disabilita la crittografia abilitata per il bucket S3 specificato.

```
Remove-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketEncryption (DeleteBucketEncryption)" on
target "s3casetestbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Per i dettagli sull'API, vedere [DeleteBucketEncryption](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3BucketInventoryConfiguration

Il seguente esempio di codice mostra come utilizzare. Remove-S3BucketInventoryConfiguration

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove l'inventario denominato 'testInventoryName' corrispondente al bucket S3 specificato.

```
Remove-S3BucketInventoryConfiguration -BucketName 'amzn-s3-demo-bucket' -InventoryId
'testInventoryName'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketInventoryConfiguration
(DeleteBucketInventoryConfiguration)" on target "amzn-s3-demo-bucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Per i dettagli sull'API, vedere [DeleteBucketInventoryConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3BucketMetricsConfiguration

Il seguente esempio di codice mostra come utilizzare. Remove-S3BucketMetricsConfiguration

Strumenti per PowerShell V4

Esempio 1: il comando rimuove il filtro delle metriche denominato “testmetrics” nel bucket S3 specificato.

```
Remove-S3BucketMetricsConfiguration -BucketName 'amzn-s3-demo-bucket' -MetricsId 'testmetrics'
```

- Per i dettagli sull'API, vedere [DeleteBucketMetricsConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3BucketPolicy

Il seguente esempio di codice mostra come utilizzare. Remove-S3BucketPolicy

Strumenti per PowerShell V4

Esempio 1: il comando rimuove la policy di bucket associata al bucket S3 specificato.

```
Remove-S3BucketPolicy -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [DeleteBucketPolicy](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3BucketReplication

Il seguente esempio di codice mostra come utilizzare. Remove-S3BucketReplication

Strumenti per PowerShell V4

Esempio 1: elimina la configurazione di replica associata al bucket denominato “amzn-s3-demo-bucket”. Nota che questa operazione richiede l'autorizzazione per l'azione s3:DeleteReplicationConfiguration . Prima di procedere con l'operazione, viene richiesta una conferma. Per eliminare la conferma, utilizza l'opzione -Force.

```
Remove-S3BucketReplication -BucketName amzn-s3-demo-bucket
```

- Per i dettagli sull'API, vedere [DeleteBucketReplication](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3BucketTagging

Il seguente esempio di codice mostra come utilizzare. Remove-S3BucketTagging

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove tutti i tag associati al bucket S3 specificato.

```
Remove-S3BucketTagging -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketTagging (DeleteBucketTagging)" on target
"amzn-s3-demo-bucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Per i dettagli sull'API, vedere [DeleteBucketTagging](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3BucketWebsite

Il seguente esempio di codice mostra come utilizzare. Remove-S3BucketWebsite

Strumenti per PowerShell V4

Esempio 1: questo comando disabilita la proprietà di hosting di siti web statici del bucket S3 specificato.

```
Remove-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketWebsite (DeleteBucketWebsite)" on target
"amzn-s3-demo-bucket".
```

```
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): Y
```

- Per i dettagli sull'API, vedere [DeleteBucketWebsite](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3CORSConfiguration

Il seguente esempio di codice mostra come utilizzare. Remove-S3CORSConfiguration

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove la configurazione CORS per il bucket S3 specificato.

```
Remove-S3CORSConfiguration -BucketName 'amzn-s3-demo-bucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3CORSConfiguration (DeleteCORSConfiguration)" on
target "amzn-s3-demo-bucket".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y
```

- Per i dettagli sull'API, vedere [Delete CORSConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3LifecycleConfiguration

Il seguente esempio di codice mostra come utilizzare. Remove-S3LifecycleConfiguration

Strumenti per PowerShell V4

Esempio 1: il comando rimuove tutte le regole del ciclo di vita per il bucket S3 specificato.

```
Remove-S3LifecycleConfiguration -BucketName 'amzn-s3-demo-bucket'
```

- Per i dettagli sull'API, vedere [DeleteLifecycleConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3MultipartUpload

Il seguente esempio di codice mostra come utilizzare. Remove-S3MultipartUpload

Strumenti per PowerShell V4

Esempio 1: questo comando interrompe i caricamenti in più parti creati prima di 5 giorni fa.

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -DaysBefore 5
```

Esempio 2: questo comando interrompe i caricamenti in più parti creati prima del 2 gennaio 2014.

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -InitiatedDate "Thursday,  
January 02, 2014"
```

Esempio 3: questo comando interrompe i caricamenti in più parti creati prima del 2 gennaio 2014 alle 10:45:37.

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -InitiatedDate "2014/01/02  
10:45:37"
```

- Per i dettagli sull'API, vedere [AbortMultipartUpload](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3Object

Il seguente esempio di codice mostra come utilizzare. Remove-S3Object

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove l'oggetto "sample.txt" dal bucket "test-files". Prima di eseguire il comando, viene richiesto di confermare l'operazione. Per disabilitare la richiesta, utilizza il parametro -Force.

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt
```

Esempio 2: questo comando rimuove la versione specificata dell'oggetto "sample.txt" dal bucket "test-files", supponendo che il bucket sia stato configurato per abilitare le versioni degli oggetti.

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -VersionId  
HLbxnx6V9omT6AQYVpks8mmFKQcejpqt
```

Esempio 3: questo comando rimuove gli oggetti “sample1.txt”, “sample2.txt” e “sample3.txt” dal bucket “test-files” in un’unica operazione batch. La risposta del servizio elencherà tutte le chiavi elaborate, indipendentemente dallo stato di riuscita o errore dell’eliminazione. Per ottenere solo gli errori relativi alle chiavi che non hanno potuto essere elaborate dal servizio, aggiungi il ReportErrorsOnly parametro - (questo parametro può essere specificato anche con l’alias -Quiet).

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -KeyCollection @( "sample1.txt",  
"sample2.txt", "sample3.txt" )
```

Esempio 4: Questo esempio utilizza un’espressione in linea con il KeyCollection parametro - per ottenere le chiavi degli oggetti da eliminare. Get-S3Object restituisce una raccolta di istanze Amazon.S3.Model.S3Object, ognuna delle quali ha un membro Key di tipo string che identifica l’oggetto.

```
Remove-S3Object -bucketname "amzn-s3-demo-bucket" -KeyCollection (Get-S3Object  
"test-files" -KeyPrefix "prefix/subprefix" | select -ExpandProperty Key)
```

Esempio 5: questo esempio ottiene tutti gli oggetti con il prefisso di chiave “prefix/subprefix” nel bucket e li elimina. Nota che gli oggetti in entrata vengono elaborati uno alla volta. Per raccolte di grandi dimensioni, è consigliabile passare la raccolta al parametro - InputObject (alias - S3ObjectCollection) del cmdlet per consentire l’eliminazione in batch con una singola chiamata al servizio.

```
Get-S3Object -BucketName "amzn-s3-demo-bucket" -KeyPrefix "prefix/subprefix" |  
Remove-S3Object -Force
```

Esempio 6: questo esempio reindirizza una raccolta di istanze Amazon.S3.Model.S3ObjectVersion che rappresentano indicatori di eliminazione al cmdlet per l’eliminazione. Nota che gli oggetti in entrata vengono elaborati uno alla volta. Per raccolte di grandi dimensioni, è consigliabile passare la raccolta al parametro - InputObject (alias -S3ObjectCollection) del cmdlet per consentire l’eliminazione in batch con una singola chiamata al servizio.

```
(Get-S3Version -BucketName "amzn-s3-demo-bucket").Versions | Where  
{$_ .IsDeleteMarker -eq "True"} | Remove-S3Object -Force
```

Esempio 7: questo script mostra come eseguire un'eliminazione in batch di un set di oggetti (in questo caso eliminare i marker) creando una matrice di oggetti da utilizzare con il parametro `-KeyAndVersionCollection`

```
$keyVersions = @()
$markers = (Get-S3Version -BucketName $BucketName).Versions | Where
{$_ .IsDeleteMarker -eq "True"}
foreach ($marker in $markers) { $keyVersions += @{ Key = $marker.Key; VersionId =
$marker.VersionId } }
Remove-S3Object -BucketName $BucketName -KeyAndVersionCollection $keyVersions -Force
```

- Per i dettagli sull'API, vedere [DeleteObjects](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3ObjectTagSet

Il seguente esempio di codice mostra come utilizzare `Remove-S3ObjectTagSet`

Strumenti per PowerShell V4

Esempio 1: questo comando rimuove tutti i tag associati all'oggetto con la chiave "testfile.txt" nel bucket S3 specificato.

```
Remove-S3ObjectTagSet -Key 'testfile.txt' -BucketName 'amzn-s3-demo-bucket' -Select
'^Key'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3ObjectTagSet (DeleteObjectTagging)" on target
"testfile.txt".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is
"Y"): Y
testfile.txt
```

- Per i dettagli sull'API, vedere [DeleteObjectTagging](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-S3PublicAccessBlock

Il seguente esempio di codice mostra come utilizzare. Remove-S3PublicAccessBlock

Strumenti per PowerShell V4

Esempio 1: questo comando disattiva l'impostazione di blocco dell'accesso pubblico per il bucket specificato.

```
Remove-S3PublicAccessBlock -BucketName 'amzn-s3-demo-bucket' -Force -Select  
'^BucketName'
```

Output:

```
amzn-s3-demo-bucket
```

- Per i dettagli sull'API, vedere [DeletePublicAccessBlock](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-S3BucketEncryption

Il seguente esempio di codice mostra come utilizzare. Set-S3BucketEncryption

Strumenti per PowerShell V4

Esempio 1: questo comando abilita la crittografia predefinita lato AES256 server con Amazon S3 Managed Keys (SSE-S3) sul bucket specificato.

```
$Encryptionconfig = @{ServerSideEncryptionByDefault =  
  @{{ServerSideEncryptionAlgorithm = "AES256"}}}  
Set-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket' -  
ServerSideEncryptionConfiguration_ServerSideEncryptionRule $Encryptionconfig
```

- Per i dettagli sull'API, vedere [PutBucketEncryption](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Test-S3Bucket

Il seguente esempio di codice mostra come utilizzare. Test-S3Bucket

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce True se il bucket esiste, False in caso contrario. Il comando restituisce True anche se il bucket non appartiene all'utente.

```
Test-S3Bucket -BucketName amzn-s3-demo-bucket
```

- Per i dettagli sull'API, vedere [Test-S3Bucket](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-S3BucketAccelerateConfiguration

Il seguente esempio di codice mostra come utilizzare. Write-S3BucketAccelerateConfiguration

Strumenti per PowerShell V4

Esempio 1: questo comando abilita l'accelerazione del trasferimento per il bucket S3 specificato.

```
$statusVal = New-Object Amazon.S3.BucketAccelerateStatus('Enabled')
Write-S3BucketAccelerateConfiguration -BucketName 'amzn-s3-demo-bucket' -
AccelerateConfiguration_Status $statusVal
```

- Per i dettagli sull'API, vedere [PutBucketAccelerateConfiguration](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-S3BucketNotification

Il seguente esempio di codice mostra come utilizzare. Write-S3BucketNotification

Strumenti per PowerShell V4

Esempio 1: questo esempio configura la configurazione dell'argomento SNS per l'evento S3 ObjectRemovedDelete e abilita la notifica per il bucket s3 specificato

```
$topic = [Amazon.S3.Model.TopicConfiguration] @{
    Id = "delete-event"
    Topic = "arn:aws:sns:eu-west-1:123456789012:topic-1"
    Event = [Amazon.S3.EventType]::ObjectRemovedDelete
}
```



```
Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -TopicConfiguration
$topic
```

Esempio 2: questo esempio abilita le notifiche ObjectCreatedAll relative al bucket specificato che lo invia alla funzione Lambda.

```
$lambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = "s3:ObjectCreated:*"
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:rdplock"
    Id = "ObjectCreated-Lambda"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".pem"}
            )
        }
    }
}

Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -
LambdaFunctionConfiguration $lambdaConfig
```

Esempio 3: questo esempio crea 2 diverse configurazioni Lambda in base a suffissi di chiave diversi e le configura entrambe in un unico comando.

```
#Lambda Config 1

$firstLambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = "s3:ObjectCreated:*"
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifynet"
    Id = "ObjectCreated-dada-ps1"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".ps1"}
            )
        }
    }
}
```

```
#Lambda Config 2

$secondlambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = [Amazon.S3.EventType]::ObjectCreatedAll
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifyssm"
    Id = "ObjectCreated-dada-json"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".json"}
            )
        }
    }
}

Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -
LambdaFunctionConfiguration $firstLambdaConfig,$secondlambdaConfig
```

- Per i dettagli sull'API, vedere [PutBucketNotification](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-S3BucketReplication

Il seguente esempio di codice mostra come utilizzare Write-S3BucketReplication

Strumenti per PowerShell V4

Esempio 1: Questo esempio imposta una configurazione di replica con un'unica regola che consente la replica nel bucket 'amzn-s3-demo-bucket' di tutti i nuovi oggetti creati con il prefisso del nome chiave "" nel bucket 'amzn-s3-demo-bucket'. TaxDocs

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
```

```

    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params

```

Esempio 2: Questo esempio imposta una configurazione di replica con più regole che consentono la replica nel bucket 'amzn-s3-demo-bucket' di qualsiasi nuovo oggetto creato con il prefisso del nome chiave "TaxDocs" o OtherDocs "». I prefissi di chiave non devono sovrapporsi.

```

$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-bucket" }

$rule2 = New-Object Amazon.S3.Model.ReplicationRule
$rule2.ID = "Rule-2"
$rule2.Status = "Enabled"
$rule2.Prefix = "OtherDocs"
$rule2.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1,$rule2
}

Write-S3BucketReplication @params

```

Esempio 3: Questo esempio aggiorna la configurazione di replica sul bucket specificato per disabilitare la regola che controlla la replica degli oggetti con il prefisso del nome chiave TaxDocs "" nel bucket 'amzn-s3-demo-bucket'.

```

$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Disabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"

```

```

    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params

```

- Per [PutBucketReplication AWS Strumenti per PowerShell](#) i dettagli sull'API, vedere in Cmdlet Reference (V4).

Write-S3BucketRequestPayment

Il seguente esempio di codice mostra come utilizzare. Write-S3BucketRequestPayment

Strumenti per PowerShell V4

Esempio 1: aggiorna la configurazione del pagamento della richiesta per il bucket denominato "amzn-s3-demo-bucket" in modo che alla persona che richiede il download dal bucket venga addebitato il costo del download. Per impostazione predefinita, il proprietario del bucket paga i download. Per riportare la richiesta di pagamento ai valori predefiniti, usa 'BucketOwner' per il parametro RequestPaymentConfiguration _Payer.

```

Write-S3BucketRequestPayment -BucketName amzn-s3-demo-bucket -
RequestPaymentConfiguration_Payer Requester

```

- Per i dettagli sull'API, vedere [PutBucketRequestPayment](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-S3BucketTagging

Il seguente esempio di codice mostra come utilizzare. Write-S3BucketTagging

Strumenti per PowerShell V4

Esempio 1: questo comando applica due tag a un bucket denominato **cloudtrail-test-2018**, ovvero un tag con una chiave Stage e un valore Test e un tag con una chiave Environment e un valore Alpha. Per verificare che i tag siano stati aggiunti al bucket, esegui **Get-S3BucketTagging -BucketName bucket_name**. I risultati dovrebbero riportare i tag applicati al bucket nel primo comando. Nota che **Write-S3BucketTagging** sovrascrive l'intero

set di tag esistente in un bucket. Per aggiungere o eliminare singoli tag, esegui i cmdlet **Add-RGTResourceTag** e **Remove-RGTResourceTag** delle API Gruppi di risorse e Tagging. In alternativa, utilizza Tag Editor nella console di AWS gestione per gestire i tag dei bucket S3.

```
Write-S3BucketTagging -BucketName amzn-s3-demo-bucket -TagSet @( @{ Key="Stage"; Value="Test" }, @{ Key="Environment"; Value="Alpha" } )
```

Esempio 2: questo comando reindirizza un bucket denominato **cloudtrail-test-2018** nel cmdlet **Write-S3BucketTagging**. Applica i tag Stage:Production e Department:Finance al bucket. Nota che **Write-S3BucketTagging** sovrascrive l'intero set di tag esistente in un bucket.

```
Get-S3Bucket -BucketName amzn-s3-demo-bucket | Write-S3BucketTagging -TagSet @( @{ Key="Stage"; Value="Production" }, @{ Key="Department"; Value="Finance" } )
```

- Per i dettagli sull'API, vedere [PutBucketTagging](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-S3BucketVersioning

Il seguente esempio di codice mostra come utilizzare Write-S3BucketVersioning

Strumenti per PowerShell V4

Esempio 1: il comando abilita il controllo delle versioni per il bucket S3 specificato.

```
Write-S3BucketVersioning -BucketName 'amzn-s3-demo-bucket' -VersioningConfig_Status Enabled
```

- Per i dettagli sull'API, vedere [PutBucketVersioning](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-S3BucketWebsite

Il seguente esempio di codice mostra come utilizzare Write-S3BucketWebsite

Strumenti per PowerShell V4

Esempio 1: il comando abilita l'hosting del sito web per il bucket specificato con il documento dell'indice denominato "index.html" e il documento degli errori denominato "error.html".

```
Write-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
-WebsiteConfiguration_IndexDocumentSuffix 'index.html' -
WebsiteConfiguration_ErrorDocument 'error.html'
```

- Per i dettagli sull'API, vedere [PutBucketWebsite](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-S3LifecycleConfiguration

Il seguente esempio di codice mostra come utilizzare Write-S3LifecycleConfiguration

Strumenti per PowerShell V4

Esempio 1: Questo esempio scrive/sostituisce la configurazione fornita in \$. NewRule Questa configurazione assicura di limitare l'ambito degli oggetti con determinati valori di prefisso e tag.

```
$NewRule = [Amazon.S3.Model.LifecycleRule] @{
    Expiration = @{
        Days= 50
    }
    Id = "Test-From-Write-cmdlet-1"
    Filter= @{
        LifecycleFilterPredicate = [Amazon.S3.Model.LifecycleAndOperator]@{
            Operands= @(
                [Amazon.S3.Model.LifecyclePrefixPredicate] @{
                    "Prefix" = "py"
                },
                [Amazon.S3.Model.LifecycleTagPredicate] @{
                    "Tag"= @{
                        "Key" = "non-use"
                        "Value" = "yes"
                    }
                }
            )
        }
    }
    "Status"= 'Enabled'
    NoncurrentVersionExpiration = @{
        NoncurrentDays = 75
    }
}
```

```
Write-S3LifecycleConfiguration -BucketName amzn-s3-demo-bucket -Configuration_Rule
$NewRule
```

Esempio 2: Questo esempio imposta più regole con il filtraggio. \$ ArchiveRule imposta gli oggetti da archiviare in 30 giorni su Glacier e 120 su. DeepArchive \$ ExpireRule scade sia la versione attuale che quella precedente in 150 giorni per gli oggetti con prefisso 'py' e tag:key 'archieved' impostato su 'yes'.

```
$ExpireRule = [Amazon.S3.Model.LifecycleRule] @{
    Expiration = @{
        Days= 150
    }
    Id = "Remove-in-150-days"
    Filter= @{
        LifecycleFilterPredicate = [Amazon.S3.Model.LifecycleAndOperator]@{
            Operands= @(
                [Amazon.S3.Model.LifecyclePrefixPredicate] @{
                    "Prefix" = "py"
                },
                [Amazon.S3.Model.LifecycleTagPredicate] @{
                    "Tag"= @{
                        "Key" = "archived"
                        "Value" = "yes"
                    }
                }
            )
        }
    }
    Status= 'Enabled'
    NoncurrentVersionExpiration = @{
        NoncurrentDays = 150
    }
}

$ArchiveRule = [Amazon.S3.Model.LifecycleRule] @{
    Expiration = $null
    Id = "Archive-to-Glacier-in-30-days"
    Filter= @{
        LifecycleFilterPredicate = [Amazon.S3.Model.LifecycleAndOperator]@{
            Operands= @(
                [Amazon.S3.Model.LifecyclePrefixPredicate] @{
                    "Prefix" = "py"
                },
```

```

    [Amazon.S3.Model.LifecycleTagPredicate] @{
        "Tag"= @{
            "Key" = "reviewed"
            "Value" = "yes"
        }
    }
)
}
}
Status = 'Enabled'
NoncurrentVersionExpiration = @{
    NoncurrentDays = 75
}
Transitions = @(
    @{
        Days = 30
        "StorageClass"= 'Glacier'
    },
    @{
        Days = 120
        "StorageClass"= [Amazon.S3.S3StorageClass]::DeepArchive
    }
)
}

Write-S3LifecycleConfiguration -BucketName amzn-s3-demo-bucket -Configuration_Rule
$ExpireRule,$ArchiveRule

```

- Per i dettagli sull'API, vedere in Cmdlet Reference (V4). [PutLifecycleConfiguration](#) AWS Strumenti per PowerShell

Write-S3Object

Il seguente esempio di codice mostra come utilizzare. Write-S3Object

Strumenti per PowerShell V4

Esempio 1: questo comando carica il file “local-sample.txt” in Amazon S3 e crea un oggetto con la chiave “sample.txt” nel bucket “test-files”.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "sample.txt" -File .\local-
sample.txt
```


Esempio 2: questo comando carica il file "sample.txt" in Amazon S3 e crea un oggetto con la chiave "sample.txt" nel bucket "test-files". Se il parametro -Key non viene fornito, il nome del file viene utilizzato come chiave dell'oggetto S3.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -File .\sample.txt
```

Esempio 3: questo comando carica il singolo file "local-sample.txt" su Amazon S3, creando un oggetto con la chiave prefix/to/sample ".txt» nel bucket «test-files».

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "prefix/to/sample.txt" -File .\local-sample.txt
```

Esempio 4: questo comando carica tutti i file nella sottodirectory «Scripts» nel bucket «test-files» e applica il prefisso chiave comune "" a ciascun oggetto. SampleScripts Ogni file caricato avrà una chiave "SampleScripts/filename" dove 'filename' varia.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder .\Scripts -KeyPrefix SampleScripts\
```

Esempio 5: questo comando carica tutti i file*.ps1 nella directory locale «Scripts» nel bucket «test-files» e applica il prefisso chiave comune "" a ciascun oggetto. SampleScripts Ogni file caricato avrà una chiave "/filename.ps1" dove 'filename' varia. SampleScripts

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder .\Scripts -KeyPrefix SampleScripts\ -SearchPattern *.ps1
```

Esempio 6: questo comando crea un nuovo oggetto S3 contenente la stringa di contenuto specificata con la chiave "sample.txt".

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "sample.txt" -Content "object contents"
```

Esempio 7: questo comando carica il file specificato (il nome del file viene utilizzato come chiave) e applica i tag specificati al nuovo oggetto.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -File "sample.txt" -TagSet @{{Key="key1";Value="value1"}},{Key="key2";Value="value2"}}
```

Esempio 8: questo comando carica in modo ricorsivo la cartella specificata e applica i tag specificati a tutti i nuovi oggetti.

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder . -KeyPrefix "TaggedFiles" -  
Recurse -TagSet @{Key="key1";Value="value1"},@{Key="key2";Value="value2"}
```

- Per i dettagli sull'API, vedere [PutObject](#) in Cmdlet Reference (V4). AWS Strumenti per PowerShell

Write-S3ObjectRetention

Il seguente esempio di codice mostra come utilizzare Write-S3ObjectRetention

Strumenti per PowerShell V4

Esempio 1: il comando abilita la modalità di conservazione della governance fino alla data “31 dicembre 2019 00:00:00” per l’oggetto “testfile.txt” nel bucket S3 specificato.

```
Write-S3ObjectRetention -BucketName 'amzn-s3-demo-bucket' -Key 'testfile.txt' -  
Retention_Mode GOVERNANCE -Retention_RetainUntilDate "2019-12-31T00:00:00"
```

- Per i dettagli sull'API, vedere [PutObjectRetention](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi CSPM di Security Hub con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando il CSPM AWS Strumenti per PowerShell V4 con Security Hub.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-SHUBFinding

Il seguente esempio di codice mostra come utilizzare. Get-SHUBFinding

Strumenti per PowerShell V4

Esempio 1: questo comando recupera i risultati del Security Hub da Amazon EC2; service.

```
$filter = New-Object -TypeName Amazon.SecurityHub.Model.AwsSecurityFindingFilters
$filter.ResourceType = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter -
Property @{
    Comparison = 'PREFIX'
    Value = 'AwsEc2'
}
Get-SHUBFinding -Filter $filter
```

Esempio 2: questo comando recupera i risultati del Security Hub dall' AWS account ID 123456789012.

```
$filter = New-Object -TypeName Amazon.SecurityHub.Model.AwsSecurityFindingFilters
$filter.AwsAccountId = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter -
Property @{
    Comparison = 'EQUALS'
    Value = '123456789012'
}
Get-SHUBFinding -Filter $filter
```

Esempio 3: questo comando recupera gli esiti della Centrale di sicurezza generati per lo standard “pci-dss”.

```
$filter = New-Object -TypeName Amazon.SecurityHub.Model.AwsSecurityFindingFilters
$filter.GeneratorId = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter -
Property @{
    Comparison = 'PREFIX'
    Value = 'pci-dss'
}
Get-SHUBFinding -Filter $filter
```

Esempio 4: questo comando recupera gli esiti della Centrale di sicurezza relativamente alla gravità critica con stato NOTIFIED del flusso di lavoro.

```
$filter = New-Object -TypeName Amazon.SecurityHub.Model.AwsSecurityFindingFilters
$filter.SeverityLabel = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter -
Property @{
    Comparison = 'EQUALS'
    Value = 'CRITICAL'
}
$filter.WorkflowStatus = New-Object -TypeName Amazon.SecurityHub.Model.StringFilter
-Property @{
    Comparison = 'EQUALS'
    Value = 'NOTIFIED'
}
Get-SHUBFinding -Filter $filter
```

- Per i dettagli sull'API, vedere [GetFindings](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon SES con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon SES.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Get-SESIdentity

Il seguente esempio di codice mostra come usare `Get-SESIdentity`.

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce un elenco contenente tutte le identità (indirizzi e-mail e domini) per un AWS account specifico, indipendentemente dallo stato di verifica.

```
Get-SESIIdentity
```

- Per i dettagli sull'API, vedere [ListIdentities](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SESSendQuota

Il seguente esempio di codice mostra come utilizzare. Get-SESSendQuota

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce i limiti di invio correnti dell'utente.

```
Get-SESSendQuota
```

- Per i dettagli sull'API, vedere [GetSendQuota](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SESSendStatistic

Il seguente esempio di codice mostra come utilizzare. Get-SESSendStatistic

Strumenti per PowerShell V4

Esempio 1: questo comando restituisce le statistiche di invio dell'utente. Il risultato è un elenco di punti di dati relativi alle ultime due settimane di attività di invio. Ogni punto dati nell'elenco contiene statistiche per un intervallo di 15 minuti.

```
Get-SESSendStatistic
```

- Per i dettagli sull'API, vedere [GetSendStatistics](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di API Amazon SES v2 con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la versione AWS Strumenti per PowerShell 4 con l'API Amazon SES v2.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Send-SES2Email

Il seguente esempio di codice mostra come usare. Send-SES2Email

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra come inviare un messaggio e-mail standard.

```
Send-SES2Email -FromEmailAddress "sender@example.com" -Destination_ToAddress  
"recipient@example.com" -Subject_Data "Email Subject" -Text_Data "Email Body"
```

- Per i dettagli sull'API, vedere [SendEmail](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon SNS con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon SNS.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Publish-SNSMessage

Il seguente esempio di codice mostra come usare Publish-SNSMessage

Strumenti per PowerShell V4

Esempio 1: Questo esempio mostra la pubblicazione di un messaggio con una singola riga MessageAttribute dichiarata.

```
Publish-SNSMessage -TopicArn "arn:aws:sns:us-west-2:123456789012:my-topic" -Message  
"Hello" -MessageAttribute  
@{'City'=[Amazon.SimpleNotificationService.Model.MessageAttributeValue]@{'DataType='String';  
StringValue ='AnyCity'}}}
```

Esempio 2: Questo esempio mostra la pubblicazione di un messaggio con più messaggi MessageAttributes dichiarati in anticipo.

```
$cityAttributeValue = New-Object  
Amazon.SimpleNotificationService.Model.MessageAttributeValue  
$cityAttributeValue.DataType = "String"  
$cityAttributeValue.StringValue = "AnyCity"  
  
$populationAttributeValue = New-Object  
Amazon.SimpleNotificationService.Model.MessageAttributeValue  
$populationAttributeValue.DataType = "Number"  
$populationAttributeValue.StringValue = "1250800"  
  
$messageAttributes = New-Object System.Collections.Hashtable  
$messageAttributes.Add("City", $cityAttributeValue)  
$messageAttributes.Add("Population", $populationAttributeValue)  
  
Publish-SNSMessage -TopicArn "arn:aws:sns:us-west-2:123456789012:my-topic" -Message  
"Hello" -MessageAttribute $messageAttributes
```

- Per informazioni dettagliate sull'API, consulta [Publish](#) nella documentazione di riferimento dei cmdlet di AWS Strumenti per PowerShell (V4).

Esempi di Amazon SQS con Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la AWS Strumenti per PowerShell V4 con Amazon SQS.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-SQSPermission

Il seguente esempio di codice mostra come usare. Add-SQSPermission

Strumenti per PowerShell V4

Esempio 1: Questo esempio consente all'utente specificato Account AWS di inviare messaggi dalla coda specificata.

```
Add-SQSPermission -Action SendMessage -AWSAccountId 80398EXAMPLE -Label  
SendMessagesFromMyQueue -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/  
MyQueue
```

- Per i dettagli sull'API, vedere [AddPermission](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Clear-SQSQueue

Il seguente esempio di codice mostra come utilizzare. Clear-SQSQueue

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina tutti i messaggi dalla coda specificata.

```
Clear-SQSQueue -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Per i dettagli sull'API, vedere [PurgeQueue](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-SQSMessageVisibility

Il seguente esempio di codice mostra come utilizzare `Edit-SQSMessageVisibility`

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica il timeout della visibilità del messaggio con l'handle di ricezione specificato nella coda specificata portandolo a 10 ore (10 ore * 60 minuti * 60 secondi = 36000 secondi).

```
Edit-SQSMessageVisibility -QueueUrl https://sqs.us-east-1.amazonaws.com/8039EXAMPLE/MyQueue -ReceiptHandle AQEBgGDh...J/Iqww== -VisibilityTimeout 36000
```

- Per i dettagli sull'API, vedere [ChangeMessageVisibility](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-SQSMessageVisibilityBatch

Il seguente esempio di codice mostra come utilizzare `Edit-SQSMessageVisibilityBatch`

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica il timeout della visibilità di 2 messaggi con gli handle di ricezione specificati nella coda specificata. Il timeout della visibilità del primo messaggio viene impostato su 10 ore (10 ore * 60 minuti * 60 secondi = 36000 secondi). Il timeout della visibilità del secondo messaggio viene impostata su 5 ore (5 ore * 60 minuti * 60 secondi = 18000 secondi).

```
$changeVisibilityRequest1 = New-Object  
    Amazon.SQS.Model.ChangeMessageVisibilityBatchRequestEntry  
$changeVisibilityRequest1.Id = "Request1"  
$changeVisibilityRequest1.ReceiptHandle = "AQEBd329...v6gl8Q=="  
$changeVisibilityRequest1.VisibilityTimeout = 36000
```

```
$changeVisibilityRequest2 = New-Object
    Amazon.SQS.Model.ChangeMessageVisibilityBatchRequestEntry
$changeVisibilityRequest2.Id = "Request2"
$changeVisibilityRequest2.ReceiptHandle = "AQEBgGDh...J/Iqww=="
$changeVisibilityRequest2.VisibilityTimeout = 18000

Edit-SQSMMessageVisibilityBatch -QueueUrl https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue -Entry $changeVisibilityRequest1,
    $changeVisibilityRequest2
```

Output:

Failed	Successful
-----	-----
{}	{Request2, Request1}

- Per i dettagli sull'API, vedere [ChangeMessageVisibilityBatch](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SQSDeadLetterSourceQueue

Il seguente esempio di codice mostra come utilizzare `Get-SQSDeadLetterSourceQueue`

Strumenti per PowerShell V4

Esempio 1: Questo esempio elenca tutte le URL di coda che si basano sulla coda specificata come coda di lettere morte.

```
Get-SQSDeadLetterSourceQueue -QueueUrl https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyOtherQueue
```

- Per i dettagli sull'API, vedere [ListDeadLetterSourceQueues](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SQSQueue

Il seguente esempio di codice mostra come utilizzare. Get-SQSQueue

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le code.

```
Get-SQSQueue
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/AnotherQueue
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/DeadLetterQueue
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyOtherQueue
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue
```

Esempio 2: questo esempio elenca qualsiasi coda che inizia con il nome specificato.

```
Get-SQSQueue -QueueNamePrefix My
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyOtherQueue
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue
```

- Per i dettagli sull'API, vedere [ListQueues](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SQSQueueAttribute

Il seguente esempio di codice mostra come utilizzare. Get-SQSQueueAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutti gli attributi della coda specificata.

```
Get-SQSQueueAttribute -AttributeName All -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

Output:

```

VisibilityTimeout           : 30
DelaySeconds                : 0
MaximumMessageSize         : 262144
MessageRetentionPeriod     : 345600
ApproximateNumberOfMessages : 0
ApproximateNumberOfMessagesNotVisible : 0
ApproximateNumberOfMessagesDelayed : 0
CreatedTimestamp           : 2/11/2015 5:53:35 PM
LastModifiedTimestamp      : 12/29/2015 2:23:17 PM
QueueARN                   : arn:aws:sqs:us-east-1:80398EXAMPLE:MyQueue
Policy                     : {"Version":"2012-10-17",
  "Id":"arn:aws:sqs:us-east-1:80398EXAMPLE:MyQueue/SQSDefaultPolicy", "Statement":
  [{"Sid":"Sid14
                                495134224EX", "Effect":"Allow", "Principal":
{"AWS":"*"}, "Action":"SQS:SendMessage", "Resource":"arn:aws:sqs:us-east-1:80
                                398EXAMPLE:MyQueue", "Condition":
{"ArnEquals":{"aws:SourceArn":"arn:aws:sns:us-east-1:80398EXAMPLE:MyTopic"}}},
{"Sid":
  "SendMessageFromMyQueue", "Effect":"Allow", "Principal":
{"AWS":"80398EXAMPLE"}, "Action":"SQS:SendMessage", "Resource":":
                                arn:aws:sqs:us-
east-1:80398EXAMPLE:MyQueue"}]}
Attributes                  : {[QueueArn, arn:aws:sqs:us-
east-1:80398EXAMPLE:MyQueue], [ApproximateNumberOfMessages, 0],
                                [ApproximateNumberOfMessagesNotVisible, 0],
                                [ApproximateNumberOfMessagesDelayed, 0]...}

```

Esempio 2: questo esempio elenca separatamente solo gli attributi specificati della coda specificata.

```

Get-SQSQueueAttribute -AttributeName MaximumMessageSize, VisibilityTimeout -QueueUrl
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue

```

Output:

```

VisibilityTimeout           : 30
DelaySeconds                : 0
MaximumMessageSize         : 262144
MessageRetentionPeriod     : 345600

```

```

ApproximateNumberOfMessages      : 0
ApproximateNumberOfMessagesNotVisible : 0
ApproximateNumberOfMessagesDelayed : 0
CreatedTimestamp                  : 2/11/2015 5:53:35 PM
LastModifiedTimestamp              : 12/29/2015 2:23:17 PM
QueueARN                          : arn:aws:sqs:us-east-1:80398EXAMPLE:MyQueue
Policy                            : {"Version":"2012-10-17",
  "Id":"arn:aws:sqs:us-east-1:80398EXAMPLE:MyQueue/SQSDefaultPolicy","Statement":
  [{"Sid":"Sid14
                                     495134224EX","Effect":"Allow","Principal":
  {"AWS":"*"},"Action":"SQS:SendMessage","Resource":"arn:aws:sqs:us-east-1:80
                                     398EXAMPLE:MyQueue","Condition":
  {"ArnEquals":{"aws:SourceArn":"arn:aws:sns:us-east-1:80398EXAMPLE:MyTopic"}}},
  {"Sid":
    "SendMessageFromMyQueue","Effect":"Allow","Principal":
  {"AWS":"80398EXAMPLE"},"Action":"SQS:SendMessage","Resource":
    "arn:aws:sqs:us-east-1:80398EXAMPLE:MyQueue"}]}
Attributes                        : {[MaximumMessageSize, 262144],
  [VisibilityTimeout, 30]}

```

- Per i dettagli sull'API, vedere [GetQueueAttributes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SQSQueueUrl

Il seguente esempio di codice mostra come utilizzare. Get-SQSQueueUrl

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca l'URL della coda con il nome specificato.

```
Get-SQSQueueUrl -QueueName MyQueue
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Per i dettagli sull'API, vedere [GetQueueUrl](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-SQSQueue

Il seguente esempio di codice mostra come utilizzare. New-SQSQueue

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una coda con il nome specificato.

```
New-SQSQueue -QueueName MyQueue
```

Output:

```
https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Per i dettagli sull'API, vedere [CreateQueue](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Receive-SQSMessage

Il seguente esempio di codice mostra come utilizzare. Receive-SQSMessage

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le informazioni per un massimo di 10 messaggi da ricevere per la coda specificata. Le informazioni conterranno i valori degli attributi del messaggio specificati, se esistenti.

```
Receive-SQSMessage -AttributeName SenderId, SentTimestamp -MessageAttributeName  
  StudentName, StudentGrade -MessageCount 10 -QueueUrl https://sqs.us-  
east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

Output:

```
Attributes           : {[SenderId, AIDAIAZKMSNQ7TEXAMPLE], [SentTimestamp,  
  1451495923744]}
```

```
Body                 : Information about John Doe's grade.
```

```
MD5ofBody            : ea572796e3c231f974fe75d89EXAMPLE
```

```
MD5ofMessageAttributes : 48c1ee811f0fe7c4e88fbe0f5EXAMPLE
```

```
MessageAttributes    : {[StudentGrade, Amazon.SQS.Model.MessageAttributeValue],  
  [StudentName, Amazon.SQS.Model.MessageAttributeValue]}
```

```
MessageId            : 53828c4b-631b-469b-8833-c093cEXAMPLE
```

```
ReceiptHandle      : AQEBpfGp...20Q5cg==
```

- Per i dettagli sull'API, vedere [ReceiveMessage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SQSMessage

Il seguente esempio di codice mostra come utilizzare. Remove-SQSMessage

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il messaggio con l'handle di ricezione specificato dalla coda specificata.

```
Remove-SQSMessage -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue  
-ReceiptHandle AQEBd329...v6gl8Q==
```

- Per i dettagli sull'API, vedere [DeleteMessage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SQSMessageBatch

Il seguente esempio di codice mostra come utilizzare. Remove-SQSMessageBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina 2 messaggi con gli handle di ricezione specificati dalla coda specificata.

```
$deleteMessageRequest1 = New-Object Amazon.SQS.Model.DeleteMessageBatchRequestEntry  
$deleteMessageRequest1.Id = "Request1"  
$deleteMessageRequest1.ReceiptHandle = "AQEBX2g4...wtJSQg=="  
  
$deleteMessageRequest2 = New-Object Amazon.SQS.Model.DeleteMessageBatchRequestEntry  
$deleteMessageRequest2.Id = "Request2"  
$deleteMessageRequest2.ReceiptHandle = "AQEBq0VY...KTsLYg=="  
  
Remove-SQSMessageBatch -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/  
MyQueue -Entry $deleteMessageRequest1, $deleteMessageRequest2
```

Output:

```
Failed      Successful
-----
{}          {Request1, Request2}
```

- Per i dettagli sull'API, vedere [DeleteMessageBatch](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SQSPermission

Il seguente esempio di codice mostra come utilizzare. Remove-SQSPermission

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove le impostazioni delle autorizzazioni con l'etichetta specificata dalla coda specificata.

```
Remove-SQSPermission -Label SendMessagesFromMyQueue -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Per i dettagli sull'API, vedere [RemovePermission](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SQSQueue

Il seguente esempio di codice mostra come utilizzare. Remove-SQSQueue

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina la coda specificata.

```
Remove-SQSQueue -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Per i dettagli sull'API, vedere [DeleteQueue](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Send-SQSMessage

Il seguente esempio di codice mostra come utilizzare. Send-SQSMessage

Strumenti per PowerShell V4

Esempio 1: questo esempio invia un messaggio con gli attributi e il corpo del messaggio specificati alla coda specificata con un ritardo di consegna del messaggio di 10 secondi.

```
$cityAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$cityAttributeValue.DataType = "String"
$cityAttributeValue.StringValue = "AnyCity"

$populationAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$populationAttributeValue.DataType = "Number"
$populationAttributeValue.StringValue = "1250800"

$messageAttributes = New-Object System.Collections.Hashtable
$messageAttributes.Add("City", $cityAttributeValue)
$messageAttributes.Add("Population", $populationAttributeValue)

Send-SQSMessage -DelayInSeconds 10 -MessageAttributes $messageAttributes -
MessageBody "Information about the largest city in Any Region." -QueueUrl https://
sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

Output:

MD5ofMessageAttributes	MD5ofMessageBody	MessageId
-----	-----	-----
1d3e51347bc042efbdf6dda31EXAMPLE c739-4d0c-818b-1820eEXAMPLE	51b0a3256d59467f973009b73EXAMPLE	c35fed8f-

- Per i dettagli sull'API, vedere [SendMessage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Send-SQSMessageBatch

Il seguente esempio di codice mostra come utilizzare. Send-SQSMessageBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio invia 2 messaggi con gli attributi e i corpi dei messaggi specificati alla coda specificata. La distribuzione viene ritardata di 15 secondi per il primo messaggio e di 10 secondi per il secondo messaggio.

```

$student1NameAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$student1NameAttributeValue.DataType = "String"
$student1NameAttributeValue.StringValue = "John Doe"

$student1GradeAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$student1GradeAttributeValue.DataType = "Number"
$student1GradeAttributeValue.StringValue = "89"

$student2NameAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$student2NameAttributeValue.DataType = "String"
$student2NameAttributeValue.StringValue = "Jane Doe"

$student2GradeAttributeValue = New-Object Amazon.SQS.Model.MessageAttributeValue
$student2GradeAttributeValue.DataType = "Number"
$student2GradeAttributeValue.StringValue = "93"

$message1 = New-Object Amazon.SQS.Model.SendMessageBatchRequestEntry
$message1.DelaySeconds = 15
$message1.Id = "FirstMessage"
$message1.MessageAttributes.Add("StudentName", $student1NameAttributeValue)
$message1.MessageAttributes.Add("StudentGrade", $student1GradeAttributeValue)
$message1.MessageBody = "Information about John Doe's grade."

$message2 = New-Object Amazon.SQS.Model.SendMessageBatchRequestEntry
$message2.DelaySeconds = 10
$message2.Id = "SecondMessage"
$message2.MessageAttributes.Add("StudentName", $student2NameAttributeValue)
$message2.MessageAttributes.Add("StudentGrade", $student2GradeAttributeValue)
$message2.MessageBody = "Information about Jane Doe's grade."

Send-SQSMessageBatch -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/
MyQueue -Entry $message1, $message2

```

Output:

Failed	Successful
-----	-----
{}	{FirstMessage, SecondMessage}

- Per i dettagli sull'API, vedere [SendMessageBatch](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Set-SQSQueueAttribute

Il seguente esempio di codice mostra come utilizzare Set-SQSQueueAttribute

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra come impostare una policy per sottoscrivere una coda a un argomento SNS. Quando si pubblica un messaggio in un argomento, viene inviato un messaggio a ogni coda sottoscritta.

```
# create the queue and topic to be associated
$qurl = New-SQSQueue -QueueName "myQueue"
$topicarn = New-SNSTopic -Name "myTopic"

# get the queue ARN to inject into the policy; it will be returned
# in the output's QueueARN member but we need to put it into a variable
# so text expansion in the policy string takes effect
$qarn = (Get-SQSQueueAttribute -QueueUrl $qurl -AttributeName "QueueArn").QueueARN

# construct the policy and inject arns
$policy = @"
{
  "Version":"2012-10-17",
  "Id": "$qarn/SQSPOLICY",
  "Statement": [
    {
      "Sid": "1",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "SQS:SendMessage",
      "Resource": "$qarn",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "$topicarn"
        }
      }
    }
  ]
}
```

```
"@  
  
# set the policy  
Set-SQSQueueAttribute -QueueUrl $qurl -Attribute @{ Policy=$policy }
```

Esempio 2: questo esempio imposta gli attributi specificati della coda specificata.

```
Set-SQSQueueAttribute -Attribute @{"DelaySeconds" = "10"; "MaximumMessageSize" =  
"131072"} -QueueUrl https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue
```

- Per i dettagli sull'API, vedere [SetQueueAttributes](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

AWS STS esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con AWS STS.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Convert-STSAuthorizationMessage

Il seguente esempio di codice mostra come utilizzare `Convert-STSAuthorizationMessage`.

Strumenti per PowerShell V4

Esempio 1: decodifica le informazioni aggiuntive contenute nel contenuto del messaggio codificato fornito e restituito in risposta a una richiesta. Le informazioni aggiuntive sono codificate perché i

dettagli dello stato di autorizzazione possono costituire informazioni con privilegi che l'utente che ha richiesto l'azione non dovrebbe vedere.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Per i dettagli sull'API, vedere [DecodeAuthorizationMessage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-STSFederationToken

Il seguente esempio di codice mostra come utilizzare. Get-STSFederationToken

Strumenti per PowerShell V4

Esempio 1: richiede un token federato valido per un'ora utilizzando "Bob" come nome dell'utente federato. Questo nome può essere usato per fare riferimento al nome utente federato in una policy basata sulle risorse (ad esempio una policy di bucket di Amazon S3). La policy IAM fornita, in formato JSON, viene utilizzata per definire le autorizzazioni disponibili per l'utente IAM. La policy fornita non può concedere più autorizzazioni di quelle concesse all'utente richiedente, e le autorizzazioni finali per l'utente federato sono il set più restrittivo in base all'intersezione tra la policy passata e la policy utente IAM.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds 3600
```

- Per i dettagli sull'API, vedere [GetFederationToken](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-STSSessionToken

Il seguente esempio di codice mostra come utilizzare. Get-STSSessionToken

Strumenti per PowerShell V4

Esempio 1: restituisce un'istanza **Amazon.Runtime.AWSCredentials** contenente credenziali temporanee valide per un determinato periodo di tempo. Le credenziali utilizzate per richiedere credenziali temporanee vengono dedotte dalle impostazioni predefinite correnti della shell. Per specificare altre credenziali, utilizzare i parametri - ProfileName o - AccessKey SecretKey /.

```
Get-STSSessionToken
```

Output:

AccessKeyId	Expiration
SecretAccessKey	SessionToken
-----	-----
-----	-----
EXAMPLEACCESSKEYID	2/16/2015 9:12:28 PM
examplesecretaccesskey...	SamPleToken.....

Esempio 2: restituisce un'istanza **Amazon.RuntimeAWSCredentials** contenente credenziali temporanee valide per un'ora. Le credenziali utilizzate per effettuare la richiesta vengono ottenute dal profilo specificato.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Output:

AccessKeyId	Expiration
SecretAccessKey	SessionToken
-----	-----
-----	-----
EXAMPLEACCESSKEYID	2/16/2015 9:12:28 PM
examplesecretaccesskey...	SamPleToken.....

Esempio 3: restituisce un'istanza **Amazon.RuntimeAWSCredentials** contenente credenziali temporanee valide per un'ora utilizzando il numero di identificazione del dispositivo MFA associato all'account le cui credenziali sono specificate nel profilo 'myprofile' e il valore fornito dal dispositivo.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber  
YourMFADeviceSerialNumber -TokenCode 123456
```

Output:

AccessKeyId	Expiration
SecretAccessKey	SessionToken

```

-----
-----
EXAMPLEACCESSKEYID          2/16/2015 9:12:28 PM
examplesecretaccesskey...    SamPleTokenN.....

```

- Per i dettagli sull'API, vedere [GetSessionToken](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Use-STSRole

Il seguente esempio di codice mostra come utilizzare. Use-STSRole

Strumenti per PowerShell V4

Esempio 1: restituisce un set di credenziali temporanee (chiave di accesso, chiave segreta e token di sessione) che possono essere utilizzate per un'ora per accedere a AWS risorse a cui l'utente richiedente potrebbe normalmente non avere accesso. Le credenziali restituite hanno le autorizzazioni consentite dalla policy di accesso del ruolo assunto e dalla policy fornita (non è possibile utilizzare la policy fornita per concedere autorizzazioni superiori a quelle definite dalla policy di accesso del ruolo assunto).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo" -
Policy "...JSON policy..." -DurationInSeconds 3600
```

Esempio 2: restituisce un set di credenziali temporanee, valide per un'ora, con le stesse autorizzazioni definite nella policy di accesso del ruolo assunto.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo" -
DurationInSeconds 3600
```

Esempio 3: restituisce un set di credenziali temporanee che forniscono il numero di serie e il token generato da un MFA associato alle credenziali utente utilizzate per eseguire il cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo" -
DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Esempio 4: restituisce un set di credenziali temporanee che hanno assunto un ruolo definito in un account cliente. Per ogni ruolo che la terza parte può assumere, l'account cliente deve creare un

ruolo utilizzando un identificatore che deve essere passato nel ExternalId parametro - ogni volta che viene assunto il ruolo.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo" -  
DurationInSeconds 3600 -ExternalId "ABC123"
```

- Per i dettagli sull'API, vedere [AssumeRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Use-STSWebIdentityRole

Il seguente esempio di codice mostra come utilizzare. Use-STSWebIdentityRole

Strumenti per PowerShell V4

Esempio 1: restituisce un set temporaneo di credenziali, valido per un'ora, per un utente che è stato autenticato con il provider di identità Login with Amazon. Le credenziali presuppongono la policy di accesso associata al ruolo identificato dall'ARN del ruolo. Facoltativamente, è possibile passare una policy JSON al parametro -Policy che perfeziona ulteriormente le autorizzazioni di accesso (non è possibile concedere più autorizzazioni di quelle disponibili nelle autorizzazioni associate al ruolo). Il valore fornito a - WebIdentityToken è l'identificatore utente univoco restituito dal provider di identità.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"  
-RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/  
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Per i dettagli sull'API, vedere [AssumeRoleWithWebIdentity](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Supporto esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. Supporto

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-ASACommunicationToCase

Il seguente esempio di codice mostra come utilizzare `Add-ASACommunicationToCase`.

Strumenti per PowerShell V4

Esempio 1: aggiunge il corpo di una comunicazione e-mail al caso specificato.

```
Add-ASACommunicationToCase -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47" -  
CommunicationBody "Some text about the case"
```

Esempio 2: aggiunge il corpo di una comunicazione e-mail al caso specificato e uno o più indirizzi e-mail contenuti nella riga CC dell'e-mail.

```
Add-ASACommunicationToCase -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47" -  
CcEmailAddress @"email1@address.com", "email2@address.com") -CommunicationBody  
"Some text about the case"
```

- Per i dettagli sull'API, vedere [AddCommunicationToCase](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASACase

Il seguente esempio di codice mostra come utilizzare `Get-ASACase`.

Strumenti per PowerShell V4

Esempio 1: restituisce i dettagli di tutti i casi di supporto.

```
Get-ASACase
```

Esempio 2: restituisce i dettagli di tutti i casi di supporto a partire dalla data e dall'ora specificate.

```
Get-ASACase -AfterTime "2013-09-10T03:06Z"
```

Esempio 3: restituisce i dettagli dei primi 10 casi di supporto, inclusi quelli risolti.

```
Get-ASACase -MaxResult 10 -IncludeResolvedCases $true
```

Esempio 4: restituisce i dettagli del caso di supporto specificato.

```
Get-ASACase -CaseIdList "case-12345678910-2013-c4c1d2bf33c5cf47"
```

Esempio 5: restituisce i dettagli dei casi di supporto specificati.

```
Get-ASACase -CaseIdList @("case-12345678910-2013-c4c1d2bf33c5cf47",  
"case-18929034710-2011-c4fdeabf33c5cf47")
```

- Per i dettagli sull'API, vedere [DescribeCases](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASACommunication

Il seguente esempio di codice mostra come utilizzare. Get-ASACommunication

Strumenti per PowerShell V4

Esempio 1: restituisce tutte le comunicazioni per il caso specificato.

```
Get-ASACommunication -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47"
```

Esempio 2: restituisce tutte le comunicazioni dalla mezzanotte UTC del 1° gennaio 2012 per il caso specificato.

```
Get-ASACommunication -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47" -AfterTime  
"2012-01-10T00:00Z"
```

- Per i dettagli sull'API, vedere [DescribeCommunications](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASAService

Il seguente esempio di codice mostra come utilizzare. Get-ASAService

Strumenti per PowerShell V4

Esempio 1: restituisce tutti i codici, i nomi e le categorie disponibili per il servizio.

```
Get-ASAService
```

Esempio 2: restituisce il nome e le categorie del servizio con il codice specificato.

```
Get-ASAService -ServiceCodeList "amazon-cloudfront"
```

Esempio 3: restituisce il nome e le categorie dei codici del servizio specificato.

```
Get-ASAService -ServiceCodeList @("amazon-cloudfront", "amazon-cloudwatch")
```

Esempio 4: restituisce il nome e le categorie (in giapponese) dei codici del servizio specificato. Attualmente sono supportati i codici di lingua inglese ("en") e giapponese ("ja").

```
Get-ASAService -ServiceCodeList @("amazon-cloudfront", "amazon-cloudwatch") -  
Language "ja"
```

- Per i dettagli sull'API, vedere [DescribeServices](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASASeverityLevel

Il seguente esempio di codice mostra come utilizzare. Get-ASASeverityLevel

Strumenti per PowerShell V4

Esempio 1: restituisce l'elenco dei livelli di gravità che possono essere assegnati a un caso AWS Support.

```
Get-ASASeverityLevel
```

Esempio 2: restituisce l'elenco dei livelli di gravità che possono essere assegnati a un caso AWS Support. I nomi dei livelli vengono restituiti in giapponese.

```
Get-ASASeverityLevel -Language "ja"
```

- Per i dettagli sull'API, vedere [DescribeSeverityLevels](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASATrustedAdvisorCheck

Il seguente esempio di codice mostra come utilizzare. Get-ASATrustedAdvisorCheck

Strumenti per PowerShell V4

Esempio 1: restituisce la raccolta dei controlli di Trusted Advisor. È necessario specificare il parametro Language che può accettare “en” per l’output in inglese o “ja” per l’output in giapponese.

```
Get-ASATrustedAdvisorCheck -Language "en"
```

- Per i dettagli sull'API, vedere [DescribeTrustedAdvisorChecks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASATrustedAdvisorCheckRefreshStatus

Il seguente esempio di codice mostra come utilizzare. Get-ASATrustedAdvisorCheckRefreshStatus

Strumenti per PowerShell V4

Esempio 1: restituisce lo stato corrente delle richieste di aggiornamento per i controlli specificati. Richiesta- ASATrusted AdvisorCheckRefresh può essere utilizzato per richiedere l'aggiornamento delle informazioni sullo stato dei controlli.

```
Get-ASATrustedAdvisorCheckRefreshStatus -CheckId @("checkid1", "checkid2")
```

- Per i dettagli sull'API, vedere [DescribeTrustedAdvisorCheckRefreshStatuses](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASATrustedAdvisorCheckResult

Il seguente esempio di codice mostra come utilizzare. Get-ASATrustedAdvisorCheckResult

Strumenti per PowerShell V4

Esempio 1: restituisce i risultati di un controllo di Trusted Advisor. L'elenco dei controlli Trusted Advisor disponibili può essere ottenuto utilizzando `Get-ASATrustedAdvisorChecks`. L'output include lo stato generale del controllo, il timestamp dell'ultimo controllo eseguito e l'ID univoco del controllo specifico. Per visualizzare i risultati in giapponese, aggiungi il parametro `-Language "ja"`.

```
Get-ASATrustedAdvisorCheckResult -CheckId "checkid1"
```

- Per i dettagli sull'API, vedere [DescribeTrustedAdvisorCheckResult](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-ASATrustedAdvisorCheckSummary

Il seguente esempio di codice mostra come utilizzare `Get-ASATrustedAdvisorCheckSummary`

Strumenti per PowerShell V4

Esempio 1: restituisce il riepilogo più recente del controllo di Trusted Advisor specificato.

```
Get-ASATrustedAdvisorCheckSummary -CheckId "checkid1"
```

Esempio 2: restituisce i riepiloghi più recenti dei controlli di Trusted Advisor specificati.

```
Get-ASATrustedAdvisorCheckSummary -CheckId @("checkid1", "checkid2")
```

- Per i dettagli sull'API, vedere [DescribeTrustedAdvisorCheckSummaries](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-ASACase

Il seguente esempio di codice mostra come utilizzare `New-ASACase`

Strumenti per PowerShell V4

Esempio 1: crea un nuovo caso nel AWS Support Center. I valori per i `CategoryCode` parametri `-ServiceCode` e `-SeverityCode` possono essere ottenuti utilizzando il `Get-ASAService` cmdlet. Il valore per il `SeverityCode` parametro - può essere ottenuto utilizzando il `Get-ASASeverityLevel` cmdlet.

Il valore del `IssueType` parametro - può essere «assistenza clienti» o «tecnico». In caso di successo, viene emesso il numero del caso AWS Support. Per impostazione predefinita, il caso verrà gestito in inglese. Per utilizzare il giapponese, aggiungi il parametro `-Language "ja"`. I `CommunicationBody` parametri `-ServiceCode`, `-CategoryCode`, `-Subject` e `-` sono obbligatori.

```
New-ASACase -ServiceCode "amazon-cloudfront" -CategoryCode "APIs" -SeverityCode "low" -Subject "subject text" -CommunicationBody "description of the case" -CcEmailAddress @"email1@domain.com", "email2@domain.com" -IssueType "technical"
```

- Per i dettagli sull'API, vedere [CreateCase](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Request-ASATrustedAdvisorCheckRefresh

Il seguente esempio di codice mostra come utilizzare. `Request-ASATrustedAdvisorCheckRefresh`

Strumenti per PowerShell V4

Esempio 1: richiede un aggiornamento del controllo di Trusted Advisor specificato.

```
Request-ASATrustedAdvisorCheckRefresh -CheckId "checkid1"
```

- Per i dettagli sull'API, vedere [RefreshTrustedAdvisorCheck](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Resolve-ASACase

Il seguente esempio di codice mostra come utilizzare. `Resolve-ASACase`

Strumenti per PowerShell V4

Esempio 1: restituisce lo stato iniziale del caso specificato e lo stato corrente dopo il completamento della chiamata per risolverlo.

```
Resolve-ASACase -CaseId "case-12345678910-2013-c4c1d2bf33c5cf47"
```

- Per i dettagli sull'API, vedere [ResolveCase](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Systems Manager con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con Systems Manager.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Add-SSMResourceTag

Il seguente esempio di codice mostra come utilizzare Add-SSMResourceTag.

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna una finestra di manutenzione con nuovi tag. Se il comando va a buon fine, non viene generato output. La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
$option1 = @{Key="Stack";Value=@"Production"}  
Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType  
"MaintenanceWindow" -Tag $option1
```

Esempio 2: con PowerShell la versione 2, è necessario utilizzare New-Object per creare ogni tag. Se il comando va a buon fine, non viene generato output.

```
$tag1 = New-Object Amazon.SimpleSystemsManagement.Model.Tag  
$tag1.Key = "Stack"  
$tag1.Value = "Production"  
  
Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType  
"MaintenanceWindow" -Tag $tag1
```

- Per i dettagli sull'API, vedere [AddTagsToResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-SSMDocumentPermission

Il seguente esempio di codice mostra come utilizzare. Edit-SSMDocumentPermission

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge le autorizzazioni di "condivisione" a tutti gli account di un documento. Se il comando va a buon fine, non viene generato output.

```
Edit-SSMDocumentPermission -Name "RunShellScript" -PermissionType "Share" -
AccountIdsToAdd all
```

Esempio 2: questo esempio aggiunge le autorizzazioni di "condivisione" a un account specifico per un documento. Se il comando va a buon fine, non viene generato output.

```
Edit-SSMDocumentPermission -Name "RunShellScriptNew" -PermissionType "Share" -
AccountIdsToAdd "123456789012"
```

- Per i dettagli sull'API, vedere [ModifyDocumentPermission](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMActivation

Il seguente esempio di codice mostra come utilizzare. Get-SSMActivation

Strumenti per PowerShell V4

Esempio 1: questo esempio fornisce dettagli sulle attivazioni sull'account.

```
Get-SSMActivation
```

Output:

```
ActivationId      : 08e51e79-1e36-446c-8e63-9458569c1363
CreatedDate       : 3/1/2017 12:01:51 AM
DefaultInstanceName : MyWebServers
Description       :
```



```
ExpirationDate      : 3/2/2017 12:01:51 AM
Expired             : False
IamRole             : AutomationRole
RegistrationLimit    : 10
RegistrationsCount  : 0
```

- Per i dettagli sull'API, vedere [DescribeActivations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMAssociation

Il seguente esempio di codice mostra come utilizzare. Get-SSMAssociation

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive l'associazione tra un'istanza e un documento.

```
Get-SSMAssociation -InstanceId "i-0000293ffd8c57862" -Name "AWS-UpdateSSMAgent"
```

Output:

```
Name                : AWS-UpdateSSMAgent
InstanceId           : i-0000293ffd8c57862
Date                : 2/23/2017 6:55:22 PM
Status.Name         : Pending
Status.Date         : 2/20/2015 8:31:11 AM
Status.Message      : temp_status_change
Status.AdditionalInfo : Additional-Config-Needed
```

- Per i dettagli sull'API, vedere [DescribeAssociation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMAssociationExecution

Il seguente esempio di codice mostra come utilizzare. Get-SSMAssociationExecution

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce le esecuzioni per l'ID di associazione fornito

```
Get-SSMAssociationExecution -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

Output:

```

AssociationId      : 123a45a0-c678-9012-3456-78901234db5e
AssociationVersion  : 2
CreatedTime        : 3/2/2019 8:53:29 AM
DetailedStatus     :
ExecutionId        : 123a45a0-c678-9012-3456-78901234db5e
LastExecutionDate  : 1/1/0001 12:00:00 AM
ResourceCountByStatus : {Success=4}
Status             : Success

```

- Per i dettagli sull'API, vedere [DescribeAssociationExecutions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMAssociationExecutionTarget

Il seguente esempio di codice mostra come utilizzare `Get-SSMAssociationExecutionTarget`

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza l'ID della risorsa e il relativo stato di esecuzione che fanno parte degli obiettivi di esecuzione dell'associazione

```

Get-SSMAssociationExecutionTarget -AssociationId 123a45a0-
c678-9012-3456-78901234db5e -ExecutionId 123a45a0-c678-9012-3456-78901234db5e |
Select-Object ResourceId, Status

```

Output:

```

ResourceId          Status
-----
i-0b1b2a3456f7a890b Success
i-01c12a45d6fc7a89f Success
i-0a1caf234f56d7dc8 Success
i-012a3fd45af6dbcfef Failed
i-0ddc1df23c4a5fb67 Success

```

Esempio 2: questo comando controlla la particolare esecuzione di una particolare automazione a partire da ieri, alla quale è associato un documento di comando. Controlla ulteriormente se l'esecuzione dell'associazione è fallita e, in tal caso, mostrerà i dettagli dell'invocazione del comando per l'esecuzione insieme all'ID dell'istanza

```
$AssociationExecution= Get-SSMAssociationExecutionTarget -
AssociationId 1c234567-890f-1aca-a234-5a678d901cb0 -ExecutionId
12345ca12-3456-2345-2b45-23456789012 |
    Where-Object {$_.LastExecutionDate -gt (Get-Date -Hour 00 -Minute
00).AddDays(-1)}

foreach ($execution in $AssociationExecution) {
    if($execution.Status -ne 'Success'){
        Write-Output "There was an issue executing the association
 $($execution.AssociationId) on $($execution.ResourceId)"
        Get-SSMCommandInvocation -CommandId $execution.OutputSource.OutputSourceId -
Detail:$true | Select-Object -ExpandProperty CommandPlugins
    }
}
```

Output:

```
There was an issue executing the association 1c234567-890f-1aca-a234-5a678d901cb0 on
i-0a1caf234f56d7dc8
```

```
Name                : aws:runPowerShellScript
Output              :
                    -----ERROR-----
                    failed to run commands: exit status 1
OutputS3BucketName  :
OutputS3KeyPrefix   :
OutputS3Region      : eu-west-1
ResponseCode        : 1
ResponseFinishDateTime : 5/29/2019 11:04:49 AM
ResponseStartDateTime  : 5/29/2019 11:04:49 AM
StandardErrorUrl     :
StandardOutputUrl    :
Status              : Failed
StatusDetails        : Failed
```

- Per i dettagli sull'API, vedere [DescribeAssociationExecutionTargets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMAssociationList

Il seguente esempio di codice mostra come utilizzare. Get-SSMAssociationList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le associazioni per un'istanza. La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
$filter1 = @{Key="InstanceId";Value=@"i-0000293ffd8c57862"}  
Get-SSMAssociationList -AssociationFilterList $filter1
```

Output:

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0  
DocumentVersion    :  
InstanceId         : i-0000293ffd8c57862  
LastExecutionDate  : 2/20/2015 8:31:11 AM  
Name              : AWS-UpdateSSMAgent  
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview  
ScheduleExpression :  
Targets            : {InstanceIds}
```

Esempio 2: questo esempio elenca tutte le associazioni per un documento di configurazione. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
$filter2 = @{Key="Name";Value=@"AWS-UpdateSSMAgent"}  
Get-SSMAssociationList -AssociationFilterList $filter2
```

Output:

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0  
DocumentVersion    :  
InstanceId         : i-0000293ffd8c57862  
LastExecutionDate  : 2/20/2015 8:31:11 AM  
Name              : AWS-UpdateSSMAgent  
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview  
ScheduleExpression :  
Targets            : {InstanceIds}
```

Esempio 3: con PowerShell la versione 2, è necessario utilizzare New-Object per creare ogni filtro.

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.AssociationFilter
```

```
$filter1.Key = "InstanceId"
$filter1.Value = "i-0000293ffd8c57862"

Get-SSMAssociationList -AssociationFilterList $filter1
```

Output:

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DocumentVersion   :
InstanceId        : i-0000293ffd8c57862
LastExecutionDate : 2/20/2015 8:31:11 AM
Name              : AWS-UpdateSSMAgent
Overview          : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets           : {InstanceIds}
```

- Per i dettagli sull'API, vedere [ListAssociations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMAssociationVersionList

Il seguente esempio di codice mostra come utilizzare. Get-SSMAssociationVersionList

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera tutte le versioni dell'associazione fornita.

```
Get-SSMAssociationVersionList -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

Output:

```
AssociationId      : 123a45a0-c678-9012-3456-78901234db5e
AssociationName     :
AssociationVersion  : 2
ComplianceSeverity :
CreatedDate        : 3/12/2019 9:21:01 AM
DocumentVersion    :
MaxConcurrency     :
MaxErrors          :
Name               : AWS-GatherSoftwareInventory
OutputLocation     :
```

```

Parameters      : {}
ScheduleExpression :
Targets         : {InstanceIds}

AssociationId    : 123a45a0-c678-9012-3456-78901234db5e
AssociationName  : test-case-1234567890
AssociationVersion : 1
ComplianceSeverity :
CreatedDate     : 3/2/2019 8:53:29 AM
DocumentVersion :
MaxConcurrency   :
MaxErrors        :
Name            : AWS-GatherSoftwareInventory
OutputLocation   :
Parameters      : {}
ScheduleExpression : rate(30minutes)
Targets         : {InstanceIds}

```

- Per i dettagli sull'API, vedere [ListAssociationVersions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMAutomationExecution

Il seguente esempio di codice mostra come utilizzare. Get-SSMAutomationExecution

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza i dettagli di un'esecuzione di automazione.

```
Get-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

Output:

```

AutomationExecutionId    : 4105a4fc-f944-11e6-9d32-8fb2db27a909
AutomationExecutionStatus : Failed
DocumentName             : AWS-UpdateLinuxAmi
DocumentVersion          : 1
ExecutionEndTime         : 2/22/2017 9:17:08 PM
ExecutionStartTime       : 2/22/2017 9:17:02 PM
FailureMessage           : Step launchInstance failed maximum allowed times. You
                          are not authorized to perform this operation. Encoded

```

```

        authorization failure message:
    B_V2QyyN7NhSZQYpmVzpEc4oSnj2GLTNYnXUHsTbqJkNMoDgubmbtthLmZyaiUYek0RIrA42-
    fv1x-04q5Fjff6glh
        Yb6TI5b0GQeeNrpwNvpDzm0-
    PSR1swlAbg9fdM9BcNjyrznsPukWpuKu9EC10u6v30XU1KC9nZ7mPlWMFZNkSioQqpWwEvMw-
    GZktsQzm67q0hUhBN0LWYhbS
        pkfiqzY-5nw3S0obx30fhd3EJa50_-
    GjV_a0nFXQJa70ik40bF0rEh3MtCSbrQT6--DvFy_FQ8TKvkIXadyVskeJI84X0F5WmA60f1pi5GI08i-
    nRfZS6oDeU
        gELBjjoFKD8s3L2aI0B6umWVxnQ0jqhQRxwJ53b54sZJ2PW3v_mtg9-
    q0CK0ezS3xfh_y0ilaUG0AZG-xjQFuvU_JZedWpla3xi-MZsmb1AifBI
        (Service: AmazonEC2; Status Code: 403; Error Code:
    UnauthorizedOperation; Request ID:
        6a002f94-ba37-43fd-99e6-39517715fce5)
Outputs      : {[createImage.ImageId,
    Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
Parameters   : {[AutomationAssumeRole,
    Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]], [InstanceIamRole,
    Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]], [SourceAmiId,
    Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
StepExecutions : {launchInstance, updateOSSoftware, stopInstance,
    createImage...}

```

Esempio 2: questo esempio elenca i dettagli della fase per l'ID di esecuzione dell'automazione specificato

```

Get-SSMAutomationExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object -ExpandProperty StepExecutions | Select-Object
    StepName, Action, StepStatus, ValidNextSteps

```

Output:

StepName	Action	StepStatus	ValidNextSteps
-----	-----	-----	-----
LaunchInstance	aws:runInstances	Success	{OSCompatibilityCheck}
OSCompatibilityCheck	aws:runCommand	Success	{RunPreUpdateScript}
RunPreUpdateScript	aws:runCommand	Success	{UpdateEC2Config}
UpdateEC2Config	aws:runCommand	Cancelled	{}
UpdateSSMAgent	aws:runCommand	Pending	{}
UpdateAWSPVDriver	aws:runCommand	Pending	{}
UpdateAWSEnaNetworkDriver	aws:runCommand	Pending	{}

UpdateAWSNVMe	aws:runCommand	Pending	{}
InstallWindowsUpdates	aws:runCommand	Pending	{}
RunPostUpdateScript	aws:runCommand	Pending	{}
RunSysprepGeneralize	aws:runCommand	Pending	{}
StopInstance	aws:changeInstanceState	Pending	{}
CreateImage	aws:createImage	Pending	{}
TerminateInstance	aws:changeInstanceState	Pending	{}

- Per i dettagli sull'API, vedere [GetAutomationExecution](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMAutomationExecutionList

Il seguente esempio di codice mostra come utilizzare. Get-SSMAutomationExecutionList

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive tutte le esecuzioni di automazione attive e terminate associate all'account.

```
Get-SSMAutomationExecutionList
```

Output:

```
AutomationExecutionId      : 4105a4fc-f944-11e6-9d32-8fb2db27a909
AutomationExecutionStatus  : Failed
DocumentName               : AWS-UpdateLinuxAmi
DocumentVersion            : 1
ExecutedBy                 : admin
ExecutionEndTime           : 2/22/2017 9:17:08 PM
ExecutionStartTime         : 2/22/2017 9:17:02 PM
LogFile                   :
Outputs                    : {[createImage.ImageId,
    Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
```

Esempio 2: Questo esempio visualizza ExecutionID, document, start/end timestamp di esecuzione per le esecuzioni con valori diversi da «Success» AutomationExecutionStatus

```
Get-SSMAutomationExecutionList | Where-Object AutomationExecutionStatus
-ne "Success" | Select-Object AutomationExecutionId, DocumentName,
```



```
AutomationExecutionStatus, ExecutionStartTime, ExecutionEndTime | Format-Table -
AutoSize
```

Output:

AutomationExecutionId	DocumentName	AutomationExecutionStatus	ExecutionStartTime	ExecutionEndTime
e1d2bad3-4567-8901-ae23-456c7c8901be	AWS-UpdateWindowsAmi	Cancelled	4/16/2019 5:37:04 AM	4/16/2019 5:47:29 AM
61234567-a7f8-90e1-2b34-567b8bf9012c	Fixed-UpdateAmi	Cancelled	4/16/2019 5:33:04 AM	4/16/2019 5:40:15 AM
91234d56-7e89-0ac1-2aee-34ea5d6a7c89	AWS-UpdateWindowsAmi		4/16/2019 5:22:46 AM	4/16/2019 5:27:29 AM

Failed

- Per i dettagli sull'API, vedere [DescribeAutomationExecutions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMAutomationStepExecution

Il seguente esempio di codice mostra come utilizzare. Get-SSMAutomationStepExecution

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza le informazioni relative a tutte le esecuzioni della fase attive e terminate in un flusso di lavoro di automazione.

```
Get-SSMAutomationStepExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object StepName, Action, StepStatus
```

Output:

StepName	Action	StepStatus
LaunchInstance	aws:runInstances	Success
OSCompatibilityCheck	aws:runCommand	Success
RunPreUpdateScript	aws:runCommand	Success
UpdateEC2Config	aws:runCommand	Cancelled
UpdateSSMAgent	aws:runCommand	Pending
UpdateAWSPVDriver	aws:runCommand	Pending
UpdateAWSEnaNetworkDriver	aws:runCommand	Pending

UpdateAWSNVMe	aws:runCommand	Pending
InstallWindowsUpdates	aws:runCommand	Pending
RunPostUpdateScript	aws:runCommand	Pending
RunSysprepGeneralize	aws:runCommand	Pending
StopInstance	aws:changeInstanceState	Pending
CreateImage	aws:createImage	Pending
TerminateInstance	aws:changeInstanceState	Pending

- Per i dettagli sull'API, vedere [DescribeAutomationStepExecutions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMAvailablePatch

Il seguente esempio di codice mostra come utilizzare. Get-SSMAvailablePatch

Strumenti per PowerShell V4

Esempio 1: questo esempio fa ottenere tutte le patch per Windows Server 2012 che presentano una gravità MSRC pari a Critica. La sintassi utilizzata in questo esempio richiede la PowerShell versione 3 o successiva.

```
$filter1 = @{Key="PRODUCT";Values=@"WindowsServer2012"}
$filter2 = @{Key="MSRC_SEVERITY";Values=@"Critical"}

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

Output:

```
Classification : SecurityUpdates
ContentUrl      : https://support.microsoft.com/en-us/kb/2727528
Description     : A security issue has been identified that could allow an
                  unauthenticated remote attacker to compromise your system and gain control
                  over it. You can help protect your system by installing this update
                  from Microsoft. After you install this update, you may have to
                  restart your system.
Id              : 1eb507be-2040-4eeb-803d-abc55700b715
KbNumber        : KB2727528
Language        : All
MsrcNumber      : MS12-072
MsrcSeverity    : Critical
Product         : WindowsServer2012
ProductFamily   : Windows
```

```
ReleaseDate      : 11/13/2012 6:00:00 PM
Title            : Security Update for Windows Server 2012 (KB2727528)
Vendor           : Microsoft
...
```

Esempio 2: con PowerShell la versione 2, è necessario utilizzare New-Object per creare ogni filtro.

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "PRODUCT"
$filter1.Values = "WindowsServer2012"
$filter2 = New-Object Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter2.Key = "MSRC_SEVERITY"
$filter2.Values = "Critical"

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

Esempio 3: questo esempio recupera tutti gli aggiornamenti rilasciati negli ultimi 20 giorni e applicabili ai prodotti corrispondenti al 2019 WindowsServer

```
Get-SSMAvailablePatch | Where-Object ReleaseDate -ge (Get-Date).AddDays(-20) |
Where-Object Product -eq "WindowsServer2019" | Select-Object ReleaseDate, Product,
Title
```

Output:

ReleaseDate	Product	Title
-----	-----	-----
4/9/2019 5:00:12 PM	WindowsServer2019	2019-04 Security Update for Adobe Flash Player for Windows Server 2019 for x64-based Systems (KB4493478)
4/9/2019 5:00:06 PM	WindowsServer2019	2019-04 Cumulative Update for Windows Server 2019 for x64-based Systems (KB4493509)
4/2/2019 5:00:06 PM	WindowsServer2019	2019-03 Servicing Stack Update for Windows Server 2019 for x64-based Systems (KB4493510)

- Per i dettagli sull'API, vedere [DescribeAvailablePatches](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMCommand

Il seguente esempio di codice mostra come utilizzare. Get-SSMCommand

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutti i comandi richiesti.

```
Get-SSMCommand
```

Output:

```
CommandId      : 4b75a163-d39a-4d97-87c9-98ae52c6be35
Comment        : Apply association with id at update time: 4cc73e42-
d5ae-4879-84f8-57e09c0efcd0
CompletedCount : 1
DocumentName   : AWS-RefreshAssociation
ErrorCount     : 0
ExpiresAfter   : 2/24/2017 3:19:08 AM
InstanceIds    : {i-0cb2b964d3e14fd9f}
MaxConcurrency : 50
MaxErrors      : 0
NotificationConfig : Amazon.SimpleSystemsManagement.Model.NotificationConfig
OutputS3BucketName :
OutputS3KeyPrefix :
OutputS3Region  :
Parameters     : {[associationIds,
  Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
RequestedDateTime : 2/24/2017 3:18:08 AM
ServiceRole     :
Status          : Success
StatusDetails   : Success
TargetCount     : 1
Targets        : {}
```

Esempio 2: questo esempio ottiene lo stato di un comando specifico.

```
Get-SSMCommand -CommandId "4b75a163-d39a-4d97-87c9-98ae52c6be35"
```

Esempio 3: questo esempio recupera tutti i comandi SSM invocati dopo il 2019-04-01T00:00:00Z

```
Get-SSMCommand -Filter @{Key="InvokedAfter";Value="2019-04-01T00:00:00Z"} | Select-
Object CommandId, DocumentName, Status, RequestedDateTime | Sort-Object -Property
RequestedDateTime -Descending
```

Output:

CommandId	DocumentName	Status	
RequestedDateTime			
-----	-----	-----	

edb1b23e-456a-7adb-aef8-90e-012ac34f	AWS-RunPowerShellScript	Cancelled	4/16/2019
5:45:23 AM			
1a2dc3fb-4567-890d-a1ad-234b5d6bc7d9	AWS-ConfigureAWSPackage	Success	4/6/2019
9:19:42 AM			
12c3456c-7e90-4f12-1232-1234f5b67893	KT-Retrieve-Cloud-Type-Win	Failed	4/2/2019
4:13:07 AM			
fe123b45-240c-4123-a2b3-234bdd567ecf	AWS-RunInspecChecks	Failed	4/1/2019
2:27:31 PM			
1eb23aa4-567d-4123-12a3-4c1c2ab34561	AWS-RunPowerShellScript	Success	4/1/2019
1:05:55 PM			
1c2f3bb4-ee12-4bc1-1a23-12345eea123e	AWS-RunInspecChecks	Failed	4/1/2019
11:13:09 AM			

- Per i dettagli sull'API, vedere [ListCommands](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMCommandInvocation

Il seguente esempio di codice mostra come utilizzare. Get-SSMCommandInvocation

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le invocazioni di un comando.

```
Get-SSMCommandInvocation -CommandId "b8eac879-0541-439d-94ec-47a80d554f44" -Detail
>true
```

Output:

```
CommandId      : b8eac879-0541-439d-94ec-47a80d554f44
CommandPlugins : {aws:runShellScript}
Comment        : IP config
DocumentName   : AWS-RunShellScript
InstanceId     : i-0cb2b964d3e14fd9f
InstanceName   :
NotificationConfig : Amazon.SimpleSystemsManagement.Model.NotificationConfig
RequestedDateTime : 2/22/2017 8:13:16 PM
```

```
ServiceRole      :
StandardErrorUrl :
StandardOutputUrl :
Status           : Success
StatusDetails    : Success
TraceOutput      :
```

Esempio 2: Questo esempio elenca le modalità di invocazione dell'id di comando CommandPlugins e1eb2e3c-ed4c-5123-45c1-234f5612345f

```
Get-SSMCommandInvocation -CommandId e1eb2e3c-ed4c-5123-45c1-234f5612345f -Detail:
$true | Select-Object -ExpandProperty CommandPlugins
```

Output:

```
Name                : aws:runPowerShellScript
Output              : Completed 17.7 KiB/17.7 KiB (40.1 KiB/s) with 1 file(s)
                    remainingdownload: s3://dd-aess-r-ctmer/KUM0.png to ..\..\programdata\KUM0.png
                    kumo available

OutputS3BucketName  :
OutputS3KeyPrefix   :
OutputS3Region      : eu-west-1
ResponseCode        : 0
ResponseFinishDateTime : 4/3/2019 11:53:23 AM
ResponseStartDateTime : 4/3/2019 11:53:21 AM
StandardErrorUrl    :
StandardOutputUrl   :
Status              : Success
StatusDetails       : Success
```

- AWS Strumenti per PowerShell Per i dettagli sull'API, vedere in Cmdlet Reference (V4).
[ListCommandInvocations](#)

Get-SSMCommandInvocationDetail

Il seguente esempio di codice mostra come utilizzare. Get-SSMCommandInvocationDetail

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza i dettagli di un comando eseguito su un'istanza.

```
Get-SSMCommandInvocationDetail -InstanceId "i-0cb2b964d3e14fd9f" -CommandId
"b8eac879-0541-439d-94ec-47a80d554f44"
```

Output:

```
CommandId           : b8eac879-0541-439d-94ec-47a80d554f44
Comment             : IP config
DocumentName        : AWS-RunShellScript
ExecutionElapsedTime : PT0.004S
ExecutionEndDateTime : 2017-02-22T20:13:16.651Z
ExecutionStartDateTime : 2017-02-22T20:13:16.651Z
InstanceId           : i-0cb2b964d3e14fd9f
PluginName           : aws:runShellScript
ResponseCode         : 0
StandardErrorContent : 
StandardErrorUrl      : 
StandardOutputContent : 
StandardOutputUrl     : 
Status               : Success
StatusDetails        : Success
```

- Per i dettagli sull'API, vedere [GetCommandInvocation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMComplianceItemList

Il seguente esempio di codice mostra come utilizzare. Get-SSMComplianceItemList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca l'elenco degli elementi di conformità per l'ID e il tipo di risorsa specificati, filtrando il tipo di conformità utilizzando "Associazione"

```
Get-SSMComplianceItemList -ResourceId i-1a2caf345f67d0dc2 -ResourceType
ManagedInstance -Filter @{Key="ComplianceType";Values="Association"}
```

Output:

```
ComplianceType      : Association
Details              : {[DocumentName, AWS-GatherSoftwareInventory], [DocumentVersion,
1]}
```

```

ExecutionSummary : Amazon.SimpleSystemsManagement.Model.ComplianceExecutionSummary
Id                : 123a45a1-c234-1234-1245-67891236db4e
ResourceId        : i-1a2caf345f67d0dc2
ResourceType      : ManagedInstance
Severity          : UNSPECIFIED
Status            : COMPLIANT
Title             :

```

- Per i dettagli sull'API, vedere [ListComplianceItems](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMComplianceSummaryList

Il seguente esempio di codice mostra come utilizzare. Get-SSMComplianceSummaryList

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce un conteggio riepilogativo delle risorse conformi e non conformi per tutti i tipi di conformità.

```
Get-SSMComplianceSummaryList
```

Output:

```

ComplianceType CompliantSummary
NonCompliantSummary
-----
-----
FleetTotal      Amazon.SimpleSystemsManagement.Model.CompliantSummary
Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Association      Amazon.SimpleSystemsManagement.Model.CompliantSummary
Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Custom:InSpec   Amazon.SimpleSystemsManagement.Model.CompliantSummary
Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Patch           Amazon.SimpleSystemsManagement.Model.CompliantSummary
Amazon.SimpleSystemsManagement.Model.NonCompliantSummary

```

- Per i dettagli sull'API, vedere [ListComplianceSummaries](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMConnectionStatus

Il seguente esempio di codice mostra come utilizzare. Get-SSMConnectionStatus

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera lo stato della connessione di Session Manager per un'istanza per determinare se è connessa e pronta a ricevere connessioni a Session Manager.

```
Get-SSMConnectionStatus -Target i-0a1caf234f12d3dc4
```

Output:

```
Status      Target
-----
Connected i-0a1caf234f12d3dc4
```

- Per i dettagli sull'API, vedere [GetConnectionStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMDefaultPatchBaseline

Il seguente esempio di codice mostra come utilizzare. Get-SSMDefaultPatchBaseline

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza la baseline delle patch predefinita.

```
Get-SSMDefaultPatchBaseline
```

Output:

```
arn:aws:ssm:us-west-2:123456789012:patchbaseline/pb-04fb4ae6142167966
```

- Per i dettagli sull'API, vedere [GetDefaultPatchBaseline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMDeployablePatchSnapshotForInstance

Il seguente esempio di codice mostra come utilizzare. Get-SSMDeployablePatchSnapshotForInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio visualizza l'istantanea corrente per la baseline delle patch utilizzata da un'istanza. Questo comando deve essere eseguito dall'istanza utilizzando le credenziali dell'istanza. Per garantire che utilizzi le credenziali dell'istanza, l'esempio passa un oggetto **Amazon.Runtime.InstanceProfileAWSCredentials** al parametro Credentials.

```
$credentials = [Amazon.Runtime.InstanceProfileAWSCredentials]::new()
Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f" -Credentials $credentials
```

Output:

```
InstanceId          SnapshotDownloadUrl
-----
i-0cb2b964d3e14fd9f https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...1692/4681775b-098f-4435...
```

Esempio 2: Questo esempio mostra come ottenere il massimo SnapshotDownloadUrl. Questo comando deve essere eseguito dall'istanza utilizzando le credenziali dell'istanza. Per garantire che utilizzi le credenziali dell'istanza, l'esempio configura la PowerShell sessione per l'utilizzo di un **Amazon.Runtime.InstanceProfileAWSCredentials** oggetto.

```
Set-AWSCredential -Credential
([Amazon.Runtime.InstanceProfileAWSCredentials]::new())
(Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f").SnapshotDownloadUrl
```

Output:

```
https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...
```

- Per i dettagli sull'API, vedere [GetDeployablePatchSnapshotForInstance](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMDocument

Il seguente esempio di codice mostra come utilizzare. Get-SSMDocument

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce il contenuto di un documento.

```
Get-SSMDocument -Name "RunShellScript"
```

Output:

```
Content  
-----  
{...
```

Esempio 2: questo esempio visualizza il contenuto completo di un documento.

```
(Get-SSMDocument -Name "RunShellScript").Content  
{  
  "schemaVersion":"2.0",  
  "description":"Run an updated script",  
  "parameters":{  
    "commands":{  
      "type":"StringList",  
      "description":"(Required) Specify a shell script or a command to run.",  
      "minItems":1,  
      "displayType":"textarea"  
    }  
  },  
  "mainSteps":[  
    {  
      "action":"aws:runShellScript",  
      "name":"runShellScript",  
      "inputs":{  
        "commands":"{{ commands }}"  
      }  
    },  
    {  
      "action":"aws:runPowerShellScript",  
      "name":"runPowerShellScript",  
      "inputs":{
```

```
        "commands": "{{ commands }}"
    }
}
]
```

- Per i dettagli sull'API, vedere [GetDocument](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMDocumentDescription

Il seguente esempio di codice mostra come utilizzare `Get-SSMDocumentDescription`

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce informazioni su un documento.

```
Get-SSMDocumentDescription -Name "RunShellScript"
```

Output:

```
CreatedDate      : 2/24/2017 5:25:13 AM
DefaultVersion   : 1
Description      : Run an updated script
DocumentType     : Command
DocumentVersion  : 1
Hash             : f775e5df4904c6fa46686c4722fae9de1950dace25cd9608ff8d622046b68d9b
HashType        : Sha256
LatestVersion    : 1
Name            : RunShellScript
Owner           : 123456789012
Parameters       : {commands}
PlatformTypes    : {Linux}
SchemaVersion    : 2.0
Sha1             :
Status          : Active
```

- Per i dettagli sull'API, vedere [DescribeDocument](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMDocumentList

Il seguente esempio di codice mostra come utilizzare. Get-SSMDocumentList

Strumenti per PowerShell V4

Esempio 1: elenca tutti i documenti di configurazione dell'account.

```
Get-SSMDocumentList
```

Output:

```
DocumentType      : Command
DocumentVersion   : 1
Name              : AWS-ApplyPatchBaseline
Owner            : Amazon
PlatformTypes     : {Windows}
SchemaVersion     : 1.2

DocumentType      : Command
DocumentVersion   : 1
Name              : AWS-ConfigureAWSPackage
Owner            : Amazon
PlatformTypes     : {Windows, Linux}
SchemaVersion     : 2.0

DocumentType      : Command
DocumentVersion   : 1
Name              : AWS-ConfigureCloudWatch
Owner            : Amazon
PlatformTypes     : {Windows}
SchemaVersion     : 1.2
...
```

Esempio 2: questo esempio recupera tutti i documenti di automazione il cui nome corrisponde a "Piattaforma"

```
Get-SSMDocumentList -DocumentFilterList @{Key="DocumentType";Value="Automation"} |
Where-Object Name -Match "Platform"
```

Output:

```

DocumentFormat : JSON
DocumentType    : Automation
DocumentVersion : 7
Name            : KT-Get-Platform
Owner           : 987654123456
PlatformTypes  : {Windows, Linux}
SchemaVersion   : 0.3
Tags            : {}
TargetType      :
VersionName     :

```

- Per i dettagli sull'API, vedere [ListDocuments](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMDocumentPermission

Il seguente esempio di codice mostra come utilizzare. Get-SSMDocumentPermission

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le versioni di un documento.

```
Get-SSMDocumentVersionList -Name "RunShellScript"
```

Output:

CreatedDate	DocumentVersion	IsDefaultVersion	Name
-----	-----	-----	----
2/24/2017 5:25:13 AM	1	True	RunShellScript

- Per i dettagli sull'API, vedere [DescribeDocumentPermission](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMDocumentVersionList

Il seguente esempio di codice mostra come utilizzare. Get-SSMDocumentVersionList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le versioni di un documento.

```
Get-SSMDocumentVersionList -Name "AWS-UpdateSSMAgent"
```

Output:

```

CreatedDate      : 6/1/2021 5:19:10 PM
DocumentFormat   : JSON
DocumentVersion  : 1
IsDefaultVersion : True
Name             : AWS-UpdateSSMAgent
Status          : Active

```

- Per i dettagli sull'API, vedere [ListDocumentVersions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMEffectiveInstanceAssociationList

Il seguente esempio di codice mostra come utilizzare. Get-SSMEffectiveInstanceAssociationList

Strumenti per PowerShell V4

Esempio 1: questo esempio descrive le associazioni efficaci per un'istanza.

```
Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -MaxResult 5
```

Output:

```

AssociationId      Content
-----
d8617c07-2079-4c18-9847-1655fc2698b0 {...

```

Esempio 2: questo esempio visualizza il contenuto delle associazioni efficaci per un'istanza.

```
(Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -MaxResult 5).Content
```

Output:

```
{
  "schemaVersion": "1.2",
  "description": "Update the Amazon SSM Agent to the latest version or specified
version.",
  "parameters": {
    "version": {
      "default": "",
      "description": "(Optional) A specific version of the Amazon SSM Agent to
install. If not specified, the agen
t will be updated to the latest version.",
      "type": "String"
    },
    "allowDowngrade": {
      "default": "false",
      "description": "(Optional) Allow the Amazon SSM Agent service to be
downgraded to an earlier version. If set
to false, the service can be upgraded to newer versions only (default). If set to
true, specify the earlier version.",
      "type": "String",
      "allowedValues": [
        "true",
        "false"
      ]
    }
  },
  "runtimeConfig": {
    "aws:updateSsmAgent": {
      "properties": [
        {
          "agentName": "amazon-ssm-agent",
          "source": "https://s3.{Region}.amazonaws.com/amazon-ssm-{Region}/
ssm-agent-manifest.json",
          "allowDowngrade": "{{ allowDowngrade }}",
          "targetVersion": "{{ version }}"
        }
      ]
    }
  }
}
```

- Per i dettagli sull'API, vedere [DescribeEffectiveInstanceAssociations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMEffectivePatchesForPatchBaseline

Il seguente esempio di codice mostra come utilizzare. Get-SSMEffectivePatchesForPatchBaseline

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le baseline delle patch, con un elenco di risultati massimo di 1.

```
Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1
```

Output:

Patch	PatchStatus
-----	-----
Amazon.SimpleSystemsManagement.Model.Patch	
Amazon.SimpleSystemsManagement.Model.PatchStatus	

Esempio 2: questo esempio visualizza lo stato delle patch per tutte baseline delle patch, con un elenco di risultati massimo di 1.

```
(Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1).PatchStatus
```

Output:

ApprovalDate	DeploymentStatus
-----	-----
12/21/2010 6:00:00 PM	APPROVED

- Per i dettagli sull'API, vedere [DescribeEffectivePatchesForPatchBaseline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMInstanceAssociationsStatus

Il seguente esempio di codice mostra come utilizzare. Get-SSMInstanceAssociationsStatus

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra i dettagli delle associazioni per un'istanza.

```
Get-SSMInstanceAssociationsStatus -InstanceId "i-0000293ffd8c57862"
```

Output:

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DetailedStatus    : Pending
DocumentVersion   : 1
ErrorCode         :
ExecutionDate      : 2/20/2015 8:31:11 AM
ExecutionSummary   : temp_status_change
InstanceId        : i-0000293ffd8c57862
Name              : AWS-UpdateSSMAgent
OutputUrl         :
Status            : Pending
```

Esempio 2: questo esempio controlla lo stato di associazione delle istanze per l'ID di istanza specificato e inoltre visualizza lo stato di esecuzione di tali associazioni

```
Get-SSMInstanceAssociationsStatus -InstanceId i-012e3cb4df567e8aa | ForEach-Object
{Get-SSMAssociationExecution -AssociationId .AssociationId}
```

Output:

```
AssociationId      : 512a34a5-c678-1234-1234-12345678db9e
AssociationVersion  : 2
CreatedTime        : 3/2/2019 8:53:29 AM
DetailedStatus     :
ExecutionId        : 512a34a5-c678-1234-1234-12345678db9e
LastExecutionDate   : 1/1/0001 12:00:00 AM
ResourceCountByStatus : {Success=9}
Status             : Success
```

- Per i dettagli sull'API, vedere [DescribeInstanceAssociationsStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMInstanceInformation

Il seguente esempio di codice mostra come utilizzare. Get-SSMInstanceInformation

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra i dettagli di ciascuna istanza.

```
Get-SSMInstanceInformation
```

Output:

```

ActivationId                :
AgentVersion                 : 2.0.672.0
AssociationOverview         :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus            : Success
ComputerName                 : ip-172-31-44-222.us-west-2.compute.internal
IamRole                      :
InstanceId                   : i-0cb2b964d3e14fd9f
IPAddress                    : 172.31.44.222
IsLatestVersion              : True
LastAssociationExecutionDate : 2/24/2017 3:18:09 AM
LastPingDateTime             : 2/24/2017 3:35:03 AM
LastSuccessfulAssociationExecutionDate : 2/24/2017 3:18:09 AM
Name                         :
PingStatus                   : ConnectionLost
PlatformName                 : Amazon Linux AMI
PlatformType                  : Linux
PlatformVersion               : 2016.09
RegistrationDate              : 1/1/0001 12:00:00 AM
ResourceType                  : EC2Instance

```

Esempio 2: Questo esempio mostra come utilizzare il parametro -Filter per filtrare i risultati solo per le istanze di AWS Systems Manager nella regione **us-east-1** con un **AgentVersion** of. **2.2.800.0** È possibile trovare un elenco di valori di chiave -Filter validi nell'argomento di riferimento dell' InstanceInformation API (https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformation.html#systemsmanager-Type-). InstanceInformation ActivationId

```
$Filters = @{
```

```

    Key="AgentVersion"
    Values="2.2.800.0"
}
Get-SSMInstanceInformation -Region us-east-1 -Filter $Filters

```

Output:

```

ActivationId                :
AgentVersion                 : 2.2.800.0
AssociationOverview         :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus           : Success
ComputerName                 : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                      :
InstanceId                   : i-EXAMPLEb0792d98ce
IPAddress                   : 10.0.0.01
IsLatestVersion             : False
LastAssociationExecutionDate : 8/16/2018 12:02:50 AM
LastPingDateTime            : 8/16/2018 7:40:27 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:02:50 AM
Name                         :
PingStatus                   : Online
PlatformName                 : Microsoft Windows Server 2016 Datacenter
PlatformType                 : Windows
PlatformVersion              : 10.0.14393
RegistrationDate             : 1/1/0001 12:00:00 AM
ResourceType                 : EC2Instance

ActivationId                :
AgentVersion                 : 2.2.800.0
AssociationOverview         :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus           : Success
ComputerName                 : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                      :
InstanceId                   : i-EXAMPLEac7501d023
IPAddress                   : 10.0.0.02
IsLatestVersion             : False
LastAssociationExecutionDate : 8/16/2018 12:00:20 AM
LastPingDateTime            : 8/16/2018 7:40:35 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:00:20 AM
Name                         :
PingStatus                   : Online

```

```

PlatformName           : Microsoft Windows Server 2016 Datacenter
PlatformType           : Windows
PlatformVersion        : 10.0.14393
RegistrationDate       : 1/1/0001 12:00:00 AM
ResourceType           : EC2Instance

```

Esempio 3: Questo esempio mostra come utilizzare il `InstanceInformationFilterList` parametro - per filtrare i risultati solo in base alle istanze di AWS Systems Manager nella regione **us-east-1** con **PlatformTypes** of **Windows** o **Linux**. È possibile trovare un elenco di valori `InstanceInformationFilterList` chiave validi nell'argomento di riferimento dell'`InstanceInformationFilter` API (https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformationFilter.html).

```

$Filters = @{
    Key="PlatformTypes"
    ValueSet=("Windows","Linux")
}
Get-SSMInstanceInformation -Region us-east-1 -InstanceInformationFilterList $Filters

```

Output:

```

ActivationId           :
AgentVersion           : 2.2.800.0
AssociationOverview    :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus      : Success
ComputerName           : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                :
InstanceId             : i-EXAMPLEb0792d98ce
IPAddress              : 10.0.0.27
IsLatestVersion       : False
LastAssociationExecutionDate : 8/16/2018 12:02:50 AM
LastPingDateTime       : 8/16/2018 7:40:27 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:02:50 AM
Name                  :
PingStatus            : Online
PlatformName          : Ubuntu Server 18.04 LTS
PlatformType          : Linux
PlatformVersion        : 18.04
RegistrationDate       : 1/1/0001 12:00:00 AM
ResourceType           : EC2Instance

```

```

ActivationId           :
AgentVersion           : 2.2.800.0
AssociationOverview    :
  Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus      : Success
ComputerName           : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                :
InstanceId             : i-EXAMPLEac7501d023
IPAddress              : 10.0.0.100
IsLatestVersion        : False
LastAssociationExecutionDate : 8/16/2018 12:00:20 AM
LastPingDateTime       : 8/16/2018 7:40:35 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:00:20 AM
Name                  :
PingStatus            : Online
PlatformName          : Microsoft Windows Server 2016 Datacenter
PlatformType          : Windows
PlatformVersion        : 10.0.14393
RegistrationDate       : 1/1/0001 12:00:00 AM
ResourceType          : EC2Instance

```

Esempio 4: questo esempio elenca le istanze e le esportazioni InstanceId gestite da ssm LastPingDateTime e in un file PlatformName csv. PingStatus

```

Get-SSMInstanceInformation | Select-Object InstanceId, PingStatus, LastPingDateTime,
PlatformName | Export-Csv Instance-details.csv -NoTypeInfoation

```

- Per i dettagli sull'API, vedere [DescribeInstanceInformation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMInstancePatch

Il seguente esempio di codice mostra come utilizzare. Get-SSMInstancePatch

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene i dettagli sulla conformità delle patch per un'istanza.

```

Get-SSMInstancePatch -InstanceId "i-08ee91c0b17045407"

```

- Per i dettagli sull'API, vedere [DescribeInstancePatches](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMInstancePatchState

Il seguente esempio di codice mostra come utilizzare. Get-SSMInstancePatchState

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene gli stati di riepilogo delle patch per un'istanza.

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407"
```

Esempio 2: questo esempio ottiene gli stati di riepilogo delle patch per due istanze.

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407","i-09a618aec652973a9"
```

- Per i dettagli sull'API, vedere [DescribeInstancePatchStates](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMInstancePatchStatesForPatchGroup

Il seguente esempio di codice mostra come utilizzare. Get-SSMInstancePatchStatesForPatchGroup

Strumenti per PowerShell V4

Esempio 1: Questo esempio ottiene gli stati del riepilogo delle patch per istanza per un gruppo di patch.

```
Get-SSMInstancePatchStatesForPatchGroup -PatchGroup "Production"
```

- Per i dettagli sull'API, vedere [DescribeInstancePatchStatesForPatchGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMInventory

Il seguente esempio di codice mostra come utilizzare. Get-SSMInventory

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene i metadati personalizzati per l'inventario.

Get-SSMInventory

Output:

```

Data
  Id
----
--
{[AWS:InstanceInformation,
 Amazon.SimpleSystemsManagement.Model.InventoryResultItem]} i-0cb2b964d3e14fd9f

```

- Per i dettagli sull'API, vedere [GetInventory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMInventoryEntriesList

Il seguente esempio di codice mostra come utilizzare. Get-SSMInventoryEntriesList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le voci di inventario personalizzate per un'istanza.

```

Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
"Custom:RackInfo"

```

Output:

```

CaptureTime    : 2016-08-22T10:01:01Z
Entries        :
{Amazon.Runtime.Internal.Util.AlwaysSendDictionary`2[System.String,System.String]}
InstanceId     : i-0cb2b964d3e14fd9f
NextToken      :
SchemaVersion  : 1.0
TypeName       : Custom:RackInfo

```

Esempio 2: questo esempio elenca i dettagli.

```

(Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
"Custom:RackInfo").Entries

```


Output:

Key	Value
---	-----
RackLocation	Bay B/Row C/Rack D/Shelf E

- Per i dettagli sull'API, vedere [ListInventoryEntries](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMInventoryEntryList

Il seguente esempio di codice mostra come utilizzare. Get-SSMInventoryEntryList

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera le voci di inventario di tipo **AWS:Network** per l'istanza.

```
Get-SSMInventoryEntryList -InstanceId mi-088dcb0ecea37b076 -TypeName AWS:Network |
Select-Object -ExpandProperty Entries
```

Output:

Key	Value
---	-----
DHCPServer	172.31.11.2
DNSServer	172.31.0.1
Gateway	172.31.11.2
IPV4	172.31.11.222
IPV6	fe12::3456:7da8:901a:12a3
MacAddress	1A:23:4E:5B:FB:67
Name	Amazon Elastic Network Adapter
SubnetMask	255.255.240.0

- Per i dettagli sull'API, vedere [Get-SSMInventoryEntryList](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMInventorySchema

Il seguente esempio di codice mostra come utilizzare. Get-SSMInventorySchema

Strumenti per PowerShell V4

Esempio 1: questo esempio restituisce un elenco di nomi dei tipi di inventario per l'account.

```
Get-SSMInventorySchema
```

- Per i dettagli sull'API, vedere [GetInventorySchema](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMLatestEC2Image

Il seguente esempio di codice mostra come utilizzare. Get-SSMLatestEC2Image

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le versioni più recenti di Windows. AMIs

```
PS Get-SSMLatestEC2Image -Path ami-windows-latest
```

Output:

Name	Value
----	-----
Windows_Server-2008-R2_SP1-English-64Bit-SQL_2012_SP4_Express ami-0e5ddd288daff4fab	
Windows_Server-2012-R2_RTM-Chinese_Simplified-64Bit-Base ami-0c5ea64e6bec1cb50	
Windows_Server-2012-R2_RTM-Chinese_Traditional-64Bit-Base ami-09775eff0bf8c113d	
Windows_Server-2012-R2_RTM-Dutch-64Bit-Base ami-025064b67e28cf5df	
...	

Esempio 2: questo esempio recupera l'ID AMI di un'immagine Amazon Linux specifica per la Regione us-west-2.

```
PS Get-SSMLatestEC2Image -Path ami-amazon-linux-latest -ImageName amzn-ami-hvm-x86_64-ebs -Region us-west-2
```

Output:

```
ami-09b92cd132204c704
```

Esempio 3: Questo esempio elenca tutte le ultime finestre AMIs che corrispondono all'espressione con caratteri jolly specificata.

```
Get-SSMLatestEC2Image -Path ami-windows-latest -ImageName *Windows*2019*English*
```

Output:

Name	Value
----	----
Windows_Server-2019-English-Full-SQL_2017_Web	ami-085e9d27da5b73a42
Windows_Server-2019-English-STIG-Core	ami-0bfd85c29148c7f80
Windows_Server-2019-English-Full-SQL_2019_Web	ami-02099560d7fb11f20
Windows_Server-2019-English-Full-SQL_2016_SP2_Standard	ami-0d7ae2d81c07bd598
...	

- Per i dettagli sull'API, vedere [Get-SSMLatest EC2 Image](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMMaintenanceWindow

Il seguente esempio di codice mostra come utilizzare. Get-SSMMaintenanceWindow

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene dettagli su una finestra di manutenzione.

```
Get-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d"
```

Output:

```
AllowUnassociatedTargets : False
CreatedDate               : 2/20/2017 6:14:05 PM
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
ModifiedDate             : 2/20/2017 6:14:05 PM
Name                     : TestMaintWin
Schedule                 : cron(0 */30 * * * ? *)
```

```
WindowId : mw-03eb9db42890fb82d
```

- Per i dettagli sull'API, vedere [GetMaintenanceWindow](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMMaintenanceWindowExecution

Il seguente esempio di codice mostra come utilizzare. Get-SSMMaintenanceWindowExecution

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le informazioni su un'attività eseguita nell'ambito di un'esecuzione di una finestra di manutenzione.

```
Get-SSMMaintenanceWindowExecution -WindowExecutionId "518d5565-5969-4cca-8f0e-da3b2a638355"
```

Output:

```
EndTime           : 2/21/2017 4:00:35 PM
StartTime          : 2/21/2017 4:00:34 PM
Status             : FAILED
StatusDetails      : One or more tasks in the orchestration failed.
TaskIds            : {ac0c6ae1-daa3-4a89-832e-d384503b6586}
WindowExecutionId  : 518d5565-5969-4cca-8f0e-da3b2a638355
```

- Per i dettagli sull'API, vedere [GetMaintenanceWindowExecution](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMMaintenanceWindowExecutionList

Il seguente esempio di codice mostra come utilizzare. Get-SSMMaintenanceWindowExecutionList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le esecuzioni per una finestra di manutenzione.

```
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d"
```

Output:

```
EndTime           : 2/20/2017 6:30:17 PM
StartTime         : 2/20/2017 6:30:16 PM
Status            : FAILED
StatusDetails     : One or more tasks in the orchestration failed.
WindowExecutionId : 6f3215cf-4101-4fa0-9b7b-9523269599c7
WindowId          : mw-03eb9db42890fb82d
```

Esempio 2: questo esempio elenca tutte le esecuzioni per una finestra di manutenzione prima di una data specificata.

```
$option1 = @{Key="ExecutedBefore";Values=@"2016-11-04T05:00:00Z"}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
$option1
```

Esempio 3: questo esempio elenca tutte le esecuzioni per una finestra di manutenzione dopo una data specificata.

```
$option1 = @{Key="ExecutedAfter";Values=@"2016-11-04T05:00:00Z"}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
$option1
```

- Per i dettagli sull'API, vedere [DescribeMaintenanceWindowExecutions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMMaintenanceWindowExecutionTask

Il seguente esempio di codice mostra come utilizzare. Get -SSMMaintenanceWindowExecutionTask

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le informazioni su un'attività che faceva parte dell'esecuzione di una finestra di manutenzione.

```
Get-SSMMaintenanceWindowExecutionTask -TaskId "ac0c6ae1-daa3-4a89-832e-d384503b6586"
-WindowExecutionId "518d5565-5969-4cca-8f0e-da3b2a638355"
```

Output:

```

EndTime           : 2/21/2017 4:00:35 PM
MaxConcurrency    : 1
MaxErrors         : 1
Priority          : 10
ServiceRole       : arn:aws:iam::123456789012:role/MaintenanceWindowsRole
StartTime         : 2/21/2017 4:00:34 PM
Status            : FAILED
StatusDetails     : The maximum error count was exceeded.
TaskArn           : AWS-RunShellScript
TaskExecutionId   : ac0c6ae1-daa3-4a89-832e-d384503b6586
TaskParameters    :
    {Amazon.Runtime.Internal.Util.AlwaysSendDictionary`2[System.String,Amazon.SimpleSystemsManag
        meterValueExpression]}
Type              : RUN_COMMAND
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355

```

- Per i dettagli sull'API, vedere [GetMaintenanceWindowExecutionTask](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMMaintenanceWindowExecutionTaskInvocationList

Il seguente esempio di codice mostra come utilizzare. Get-SSMMaintenanceWindowExecutionTaskInvocationList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le invocazioni per un'attività eseguita nell'ambito di un'esecuzione di una finestra di manutenzione.

```

Get-SSMMaintenanceWindowExecutionTaskInvocationList -TaskId "ac0c6ae1-
daa3-4a89-832e-d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"

```

Output:

```

EndTime           : 2/21/2017 4:00:34 PM
ExecutionId       :
InvocationId      : e274b6e1-fe56-4e32-bd2a-8073c6381d8b
OwnerInformation  :
Parameters       : {"documentName":"AWS-RunShellScript","instanceIds":
["i-0000293ffd8c57862"],"parameters":{"commands":["df"]},"maxConcurrency":"1",

```

```

        "maxErrors":"1"}
StartTime      : 2/21/2017 4:00:34 PM
Status         : FAILED
StatusDetails  : The instance IDs list contains an invalid entry.
TaskExecutionId : ac0c6ae1-daa3-4a89-832e-d384503b6586
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355
WindowTargetId  :

```

- Per i dettagli sull'API, vedere [DescribeMaintenanceWindowExecutionTaskInvocations](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMMaintenanceWindowExecutionTaskList

Il seguente esempio di codice mostra come utilizzare. Get-SSMMaintenanceWindowExecutionTaskList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le attività associate all'esecuzione di una finestra di manutenzione.

```

Get-SSMMaintenanceWindowExecutionTaskList -WindowExecutionId
"518d5565-5969-4cca-8f0e-da3b2a638355"

```

Output:

```

EndTime       : 2/21/2017 4:00:35 PM
StartTime     : 2/21/2017 4:00:34 PM
Status        : SUCCESS
TaskArn       : AWS-RunShellScript
TaskExecutionId : ac0c6ae1-daa3-4a89-832e-d384503b6586
TaskType      : RUN_COMMAND
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355

```

- Per i dettagli sull'API, vedere [DescribeMaintenanceWindowExecutionTasks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMMaintenanceWindowList

Il seguente esempio di codice mostra come utilizzare. Get-SSMMaintenanceWindowList

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le finestre di manutenzione dell'account.

```
Get-SSMMaintenanceWindowList
```

Output:

```
Cutoff      : 1
Duration    : 4
Enabled     : True
Name        : My-First-Maintenance-Window
WindowId    : mw-06d59c1a07c022145
```

- Per i dettagli sull'API, vedere [DescribeMaintenanceWindows](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMMaintenanceWindowTarget

Il seguente esempio di codice mostra come utilizzare. Get-SSMMaintenanceWindowTarget

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutti gli obiettivi per una finestra di manutenzione.

```
Get-SSMMaintenanceWindowTarget -WindowId "mw-06cf17cbefcb4bf4f"
```

Output:

```
OwnerInformation : Single instance
ResourceType     : INSTANCE
Targets          : {InstanceIds}
WindowId         : mw-06cf17cbefcb4bf4f
WindowTargetId   : 350d44e6-28cc-44e2-951f-4b2c985838f6

OwnerInformation : Two instances in a list
ResourceType     : INSTANCE
Targets          : {InstanceIds}
WindowId         : mw-06cf17cbefcb4bf4f
WindowTargetId   : e078a987-2866-47be-bedd-d9cf49177d3a
```


- Per i dettagli sull'API, vedere [DescribeMaintenanceWindowTargets](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMMaintenanceWindowTaskList

Il seguente esempio di codice mostra come utilizzare. Get-SSMMaintenanceWindowTaskList
Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le attività per una finestra di manutenzione.

```
Get-SSMMaintenanceWindowTaskList -WindowId "mw-06cf17cbefcb4bf4f"
```

Output:

```
LoggingInfo      :  
MaxConcurrency   : 1  
MaxErrors        : 1  
Priority          : 10  
ServiceRoleArn   : arn:aws:iam::123456789012:role/MaintenanceWindowsRole  
Targets          : {InstanceIds}  
TaskArn          : AWS-RunShellScript  
TaskParameters   : {[commands,  
  Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression]]  
Type             : RUN_COMMAND  
WindowId         : mw-06cf17cbefcb4bf4f  
WindowTaskId     : a23e338d-ff30-4398-8aa3-09cd052ebf17
```

- Per i dettagli sull'API, vedere [DescribeMaintenanceWindowTasks](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMParameterHistory

Il seguente esempio di codice mostra come utilizzare. Get-SSMParameterHistory
Strumenti per PowerShell V4

Esempio 1: questo esempio elenca la cronologia dei valori di un parametro.

```
Get-SSMParameterHistory -Name "Welcome"
```

Output:

```
Description      :  
KeyId            :  
LastModifiedDate : 3/3/2017 6:55:25 PM  
LastModifiedUser : arn:aws:iam::123456789012:user/admin  
Name             : Welcome  
Type             : String  
Value            : helloWorld
```

- Per i dettagli sull'API, vedere [GetParameterHistory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMParameterList

Il seguente esempio di codice mostra come utilizzare. `Get-SSMParameterList`

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutti i parametri.

```
Get-SSMParameterList
```

Output:

```
Description      :  
KeyId            :  
LastModifiedDate : 3/3/2017 6:58:23 PM  
LastModifiedUser : arn:aws:iam::123456789012:user/admin  
Name             : Welcome  
Type             : String
```

- Per i dettagli sull'API, vedere [DescribeParameters](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMParameterValue

Il seguente esempio di codice mostra come utilizzare. `Get-SSMParameterValue`

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i valori di un parametro.

```
Get-SSMParameterValue -Name "Welcome"
```

Output:

```
InvalidParameters Parameters
-----
{}                  {Welcome}
```

Esempio 2: questo esempio elenca i dettagli del valore.

```
(Get-SSMParameterValue -Name "Welcome").Parameters
```

Output:

```
Name      Type      Value
----      -
Welcome   String    Good day, Sunshine!
```

- Per i dettagli sull'API, vedere [GetParameters](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMPatchBaseline

Il seguente esempio di codice mostra come utilizzare. Get-SSMPatchBaseline

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca tutte le baseline delle patch.

```
Get-SSMPatchBaseline
```

Output:

```
BaselineDescription                                     BaselineId
-----
                                     BaselineName
-----
```

```
Default Patch Baseline Provided by AWS.                arn:aws:ssm:us-
west-2:123456789012:patchbaseline/pb-04fb4ae6142167966 AWS-DefaultP...
Baseline containing all updates approved for production systems pb-045f10b4f382baeda
Production-B...
Baseline containing all updates approved for production systems pb-0a2f1059b670ebd31
Production-B...
```

Esempio 2: Questo esempio elenca tutte le linee di base delle patch fornite da AWS. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
$filter1 = @{"Key"="OWNER";Values=@("AWS")}
```

Output:

```
Get-SSMPatchBaseline -Filter $filter1
```

Esempio 3: questo esempio elenca tutte le baseline delle patch con l'utente come proprietario. La sintassi utilizzata da questo esempio richiede la PowerShell versione 3 o successiva.

```
$filter1 = @{"Key"="OWNER";Values=@("Self")}
```

Output:

```
Get-SSMPatchBaseline -Filter $filter1
```

Esempio 4: con PowerShell la versione 2, è necessario utilizzare New-Object per creare ogni tag.

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "OWNER"
$filter1.Values = "AWS"

Get-SSMPatchBaseline -Filter $filter1
```

Output:

BaselineDescription	BaselineId
BaselineName	DefaultBaselin
	e

```
-----
Default Patch Baseline Provided by AWS. arn:aws:ssm:us-
west-2:123456789012:patchbaseline/pb-04fb4ae6142167966 AWS-DefaultPatchBaseline True
-----
```

- Per i dettagli sull'API, vedere [DescribePatchBaselines](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMPatchBaselineDetail

Il seguente esempio di codice mostra come utilizzare. Get-SSMPatchBaselineDetail

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra i dettagli di una baseline delle patch.

```
Get-SSMPatchBaselineDetail -BaselineId "pb-03da896ca3b68b639"
```

Output:

```
ApprovalRules    : Amazon.SimpleSystemsManagement.Model.PatchRuleGroup
ApprovedPatches  : {}
BaselineId       : pb-03da896ca3b68b639
CreatedDate      : 3/3/2017 5:02:19 PM
Description       : Baseline containing all updates approved for production systems
GlobalFilters    : Amazon.SimpleSystemsManagement.Model.PatchFilterGroup
ModifiedDate     : 3/3/2017 5:02:19 PM
Name             : Production-Baseline
PatchGroups      : {}
RejectedPatches  : {}
```

- Per i dettagli sull'API, vedere [GetPatchBaseline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMPatchBaselineForPatchGroup

Il seguente esempio di codice mostra come utilizzare. Get-SSMPatchBaselineForPatchGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra la baseline delle patch per un gruppo di patch.

```
Get-SSMPatchBaselineForPatchGroup -PatchGroup "Production"
```

Output:

BaselineId	PatchGroup
-----	-----
pb-045f10b4f382baeda	Production

- Per i dettagli sull'API, vedere [GetPatchBaselineForPatchGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMPatchGroup

Il seguente esempio di codice mostra come utilizzare. Get-SSMPatchGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca le registrazioni dei gruppi di patch.

```
Get-SSMPatchGroup
```

Output:

BaselineIdentity	PatchGroup
-----	-----
Amazon.SimpleSystemsManagement.Model.PatchBaselineIdentity	Production

- Per i dettagli sull'API, vedere [DescribePatchGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMPatchGroupState

Il seguente esempio di codice mostra come utilizzare. Get-SSMPatchGroupState

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene il riepilogo di alto livello sulla conformità delle patch per un gruppo di patch.

```
Get-SSMPatchGroupState -PatchGroup "Production"
```

Output:

```
Instances : 4
InstancesWithFailedPatches : 1
InstancesWithInstalledOtherPatches : 4
InstancesWithInstalledPatches : 3
InstancesWithMissingPatches : 0
InstancesWithNotApplicablePatches : 0
```

- Per i dettagli sull'API, vedere [DescribePatchGroupState](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMResourceComplianceSummaryList

Il seguente esempio di codice mostra come utilizzare. Get-SSMResourceComplianceSummaryList

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene un conteggio riepilogativo a livello di risorsa. Il riepilogo include informazioni sullo stato conforme e non conforme e sui conteggi dettagliati di gravità degli elementi di conformità per i prodotti che corrispondono a "Windows10". Poiché l' MaxResult impostazione predefinita è 100 se il parametro non è specificato e questo valore non è valido, il MaxResult parametro viene aggiunto e il valore è impostato su 50.

```
$FilterValues = @{
    "Key"="Product"
    "Type"="EQUAL"
    "Values"="Windows10"
}

Get-SSMResourceComplianceSummaryList -Filter $FilterValues -MaxResult 50
```

- Per i dettagli sull'API, vedere [ListResourceComplianceSummaries](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-SSMResourceTag

Il seguente esempio di codice mostra come utilizzare. Get-SSMResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i tag per una finestra di manutenzione.

```
Get-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType  
"MaintenanceWindow"
```

Output:

Key	Value
---	-----
Stack	Production

- Per i dettagli sull'API, vedere [ListTagsForResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-SSMActivation

Il seguente esempio di codice mostra come utilizzare. New-SSMActivation

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un'istanza gestita.

```
New-SSMActivation -DefaultInstanceName "MyWebServers" -IamRole "SSMAutomationRole" -  
RegistrationLimit 10
```

Output:

ActivationCode	ActivationId
-----	-----
KWChh0xBTiWdCkE9B1KC	08e51e79-1e36-446c-8e63-9458569c1363

- Per i dettagli sull'API, vedere [CreateActivation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-SSMAssociation

Il seguente esempio di codice mostra come utilizzare. New-SSMAssociation

Strumenti per PowerShell V4

Esempio 1: Questo esempio associa un documento di configurazione a un'istanza, utilizzando instance. IDs

```
New-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-UpdateSSMAgent"
```

Output:

```
Name                : AWS-UpdateSSMAgent
InstanceId           : i-0000293ffd8c57862
Date                : 2/23/2017 6:55:22 PM
Status.Name         : Associated
Status.Date         : 2/20/2015 8:31:11 AM
Status.Message      : Associated with AWS-UpdateSSMAgent
Status.AdditionalInfo :
```

Esempio 2: questo esempio associa un documento di configurazione a un'istanza, utilizzando destinazioni.

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
New-SSMAssociation -Name "AWS-UpdateSSMAgent" -Target $target
```

Output:

```
Name                : AWS-UpdateSSMAgent
InstanceId           :
Date                : 3/1/2017 6:22:21 PM
Status.Name         :
Status.Date         :
Status.Message      :
Status.AdditionalInfo :
```

Esempio 3: questo esempio associa un documento di configurazione a un'istanza, utilizzando destinazioni e parametri.

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
$params = @{
    "action"="configure"
    "mode"="ec2"
```

```

    "optionalConfigurationSource"="ssm"
    "optionalConfigurationLocation"=""
    "optionalRestart"="yes"
}
New-SSMAssociation -Name "Configure-CloudWatch" -AssociationName "CWConfiguration" -
Target $target -Parameter $params

```

Output:

```

Name                : Configure-CloudWatch
InstanceId           :
Date                : 5/17/2018 3:17:44 PM
Status.Name         :
Status.Date         :
Status.Message      :
Status.AdditionalInfo :

```

Esempio 4: questo esempio crea un'associazione con tutte le istanze della regione, con **AWS-GatherSoftwareInventory**. Fornisce inoltre file personalizzati e posizioni di registro nei parametri da raccogliere

```

$params =
    [Collections.Generic.Dictionary[String,Collections.Generic.List[String]]]::new()
$params["windowsRegistry"] = '[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon
\MachineImage","Recursive":false,"ValueNames":["AMIName"]}]'
$params["files"] = '[{"Path":"C:\Program Files","Pattern":
["*.exe"],"Recursive":true}, {"Path":"C:\ProgramData","Pattern":
["*.log"],"Recursive":true}]'
New-SSMAssociation -AssociationName new-in-mum -Name AWS-GatherSoftwareInventory
-Target @{Key="instanceids";Values="*"} -Parameter $params -region ap-south-1 -
ScheduleExpression "rate(720 minutes)"

```

Output:

```

Name                : AWS-GatherSoftwareInventory
InstanceId           :
Date                : 6/9/2019 8:57:56 AM
Status.Name         :
Status.Date         :
Status.Message      :
Status.AdditionalInfo :

```

- Per i dettagli sull'API, vedere [CreateAssociation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-SSMAssociationFromBatch

Il seguente esempio di codice mostra come utilizzare. New-SSMAssociationFromBatch

Strumenti per PowerShell V4

Esempio 1: questo esempio associa un documento di configurazione a più istanze. L'output restituisce un elenco di operazioni riuscite e non riuscite, se applicabile.

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@"AWS-UpdateSSMAgent"}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@"AWS-UpdateSSMAgent"}
New-SSMAssociationFromBatch -Entry $option1,$option2
```

Output:

```
Failed    Successful
-----
{}        {Amazon.SimpleSystemsManagement.Model.FailedCreateAssociation,
Amazon.SimpleSystemsManagement.Model.FailedCreateAsso...
```

Esempio 2: questo esempio mostrerà tutti i dettagli di un'operazione riuscita.

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@"AWS-UpdateSSMAgent"}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@"AWS-UpdateSSMAgent"}
(New-SSMAssociationFromBatch -Entry $option1,$option2).Successful
```

- Per i dettagli sull'API, vedere [CreateAssociationBatch](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-SSMDocument

Il seguente esempio di codice mostra come utilizzare. New-SSMDocument

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un documento nell'account. Il documento deve essere in formato JSON. Per ulteriori informazioni sulla scrittura di un documento di configurazione, consulta il Documento di configurazione nella Documentazione di riferimento API SSM.

```
New-SSMDocument -Content (Get-Content -Raw "c:\temp\RunShellScript.json") -Name  
"RunShellScript" -DocumentType "Command"
```

Output:

```
CreatedDate      : 3/1/2017 1:21:33 AM  
DefaultVersion  : 1  
Description      : Run an updated script  
DocumentType    : Command  
DocumentVersion : 1  
Hash            : 1d5ce820e999ff051eb4841ed887593daf77120fd76cae0d18a53cc42e4e22c1  
HashType        : Sha256  
LatestVersion   : 1  
Name            : RunShellScript  
Owner           : 809632081692  
Parameters      : {commands}  
PlatformTypes   : {Linux}  
SchemaVersion   : 2.0  
Sha1            :  
Status          : Creating
```

- Per i dettagli sull'API, vedere [CreateDocument](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-SSMMaintenanceWindow

Il seguente esempio di codice mostra come utilizzare New-SSMMaintenanceWindow

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova finestra di manutenzione con il nome specificato che viene eseguita alle 16:00 di ogni martedì per 4 ore, con un limite di 1 ora e che consente obiettivi non associati.

```
New-SSMMaintenanceWindow -Name "MyMaintenanceWindow" -Duration 4 -Cutoff 1 -
AllowUnassociatedTarget $true -Schedule "cron(0 16 ? * TUE *)"
```

Output:

```
mw-03eb53e1ea7383998
```

- Per i dettagli sull'API, vedere [CreateMaintenanceWindow](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-SSMPatchBaseline

Il seguente esempio di codice mostra come utilizzare New-SSMPatchBaseline

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una baseline delle patch che approva le patch, sette giorni dopo che sono state rilasciate da Microsoft, per istanze gestite che eseguono Windows Server 2019 in un ambiente di produzione.

```
$rule = New-Object Amazon.SimpleSystemsManagement.Model.PatchRule
$rule.ApproveAfterDays = 7

$ruleFilters = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilterGroup

$patchFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$patchFilter.Key="PRODUCT"
$patchFilter.Values="WindowsServer2019"

$severityFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$severityFilter.Key="MSRC_SEVERITY"
$severityFilter.Values.Add("Critical")
$severityFilter.Values.Add("Important")
$severityFilter.Values.Add("Moderate")

$classificationFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$classificationFilter.Key = "CLASSIFICATION"
$classificationFilter.Values.Add( "SecurityUpdates" )
$classificationFilter.Values.Add( "Updates" )
$classificationFilter.Values.Add( "UpdateRollups" )
$classificationFilter.Values.Add( "CriticalUpdates" )
```

```
$ruleFilters.PatchFilters.Add($severityFilter)
$ruleFilters.PatchFilters.Add($classificationFilter)
$ruleFilters.PatchFilters.Add($patchFilter)
$rule.PatchFilterGroup = $ruleFilters

New-SSMPatchBaseline -Name "Production-Baseline-Windows2019" -Description "Baseline
containing all updates approved for production systems" -ApprovalRules_PatchRule
$rule
```

Output:

```
pb-0z4z6221c4296b23z
```

- Per i dettagli sull'API, vedere [CreatePatchBaseline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-SSMDefaultPatchBaseline

Il seguente esempio di codice mostra come utilizzare. Register-SSMDefaultPatchBaseline

Strumenti per PowerShell V4

Esempio 1: questo esempio registra una baseline delle patch come baseline delle patch predefinita.

```
Register-SSMDefaultPatchBaseline -BaselineId "pb-03da896ca3b68b639"
```

Output:

```
pb-03da896ca3b68b639
```

- Per i dettagli sull'API, vedere [RegisterDefaultPatchBaseline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-SSMPatchBaselineForPatchGroup

Il seguente esempio di codice mostra come utilizzare. Register-SSMPatchBaselineForPatchGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio registra una baseline delle patch per un gruppo di patch.

```
Register-SSMPatchBaselineForPatchGroup -BaselineId "pb-03da896ca3b68b639" -
PatchGroup "Production"
```

Output:

```
BaselineId      PatchGroup
-----
pb-03da896ca3b68b639 Production
```

- Per i dettagli sull'API, vedere [RegisterPatchBaselineForPatchGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-SSMTargetWithMaintenanceWindow

Il seguente esempio di codice mostra come utilizzare. Register-SSMTargetWithMaintenanceWindow

Strumenti per PowerShell V4

Esempio 1: questo esempio registra un'istanza con una finestra di manutenzione.

```
$option1 = @{Key="InstanceIds";Values=@("i-0000293ffd8c57862")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
$option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

Output:

```
d8e47760-23ed-46a5-9f28-927337725398
```

Esempio 2: questo esempio registra più istanze con una finestra di manutenzione.

```
$option1 =
@{Key="InstanceIds";Values=@("i-0000293ffd8c57862","i-0cb2b964d3e14fd9f")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
$option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

Output:

```
6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

Esempio 3: Questo esempio registra un'istanza con una finestra di manutenzione utilizzando EC2 i tag.

```
$option1 = @{Key="tag:Environment";Values=@"Production"}}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
$option1 -OwnerInformation "Production Web Servers" -ResourceType "INSTANCE"
```

Output:

```
2994977e-aefb-4a71-beac-df620352f184
```

- Per i dettagli sull'API, vedere [RegisterTargetWithMaintenanceWindow](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-SSMTaskWithMaintenanceWindow

Il seguente esempio di codice mostra come utilizzare. Register-SSMTaskWithMaintenanceWindow

Strumenti per PowerShell V4

Esempio 1: questo esempio registra un'attività con una finestra di manutenzione utilizzando un ID di istanza. L'output è l'ID dell'attività.

```
$parameters = @{}
$parameterValues = New-Object
Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression
$parameterValues.Values = @"Install"@
$parameters.Add("Operation", $parameterValues)

Register-SSMTaskWithMaintenanceWindow -WindowId "mw-03a342e62c96d31b0"
-ServiceRoleArn "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
-MaxConcurrency 1 -MaxError 1 -TaskArn "AWS-RunShellScript" -Target
@{ Key="InstanceIds";Values="i-0000293ffd8c57862" } -TaskType "RUN_COMMAND" -
Priority 10 -TaskParameter $parameters
```


Output:

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

Esempio 2: questo esempio registra un'attività con una finestra di manutenzione utilizzando un ID di destinazione. L'output è l'ID dell'attività.

```
$parameters = @{}
$parameterValues = New-Object
    Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression
$parameterValues.Values = @("Install")
$parameters.Add("Operation", $parameterValues)

register-ssmtaskwithmaintenancewindow -WindowId "mw-03a342e62c96d31b0"
    -ServiceRoleArn "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
    -MaxConcurrency 1 -MaxError 1 -TaskArn "AWS-RunShellScript" -Target
    @{ Key="WindowTargetIds";Values="350d44e6-28cc-44e2-951f-4b2c985838f6" } -TaskType
    "RUN_COMMAND" -Priority 10 -TaskParameter $parameters
```

Output:

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

Esempio 3: questo esempio crea un oggetto parametro per il documento di comando run **AWS-RunPowerShellScript** e crea un'attività con una determinata finestra di manutenzione utilizzando l'ID di destinazione. L'output restituito è l'ID dell'attività.

```
$parameters =
    [Collections.Generic.Dictionary[String,Collections.Generic.List[String]]]::new()
$parameters.Add("commands",@("ipconfig","dir env:\computername"))
$parameters.Add("executionTimeout",@(3600))

$props = @{
    WindowId = "mw-0123e4cce56ff78ae"
    ServiceRoleArn = "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
    MaxConcurrency = 1
    MaxError = 1
    TaskType = "RUN_COMMAND"
    TaskArn = "AWS-RunPowerShellScript"
    Target = @{Key="WindowTargetIds";Values="fe1234ea-56d7-890b-12f3-456b789bee0f"}
    Priority = 1
```

```

    RunCommand_Parameter = $parameters
    Name = "set-via-cmdlet"
}

Register-SSMTaskWithMaintenanceWindow @props

```

Output:

```
f1e2ef34-5678-12e3-456a-12334c5c6cbe
```

Esempio 4: Questo esempio registra un'attività di AWS Systems Manager Automation utilizzando un documento denominato **Create-Snapshots**.

```

$automationParameters = @{}
$automationParameters.Add( "instanceId", @("{ TARGET_ID }") )
$automationParameters.Add( "AutomationAssumeRole",
    @("{arn:aws:iam::111111111111:role/AutomationRole}") )
$automationParameters.Add( "SnapshotTimeout", @("PT20M") )
Register-SSMTaskWithMaintenanceWindow -WindowId mw-123EXAMPLE456`
    -ServiceRoleArn "arn:aws:iam::123456789012:role/MW-Role"`
    -MaxConcurrency 1 -MaxError 1 -TaskArn "CreateVolumeSnapshots"`
    -Target @{ Key="WindowTargetIds";Values="4b5acdf4-946c-4355-
bd68-4329a43a5fd1" }`
    -TaskType "AUTOMATION"`
    -Priority 4`
    -Automation_DocumentVersion '$DEFAULT' -Automation_Parameter
$automationParameters -Name "Create-Snapshots"

```

- Per i dettagli sull'API, vedere [RegisterTaskWithMaintenanceWindow](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SSMActivation

Il seguente esempio di codice mostra come utilizzare. Remove-SSMActivation

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina un'attivazione. Se il comando va a buon fine, non viene generato output.

```
Remove-SSMActivation -ActivationId "08e51e79-1e36-446c-8e63-9458569c1363"
```

- Per i dettagli sull'API, vedere [DeleteActivation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SSMAssociation

Il seguente esempio di codice mostra come utilizzare. Remove-SSMAssociation

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina l'associazione tra un'istanza e un documento. Se il comando va a buon fine, non viene generato output.

```
Remove-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-UpdateSSMAgent"
```

- Per i dettagli sull'API, vedere [DeleteAssociation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SSMDocument

Il seguente esempio di codice mostra come utilizzare. Remove-SSMDocument

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina un documento. Se il comando va a buon fine, non viene generato output.

```
Remove-SSMDocument -Name "RunShellScript"
```

- Per i dettagli sull'API, vedere [DeleteDocument](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SSMMaintenanceWindow

Il seguente esempio di codice mostra come utilizzare. Remove-SSMMaintenanceWindow

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove una finestra di manutenzione.

```
Remove-SSMMaintenanceWindow -WindowId "mw-06d59c1a07c022145"
```

Output:

```
mw-06d59c1a07c022145
```

- Per i dettagli sull'API, vedere [DeleteMaintenanceWindow](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SSMParameter

Il seguente esempio di codice mostra come utilizzare. Remove-SSMParameter

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina un parametro. Se il comando va a buon fine, non viene generato output.

```
Remove-SSMParameter -Name "helloWorld"
```

- Per i dettagli sull'API, vedere [DeleteParameter](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SSMPatchBaseline

Il seguente esempio di codice mostra come utilizzare. Remove-SSMPatchBaseline

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina una baseline delle patch.

```
Remove-SSMPatchBaseline -BaselineId "pb-045f10b4f382baeda"
```

Output:

```
pb-045f10b4f382baeda
```

- Per i dettagli sull'API, vedere [DeletePatchBaseline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-SSMResourceTag

Il seguente esempio di codice mostra come utilizzare. Remove-SSMResourceTag

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove un tag da una finestra di manutenzione. Se il comando va a buon fine, non viene generato output.

```
Remove-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType  
"MaintenanceWindow" -TagKey "Production"
```

- Per i dettagli sull'API, vedere [RemoveTagsFromResource](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Send-SSMCommand

Il seguente esempio di codice mostra come utilizzare. Send-SSMCommand

Strumenti per PowerShell V4

Esempio 1: questo esempio esegue un comando echo su un'istanza di destinazione.

```
Send-SSMCommand -DocumentName "AWS-RunPowerShellScript" -Parameter @{commands =  
"echo helloWorld"} -Target @{Key="instanceids";Values=@"i-0cb2b964d3e14fd9f"}}
```

Output:

```
CommandId           : d8d190fc-32c1-4d65-a0df-ff5ff3965524  
Comment             :  
CompletedCount      : 0  
DocumentName        : AWS-RunPowerShellScript  
ErrorCount          : 0  
ExpiresAfter        : 3/7/2017 10:48:37 PM  
InstanceIds         : {}  
MaxConcurrency      : 50  
MaxErrors           : 0  
NotificationConfig  : Amazon.SimpleSystemsManagement.Model.NotificationConfig  
OutputS3BucketName  :  
OutputS3KeyPrefix   :  
OutputS3Region      :
```

```
Parameters      : {[commands,
  Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
RequestedDateTime : 3/7/2017 9:48:37 PM
ServiceRole      :
Status           : Pending
StatusDetails    : Pending
TargetCount      : 0
Targets          : {instanceids}
```

Esempio 2: questo esempio mostra come eseguire un comando che accetta parametri annidati.

```
Send-SSMCommand -DocumentName "AWS-RunRemoteScript" -Parameter
@{ sourceType="GitHub";sourceInfo='{ "owner": "me","repository": "amazon-
ssm","path": "Examples/Install-Win32openSSH"}'; "commandLine"=".\\Install-
Win32openSSH.ps1"} -InstanceId i-0cb2b964d3e14fd9f
```

- Per i dettagli sull'API, vedere [SendCommand](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-SSMAutomationExecution

Il seguente esempio di codice mostra come utilizzare. Start-SSMAutomationExecution

Strumenti per PowerShell V4

Esempio 1: questo esempio esegue un documento che specifica un ruolo di automazione, un ID di origine AMI e un ruolo di EC2 istanza Amazon.

```
Start-SSMAutomationExecution -DocumentName AWS-UpdateLinuxAmi -
Parameter @{'AutomationAssumeRole'='arn:aws:iam::123456789012:role/
SSMAutomationRole'; 'SourceAmiId'='ami-f173cc91'; 'InstanceIamRole'='EC2InstanceRole'}
```

Output:

```
3a532a4f-0382-11e7-9df7-6f11185f6dd1
```

- Per i dettagli sull'API, vedere [StartAutomationExecution](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Start-SSMSession

Il seguente esempio di codice mostra come utilizzare. Start-SSMSession

Strumenti per PowerShell V4

Esempio 1: questo esempio avvia una connessione a una destinazione per una sessione di Session Manager, abilitando il port forwarding.

```
Start-SSMSession -Target 'i-064578e5e7454488f' -DocumentName 'AWS-  
StartPortForwardingSession' -Parameter @{ localPortNumber = '8080'; portNumber =  
'80' }
```

Output:

SessionId	StreamUrl
-----	-----
random-id0	wss://ssmmessages.amazonaws.com/v1/data-channel/random-id

- Per i dettagli sull'API, vedere [StartSession](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-SSMAutomationExecution

Il seguente esempio di codice mostra come utilizzare. Stop-SSMAutomationExecution

Strumenti per PowerShell V4

Esempio 1: questo esempio interrompe un'esecuzione di automazione. Se il comando va a buon fine, non viene generato output.

```
Stop-SSMAutomationExecution -AutomationExecutionId "4105a4fc-  
f944-11e6-9d32-8fb2db27a909"
```

- Per i dettagli sull'API, vedere [StopAutomationExecution](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-SSMCommand

Il seguente esempio di codice mostra come utilizzare. Stop-SSMCommand

Strumenti per PowerShell V4

Esempio 1: questo esempio tenta di annullare un comando. Se l'operazione va a buon fine, non viene generato output.

```
Stop-SSMCommand -CommandId "9ded293e-e792-4440-8e3e-7b8ec5feaa38"
```

- Per i dettagli sull'API, vedere [CancelCommand](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-SSMManagedInstance

Il seguente esempio di codice mostra come utilizzare. Unregister-SSMManagedInstance

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la registrazione di un'istanza gestita. Se il comando va a buon fine, non viene generato output.

```
Unregister-SSMManagedInstance -InstanceId "mi-08ab247cdf1046573"
```

- Per i dettagli sull'API, vedere [DeregisterManagedInstance](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-SSMPatchBaselineForPatchGroup

Il seguente esempio di codice mostra come utilizzare. Unregister-SSMPatchBaselineForPatchGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la registrazione di un gruppo di patch da una baseline delle patch.

```
Unregister-SSMPatchBaselineForPatchGroup -BaselineId "pb-045f10b4f382baeda" -  
PatchGroup "Production"
```

Output:

BaselineId	PatchGroup


```
-----
pb-045f10b4f382baeda Production
```

- Per i dettagli sull'API, vedere [DeregisterPatchBaselineForPatchGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-SSMTargetFromMaintenanceWindow

Il seguente esempio di codice mostra come utilizzare. `Unregister-SSMTargetFromMaintenanceWindow`

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove una destinazione da una finestra di manutenzione.

```
Unregister-SSMTargetFromMaintenanceWindow -WindowTargetId "6ab5c208-9fc4-4697-84b7-b02a6cc25f7d" -WindowId "mw-06cf17cbefcb4bf4f"
```

Output:

```
WindowId           WindowTargetId
-----
mw-06cf17cbefcb4bf4f 6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

- Per i dettagli sull'API, vedere [DeregisterTargetFromMaintenanceWindow](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-SSMTaskFromMaintenanceWindow

Il seguente esempio di codice mostra come utilizzare. `Unregister-SSMTaskFromMaintenanceWindow`

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove un'attività da una finestra di manutenzione.

```
Unregister-SSMTaskFromMaintenanceWindow -WindowTaskId "f34a2c47-ddfd-4c85-a88d-72366b69af1b" -WindowId "mw-03a342e62c96d31b0"
```

Output:

WindowId	WindowTaskId
-----	-----
mw-03a342e62c96d31b0	f34a2c47-ddfd-4c85-a88d-72366b69af1b

- Per i dettagli sull'API, vedere [DeregisterTaskFromMaintenanceWindow](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-SSMAssociation

Il seguente esempio di codice mostra come utilizzare. Update-SSMAssociation

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna un'associazione con una nuova versione del documento.

```
Update-SSMAssociation -AssociationId "93285663-92df-44cb-9f26-2292d4ecc439" -
DocumentVersion "1"
```

Output:

```
Name           : AWS-UpdateSSMAgent
InstanceId      :
Date           : 3/1/2017 6:22:21 PM
Status.Name     :
Status.Date     :
Status.Message  :
Status.AdditionalInfo :
```

- Per i dettagli sull'API, vedere [UpdateAssociation](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-SSMAssociationStatus

Il seguente esempio di codice mostra come utilizzare. Update-SSMAssociationStatus

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna lo stato dell'associazione tra un'istanza e un documento di configurazione.

```
Update-SSMAssociationStatus -Name "AWS-UpdateSSMAgent" -InstanceId
    "i-0000293ffd8c57862" -AssociationStatus_Date "2015-02-20T08:31:11Z"
    -AssociationStatus_Name "Pending" -AssociationStatus_Message
    "temporary_status_change" -AssociationStatus_AdditionalInfo "Additional-Config-
    Needed"
```

Output:

```
Name           : AWS-UpdateSSMAgent
InstanceId      : i-0000293ffd8c57862
Date           : 2/23/2017 6:55:22 PM
Status.Name     : Pending
Status.Date     : 2/20/2015 8:31:11 AM
Status.Message  : temporary_status_change
Status.AdditionalInfo : Additional-Config-Needed
```

- Per i dettagli sull'API, vedere [UpdateAssociationStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-SSMDocument

Il seguente esempio di codice mostra come utilizzare. Update-SSMDocument

Strumenti per PowerShell V4

Esempio 1: questo esempio crea una nuova versione di un documento con il contenuto aggiornato del file json specificato. Il documento deve essere in formato JSON. È possibile ottenere la versione del documento con il cmdlet «Get-SSMDocumentVersionList».

```
Update-SSMDocument -Name RunShellScript -DocumentVersion "1" -Content (Get-Content -
Raw "c:\temp\RunShellScript.json")
```

Output:

```
CreatedDate     : 3/1/2017 2:59:17 AM
DefaultVersion  : 1
Description     : Run an updated script
DocumentType    : Command
DocumentVersion : 2
Hash            : 1d5ce820e999ff051eb4841ed887593daf77120fd76cae0d18a53cc42e4e22c1
HashType       : Sha256
```

```

LatestVersion    : 2
Name             : RunShellScript
Owner            : 809632081692
Parameters       : {commands}
PlatformTypes   : {Linux}
SchemaVersion    : 2.0
Sha1             :
Status           : Updating

```

- Per i dettagli sull'API, vedere [UpdateDocument](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-SSMDocumentDefaultVersion

Il seguente esempio di codice mostra come utilizzare. Update-SSMDocumentDefaultVersion

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna la versione predefinita di un documento. È possibile ottenere le versioni dei documenti disponibili con il cmdlet «Get- SSMDocumentVersionList».

```
Update-SSMDocumentDefaultVersion -Name "RunShellScript" -DocumentVersion "2"
```

Output:

```

DefaultVersion Name
-----
2              RunShellScript

```

- Per i dettagli sull'API, vedere [UpdateDocumentDefaultVersion](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-SSMMaintenanceWindow

Il seguente esempio di codice mostra come utilizzare. Update-SSMMaintenanceWindow

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna il nome di una finestra di manutenzione.

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Name "My-Renamed-MW"
```

Output:

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

Esempio 2: questo esempio abilita una finestra di manutenzione.

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $true
```

Output:

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

Esempio 3: questo esempio disabilita una finestra di manutenzione.

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $false
```

Output:

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : False
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

- Per i dettagli sull'API, vedere [UpdateMaintenanceWindow](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-SSManagedInstanceRole

Il seguente esempio di codice mostra come utilizzare. Update-SSManagedInstanceRole

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiorna il ruolo di un'istanza gestita. Se il comando va a buon fine, non viene generato output.

```
Update-SSManagedInstanceRole -InstanceId "mi-08ab247cdf1046573" -IamRole  
"AutomationRole"
```

- Per i dettagli sull'API, vedere [UpdateManagedInstanceRole](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Update-SSMPatchBaseline

Il seguente esempio di codice mostra come utilizzare. Update-SSMPatchBaseline

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge due patch come rifiutate e una patch approvata a una baseline delle patch esistente.

```
Update-SSMPatchBaseline -BaselineId "pb-03da896ca3b68b639" -RejectedPatch  
"KB2032276", "MS10-048" -ApprovedPatch "KB2124261"
```

Output:

```
ApprovalRules      : Amazon.SimpleSystemsManagement.Model.PatchRuleGroup  
ApprovedPatches    : {KB2124261}  
BaselineId         : pb-03da896ca3b68b639  
CreatedDate        : 3/3/2017 5:02:19 PM  
Description         : Baseline containing all updates approved for production systems  
GlobalFilters       : Amazon.SimpleSystemsManagement.Model.PatchFilterGroup  
ModifiedDate       : 3/3/2017 5:22:10 PM  
Name               : Production-Baseline  
RejectedPatches    : {KB2032276, MS10-048}
```

- Per i dettagli sull'API, vedere [UpdatePatchBaseline](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-SSMComplianceItem

Il seguente esempio di codice mostra come utilizzare Write-SSMComplianceItem

Strumenti per PowerShell V4

Esempio 1: questo esempio scrive un elemento di conformità personalizzato per l'istanza gestita specificata

```
$item = [Amazon.SimpleSystemsManagement.Model.ComplianceItemEntry]::new()
$item.Id = "07Jun2019-3"
$item.Severity="LOW"
$item.Status="COMPLIANT"
$item.Title="Fin-test-1 - custom"
Write-SSMComplianceItem -ResourceId mi-012dc3ecea45b678 -ComplianceType
    Custom:VSSCompliant2 -ResourceType ManagedInstance -Item $item -
    ExecutionSummary_ExecutionTime "07-Jun-2019"
```

- Per i dettagli sull'API, vedere [PutComplianceItems](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-SSMInventory

Il seguente esempio di codice mostra come utilizzare Write-SSMInventory

Strumenti per PowerShell V4

Esempio 1: questo esempio assegna le informazioni sulla posizione del rack a un'istanza. Se il comando va a buon fine, non viene generato output.

```
$data = New-Object
    "System.Collections.Generic.Dictionary[System.String,System.String]"
$data.Add("RackLocation", "Bay B/Row C/Rack D/Shelf F")

$items = New-Object
    "System.Collections.Generic.List[System.Collections.Generic.Dictionary[System.String,
    System.String]]"
$items.Add($data)

$customInventoryItem = New-Object Amazon.SimpleSystemsManagement.Model.InventoryItem
$customInventoryItem.CaptureTime = "2016-08-22T10:01:01Z"
$customInventoryItem.Content = $items
```

```
$customInventoryItem.TypeName = "Custom:TestRackInfo2"
$customInventoryItem.SchemaVersion = "1.0"

$inventoryItems = @($customInventoryItem)

Write-SSMInventory -InstanceId "i-0cb2b964d3e14fd9f" -Item $inventoryItems
```

- Per i dettagli sull'API, vedere [PutInventory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Write-SSMParameter

Il seguente esempio di codice mostra come utilizzare `Write-SSMParameter`

Strumenti per PowerShell V4

Esempio 1: questo esempio crea un parametro. Se il comando va a buon fine, non viene generato output.

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "helloWorld"
```

Esempio 2: questo esempio modifica un parametro. Se il comando va a buon fine, non viene generato output.

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "Good day, Sunshine!" -
Overwrite $true
```

- Per i dettagli sull'API, vedere [PutParameter](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Esempi di Amazon Translate con Tools for PowerShell V4

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando la versione AWS Strumenti per PowerShell V4 con Amazon Translate.

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

ConvertTo-TRNTargetLanguage

Il seguente esempio di codice mostra come usare `ConvertTo-TRNTargetLanguage`.

Strumenti per PowerShell V4

Esempio 1: converte il testo inglese specificato in francese. Il testo da convertire può anche essere passato come parametro `-Text`.

```
"Hello World" | ConvertTo-TRNTargetLanguage -SourceLanguageCode en -  
TargetLanguageCode fr
```

- Per i dettagli sull'API, vedere [TranslateText](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

AWS WAFV2 esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. AWS WAFV2

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

New-WAF2WebACL

Il seguente esempio di codice mostra come utilizzare `New-WAF2WebACL`.

Strumenti per PowerShell V4

Esempio 1: questo comando crea una nuova lista di controllo degli accessi (ACL) web denominata "waf-test". Tieni presente che, secondo la documentazione dell'API di servizio, 'DefaultAction' è una proprietà obbligatoria. Pertanto, è necessario specificare il valore per '- DefaultAction _Allow' e/or '- DefaultAction _Block. Poiché '- DefaultAction _Allow' e '- DefaultAction _Block' non sono le proprietà richieste, il valore '@ {}' potrebbe essere usato come segnaposto come mostrato nell'esempio precedente.

```
New-WAF2WebACL -Name "waf-test" -Scope REGIONAL -Region eu-west-1 -VisibilityConfig_CloudWatchMetricsEnabled $true -
VisibilityConfig_SampledRequestsEnabled $true -VisibilityConfig_MetricName "waf-
test" -Description "Test" -DefaultAction-Allow @{}
```

Output:

```
ARN          : arn:aws:wafv2:eu-west-1:139480602983:regional/webacl/waf-
test/19460b3f-db14-4b9a-8e23-a417e1eb007f
Description  : Test
Id           : 19460b3f-db14-4b9a-8e23-a417e1eb007f
LockToken    : 5a0cd5eb-d911-4341-b313-b429e6d6b6ab
Name         : waf-test
```

- Per i dettagli sull'API, vedere [CreateWebAcl](#) in Cmdlet Reference (V4).AWS Strumenti per PowerShell

WorkSpaces esempi di utilizzo di Tools for V4 PowerShell

I seguenti esempi di codice mostrano come eseguire azioni e implementare scenari comuni utilizzando AWS Strumenti per PowerShell V4 con. WorkSpaces

Le azioni sono estratti di codice da programmi più grandi e devono essere eseguite nel contesto. Sebbene le azioni mostrino come richiamare le singole funzioni del servizio, è possibile visualizzarle contestualizzate negli scenari correlati.

Ogni esempio include un link al codice sorgente completo, in cui vengono fornite le istruzioni su come configurare ed eseguire il codice nel contesto.

Argomenti

- [Azioni](#)

Azioni

Approve-WKSIpRule

Il seguente esempio di codice mostra come utilizzare `Approve-WKSIpRule`.

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge regole a un gruppo di IP esistente

```
$Rule = @(
@{IPRule = "10.1.0.0/0"; RuleDesc = "First Rule Added"},
@{IPRule = "10.2.0.0/0"; RuleDesc = "Second Rule Added"}
)

Approve-WKSIpRule -GroupId wsipg-abcnx2fcw -UserRule $Rule
```

- Per i dettagli sull'API, vedere [AuthorizeIpRules](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Copy-WKSWorkspaceImage

Il seguente esempio di codice mostra come utilizzare `Copy-WKSWorkspaceImage`.

Strumenti per PowerShell V4

Esempio 1: questo esempio copia l'immagine dell'area di lavoro con l'ID specificato da us-west-2 nella regione corrente con il nome "" CopiedImageTest

```
Copy-WKSWorkspaceImage -Name CopiedImageTest -SourceRegion us-west-2 -SourceImageId
wsi-djfoedhw6
```

Output:

```
wsi-456abaqfe
```

- Per i dettagli sull'API, vedere [CopyWorkspaceImage](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-WKSClientProperty

Il seguente esempio di codice mostra come utilizzare. Edit-WKSClientProperty

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita la riconnessione per il client Workspaces

```
Edit-WKSClientProperty -Region us-west-2 -ClientProperties_ReconnectEnabled  
"ENABLED" -ResourceId d-123414a369
```

- Per i dettagli sull'API, vedere [ModifyClientProperties](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-WKSSelfServicePermission

Il seguente esempio di codice mostra come utilizzare. Edit-WKSSelfServicePermission

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita le autorizzazioni self-service per modificare il tipo di calcolo e aumentare le dimensioni del volume per la directory specificata

```
Edit-WKSSelfservicePermission -Region us-west-2 -ResourceId  
d-123454a369 -SelfservicePermissions_ChangeComputeType ENABLED -  
SelfservicePermissions_IncreaseVolumeSize ENABLED
```

- Per i dettagli sull'API, vedere [ModifySelfservicePermissions](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-WKSWorkspaceAccessProperty

Il seguente esempio di codice mostra come utilizzare. Edit-WKSWorkspaceAccessProperty

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita l'accesso allo spazio di lavoro su Android e Chrome OS per la directory specificata

```
Edit-WKSWorkspaceAccessProperty -Region us-west-2 -ResourceId  
d-123454a369 -WorkspaceAccessProperties_DeviceTypeAndroid ALLOW -  
WorkspaceAccessProperties_DeviceTypeChromeOs ALLOW
```

- Per i dettagli sull'API, vedere [ModifyWorkspaceAccessProperties](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-WKSWorkspaceCreationProperty

Il seguente esempio di codice mostra come utilizzare. Edit-WKSWorkspaceCreationProperty

Strumenti per PowerShell V4

Esempio 1: questo esempio abilita le opzioni Accesso a Internet e Modalità di manutenzione su true come valori predefiniti durante la creazione di uno spazio di lavoro

```
Edit-WKSWorkspaceCreationProperty -Region us-west-2 -ResourceId  
d-123454a369 -WorkspaceCreationProperties_EnableInternetAccess $true -  
WorkspaceCreationProperties_EnableMaintenanceMode $true
```

- Per i dettagli sull'API, vedere [ModifyWorkspaceCreationProperties](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-WKSWorkspaceProperty

Il seguente esempio di codice mostra come utilizzare. Edit-WKSWorkspaceProperty

Strumenti per PowerShell V4

Esempio 1: questo esempio imposta la proprietà della modalità di esecuzione dello spazio di lavoro sull'arresto automatico dello spazio di lavoro specificato

```
Edit-WKSWorkspaceProperty -WorkspaceId ws-w361s100v -Region us-west-2 -  
WorkspaceProperties_RunningMode AUTO_STOP
```

- Per i dettagli sull'API, vedere [ModifyWorkspaceProperties](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Edit-WKSWorkspaceState

Il seguente esempio di codice mostra come utilizzare. Edit-WKSWorkspaceState

Strumenti per PowerShell V4

Esempio 1: questo esempio modifica lo stato dello spazio di lavoro specificato in Available

```
Edit-WKSWorkspaceState -WorkspaceId ws-w361s100v -Region us-west-2 -WorkspaceState  
AVAILABLE
```

- Per i dettagli sull'API, vedere [ModifyWorkspaceState](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-WKSClientProperty

Il seguente esempio di codice mostra come utilizzare. Get-WKSClientProperty

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene le proprietà del client Workspace per la directory specificata

```
Get-WKSClientProperty -ResourceId d-223562a123
```

- Per i dettagli sull'API, vedere [DescribeClientProperties](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-WKSIpGroup

Il seguente esempio di codice mostra come utilizzare. Get-WKSIpGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio ottiene i dettagli del gruppo di IP specificato nella Regione specificata

```
Get-WKSIpGroup -Region us-east-1 -GroupId wsipg-8m1234v45
```

Output:

```
GroupDesc GroupId      GroupName UserRules
-----
wsipg-8m1234v45 TestGroup {Amazon.WorkSpaces.Model.IpRuleItem,
Amazon.WorkSpaces.Model.IpRuleItem}
```

- Per i dettagli sull'API, vedere [DescribeIpGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-WKSTag

Il seguente esempio di codice mostra come utilizzare. Get-WKSTag

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera il tag per lo spazio di lavoro specificato

```
Get-WKSTag -WorkspaceId ws-w361s234r -Region us-west-2
```

Output:

```
Key      Value
---      -
auto-delete no
purpose   Workbench
```

- Per i dettagli sull'API, vedere [DescribeTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-WKSWorkspace

Il seguente esempio di codice mostra come utilizzare. Get-WKSWorkspace

Strumenti per PowerShell V4

Esempio 1: recupera i dettagli di tutti gli accessi WorkSpaces alla pipeline.

```
Get-WKSWorkspace
```

Output:

```

BundleId                : wsb-1a2b3c4d
ComputerName            :
DirectoryId             : d-1a2b3c4d
ErrorCode               :
ErrorMessage            :
IpAddress               :
RootVolumeEncryptionEnabled : False
State                   : PENDING
SubnetId                :
Username                : myuser
UserVolumeEncryptionEnabled : False
VolumeEncryptionKey     :
WorkspaceId             : ws-1a2b3c4d
WorkspaceProperties      : Amazon.WorkSpaces.Model.WorkspaceProperties

```

Esempio 2: questo comando mostra i valori delle proprietà secondarie di **WorkspaceProperties** per uno spazio di lavoro nella Regione **us-west-2**. Per ulteriori informazioni sulle proprietà secondarie di **WorkspaceProperties**, vedere https://docs.aws.amazon.com/workspaces/latest/api/API_WorkspaceProperties.html.

```
(Get-WKSWorkspace -Region us-west-2 -WorkspaceId ws-xdaf7hc9s).WorkspaceProperties
```

Output:

```

ComputeTypeName          : STANDARD
RootVolumeSizeGib       : 80
RunningMode              : AUTO_STOP
RunningModeAutoStopTimeoutInMinutes : 60
UserVolumeSizeGib       : 50

```

Esempio 3: questo comando mostra i valori delle proprietà secondarie **RootVolumeSizeGib** di **WorkspaceProperties** per uno spazio di lavoro nella Regione **us-west-2**. La dimensione del volume root, in GiB, è 80.

```
(Get-WKSWorkspace -Region us-west-2 -WorkspaceId ws-xdaf7hc9s).WorkspaceProperties.RootVolumeSizeGib
```

Output:

```
80
```


- Per i dettagli sull'API, vedere [DescribeWorkspaces](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-WKSWorkspaceBundle

Il seguente esempio di codice mostra come utilizzare. Get-WKSWorkspaceBundle

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera i dettagli di tutti i bundle dello spazio di lavoro nella Regione corrente

```
Get-WKSWorkspaceBundle
```

Output:

```
BundleId       : wsb-sfhgfv342
ComputeType    : Amazon.WorkSpaces.Model.ComputeType
Description     : This bundle is custom
ImageId        : wsi-235aeqges
LastUpdatedTime : 12/26/2019 06:44:07
Name           : CustomBundleTest
Owner          : 233816212345
RootStorage    : Amazon.WorkSpaces.Model.RootStorage
UserStorage    : Amazon.WorkSpaces.Model.UserStorage
```

- Per i dettagli sull'API, vedere [DescribeWorkspaceBundles](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-WKSWorkspaceDirectory

Il seguente esempio di codice mostra come utilizzare. Get-WKSWorkspaceDirectory

Strumenti per PowerShell V4

Esempio 1: questo esempio elenca i dettagli delle directory registrate

```
Get-WKSWorkspaceDirectory
```

Output:

```

Alias                : TestWorkspace
CustomerUserName     : Administrator
DirectoryId          : d-123414a369
DirectoryName        : TestDirectory.com
DirectoryType        : MicrosoftAD
DnsIpAddresses       : {172.31.43.45, 172.31.2.97}
IamRoleId            : arn:aws:iam::761234567801:role/workspaces_RoleDefault
IpGroupIds           : {}
RegistrationCode      : WSpdx+4RRT43
SelfservicePermissions : Amazon.WorkSpaces.Model.SelfservicePermissions
State                : REGISTERED
SubnetIds            : {subnet-1m3m7b43, subnet-ard11aba}
Tenancy              : SHARED
WorkspaceAccessProperties : Amazon.WorkSpaces.Model.WorkspaceAccessProperties
WorkspaceCreationProperties :
    Amazon.WorkSpaces.Model.DefaultWorkspaceCreationProperties
WorkspaceSecurityGroupId : sg-0ed2441234a123c43

```

- Per i dettagli sull'API, vedere [DescribeWorkspaceDirectories](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-WKSWorkspaceImage

Il seguente esempio di codice mostra come utilizzare. Get-WKSWorkspaceImage

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera tutti i dettagli di tutte le immagini nella Regione

```
Get-WKSWorkspaceImage
```

Output:

```

Description      : This image is copied from another image
ErrorCode        :
ErrorMessage     :
ImageId          : wsi-345ahdjgo
Name             : CopiedImageTest
OperatingSystem  : Amazon.WorkSpaces.Model.OperatingSystem
RequiredTenancy  : DEFAULT
State            : AVAILABLE

```

- Per i dettagli sull'API, vedere [DescribeWorkspaceImages](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-WKSWorkspaceSnapshot

Il seguente esempio di codice mostra come utilizzare. Get-WKSWorkspaceSnapshot

Strumenti per PowerShell V4

Esempio 1: questo esempio mostra il timestamp dello snapshot più recente creato per lo spazio di lavoro specificato

```
Get-WKSWorkspaceSnapshot -WorkspaceId ws-w361s100v
```

Output:

RebuildSnapshots	RestoreSnapshots
-----	-----
{Amazon.WorkSpaces.Model.Snapshot}	{Amazon.WorkSpaces.Model.Snapshot}

- Per i dettagli sull'API, vedere [DescribeWorkspaceSnapshots](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Get-WKSWorkspacesConnectionStatus

Il seguente esempio di codice mostra come utilizzare. Get-WKSWorkspacesConnectionStatus

Strumenti per PowerShell V4

Esempio 1: questo esempio recupera lo stato della connessione per lo spazio di lavoro specificato

```
Get-WKSWorkspacesConnectionStatus -WorkspaceId ws-w123s234r
```

- Per i dettagli sull'API, vedere [DescribeWorkspacesConnectionStatus](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-WKSIpGroup

Il seguente esempio di codice mostra come utilizzare. New-WKSIpGroup

Strumenti per PowerShell V4

Esempio 1: Questo esempio crea un gruppo Ip vuoto denominato FreshEmptyIpGroup

```
New-WKSipGroup -GroupName "FreshNewIPGroup"
```

Output:

```
wsipg-w45rty4ty
```

- Per i dettagli sull'API, vedere [CreateIpGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-WKSTag

Il seguente esempio di codice mostra come utilizzare. New-WKSTag

Strumenti per PowerShell V4

Esempio 1: questo esempio aggiunge un nuovo tag a uno spazio di lavoro denominato **ws-wsname**. Il tag include la chiave "Name" e il valore chiave **AWS_Workspace**.

```
$tag = New-Object Amazon.WorkSpaces.Model.Tag
$tag.Key = "Name"
$tag.Value = "AWS_Workspace"
New-WKSTag -Region us-west-2 -WorkspaceId ws-wsname -Tag $tag
```

Esempio 2: questo esempio aggiunge più tag a uno spazio di lavoro denominato **ws-wsname**. Un tag include la chiave "Name" e il valore di chiave **AWS_Workspace**, mentre l'altro tag include una chiave "Stage" e il valore di chiave "Test".

```
$tag = New-Object Amazon.WorkSpaces.Model.Tag
$tag.Key = "Name"
$tag.Value = "AWS_Workspace"

$tag2 = New-Object Amazon.WorkSpaces.Model.Tag
$tag2.Key = "Stage"
$tag2.Value = "Test"
New-WKSTag -Region us-west-2 -WorkspaceId ws-wsname -Tag $tag,$tag2
```

- Per i dettagli sull'API, vedere [CreateTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

New-WKSWorkspace

Il seguente esempio di codice mostra come utilizzare. New-WKSWorkspace

Strumenti per PowerShell V4

Esempio 1: crea un file Workspace per il pacchetto, la directory e l'utente forniti.

```
New-WKSWorkspace -Workspace @{ "BundleID" = "wsb-1a2b3c4d"; "DirectoryId" =  
"d-1a2b3c4d"; "UserName" = "USERNAME" }
```

Esempio 2: Questo esempio crea più WorkSpaces

```
New-WKSWorkspace -Workspace @{ "BundleID" = "wsb-1a2b3c4d"; "DirectoryId"  
= "d-1a2b3c4d"; "UserName" = "USERNAME_1"}, @{ "BundleID" = "wsb-1a2b3c4d";  
"DirectoryId" = "d-1a2b3c4d"; "UserName" = "USERNAME_2" }
```

- Per i dettagli sull'API, vedere [CreateWorkspaces](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-WKSIpGroup

Il seguente esempio di codice mostra come utilizzare. Register-WKSIpGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio registra il gruppo di IP specificato con la directory specificata

```
Register-WKSIpGroup -GroupId wsipg-23ahsdres -DirectoryId d-123412e123
```

- Per i dettagli sull'API, vedere [AssociateIpGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Register-WKSWorkspaceDirectory

Il seguente esempio di codice mostra come utilizzare. Register-WKSWorkspaceDirectory

Strumenti per PowerShell V4

Esempio 1: questo esempio registra la directory specificata per il servizio Workspaces

```
Register-WKSSpaceDirectory -DirectoryId d-123412a123 -EnableWorkDoc $false
```

- Per i dettagli sull'API, vedere [RegisterWorkspaceDirectory](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-WKSIpGroup

Il seguente esempio di codice mostra come utilizzare. Remove-WKSIpGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio elimina il gruppo di IP specificato

```
Remove-WKSIpGroup -GroupId wsipg-32fhgtred
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-WKSIpGroup (DeleteIpGroup)" on target
"wsipg-32fhgtred".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Per i dettagli sull'API, vedere [DeleteIpGroup](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-WKSTag

Il seguente esempio di codice mostra come utilizzare. Remove-WKSTag

Strumenti per PowerShell V4

Esempio 1: questo esempio rimuove il tag associato allo spazio di lavoro

```
Remove-WKSTag -ResourceId ws-w10b3abcd -TagKey "Type"
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-WKSTag (DeleteTags)" on target "ws-w10b3abcd".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"): Y
```

- Per i dettagli sull'API, vedere [DeleteTags](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Remove-WKSWorkspace

Il seguente esempio di codice mostra come utilizzare. Remove-WKSWorkspace

Strumenti per PowerShell V4

Esempio 1: termina più elementi WorkSpaces. L'uso dello switch -Force impedisce al cmdlet di richiedere la conferma.

```
Remove-WKSWorkspace -WorkspaceId "ws-1a2b3c4d5", "ws-6a7b8c9d0" -Force
```

Esempio 2: recupera la raccolta di tutti i file WorkSpaces e reindirizza il WorkspaceId parametro - IDs di Remove-WKSWorkspace, terminando tutti i file. WorkSpaces Il cmdlet verrà visualizzato prima che ciascuna di esse venga terminata. Workspace Per disattivare la richiesta di conferma, aggiungi il parametro -Force.

```
Get-WKSWorkspaces | Remove-WKSWorkspace
```

Esempio 3: questo esempio mostra come passare TerminateRequest oggetti che definiscono l' WorkSpaces oggetto da terminare. Il cmdlet richiederà una conferma prima di procedere, a meno che non venga specificato anche il parametro switch -Force.

```
$arrRequest = @()
$request1 = New-Object Amazon.WorkSpaces.Model.TerminateRequest
$request1.WorkspaceId = 'ws-12345678'
$arrRequest += $request1
$request2 = New-Object Amazon.WorkSpaces.Model.TerminateRequest
$request2.WorkspaceId = 'ws-abcdefgh'
$arrRequest += $request2
```

```
Remove-WKSWorkspace -Request $arrRequest
```

- Per i dettagli sull'API, vedere [TerminateWorkspaces](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Reset-WKSWorkspace

Il seguente esempio di codice mostra come utilizzare. Reset-WKSWorkspace

Strumenti per PowerShell V4

Esempio 1: ricostruisce quanto specificato. Workspace

```
Reset-WKSWorkspace -WorkspaceId "ws-1a2b3c4d"
```

Esempio 2: recupera la raccolta di tutti i dati WorkSpaces e invia il WorkspaceId parametro - di Reset- IDs al parametro -WKSWorkspace, provocandone la ricostruzione WorkSpaces .

```
Get-WKSWorkspaces | Reset-WKSWorkspace
```

- Per i dettagli sull'API, vedere [RebuildWorkspaces](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Restart-WKSWorkspace

Il seguente esempio di codice mostra come utilizzare. Restart-WKSWorkspace

Strumenti per PowerShell V4

Esempio 1: riavvia il file specificato. Workspace

```
Restart-WKSWorkspace -WorkspaceId "ws-1a2b3c4d"
```

Esempio 2: riavvia più volte. WorkSpaces

```
Restart-WKSWorkspace -WorkspaceId "ws-1a2b3c4d","ws-5a6b7c8d"
```

Esempio 3: recupera la raccolta di tutti i WorkSpaces file e invia il IDs WorkspaceId parametro - di Restart-WKSWorkspace, provocandone il WorkSpaces riavvio.

Get-WKSWorkspaces | Restart-WKSWorkspace

- Per i dettagli sull'API, vedere [RebootWorkspaces](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Stop-WKSWorkspace

Il seguente esempio di codice mostra come utilizzare. Stop-WKSWorkspace

Strumenti per PowerShell V4

Esempio 1: arresta più volte WorkSpaces.

```
Stop-WKSWorkspace -WorkspaceId "ws-1a2b3c4d5","ws-6a7b8c9d0"
```

Esempio 2: recupera l'insieme di tutti i file WorkSpaces e invia IDs il WorkspaceId parametro - di Stop - WKSWorkspace che WorkSpaces causa l'interruzione.

Get-WKSWorkspaces | Stop-WKSWorkspace

Esempio 3: Questo esempio mostra come passare StopRequest oggetti che definiscono il punto WorkSpaces da fermare.

```
$arrRequest = @()
$request1 = New-Object Amazon.WorkSpaces.Model.StopRequest
$request1.WorkspaceId = 'ws-12345678'
$arrRequest += $request1
$request2 = New-Object Amazon.WorkSpaces.Model.StopRequest
$request2.WorkspaceId = 'ws-abcdefgh'
$arrRequest += $request2
Stop-WKSWorkspace -Request $arrRequest
```

- Per i dettagli sull'API, vedere [StopWorkspaces](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Unregister-WKSIpGroup

Il seguente esempio di codice mostra come utilizzare. Unregister-WKSIpGroup

Strumenti per PowerShell V4

Esempio 1: questo esempio annulla la registrazione del gruppo di IP specificato con la directory specificata

```
Unregister-WKSIPGroup -GroupId wsipg-12abcdphq -DirectoryId d-123454b123
```

- Per i dettagli sull'API, vedere [DisassociateIpGroups](#) in AWS Strumenti per PowerShell Cmdlet Reference (V4).

Sicurezza per questo AWS prodotto o servizio

La sicurezza cloud di Amazon Web Services (AWS) è la priorità più alta. In qualità di cliente AWS, è possibile trarre vantaggio da un'architettura di data center e di rete progettata per soddisfare i requisiti delle organizzazioni più esigenti a livello di sicurezza. La sicurezza è una responsabilità condivisa tra AWS e te. Il [modello di responsabilità condivisa](#) descrive questo come sicurezza del cloud e sicurezza nel cloud.

Security of the Cloud: AWS è responsabile della protezione dell'infrastruttura che gestisce tutti i servizi offerti nel AWS Cloud e della fornitura di servizi che è possibile utilizzare in modo sicuro. La nostra responsabilità in AWS materia di sicurezza è la massima priorità e l'efficacia della nostra sicurezza viene regolarmente testata e verificata da revisori di terze parti nell'ambito dei Programmi di [AWS conformità](#).

Sicurezza nel cloud: la responsabilità dell'utente è determinata dal AWS servizio utilizzato e da altri fattori, tra cui la sensibilità dei dati, i requisiti dell'organizzazione e le leggi e i regolamenti applicabili.

Questo AWS prodotto o servizio segue il [modello di responsabilità condivisa](#) attraverso i servizi specifici di Amazon Web Services (AWS) che supporta. Per informazioni sulla sicurezza dei AWS servizi, consulta la [pagina della documentazione sulla sicurezza del AWS servizio](#) e [AWS i servizi che rientrano nell'ambito delle iniziative di AWS conformità previste dal programma di conformità](#).

Argomenti

- [Protezione dei dati in questo AWS prodotto o servizio](#)
- [Identity and Access Management](#)
- [Convalida della conformità per questo AWS prodotto o servizio](#)
- [Applicazione di una versione TLS minima negli Strumenti per PowerShell](#)
- [Considerazioni aggiuntive sulla sicurezza per gli strumenti per PowerShell](#)

Protezione dei dati in questo AWS prodotto o servizio

Il modello di [responsabilità AWS condivisa modello](#) di si applica alla protezione dei dati in questo AWS prodotto o servizio. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i Cloud AWS. L'utente è responsabile del controllo dei contenuti ospitati su questa infrastruttura. L'utente è inoltre responsabile della configurazione della

protezione e delle attività di gestione per i Servizi AWS utilizzati. Per ulteriori informazioni sulla privacy dei dati, vedi le [Domande frequenti sulla privacy dei dati](#). Per informazioni sulla protezione dei dati in Europa, consulta il post del blog relativo al [Modello di responsabilità condivisa AWS e GDPR](#) nel Blog sulla sicurezza AWS .

Ai fini della protezione dei dati, consigliamo di proteggere Account AWS le credenziali e configurare i singoli utenti con AWS IAM Identity Center o AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- Usa SSL/TLS per comunicare con le risorse. AWS È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Configura l'API e la registrazione delle attività degli utenti con. AWS CloudTrail Per informazioni sull'utilizzo dei CloudTrail percorsi per acquisire AWS le attività, consulta [Lavorare con i CloudTrail percorsi](#) nella Guida per l'AWS CloudTrail utente.
- Utilizza soluzioni di AWS crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.
- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se hai bisogno di moduli crittografici convalidati FIPS 140-3 per accedere AWS tramite un'interfaccia a riga di comando o un'API, usa un endpoint FIPS. Per ulteriori informazioni sugli endpoint FIPS disponibili, consulta il [Federal Information Processing Standard \(FIPS\) 140-3](#).

Ti consigliamo di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando si lavora con questo AWS prodotto o servizio o altro Servizi AWS utilizzando la console, l'API o. AWS CLI AWS SDKs I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono essere utilizzati per i la fatturazione o i log di diagnostica. Quando fornisci un URL a un server esterno, ti suggeriamo vivamente di non includere informazioni sulle credenziali nell'URL per convalidare la tua richiesta al server.

Crittografia dei dati

Una caratteristica fondamentale di qualsiasi servizio sicuro è che le informazioni vengano crittografate quando non sono utilizzate attivamente.

Crittografia dei dati inattivi

Di per sé AWS Strumenti per PowerShell non memorizza alcun dato del cliente diverso dalle credenziali necessarie per interagire con i AWS servizi per conto dell'utente.

Se utilizzi il AWS Strumenti per PowerShell per richiamare un AWS servizio che trasmette i dati dei clienti al computer locale per l'archiviazione, consulta il capitolo Sicurezza e conformità della Guida per l'utente del servizio per informazioni su come tali dati vengono archiviati, protetti e crittografati.

Crittografia in transito

Per impostazione predefinita, tutti i dati trasmessi dal computer client che esegue gli endpoint AWS Strumenti per PowerShell e del AWS servizio vengono crittografati inviandoli tramite una connessione HTTPS/TLS.

Non devi fare nulla per abilitare l'uso di HTTPS/TLS. È sempre abilitato.

Identity and Access Management

AWS Identity and Access Management (IAM) è un software Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle AWS risorse. Gli amministratori IAM controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (disporre delle autorizzazioni) a utilizzare le risorse. AWS IAM è uno Servizio AWS strumento che puoi utilizzare senza costi aggiuntivi.

Argomenti

- [Destinatari](#)
- [Autenticazione con identità](#)
- [Gestione dell'accesso con l'utilizzo delle policy](#)
- [Come Servizi AWS lavorare con IAM](#)
- [Risoluzione dei problemi di AWS identità e accesso](#)

Destinatari

Il modo in cui usi AWS Identity and Access Management (IAM) varia a seconda del lavoro che AWS svolgi.

Utente del servizio: se lo utilizzi Servizi AWS per svolgere il tuo lavoro, l'amministratore ti fornisce le credenziali e le autorizzazioni necessarie. Man mano che utilizzi più AWS funzionalità per svolgere il tuo lavoro, potresti aver bisogno di autorizzazioni aggiuntive. La comprensione della gestione dell'accesso consente di richiedere le autorizzazioni corrette all'amministratore. Se non riesci ad accedere a una funzionalità di AWS, consulta [Risoluzione dei problemi di AWS identità e accesso](#) o consulta la guida per l'utente della funzionalità Servizio AWS che stai utilizzando.

Amministratore del servizio: se sei responsabile delle AWS risorse della tua azienda, probabilmente hai pieno accesso a AWS. È tuo compito determinare a quali AWS funzionalità e risorse devono accedere gli utenti del servizio. È quindi necessario inviare le richieste all'amministratore IAM per modificare le autorizzazioni degli utenti del servizio. Esaminare le informazioni contenute in questa pagina per comprendere i concetti di base relativi a IAM. Per saperne di più su come la tua azienda può utilizzare IAM con AWS, consulta la guida per l'utente del Servizio AWS software che stai utilizzando.

Amministratore IAM: un amministratore IAM potrebbe essere interessato a ottenere dei dettagli su come scrivere policy per gestire l'accesso a AWS. Per visualizzare esempi di policy AWS basate sull'identità che puoi utilizzare in IAM, consulta la guida per l'utente di quella Servizio AWS che stai utilizzando.

Autenticazione con identità

L'autenticazione è il modo in cui accedi AWS utilizzando le tue credenziali di identità. Devi autenticarti come utente IAM o assumendo un ruolo IAM. Utente root dell'account AWS

Puoi accedere come identità federata utilizzando credenziali provenienti da una fonte di identità come AWS IAM Identity Center (IAM Identity Center), autenticazione Single Sign-On o credenziali. Google/Facebook Per ulteriori informazioni sull'accesso, consulta [Come accedere al tuo nella Guida per l'utente Account AWS](#).Accedi ad AWS

Per l'accesso programmatico, AWS fornisce un SDK e una CLI per firmare crittograficamente le richieste. Per ulteriori informazioni, consulta [AWS Signature Version 4 per le richieste API](#) nella IAM User Guide.

Account AWS utente root

Quando si crea un Account AWS, si inizia con un'identità di accesso denominata utente Account AWS root che ha accesso completo a tutte Servizi AWS le risorse. Ti consigliamo vivamente di non utilizzare l'utente root per le attività quotidiane. Per le attività che richiedono credenziali utente root, consulta [Attività che richiedono credenziali utente root nella Guida](#) per l'utente IAM.

Identità federata

Come best practice, richiedi agli utenti umani di utilizzare la federazione con un provider di identità per accedere Servizi AWS utilizzando credenziali temporanee.

Un'identità federata è un utente della directory aziendale, del provider di identità Web o Directory Service che accede Servizi AWS utilizzando le credenziali di una fonte di identità. Le identità federate assumono ruoli che forniscono credenziali temporanee.

Per la gestione centralizzata degli accessi, consigliamo. AWS IAM Identity Center Per ulteriori informazioni, consulta [Cos'è IAM Identity Center?](#) nella Guida per l'utente di AWS IAM Identity Center

Utenti e gruppi IAM

Un [utente IAM](#) è un'identità con autorizzazioni specifiche per una singola persona o applicazione. Consigliamo di utilizzare credenziali temporanee anziché utenti IAM con credenziali a lungo termine. Per ulteriori informazioni, consulta [Richiedere agli utenti umani di utilizzare la federazione con un provider di identità per accedere AWS utilizzando credenziali temporanee](#) nella Guida per l'utente IAM.

Un [gruppo IAM](#) specifica una raccolta di utenti IAM e semplifica la gestione delle autorizzazioni per grandi gruppi di utenti. Per ulteriori informazioni, consulta [Casi d'uso per utenti IAM nella Guida](#) per l'utente IAM.

Ruoli IAM

Un [ruolo IAM](#) è un'identità con autorizzazioni specifiche che fornisce credenziali temporanee. Puoi assumere un ruolo [passando da un utente a un ruolo IAM \(console\)](#) o chiamando un'operazione AWS CLI o AWS API. Per ulteriori informazioni, consulta [Metodi per assumere un ruolo](#) nella Guida per l'utente IAM.

I ruoli IAM sono utili per l'accesso federato degli utenti, le autorizzazioni utente IAM temporanee, l'accesso tra account, l'accesso tra servizi e le applicazioni in esecuzione su Amazon. EC2 Per maggiori informazioni, consultare [Accesso a risorse multi-account in IAM](#) nella Guida per l'utente di IAM.

Gestione dell'accesso con l'utilizzo delle policy

Puoi controllare l'accesso AWS creando policy e collegandole a identità o risorse. AWS Una policy definisce le autorizzazioni quando è associata a un'identità o a una risorsa. AWS valuta queste

politiche quando un preside effettua una richiesta. La maggior parte delle politiche viene archiviata AWS come documenti JSON. Per ulteriori informazioni sui documenti relativi alle policy JSON, consulta [Overview of JSON policy](#) nella IAM User Guide.

Utilizzando le policy, gli amministratori specificano chi ha accesso a cosa definendo quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Un amministratore IAM crea le policy IAM e le aggiunge ai ruoli, che gli utenti possono quindi assumere. Le policy IAM definiscono le autorizzazioni indipendentemente dal metodo utilizzato per eseguire l'operazione.

Policy basate sull'identità

Le policy basate sull'identità sono documenti di policy di autorizzazione JSON che alleggi a un'identità (utente, gruppo o ruolo). Queste politiche controllano quali azioni possono eseguire le identità, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consultare [Definizione di autorizzazioni personalizzate IAM con policy gestite dal cliente](#) nella Guida per l'utente di IAM.

Le politiche basate sull'identità possono essere politiche in linea (incorporate direttamente in una singola identità) o politiche gestite (politiche autonome collegate a più identità). Per scoprire come scegliere tra politiche gestite e politiche in linea, consulta Choose [between managed policy e inline policy nella IAM User Guide](#).

Policy basate sulle risorse

Le policy basate su risorse sono documenti di policy JSON che è possibile collegare a una risorsa. Gli esempi includono le policy di trust dei ruoli IAM e le policy dei bucket di Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare l'accesso a una risorsa specifica. In una policy basata sulle risorse è obbligatorio [specificare un'entità principale](#).

Le policy basate su risorse sono policy inline che risiedono in tale servizio. Non puoi utilizzare le policy AWS gestite di IAM in una policy basata sulle risorse.

Liste di controllo degli accessi (ACLs)

Le liste di controllo degli accessi (ACLs) controllano quali principali (membri dell'account, utenti o ruoli) dispongono delle autorizzazioni per accedere a una risorsa. ACLs sono simili alle politiche basate sulle risorse, sebbene non utilizzino il formato del documento di policy JSON.

Amazon S3 e Amazon VPC sono esempi di servizi che supportano. AWS WAF ACLs Per ulteriori informazioni ACLs, consulta la [panoramica della lista di controllo degli accessi \(ACL\)](#) nella Amazon Simple Storage Service Developer Guide.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi che possono impostare le autorizzazioni massime concesse dai tipi di policy più comuni:

- Limiti delle autorizzazioni: imposta le autorizzazioni massime che una policy basata sull'identità può concedere a un'entità IAM. Per ulteriori informazioni, consulta [Limiti delle autorizzazioni per le entità IAM](#) nella Guida per l'utente di IAM .
- Politiche di controllo del servizio (SCPs): specifica le autorizzazioni massime per un'organizzazione o un'unità organizzativa in. AWS Organizations Per ulteriori informazioni, consultare [Policy di controllo dei servizi](#) nella Guida per l'utente di AWS Organizations .
- Politiche di controllo delle risorse (RCPs): imposta le autorizzazioni massime disponibili per le risorse nei tuoi account. Per ulteriori informazioni, consulta [Politiche di controllo delle risorse \(RCPs\)](#) nella Guida per l'AWS Organizations utente.
- Criteri di sessione: criteri avanzati passati come parametro durante la creazione di una sessione temporanea per un ruolo o un utente federato. Per maggiori informazioni, consultare [Policy di sessione](#) nella Guida per l'utente di IAM.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per sapere come si AWS determina se consentire una richiesta quando sono coinvolti più tipi di policy, consulta la [logica di valutazione delle policy](#) nella IAM User Guide.

Come Servizi AWS lavorare con IAM

Per avere una visione di alto livello di come Servizi AWS funziona la maggior parte delle funzionalità IAM, consulta [AWS i servizi che funzionano con IAM nella IAM](#) User Guide.

Per scoprire come utilizzare uno specifico Servizio AWS con IAM, consulta la sezione sulla sicurezza della Guida per l'utente del servizio pertinente.

Risoluzione dei problemi di AWS identità e accesso

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare quando lavori con un AWS IAM.

Argomenti

- [Non sono autorizzato a eseguire alcuna azione in AWS](#)
- [Non sono autorizzato a eseguire iam: PassRole](#)
- [Voglio consentire a persone esterne a me di accedere Account AWS alle mie AWS risorse](#)

Non sono autorizzato a eseguire alcuna azione in AWS

Se ricevi un errore che indica che non sei autorizzato a eseguire un'operazione, le tue policy devono essere aggiornate per poter eseguire l'operazione.

L'errore di esempio seguente si verifica quando l'utente IAM `mateojackson` prova a utilizzare la console per visualizzare i dettagli relativi a una risorsa `my-example-widget` fittizia ma non dispone di autorizzazioni `aws:GetWidget` fittizie.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
aws:GetWidget on resource: my-example-widget
```

In questo caso, la policy per l'utente `mateojackson` deve essere aggiornata per consentire l'accesso alla risorsa `my-example-widget` utilizzando l'azione `aws:GetWidget`.

Se hai bisogno di aiuto, contatta il tuo AWS amministratore. L'amministratore è la persona che ha fornito le credenziali di accesso.

Non sono autorizzato a eseguire iam: PassRole

Se ricevi un errore indicante che non sei autorizzato a eseguire l'azione `iam:PassRole`, le tue policy devono essere aggiornate per poter passare un ruolo ad AWS.

Alcuni Servizi AWS consentono di passare un ruolo esistente a quel servizio invece di creare un nuovo ruolo di servizio o un ruolo collegato al servizio. Per eseguire questa operazione, è necessario disporre delle autorizzazioni per trasmettere il ruolo al servizio.

L'errore di esempio seguente si verifica quando un utente IAM denominato `marymajor` cerca di utilizzare la console per eseguire un'operazione in AWS. Tuttavia, l'operazione richiede che il servizio

disponga delle autorizzazioni concesse da un ruolo di servizio. Mary non dispone delle autorizzazioni per trasmettere il ruolo al servizio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

In questo caso, le policy di Mary devono essere aggiornate per poter eseguire l'operazione `iam:PassRole`.

Se hai bisogno di aiuto, contatta il tuo AWS amministratore. L'amministratore è la persona che ha fornito le credenziali di accesso.

Voglio consentire a persone esterne a me di accedere Account AWS alle mie AWS risorse

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per i servizi che supportano politiche basate sulle risorse o liste di controllo degli accessi (ACLs), puoi utilizzare tali politiche per concedere alle persone l'accesso alle tue risorse.

Per maggiori informazioni, consultare gli argomenti seguenti:

- Per sapere se AWS supporta queste funzionalità, consulta [Come Servizi AWS lavorare con IAM](#)
- Per scoprire come fornire l'accesso alle tue risorse attraverso Account AWS le risorse di tua proprietà, consulta [Fornire l'accesso a un utente IAM in un altro Account AWS di tua proprietà](#) nella IAM User Guide.
- Per scoprire come fornire l'accesso alle tue risorse a terze parti Account AWS, consulta [Fornire l'accesso a soggetti Account AWS di proprietà di terze parti](#) nella Guida per l'utente IAM.
- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consultare [Fornire l'accesso a utenti autenticati esternamente \(Federazione delle identità\)](#) nella Guida per l'utente di IAM.
- Per informazioni sulle differenze di utilizzo tra ruoli e policy basate su risorse per l'accesso multi-account, consulta [Accesso a risorse multi-account in IAM](#) nella Guida per l'utente IAM.

Convalida della conformità per questo AWS prodotto o servizio

Per sapere se un Servizio AWS programma rientra nell'ambito di specifici programmi di conformità, consulta Servizi AWS la sezione [Scope by Compliance Program Servizi AWS](#) e scegli il programma

di conformità che ti interessa. Per informazioni generali, consulta Programmi di [AWS conformità Programmi](#) di di .

È possibile scaricare report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta [Scaricamento dei report in AWS Artifact](#) .

La vostra responsabilità di conformità durante l'utilizzo Servizi AWS è determinata dalla sensibilità dei dati, dagli obiettivi di conformità dell'azienda e dalle leggi e dai regolamenti applicabili. Per ulteriori informazioni sulla responsabilità di conformità durante l'utilizzo Servizi AWS, consulta la [Documentazione AWS sulla sicurezza](#).

Questo AWS prodotto o servizio segue il [modello di responsabilità condivisa](#) attraverso i servizi specifici di Amazon Web Services (AWS) che supporta. Per informazioni sulla sicurezza dei AWS servizi, consulta la [pagina della documentazione sulla sicurezza del AWS servizio](#) e [AWS i servizi che rientrano nell'ambito delle iniziative di AWS conformità previste dal programma di conformità](#).

Applicazione di una versione TLS minima negli Strumenti per PowerShell

Per aumentare la sicurezza durante la comunicazione con AWS i servizi, è necessario configurare gli strumenti in modo PowerShell da utilizzare la versione TLS appropriata. Per informazioni su come eseguire questa operazione, consulta [Enforcing a minimum TLS version](#) (Applicazione di una versione minima di TLS) nella [Guida per gli sviluppatori di AWS SDK per .NET](#).

Considerazioni aggiuntive sulla sicurezza per gli strumenti per PowerShell

Questo argomento contiene considerazioni sulla sicurezza oltre agli argomenti sulla sicurezza trattati nelle sezioni precedenti.

Registrazione di informazioni sensibili

Alcune operazioni di questo strumento potrebbero restituire informazioni che potrebbero essere considerate sensibili, incluse le informazioni provenienti da variabili di ambiente. L'esposizione di queste informazioni potrebbe rappresentare un rischio per la sicurezza in determinati scenari; ad esempio, le informazioni potrebbero essere incluse nei registri di integrazione e distribuzione continua (CI/CD). È quindi importante verificare quando includere tale output nei log e sopprimerlo quando non

è necessario. Per ulteriori informazioni sulla protezione dei dati sensibili, vedere. [Protezione dei dati in questo AWS prodotto o servizio](#)

Prendi in considerazione le seguenti best practice:

- Non utilizzate variabili di ambiente per memorizzare valori sensibili per le vostre risorse serverless. Fate invece in modo che il codice serverless recuperi a livello di codice il segreto da un archivio di segreti (ad esempio,). Gestione dei segreti AWS
- Esamina il contenuto dei log di compilazione per assicurarti che non contengano informazioni riservate. Prendi in considerazione approcci come il piping to /dev/null o l'acquisizione dell'output come bash o PowerShell variabile per sopprimere gli output dei comandi.
- Prendi in considerazione l'accesso ai tuoi log e definisci l'accesso in modo appropriato per il tuo caso d'uso.

Riferimento al cmdlet per gli strumenti per PowerShell

Tools for PowerShell fornisce cmdlet che è possibile utilizzare per accedere ai servizi. AWS Per vedere quali cmdlet sono disponibili, consulta [AWS Strumenti per PowerShell Cmdlet Reference](#).

Cronologia dei documenti

Questo argomento descrive notevoli modifiche alla documentazione per gli AWS Strumenti per PowerShell.

Abbiamo inoltre aggiornato la documentazione periodicamente in risposta ai feedback dei clienti. Per inviare feedback su un argomento, utilizza i pulsanti di feedback accanto alla domanda "Questa pagina ti è stata utile?" nella parte inferiore di ogni pagina.

[Per ulteriori informazioni sulle modifiche e gli aggiornamenti di AWS Strumenti per PowerShell, consulta le note di rilascio.](#) Per ricevere notifiche sugli aggiornamenti di questa documentazione, è possibile sottoscrivere un [feed RSS](#).

Modifica	Descrizione	Data
Cosa c'è di nuovo	End-of-support è stato annunciato per la V4 del AWS Strumenti per PowerShell.	17 settembre 2025
Cosa c'è di nuovo	La versione 5 (V5) di AWS Strumenti per PowerShell è generalmente disponibile! Per ulteriori informazioni, consulta la Guida per l'AWS Strumenti per PowerShell utente (V5) , in particolare l'argomento per la migrazione alla V5.	23 giugno 2025
Cosa c'è di nuovo	Rilasciati i contenuti di anteprima per la V5 del AWS Strumenti per PowerShell.	13 giugno 2025
Cosa c'è di nuovo	Pubblicata la guida per l'utente per la versione 5 (anteprima) di AWS Strumenti per PowerShell.	28 maggio 2025

Osservabilità	Annunciata la versione GA per l'osservabilità	10 febbraio 2025
Cosa c'è di nuovo	Sono state aggiunte informazioni sul nuovo comportamento predefinito per la protezione dell'integrità.	15 gennaio 2025
Cosa c'è di nuovo	Sono state aggiunte informazioni sulla prima versione di anteprima della AWS Strumenti per PowerShell versione 5.	18 novembre 2024
Pipelining, output e iterazione	Sostituito il contenuto relativo a \$AWSHistory, che è stato obsoleto.	10 ottobre 2024
Osservabilità	Sono state aggiunte informazioni di anteprima sull'osservabilità in AWS Strumenti per PowerShell, che consentono la raccolta di dati di telemetria.	13 settembre 2024
Installazione di su Windows AWS Strumenti per PowerShell	Sono state aggiunte informazioni sullo sblocco dei file ZIP prima dell'estrazione del contenuto.	5 agosto 2024
Informazioni su EC2 -Classic	Informazioni rimosse su EC2 -Classic, che è stato ritirato.	1° agosto 2024
Codici di esempio	È incluso un capitolo con esempi di cmdlet.	17 aprile 2024
Considerazioni aggiuntive sulla sicurezza	Informazioni incluse sulla potenziale registrazione di dati sensibili.	16 aprile 2024

Configura l'autenticazione dello strumento con AWS	Sono state aggiunte informazioni sul supporto per SSO in AWS Strumenti per PowerShell.	15 marzo 2024
Riferimento al cmdlet per il programma Tools for PowerShell	È stata aggiunta una sezione con un collegamento al riferimento Tools for PowerShell cmdlet.	17 novembre 2023
Sono stati inclusi ulteriori aggiornamenti delle best practice IAM	Guida aggiornata per l'allineamento alle best practice IAM. Per ulteriori informazioni, consulta Best practice per la sicurezza in IAM .	12 ottobre 2023
Installazione su Windows	Sono state rimosse le informazioni sull'installazione degli strumenti per Windows PowerShell utilizzando l'MSI, che è diventato obsoleto.	25 settembre 2023
Aggiornamenti delle best practice di IAM	Guida aggiornata per l'allineamento alle best practice IAM. Per ulteriori informazioni, consulta Best practice per la sicurezza in IAM .	8 settembre 2023
Pipelining e \$ AWSHistory	È stato aggiunto il parametro IncludeSensitiveData al cmdlet Set-AWSHistoryConfiguration.	9 marzo 2023
Utilizzo del ClientConfig parametro nei cmdlet	Sono state aggiunte informazioni sul supporto per il ClientConfig parametro.	28 ottobre 2022

[Avvio di un' EC2 istanza
Amazon tramite Windows
PowerShell](#)

Sono state aggiunte note sul
ritiro di EC2 -Classic.

26 luglio 2022

[AWS Strumenti per PowerShel
l Versione 4](#)

Sono state aggiunte informazi
oni sulla versione 4, incluse le
istruzioni di installazione per
[Windows](#) e [Linux/macOS](#), e un
argomento di [migrazione](#) che
descrive le differenze rispetto
alla versione 3 e introduce
nuove funzionalità.

21 novembre 2019

[AWS Strumenti per PowerShel
l 3.3.563](#)

Sono state aggiunte informazi
oni su come installare e
utilizzare la versione di
anteprima del modulo
`AWS.Tools.Common` .
Questo nuovo modulo
suddivide il vecchio pacchetto
monolitico in un modulo
condiviso e un modulo per
servizio. AWS

18 ottobre 2019

[AWS Strumenti per PowerShel
l 3.3.343.0](#)

Sono state aggiunte informazi
oni alla sezione [Utilizzo della
AWS Strumenti per PowerShel
l sezione che](#) introduce gli
AWS Lambda strumenti per
gli sviluppatori PowerShell
Core PowerShell per creare
funzioni. AWS Lambda

11 settembre 2018

[AWS Tools for Windows PowerShell 3.1.31.0](#)

Sono state aggiunte informazioni alla sezione [Nozioni di base](#) riguardo nuovi cmdlet che utilizzano Security Assertion Markup Language (SAML) per supportare la configurazione di identità federate per gli utenti.

1 dicembre 2015

[AWS Tools for Windows PowerShell 2,3,19](#)

Sono state aggiunte informazioni alla sezione [Cmdlets Discovery and Aliases](#) sul nuovo Get-AWSCmdletName cmdlet che possono aiutare gli utenti a trovare più facilmente i cmdlet desiderati. AWS

5 febbraio 2015

[AWS Tools for Windows](#)

[PowerShell 1.1.1.0](#)

15 maggio 2013

L'output della raccolta dai cmdlet viene sempre enumerato nella pipeline. PowerShell È disponibile supporto automatico per le chiamate di servizio impaginabili. La nuova variabile \$AWSHistory shell raccoglie le risposte di servizio e, facoltativamente, le richieste di assistenza. AWSRegion le istanze utilizzano il campo Region invece di SystemName e facilitare la pipelining. Remove-S3Bucket supporta un'DeleteObjects opzione - switch. Risolto il problema di usabilità con Set-AWSCredentials Inizializza: AWSDefaults riporta da dove ha ottenuto le credenziali e i dati della regione. Stop-EC2Instance accetta Amazon. EC2Instances .Model.Reservation come input. I tipi di parametro Generic List<T> sono sostituiti con i tipi di array (T[]). I cmdlet che eliminano o terminano le risorse richiedono o una conferma prima dell'eliminazione. Write-S3Object supporta contenuti di testo in linea da caricare su Amazon S3.

[AWS Tools for Windows PowerShell 1.0.1.0](#)

21 dicembre 2012

La posizione di installazione del PowerShell modulo Tools for Windows è stata modificata in modo che gli ambienti che utilizzano Windows PowerShell versione 3 possano sfruttare il caricamento automatico. Il modulo e i file di supporto sono ora installati in una sottocartella `AWSPowerShell` in `AWS ToolsPowerShell`. I file delle versioni precedenti esistenti nella cartella `AWS ToolsPowerShell` vengono rimossi automaticamente dal programma di installazione. Il `PSModulePath` per Windows PowerShell (tutte le versioni) viene aggiornato in questa versione per contenere la cartella principale del modulo (`AWS ToolsPowerShell`). Per i sistemi con Windows PowerShell versione 2, la scorciatoia del menu Start viene aggiornata per importare il modulo dalla nuova posizione e quindi eseguirlo `Initialize-AWSDefaults`. Per i sistemi con Windows PowerShell versione 3, la scorciatoia del menu Start viene aggiornata per rimuovere il `Import-Module` comando,

lasciando solo. `Initialize-AWSDefaults`. Se hai modificato il PowerShell I profilo per eseguire uno `Import-Module` dei `AWSPowerShell.psd1` file, dovrai aggiornarlo in modo che punti alla nuova posizione del file (oppure, se usi la PowerShell versione 3, rimuovere l'`Import-Module` istruzione in quanto non è più necessaria). Come risultato di queste modifiche, il PowerShell modulo Tools for Windows è ora elencato come modulo disponibile durante l'esecuzione `Get-Module -ListAvailable`. Inoltre, per gli utenti di Windows PowerShell versione 3, l'esecuzione di qualsiasi cmdlet esportato dal modulo caricherà automaticamente il modulo nella PowerShell shell corrente senza doverlo prima utilizzare. `Import-Module` Questo abilita l'utilizzo interattivo dei cmdlet su un sistema con una policy di esecuzione che non consente l'esecuzione di script.

[AWS Tools for Windows PowerShell 1.0.0.0](#)

Rilascio iniziale

6 dicembre 2012

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.