Guida per gli sviluppatori

AWS App Runner

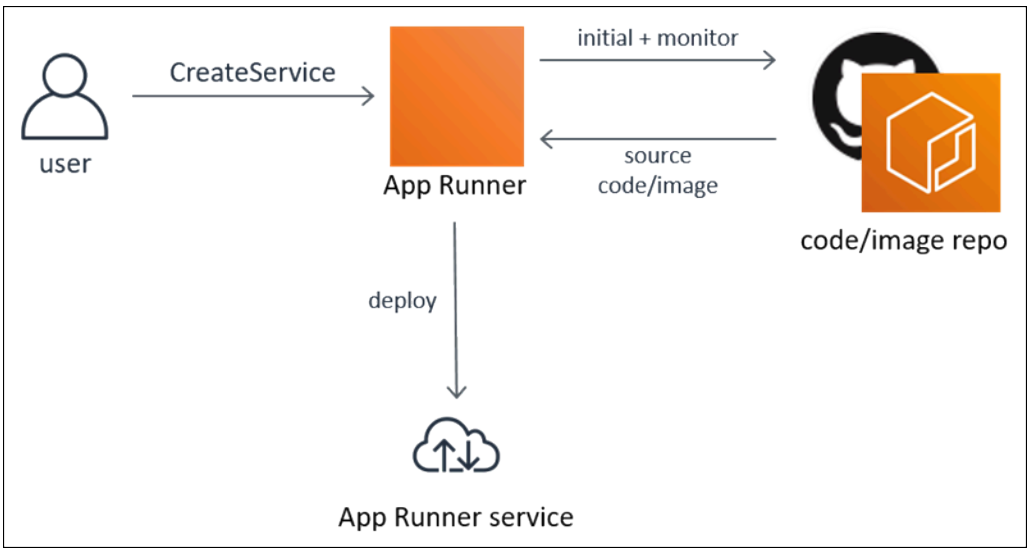
AWS App Runner: Guida per gli sviluppatori

Table of Contents

Che cos'è AWS App Runner?

AWS App Runner è un AWS servizio che offre un modo rapido, semplice ed economico per eseguire la distribuzione dal codice sorgente o da un'immagine del contenitore direttamente a un'applicazione Web scalabile e sicura nel cloud. AWS Non è necessario apprendere nuove tecnologie, decidere quale servizio di elaborazione utilizzare o sapere come effettuare il provisioning e configurare le risorse. AWS

App Runner si connette direttamente al tuo archivio di codice o immagini. Fornisce una pipeline di integrazione e distribuzione automatica con operazioni completamente gestite, alte prestazioni, scalabilità e sicurezza.



A chi è rivolto App Runner?

Se sei uno sviluppatore, puoi utilizzare App Runner per semplificare il processo di distribuzione di una nuova versione del tuo archivio di codice o immagini.

Per i team operativi, App Runner consente le distribuzioni automatiche ogni volta che un commit viene inviato al repository di codice o una nuova versione dell'immagine del contenitore viene inserita nell'archivio di immagini.

Accesso ad App Runner

È possibile definire e configurare le distribuzioni del servizio App Runner utilizzando una delle seguenti interfacce:

- Console App Runner: fornisce un'interfaccia web per la gestione dei servizi App Runner.
- API App Runner: fornisce un' RESTful API per eseguire azioni App Runner. Per ulteriori informazioni, consulta la Documentazione di riferimento delle API AWS App Runner.
- AWS Command Line Interface (AWS CLI): fornisce comandi per un'ampia gamma di AWS servizi, tra cui Amazon VPC, ed è supportato su Windows, macOS e Linux. Per ulteriori informazioni, consulta AWS Command Line Interface.
- AWS SDKs— Fornisce informazioni specifiche per la lingua APIs e si occupa di molti dettagli di connessione, come il calcolo delle firme, la gestione dei tentativi di richiesta e la gestione degli errori. Per ulteriori informazioni, consulta AWS SDKs.

Prezzi per App Runner

App Runner offre un modo conveniente per eseguire l'applicazione. Paghi solo per le risorse utilizzate dal servizio App Runner. Il servizio si riduce a un numero inferiore di istanze di elaborazione quando il traffico delle richieste è inferiore. Hai il controllo sulle impostazioni di scalabilità: il numero più basso e più alto di istanze assegnate e il carico massimo gestito da un'istanza.

Per ulteriori informazioni sul ridimensionamento automatico di App Runner, consulta. the section called “Dimensionamento automatico”

Per informazioni sui prezzi, consulta Prezzi di AWS App Runner.

Cosa c'è dopo

Scopri come iniziare a usare App Runner nei seguenti argomenti:

- Configurazione— Completa i passaggi prerequisiti per l'utilizzo di App Runner.
- Nozioni di base— Distribuisci la tua prima applicazione su App Runner.

Configurazione per App Runner

Se sei un nuovo AWS cliente, completa i prerequisiti di configurazione elencati in questa pagina prima di iniziare a utilizzare. AWS App Runner

Per queste procedure di configurazione, utilizzi il servizio AWS Identity and Access Management (IAM). Per informazioni complete su IAM, consulta i seguenti materiali di riferimento:

- AWS Identity and Access Management (IAM)
- Guida per l'utente di IAM

Iscriviti per un Account AWS

Se non ne hai uno Account AWS, completa i seguenti passaggi per crearne uno.

Per iscriverti a un Account AWS

1. Apri la https://portal.aws.amazon.com/billing/registrazione.
2. Segui le istruzioni online.

   Parte della procedura di registrazione prevede la ricezione di una telefonata o di un messaggio di testo e l'immissione di un codice di verifica sulla tastiera del telefono.

   Quando ti iscrivi a un Account AWS, Utente root dell'account AWSviene creato un. L'utente root dispone dell'accesso a tutte le risorse e tutti i Servizi AWS nell'account. Come best practice di sicurezza, assegna l'accesso amministrativo a un utente e utilizza solo l'utente root per eseguire attività che richiedono l'accesso di un utente root.

AWS ti invia un'email di conferma dopo il completamento della procedura di registrazione. In qualsiasi momento, puoi visualizzare l'attività corrente del tuo account e gestirlo accedendo a https://aws.amazon.com/e scegliendo Il mio account.

Crea un utente con accesso amministrativo

Dopo esserti registrato Account AWS, proteggi Utente root dell'account AWS AWS IAM Identity Center, abilita e crea un utente amministrativo in modo da non utilizzare l'utente root per le attività quotidiane.

Proteggi i tuoi Utente root dell'account AWS

1. Accedi AWS Management Consolecome proprietario dell'account scegliendo Utente root e inserendo il tuo indirizzo Account AWS email. Nella pagina successiva, inserisci la password.

   Per informazioni sull'accesso utilizzando un utente root, consulta la pagina Signing in as the root user della Guida per l'utente di Accedi ad AWS .
2. Abilita l'autenticazione a più fattori (MFA) per l'utente root.

   Per istruzioni, consulta Abilitare un dispositivo MFA virtuale per l'utente Account AWS root (console) nella Guida per l'utente IAM.

Crea un utente con accesso amministrativo

1. Abilita Centro identità IAM.

   Per istruzioni, consulta Abilitazione di AWS IAM Identity Center nella Guida per l'utente di AWS IAM Identity Center .
2. In IAM Identity Center, assegna l'accesso amministrativo a un utente.

   Per un tutorial sull'utilizzo di IAM Identity Center directory come fonte di identità, consulta Configurare l'accesso utente con l'impostazione predefinita IAM Identity Center directory nella Guida per l'AWS IAM Identity Center utente.

Accesso come utente amministratore

- Per accedere con l'utente IAM Identity Center, utilizza l'URL di accesso che è stato inviato al tuo indirizzo e-mail quando hai creato l'utente IAM Identity Center.

  Per informazioni sull'accesso utilizzando un utente IAM Identity Center, consulta AWS Accedere al portale di accesso nella Guida per l'Accedi ad AWS utente.

Assegna l'accesso a ulteriori utenti

1. In IAM Identity Center, crea un set di autorizzazioni conforme alla best practice dell'applicazione di autorizzazioni con il privilegio minimo.

Segui le istruzioni riportate nella pagina Creazione di un set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .

2. Assegna al gruppo prima gli utenti e poi l'accesso con autenticazione unica (Single Sign-On).

   Per istruzioni, consulta Aggiungere gruppi nella Guida per l'utente di AWS IAM Identity Center .

Concessione dell'accesso programmatico

Gli utenti hanno bisogno di un accesso programmatico se vogliono interagire con l' AWS AWS Management Console esterno di. Il modo per concedere l'accesso programmatico dipende dal tipo di utente che accede. AWS

Per fornire agli utenti l'accesso programmatico, scegli una delle seguenti opzioni.

Quale utente necessita dell'accesso programmatico?	Per	Come
Identità della forza lavoro (Utenti gestiti nel centro identità IAM)	Utilizza credenziali temporane e per firmare le richieste programmatiche a AWS CLI,, AWS SDKs o. AWS APIs	Segui le istruzioni per l'interfa ccia che desideri utilizzare. • Per la AWS CLI, vedere Configurazione dell'uso AWS IAM Identity Center nella AWS CLI Guida per l'utente.AWS Command Line Interface • Per AWS SDKs gli strumenti e AWS APIs, consulta l'autenticazione di IAM Identity Center nella Guida di riferimento AWS SDKs and Tools.
IAM	Utilizza credenziali temporane e per firmare le richieste programmatiche a AWS CLI, AWS SDKs, o. AWS APIs	Seguendo le istruzioni riportate in Utilizzo delle credenziali temporanee con le

Quale utente necessita dell'accesso programmatico?	Per	Come
		AWS risorse nella Guida per l'utente IAM.
IAM	(Non consigliato) Utilizza credenziali a lungo termine per firmare richieste programmatiche a AWS CLI,, AWS SDKs o. AWS APIs	Segui le istruzioni per l'interfaccia che desideri utilizzare. • Per la AWS CLI, consulta Autenticazione tramite credenziali utente IAM nella Guida per l'utente.AWS Command Line Interface • Per gli strumenti AWS SDKs e gli strumenti, consulta Autenticazione tramite credenziali a lungo termine nella Guida di riferimento agli strumenti e agli AWS SDKs strumenti. • Per AWS APIs, consulta la sezione Gestione delle chiavi di accesso per gli utenti IAM nella Guida per l'utente IAM.

Cosa c'è dopo

Hai completato i passaggi preliminari. Per distribuire la tua prima applicazione su App Runner, consulta. Nozioni di base

Guida introduttiva a App Runner

AWS App Runner è un AWS servizio che offre un modo rapido, semplice ed economico per trasformare l'immagine o il codice sorgente di un contenitore esistente direttamente in un servizio Web in esecuzione in. Cloud AWS

Questo tutorial spiega come utilizzare l'applicazione AWS App Runner per distribuire l'applicazione su un servizio App Runner. Descrive la configurazione del codice sorgente e della distribuzione, la build del servizio e il runtime del servizio. Mostra anche come distribuire una versione del codice, apportare una modifica alla configurazione e visualizzare i log. Infine, il tutorial mostra come ripulire le risorse create seguendo le procedure del tutorial.

Argomenti

Prerequisiti

Prima di iniziare il tutorial, assicurati di eseguire le seguenti azioni:

1. Completa i passaggi di configurazione inConfigurazione.
2. Decidi se lavorare con un repository o un GitHub repository Bitbucket.
   - Per lavorare con un Bitbucket, devi prima creare un account Bitbucket, se non ne hai già uno. Se non conosci Bitbucket, consulta la sezione Guida introduttiva a Bitbucket nella documentazione di Bitbucket Cloud.
   - Con cui lavorare GitHub, crea un GitHubaccount, se non ne hai già uno. Se non l'hai mai fatto GitHub, vedi Guida introduttiva GitHub nei GitHubDocumenti.

> **Note**
>
> Puoi creare connessioni a più fornitori di repository dal tuo account. Quindi, se desideri eseguire la distribuzione sia da un repository che da un GitHub repository Bitbucket, puoi ripetere questa procedura. La prossima volta, crea un nuovo servizio App Runner e crea una nuova connessione all'account per l'altro provider di repository.

3. Crea un repository nell'account del provider del repository. Questo tutorial utilizza il nome del repository. `python-hello` Crea file nella directory principale del repository, con i nomi e il contenuto specificati negli esempi seguenti.

File per il repository di **`python-hello`** esempio

Example requirements.txt

```
pyramid==2.0
```

Example server.py

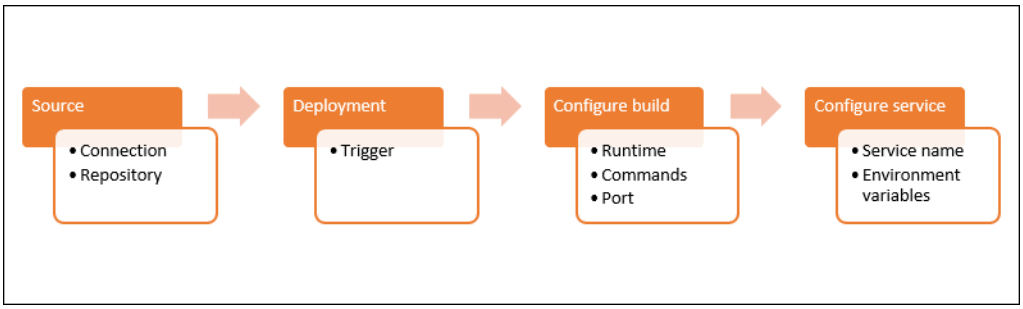
```
from wsgiref.simple_server import make_server
from pyramid.config import Configurator
from pyramid.response import Response
import os

def hello_world(request):
    name = os.environ.get('NAME')
    if name == None or len(name) == 0:
        name = "world"
    message = "Hello, " + name + "!\n"
    return Response(message)

if __name__ == '__main__':
    port = int(os.environ.get("PORT"))
    with Configurator() as config:
        config.add_route('hello', '/')
        config.add_view(hello_world, route_name='hello')
        app = config.make_wsgi_app()
    server = make_server('0.0.0.0', port, app)
    server.serve_forever()
```

Passaggio 1: creare un servizio App Runner

In questo passaggio, crei un servizio App Runner basato sull'archivio di codice sorgente di esempio su cui hai creato GitHub o di cui Bitbucket fa parte. the section called “Prerequisiti” L'esempio contiene un semplice sito Web in Python. Questi sono i passaggi principali da seguire per creare un servizio:

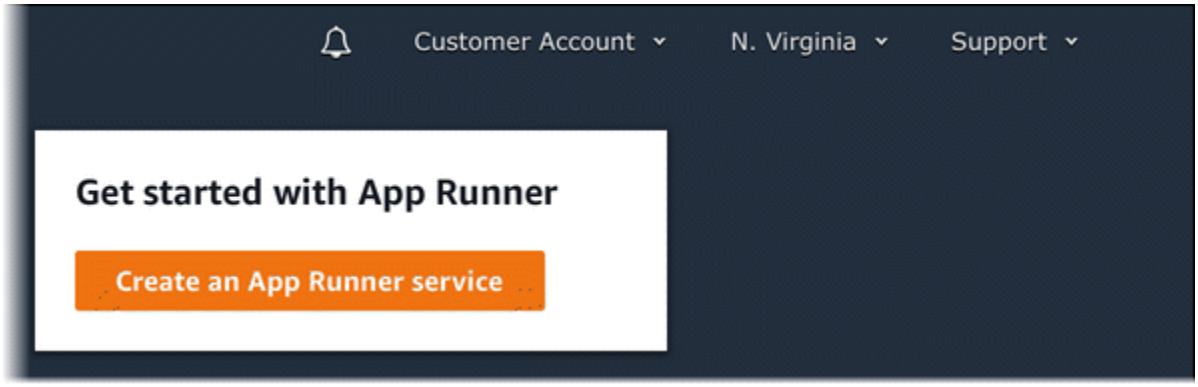
1. Configura il tuo codice sorgente.
2. Configura la distribuzione del codice sorgente.
3. Configura la build dell'applicazione.
4. Configura il tuo servizio.
5. Rivedi e conferma.

Il diagramma seguente illustra i passaggi per la creazione di un servizio App Runner:

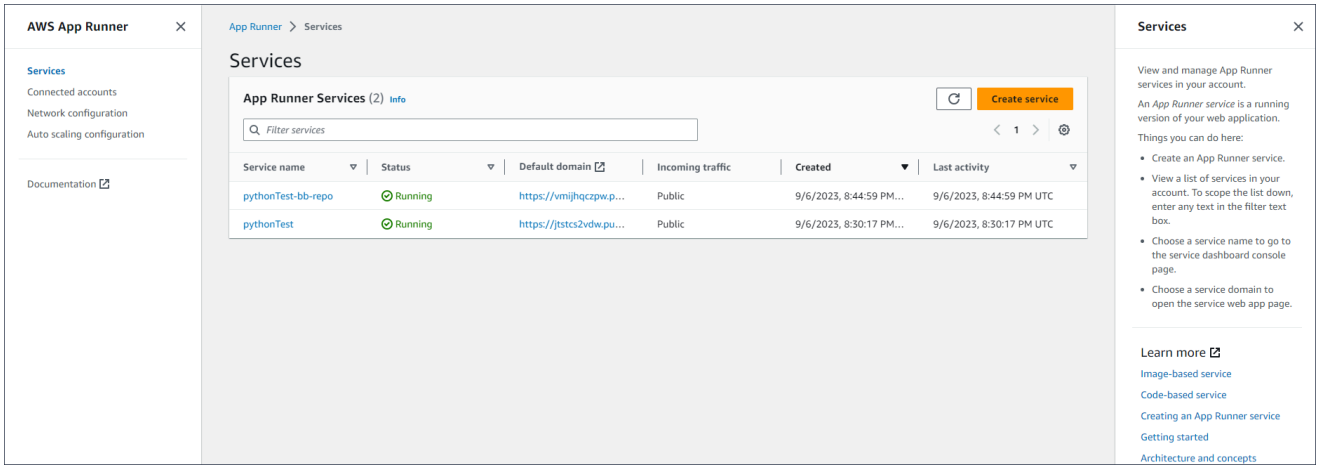


Per creare un servizio App Runner basato su un repository di codice sorgente

1. Configura il tuo codice sorgente.
   a. Apri la console App Runner e, nell'elenco Regioni, seleziona il tuo Regione AWS.
   b. Se Account AWS non dispone ancora di alcun servizio App Runner, viene visualizzata la home page della console. Scegli Crea un servizio App Runner.



Se Account AWS dispone di servizi esistenti, viene visualizzata la pagina Servizi con un elenco dei servizi. Selezionare Create service (Crea servizio).



c. Nella pagina Origine e distribuzione, nella sezione Sorgente, per Tipo di archivio, scegli Archivio del codice sorgente.

d. Seleziona un tipo di provider. Scegli uno dei due GitHubo Bitbucket.

e. Quindi scegli Aggiungi nuovo. Se richiesto, fornisci le tue credenziali GitHub o quelle di Bitbucket.

f. Scegli la prossima serie di passaggi in base al tipo di provider selezionato in precedenza.

> **Note**
>
> I seguenti passaggi per installare il connettore AWS sul tuo GitHub account sono passaggi una tantum. GitHub Puoi riutilizzare la connessione per creare più servizi App Runner basati sui repository di questo account. Quando disponi di una connessione esistente, sceglila e passa alla selezione del repository.
> Lo stesso vale per il connettore AWS per il tuo account Bitbucket. Se utilizzi entrambi GitHub e Bitbucket come repository di codice sorgente per i tuoi servizi

App Runner, dovrai installare un AWS Connector per ogni provider. Potrai quindi riutilizzare ogni connettore per creare altri servizi App Runner.

- Per GitHub, segui questi passaggi.

  i. Nella schermata successiva, inserisci un nome di connessione.

  ii. Se è la prima volta che lo usi GitHub con App Runner, seleziona Installa un altro.

  iii. Nella finestra di GitHub dialogo AWS Connector for, se richiesto, scegli il nome del tuo GitHub account.

  iv. Se viene richiesto di autorizzare il AWS connettore per GitHub, scegli Autorizza connessioni AWS.

  v. Nella finestra di GitHub dialogo Installa AWS Connector for, scegli Installa.

  Il nome del tuo account appare come l'GitHub account/organizzazione selezionato. Ora puoi scegliere un repository nel tuo account.

  vi. Per Repository, scegli il repository di esempio che hai creato,. `python-hello` Per Branch, scegli il nome del ramo predefinito del tuo repository (ad esempio, main).

  vii. Lascia la directory Source con il valore predefinito. La directory predefinita è la radice del repository. Il codice sorgente è stato memorizzato nella directory principale del repository nei passaggi precedenti relativi ai prerequisiti.

- Per Bitbucket, segui questi passaggi.

  i. Nella schermata successiva, inserisci un nome di connessione.

  ii. Se è la prima volta che usi Bitbucket con App Runner, seleziona Installa un altro.

  iii. Nella finestra di dialogo per le AWS CodeStar richieste di accesso, puoi selezionare il tuo spazio di lavoro e concedere l'accesso all' AWS CodeStar integrazione con Bitbucket. Seleziona il tuo spazio di lavoro, quindi seleziona Concedi l'accesso.

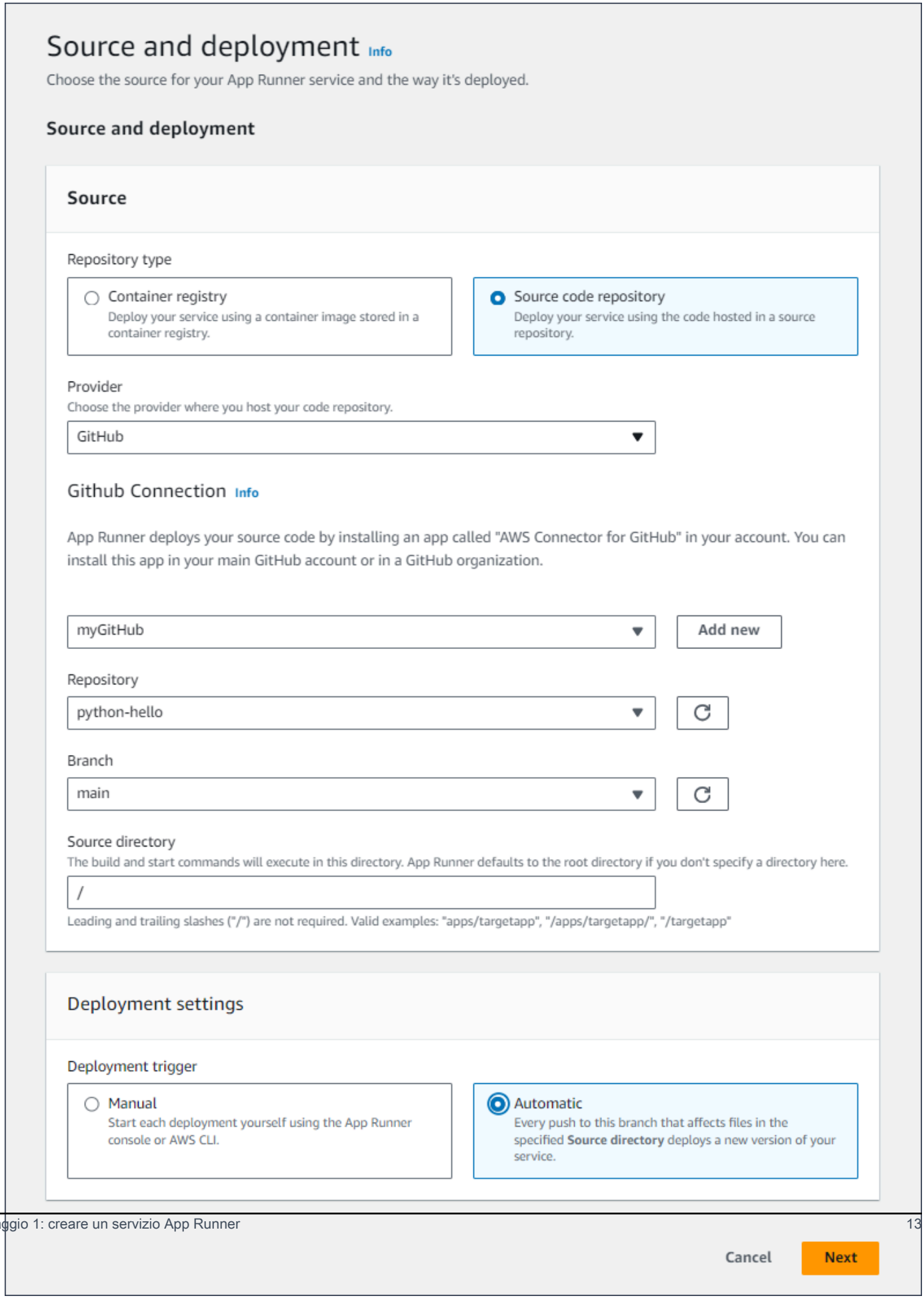
  iv. Successivamente verrai reindirizzato alla AWS console. Verifica che l'applicazione Bitbucket sia impostata sull'area di lavoro Bitbucket corretta e seleziona Avanti.

  v. Per Repository, scegli il repository di esempio che hai creato,. `python-hello` Per Branch, scegli il nome del ramo predefinito del tuo repository (ad esempio, main).

vi. Lascia la directory Source con il valore predefinito. La directory predefinita è la radice del repository. Il codice sorgente è stato memorizzato nella directory principale del repository nei passaggi precedenti relativi ai prerequisiti.

2. Configura le distribuzioni: nella sezione Impostazioni di distribuzione, scegli Automatico, quindi scegli Avanti.

> **Note**
>
> Con la distribuzione automatica, ogni nuovo commit nella directory di origine del repository distribuisce automaticamente una nuova versione del servizio.


Source and deployment Info
Choose the source for your App Runner service and the way it's deployed.
Source and deployment
Source
Repository type
Container registry
Deploy your service using a container image stored in a container registry.
Source code repository
Deploy your service using the code hosted in a source repository.
Provider
Choose the provider where you host your code repository.
GitHub
Github Connection Info
App Runner deploys your source code by installing an app called "AWS Connector for GitHub" in your account. You can install this app in your main GitHub account or in a GitHub organization.
myGitHub
Add new
Repository
python-hello
Branch
main
Source directory
The build and start commands will execute in this directory. App Runner defaults to the root directory if you don't specify a directory here.
/
Leading and trailing slashes ("/") are not required. Valid examples: "apps/targetapp", "/apps/targetapp/", "/targetapp"
Deployment settings
Deployment trigger
Manual
Start each deployment yourself using the App Runner console or AWS CLI.
Automatic
Every push to this branch that affects files in the specified Source directory deploys a new version of your service.
Cancel
Next

3. Configura la build dell'applicazione.

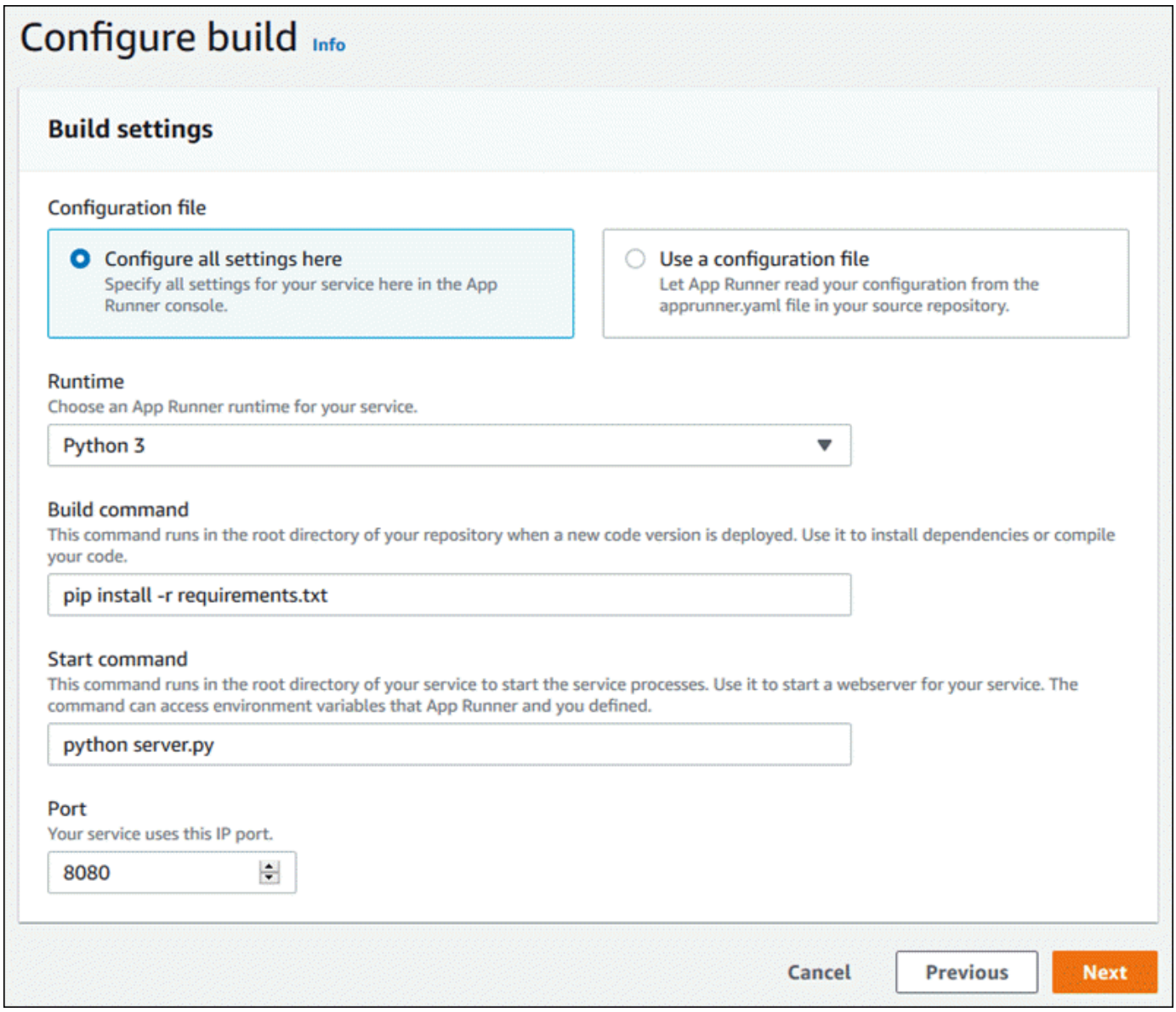
   a. Nella pagina Configura build, per File di configurazione, scegli Configura tutte le impostazioni qui.

   b. Fornisci le seguenti impostazioni di build:

      - Runtime: scegli Python 3.
      - Comando Build — Invio**`pip install -r requirements.txt`**.
      - Comando di avvio: Invio**`python server.py`**.
      - Porta: Invio**`8080`**.

   c. Scegli Next (Successivo).

> **Note**
>
> Il runtime Python 3 crea un'immagine Docker utilizzando un'immagine Python 3 di base e il codice Python di esempio. Quindi avvia un servizio che esegue un'istanza contenitore di questa immagine.



4. Configura il tuo servizio.

   a. Nella pagina Configura servizio, nella sezione Impostazioni del servizio, inserisci un nome di servizio.

   b. In Variabili di ambiente, seleziona Aggiungi variabile di ambiente. Fornisci i seguenti valori per la variabile di ambiente.

      - Fonte: scegli Testo normale
      - Nome della variabile di ambiente — **`NAME`**
      - Valore della variabile di ambiente: qualsiasi nome (ad esempio, il nome).

> **Note**
>
> L'applicazione di esempio legge il nome impostato in questa variabile di ambiente e lo visualizza sulla pagina Web.

c. Scegli Next (Successivo).

Configure service Info

Service settings

Service name

python-test

Virtual CPU & memory

1 vCPU | 2 GB

Environment variables — *optional* Info

Add environment variables in plain text or reference them from Secrets Manager and SSM Parameter Store. Update IAM Policies using the IAM Policy template given below to securely reference secrets and configurations as environment variables.

No environment variables have been configured.

Add environment variable

You can add up to 50 more items.

▶ IAM policy templates

▶ Auto scaling Info

Configure automatic scaling behavior.

▶ Health check Info

Configure load balancer health checks.

▶ Security Info

Specify an Instance role and an AWS KMS encryption key

▶ Networking Info

Configure the way your service communicates with other applications, services, and resources.

▶ Observability

Configure observability tooling.

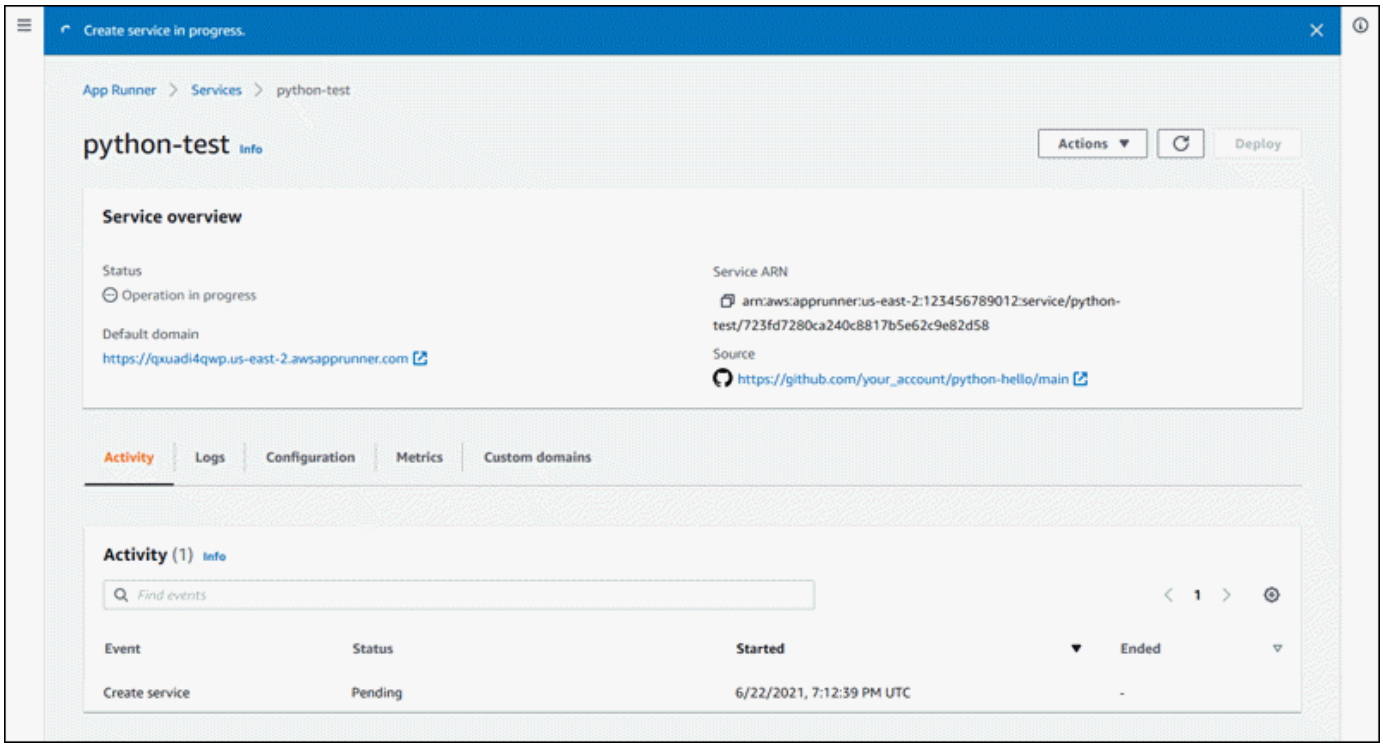
▼ Tags Info

Use tags to search and filter your resources, track your AWS costs, and control access permissions.

Tags — *optional*

5. Nella pagina Rivedi e crea, verifica tutti i dettagli che hai inserito, quindi scegli Crea e distribuisci.

   Se il servizio viene creato correttamente, la console mostra la dashboard del servizio, con una panoramica del servizio del nuovo servizio.



6. Verifica che il servizio sia in esecuzione.

   a. Nella pagina del pannello di controllo del servizio, attendi che lo stato del servizio sia in esecuzione.

   b. Scegli il valore di dominio predefinito: è l'URL del sito Web del tuo servizio.

> **Note**
>
> Per aumentare la sicurezza delle tue applicazioni App Runner, il dominio*.awsapprunner.com è registrato nella Public Suffix List (PSL). Per una maggiore sicurezza, ti consigliamo di utilizzare i cookie con un `__Host-` prefisso se hai bisogno di impostare cookie sensibili nel nome di dominio predefinito per le tue applicazioni App Runner. Questa pratica ti aiuterà a difendere il tuo dominio dai tentativi CSRF (cross-site request forgery). Per ulteriori informazioni, consulta la pagina Impostazione cookie nella pagina Mozilla Developer Network.

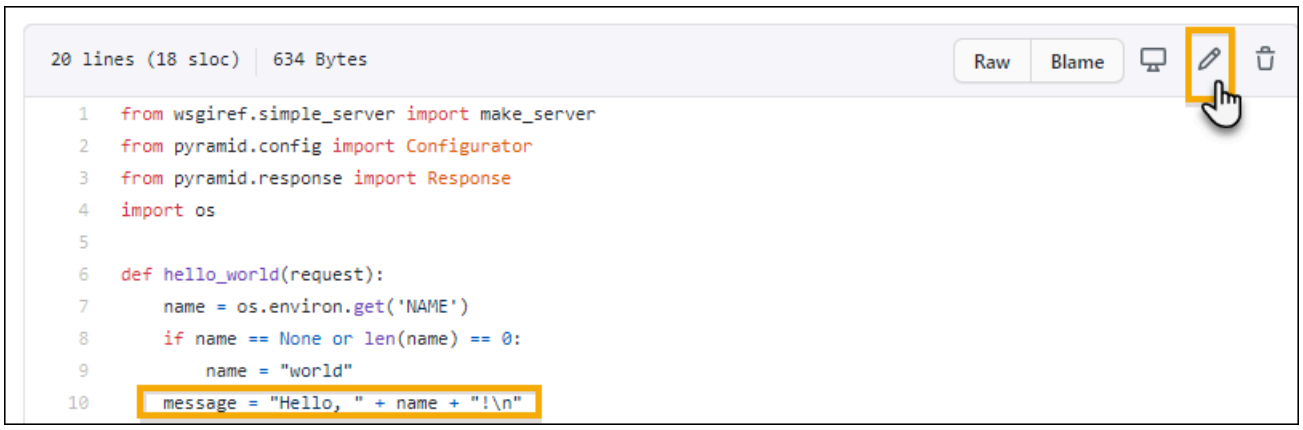
Viene visualizzata una pagina Web: Hello,! ***your name***

Fase 2: Modificare il codice di servizio

In questo passaggio, apporti una modifica al codice nella directory dei sorgenti del repository. La funzionalità CI/CD di App Runner crea e distribuisce automaticamente la modifica al servizio.

Per apportare una modifica al codice di servizio

1. Vai al tuo repository di esempio.
2. Modifica il file denominato`server.py`.
3. Nell'espressione assegnata alla variabile`message`, modificate il testo `Hello` in`Good morning`.
4. Salvate e salvate le modifiche nel repository.
5. I passaggi seguenti illustrano la modifica del codice di servizio in un GitHub repository.
    a. Passa al repository di esempio GitHub .
    b. Scegli il nome del file `server.py` per accedere a quel file.
    c. Scegli Modifica questo file (l'icona a forma di matita).
    d. Nell'espressione assegnata alla variabile`message`, modifica il testo `Hello` in`Good morning`.



e. Scegliere Commit changes (Applica modifiche).

6. Il nuovo commit inizia a essere distribuito per il servizio App Runner. Nella pagina del dashboard del servizio, lo stato del servizio cambia in Operazione in corso.

   Attendi il termine della distribuzione. Nella pagina del dashboard del servizio, lo stato del servizio dovrebbe tornare a Running.

7. Verifica che l'implementazione sia riuscita: aggiorna la scheda del browser in cui viene visualizzata la pagina web del servizio.

   La pagina ora mostra il messaggio modificato: Buongiorno,! *your name*

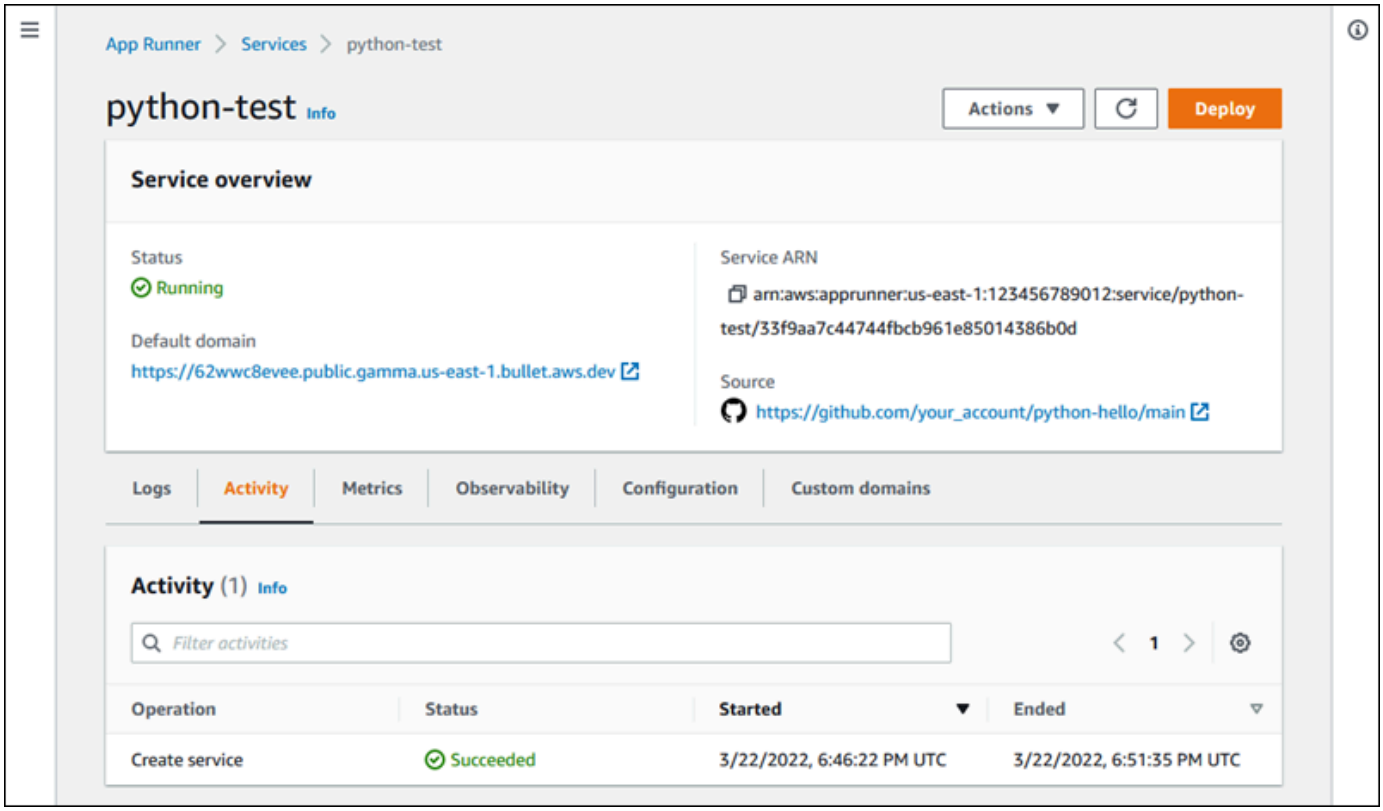
Passaggio 3: apportare una modifica alla configurazione

In questo passaggio, si apporta una modifica al valore della variabile di **NAME** ambiente, per dimostrare una modifica alla configurazione del servizio.

Per modificare il valore di una variabile di ambiente

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua Regione AWS.
2. Nel pannello di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

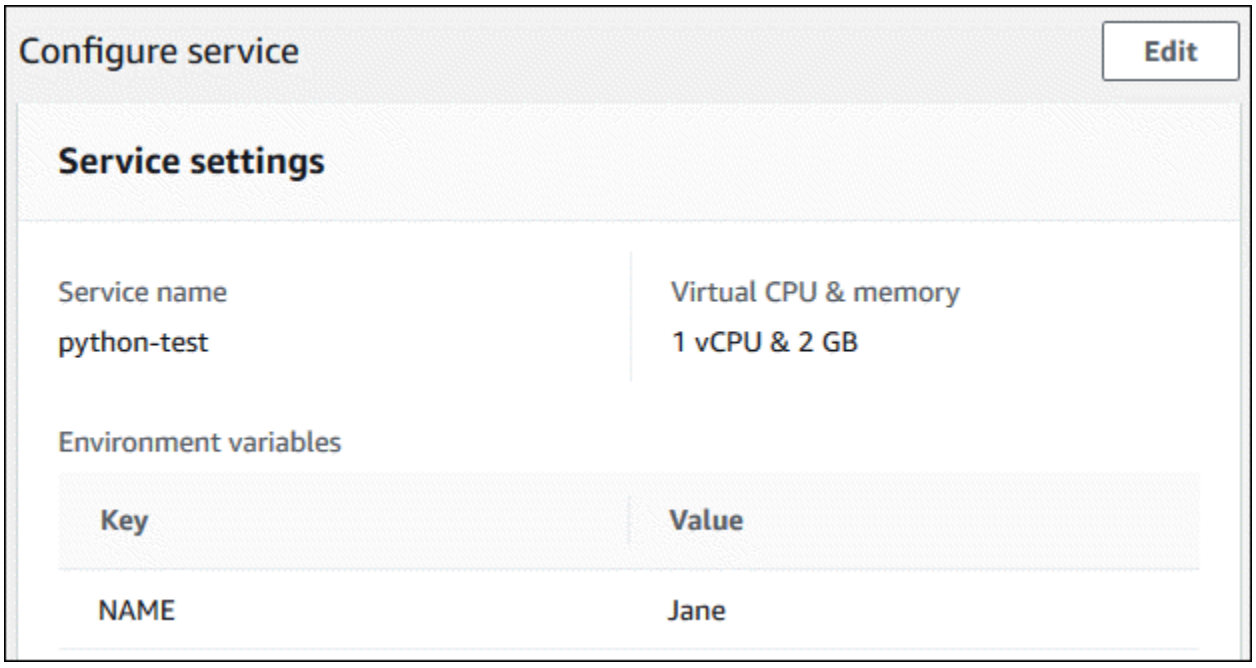
   La console mostra la dashboard del servizio con una panoramica del servizio.



3. Nella pagina del dashboard del servizio, scegli la scheda Configurazione.

   La console mostra le impostazioni di configurazione del servizio in diverse sezioni.

4. Nella sezione Configura servizio, scegli Modifica.



5. Per la variabile di ambiente con la chiave**NAME**, modifica il valore con un nome diverso.

6. Scegli Apply changes (Applica modifiche).

   App Runner avvia il processo di aggiornamento. Nella pagina del dashboard del servizio, lo stato del servizio cambia in Operazione in corso.

7. Attendi il termine dell'aggiornamento. Nella pagina del pannello di controllo del servizio, lo stato del servizio dovrebbe tornare in esecuzione.

8. Verifica che l'aggiornamento sia avvenuto correttamente: aggiorna la scheda del browser in cui viene visualizzata la pagina web del servizio.

   La pagina ora mostra il nome modificato: Buongiorno,! *new name*

Passaggio 4: Visualizza i registri del tuo servizio

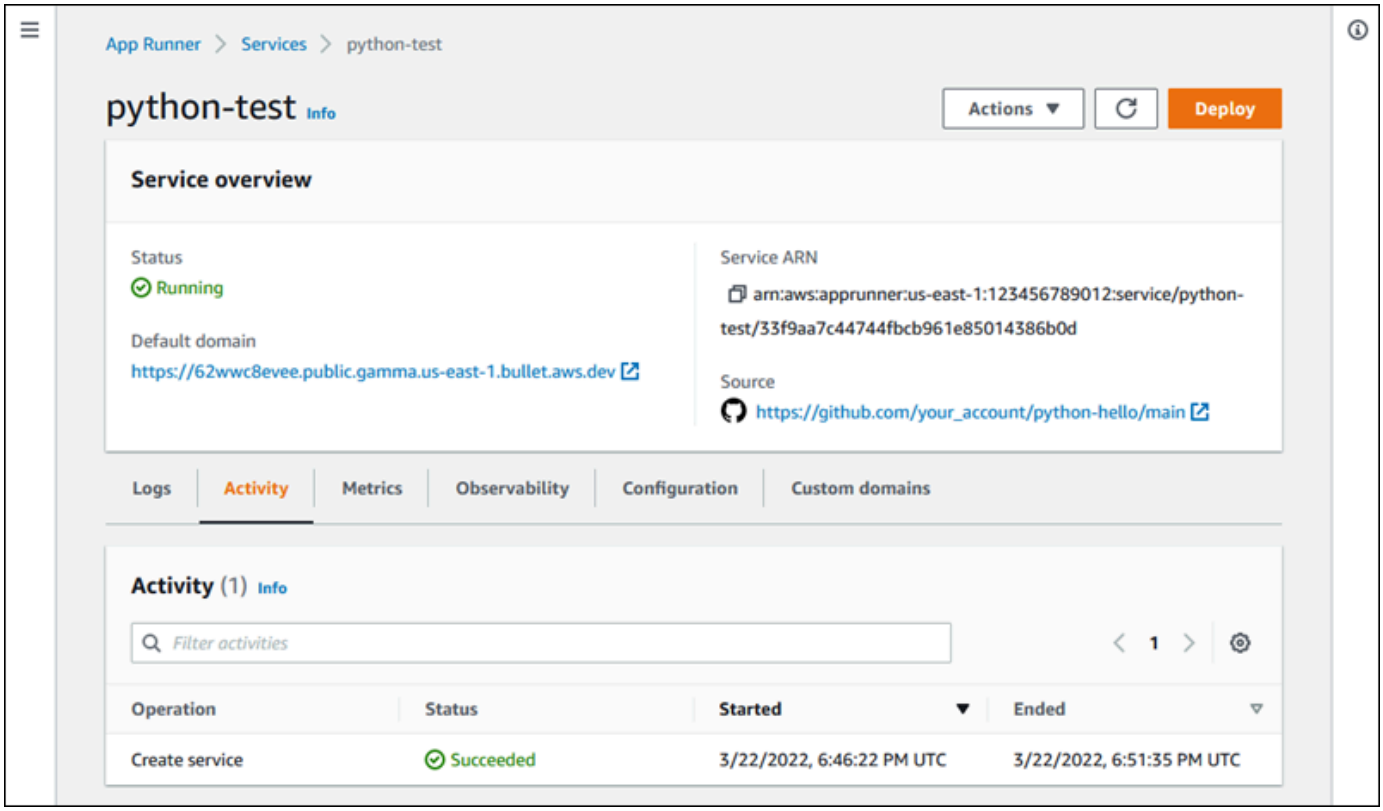
In questo passaggio, si utilizza la console App Runner per visualizzare i registri del servizio App Runner. App Runner trasmette i log ad Amazon CloudWatch Logs (CloudWatch Logs) e li visualizza sulla dashboard del tuo servizio. Per informazioni sui log di App Runner, consulta. the section called “Registri (registri) CloudWatch ”

Per visualizzare i registri del servizio

1. Apri la console App Runner e, nell'elenco Regioni, seleziona il tuo. Regione AWS

2. Nel pannello di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

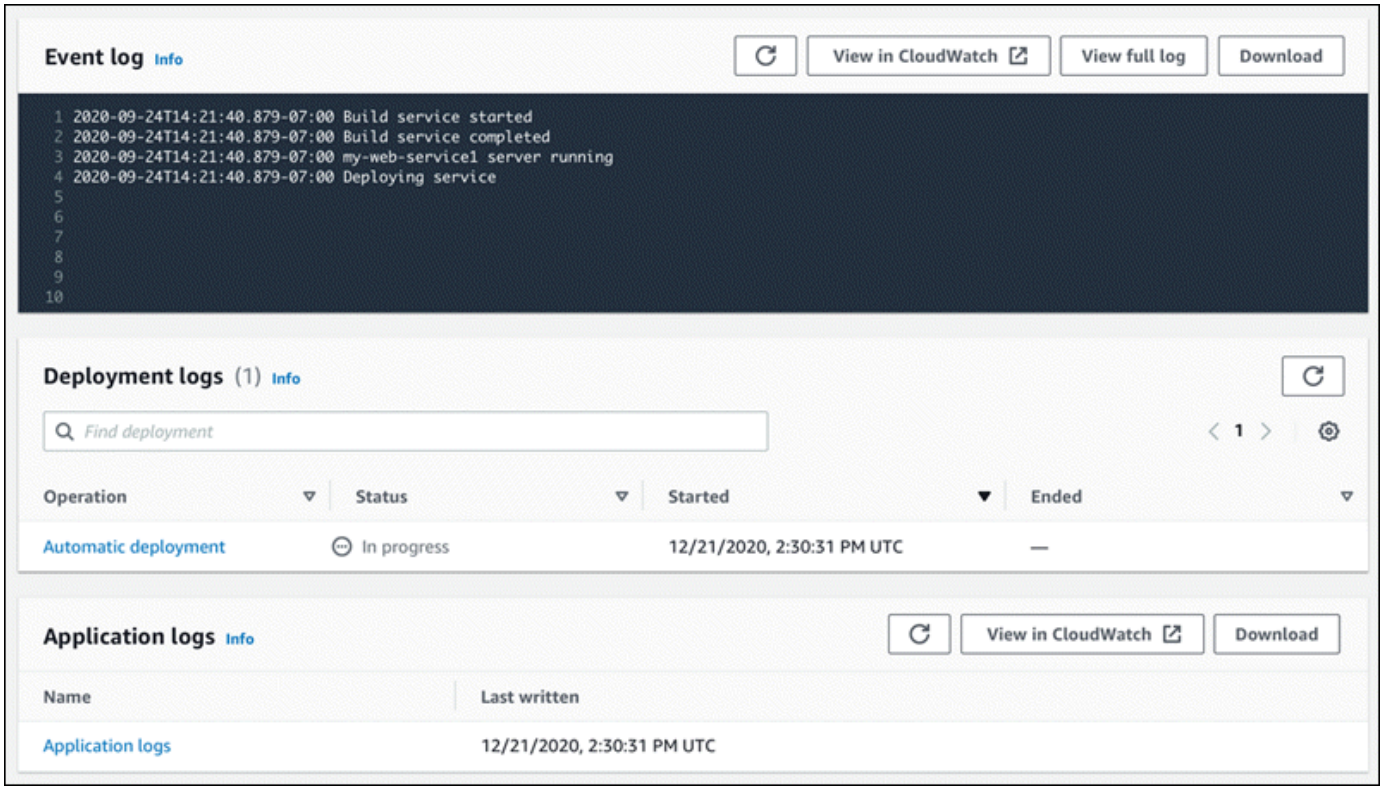
   La console mostra la dashboard del servizio con una panoramica del servizio.



3. Nella pagina del dashboard del servizio, scegli la scheda Registri.

   La console mostra alcuni tipi di log in diverse sezioni:

   - Registro eventi: attività nel ciclo di vita del servizio App Runner. La console mostra gli eventi più recenti.
   - Registri di distribuzione: invia le distribuzioni del repository al servizio App Runner. La console visualizza un flusso di log separato per ogni distribuzione.
   - Registri delle applicazioni: l'output dell'applicazione Web distribuita nel servizio App Runner. La console combina l'output di tutte le istanze in esecuzione in un unico flusso di log.



4. Per trovare implementazioni specifiche, scorri l'elenco dei log di distribuzione inserendo un termine di ricerca. È possibile cercare qualsiasi valore visualizzato nella tabella.
5. Per visualizzare il contenuto di un registro, scegli Visualizza registro completo (registro eventi) o il nome del flusso di registro (registri di distribuzione e applicazione).
6. Scegli Scarica per scaricare un registro. Per un flusso di log di distribuzione, seleziona prima un flusso di log.
7. Scegli Visualizza in CloudWatch per aprire la CloudWatch console e utilizzare tutte le sue funzionalità per esplorare i log di servizio di App Runner. Per un flusso di log di distribuzione, seleziona prima un flusso di log.

> **Note**
>
> La CloudWatch console è particolarmente utile se si desidera visualizzare i log delle applicazioni di istanze specifiche anziché il registro dell'applicazione combinato.

Fase 5: rimozione

Ora hai imparato come creare un servizio App Runner, visualizzare i log e apportare alcune modifiche. In questo passaggio, elimini il servizio per rimuovere le risorse che non ti servono più.

Per eliminare il servizio

1. Nella pagina del pannello di controllo del servizio, scegli Azioni, quindi scegli Elimina servizio.
2. Nella finestra di dialogo di conferma, inserisci il testo richiesto, quindi scegli Elimina.

   Risultato: la console accede alla pagina Servizi. Il servizio che hai appena eliminato mostra lo stato ELIMINAZIONE. Poco tempo dopo scompare dall'elenco.

Prendi in considerazione anche l'eliminazione delle connessioni GitHub e di Bitbucket che hai creato come parte di questo tutorial. Per ulteriori informazioni, consulta the section called “Connessioni”.

Cosa c'è dopo

Ora che hai distribuito il tuo primo servizio App Runner, scopri di più nei seguenti argomenti:

- Architettura e concetti— L'architettura, i concetti principali e le AWS risorse relative ad App Runner.
- Servizio basato su immaginie Servizio basato su codice — I due tipi di sorgenti applicative che App Runner può implementare.
- Sviluppo per App Runner— Cose da sapere quando si sviluppa o si migra il codice dell'applicazione da distribuire su App Runner.
- Console App Runner— Gestisci e monitora il tuo servizio utilizzando la console App Runner.
- Gestione del servizio— Gestisci il ciclo di vita del tuo servizio App Runner.
- Osservabilità— Ottieni visibilità sulle operazioni del servizio App Runner monitorando le metriche, leggendo i log, gestendo gli eventi, tracciando le chiamate di servizio e tracciando gli eventi delle applicazioni come le chiamate HTTP.
- File di configurazione di App Runner— Un modo basato sulla configurazione per specificare le opzioni per il comportamento di compilazione e di esecuzione del servizio App Runner.
- API App Runner— Utilizza l'API (Application Programming Interface) di App Runner per creare, leggere, aggiornare ed eliminare le risorse App Runner.
- Sicurezza— I diversi modi in cui AWS garantisci la sicurezza del cloud mentre utilizzi App Runner e altri servizi.

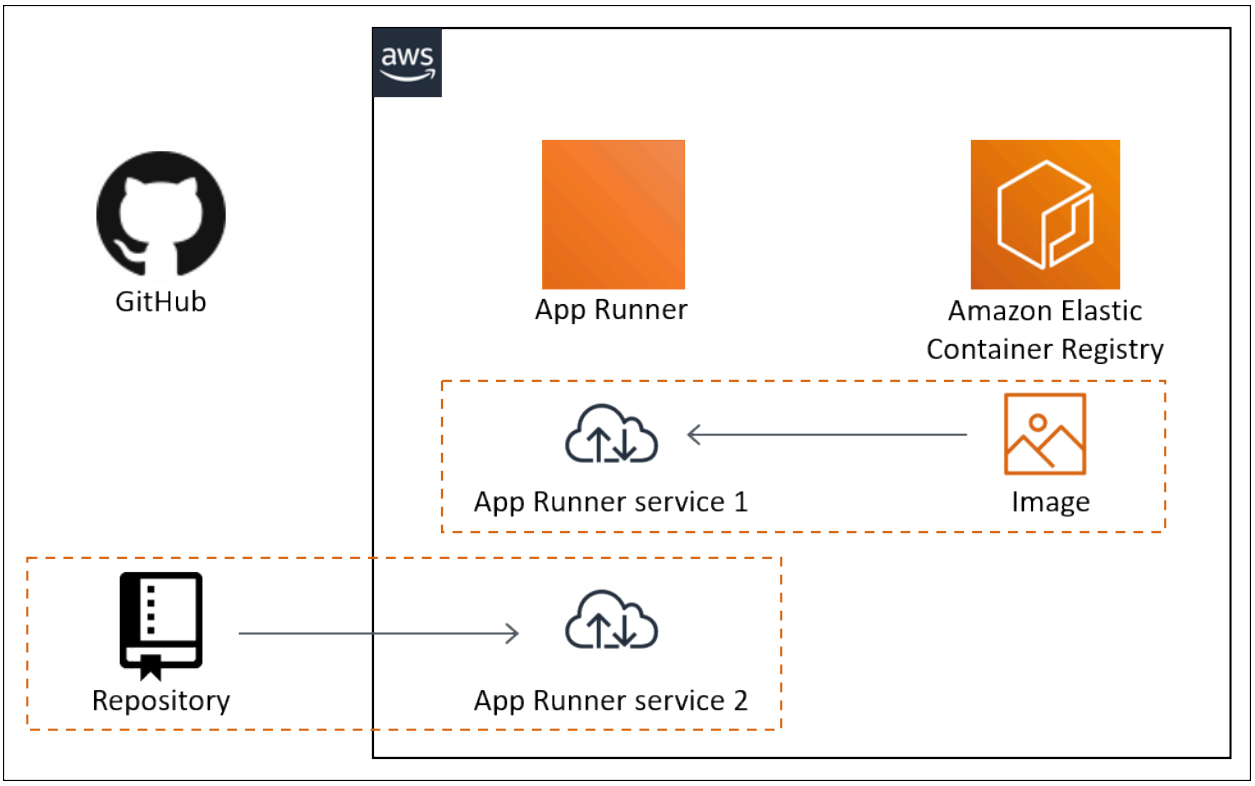
Architettura e concetti di App Runner

AWS App Runner prende il codice sorgente o l'immagine sorgente da un repository e crea e gestisce un servizio Web funzionante per te in. Cloud AWS In genere, è necessario chiamare una sola azione di App Runner per creare il servizio. CreateService

Con un archivio di immagini di origine, fornisci un'immagine del ready-to-use contenitore che App Runner può distribuire per eseguire il tuo servizio web. Con un repository di codice sorgente, fornisci il codice e le istruzioni per creare ed eseguire un servizio Web e scegli come destinazione un ambiente di runtime specifico. App Runner supporta diverse piattaforme di programmazione, ognuna con uno o più runtime gestiti per le versioni principali della piattaforma.

Al momento, App Runner può recuperare il codice sorgente da un Bitbucket o da un GitHubrepository oppure può recuperare l'immagine sorgente da Amazon Elastic Container Registry (Amazon ECR) Elastic ECR) del tuo. Account AWS

Il diagramma seguente mostra una panoramica dell'architettura del servizio App Runner. Nel diagramma sono presenti due servizi di esempio: uno distribuisce il codice sorgente da GitHub e l'altro distribuisce un'immagine sorgente da Amazon ECR. Lo stesso flusso si applica al repository Bitbucket.



Concetti di App Runner

Di seguito sono riportati i concetti chiave relativi al servizio Web in esecuzione in App Runner:

- Servizio App Runner: una AWS risorsa che App Runner utilizza per distribuire e gestire l'applicazione in base al repository di codice sorgente o all'immagine del contenitore. Un servizio App Runner è una versione in esecuzione dell'applicazione. Per ulteriori informazioni sulla creazione di un servizio, vederethe section called “Creazione”.
- Tipo di sorgente: il tipo di repository di origine fornito per la distribuzione del servizio App Runner: codice sorgente o immagine sorgente.
- Provider di repository: il servizio di repository che contiene l'origine dell'applicazione (ad esempio GitHub, Bitbucket o Amazon ECR).
- Connessione App Runner: una AWS risorsa che consente a App Runner di accedere a un account del provider di repository (ad esempio, un account o un'organizzazione). GitHub Per ulteriori informazioni sulle connessioni, consulta the section called “Connessioni”.
- Runtime: un'immagine di base per la distribuzione di un repository di codice sorgente. App Runner offre una varietà di runtime gestiti per diverse piattaforme e versioni di programmazione. Per ulteriori informazioni, consulta Servizio basato su codice.
- Distribuzione: un'azione che applica una versione del repository di origine (codice o immagine) a un servizio App Runner. La prima distribuzione al servizio avviene come parte della creazione del servizio. Le distribuzioni successive possono avvenire in due modi:
  - Distribuzione automatica: funzionalità CI/CD. È possibile configurare un servizio App Runner per creare automaticamente (per il codice sorgente) e distribuire ogni versione dell'applicazione così come appare nel repository. Può trattarsi di un nuovo commit in un archivio di codice sorgente o di una nuova versione dell'immagine in un archivio di immagini di origine.
  - Distribuzione manuale: una distribuzione del servizio App Runner che avvii esplicitamente.
- Dominio personalizzato: dominio associato al servizio App Runner. Gli utenti dell'applicazione Web possono utilizzare questo dominio per accedere al servizio Web anziché al sottodominio App Runner predefinito. Per ulteriori informazioni, consulta the section called “Nomi di dominio personalizzati”.

> **Note**
>
> Per aumentare la sicurezza delle applicazioni App Runner, il dominio*.awsapprunner.com è registrato nella Public Suffix List (PSL). Per una maggiore sicurezza, ti consigliamo di

utilizzare i cookie con un `__Host-` prefisso se hai bisogno di impostare cookie sensibili nel nome di dominio predefinito per le tue applicazioni App Runner. Questa pratica ti aiuterà a difendere il tuo dominio dai tentativi CSRF (cross-site request forgery). Per ulteriori informazioni, consulta la pagina Impostazione cookie nella pagina Mozilla Developer Network.

- Manutenzione: un'attività che App Runner esegue occasionalmente sull'infrastruttura che esegue il servizio App Runner. Quando la manutenzione è in corso, lo stato del servizio cambia temporaneamente in `OPERATION_IN_PROGRESS` (Operazione in corso nella console) per alcuni minuti. Le azioni sul servizio (ad esempio, distribuzione, aggiornamento della configurazione, pausa/ripresa o eliminazione) vengono bloccate durante questo periodo. Riprova l'azione qualche minuto dopo, quando lo stato del servizio torna a essere. `RUNNING`

  > **Note**
  >
  > Se l'azione fallisce, ciò non significa che il servizio App Runner non sia attivo. L'applicazione è attiva e continua a gestire le richieste. È improbabile che il servizio subisca interruzioni.

  In particolare, App Runner esegue la migrazione del servizio se rileva problemi nell'hardware sottostante che ospita il servizio. Per evitare interruzioni del servizio, App Runner distribuisce il servizio su un nuovo set di istanze e sposta il traffico verso di esse (una distribuzione blu-verde). Occasionalmente potresti notare un leggero aumento temporaneo degli addebiti.

Configurazioni supportate da App Runner

Quando si configura un servizio App Runner, si specifica la configurazione virtuale della CPU e della memoria da allocare al servizio. Paghi in base alla configurazione di elaborazione selezionata. Per ulteriori informazioni sui prezzi, consultare Prezzi di AWS Resource Groups.

La tabella seguente fornisce informazioni sulle configurazioni di vCPU e memoria supportate da App Runner:

CPU	Memoria
0,25 vCPU	0,5 GB

CPU	Memoria
0,25 vCPU	1 GB
0,5 vCPU	1 GB
1 vCPU	2 GB
1 vCPU	3 GB
1 vCPU	4 GB
2 vCPU	4 GB
2 vCPU	6 GB
4 vCPU	8 GB
4 vCPU	10 GB
4 vCPU	12 GB

Risorse App Runner

Quando usi App Runner, crei e gestisci alcuni tipi di risorse nel tuo. Account AWS Queste risorse vengono utilizzate per accedere al codice e gestire i servizi.

La tabella seguente fornisce una panoramica di queste risorse:

Nome risorsa	Descrizione
Service	Rappresenta una versione in esecuzione dell'applicazione. Gran parte del resto di questa guida descrive i tipi di servizio, la gestione, la configurazione e il monitoraggio. ARN: `arn:aws:apprunner:` *`region`*`:`*`account-id`* `:service/` *`service-name`* `[/`*`service-id`* `]`
Connection	Fornisce ai servizi App Runner l'accesso a repository privati archiviat i presso provider di terze parti. Esiste come risorsa separata per la

Nome risorsa	Descrizione
	condivisione tra più servizi. Per ulteriori informazioni sulle connessioni, consulta the section called “Connessioni”. ARN: `arn:aws:apprunner:` *`region`*`:`*`account-id`* `:connection/` *`connection-name`* `[/`*`connection-id`* `]`
AutoScalingConfiguration	Fornisce ai servizi App Runner impostazioni che controllano il ridimensionamento automatico dell'applicazione. Esiste come risorsa separata per la condivisione tra più servizi. Per ulteriori informazioni sul dimensionamento automatico, consulta the section called “Dimensionamento automatico”. ARN: `arn:aws:apprunner:` *`region`*`:`*`account-id`* `:autoscalingconfiguration/` *`config-name`* `[/`*`config-revision`* `[/`*`config-id`* `]]`
ObservabilityConfiguration	Configura funzionalità aggiuntive di osservabilità delle applicazioni per i servizi App Runner. Esiste come risorsa separata per la condivisione tra più servizi. Per ulteriori informazioni sulla configurazione dell'osservabilità, vederethe section called “Configurazione dell'osservabilità”. ARN: `arn:aws:apprunner:` *`region`*`:`*`account-id`* `:observabilityconfiguration/` *`config-name`* `[/`*`config-revision`* `[/`*`config-id`* `]]`
VpcConnector	Configura le impostazioni VPC per i tuoi servizi App Runner. Esiste come risorsa separata per la condivisione tra più servizi. Per ulteriori informazioni sulla funzionalità VPC, vedere. the section called “Traffico in uscita” ARN: `arn:aws:apprunner:` *`region`*`:`*`account-id`* `:vpcconnector/` *`connector-name`* `[/`*`connector-revision`* `[/`*`connector-id`* `]]`

Nome risorsa	Descrizione
VpcIngressConnection	È una AWS App Runner risorsa utilizzata per configurare il traffico in entrata. Stabilisce una connessione tra un endpoint di interfaccia VPC e il servizio App Runner, per rendere il servizio App Runner accessibi le solo dall'interno di un Amazon VPC. Per ulteriori informazioni sulla funzionalità di Connection, consulta. VPCIngress the section called “Abilita l'endpoint privato” ARN: `arn:aws:apprunner:` *`region`*`:`*`account-id`*` :vpcingressconnection/` *`vpc-ingress-connection-name`* ` [/`*`connector-id`*` ]]`

Quote di risorse di App Runner

AWS impone al tuo account alcune quote (note anche come limiti) per l' AWS utilizzo delle risorse in ciascuno di essi. Regione AWS La tabella seguente elenca le quote relative alle risorse di App Runner. Le quote sono elencate anche negli AWS App Runner endpoint e le quote in. Riferimenti generali di AWS

Quota di risorse		Descrizione	Valore predefinito	Regolabile?
Services		Il numero massimo di servizi che puoi creare nel tuo account per ciascuno Regione AWS.	30	✓ Sì
Connections		Il numero massimo di connessioni che puoi creare nel tuo account per ciascuna di esse Regione AWS. Puoi utilizzare una singola connessione in più servizi.	10	✓ Sì
Auto scaling configurations	nomi	Il numero massimo di nomi univoci che puoi avere nelle configurazioni di ridimensionamento automatico che crei nel tuo account per ciascuna di esse. Regione AWS Puoi utilizzare una	10	✓ Sì

Quota di risorse		Descrizione	Valore predefinito	Regolabile?
		singola configurazione di scalabilità automatica in più servizi.		
	revisioni per nome	Il numero massimo di revisioni della configurazione con scalabilità automatica che puoi creare nel tuo account per ciascuna Regione AWS per ogni nome univoco. È possibile utilizzare un'unica revisione della configurazione con scalabilità automatica in più servizi.	5	✗ No
Observability configurations	nomi	Il numero massimo di nomi univoci che puoi avere nelle configurazioni di osservabilità che crei nel tuo account per ciascuna di esse. Regione AWS Puoi utilizzare una singola configurazione di osservabilità in più servizi.	10	✓ Sì
	revisioni per nome	Il numero massimo di revisioni della configurazione di osservabilità che puoi creare nel tuo account per ciascuna Regione AWS per ogni nome univoco. È possibile utilizzare una singola revisione della configurazione di osservabilità in più servizi.	10	✗ No
VPC connectors		Il numero massimo di connettori VPC che puoi creare nel tuo account per ciascuno. Regione AWS Puoi utilizzare un singolo connettore VPC in più servizi.	10	✓ Sì
VPC Ingress Connection		Il numero massimo di connessioni di ingresso VPC che puoi creare nel tuo account per ciascuna. Regione AWS Puoi utilizzare una singola connessione di ingresso VPC per accedere a più servizi App Runner.	1	✗ No

La maggior parte delle quote è regolabile ed è possibile richiedere un aumento della quota. Per ulteriori informazioni, consultare Richiesta di un aumento di quota nella Guida per l'utente di Service Quotas.

Servizio App Runner basato su un'immagine sorgente

Puoi utilizzarli AWS App Runner per creare e gestire servizi basati su due tipi di fonti di servizio fondamentalmente diversi: codice sorgente e immagine sorgente. Indipendentemente dal tipo di sorgente, App Runner si occupa dell'avvio, dell'esecuzione, della scalabilità e del bilanciamento del carico del servizio. Puoi utilizzare la CI/CD funzionalità di App Runner per tenere traccia delle modifiche all'immagine o al codice sorgente. Quando App Runner rileva una modifica, crea automaticamente (per il codice sorgente) e distribuisce la nuova versione nel servizio App Runner.

Questo capitolo descrive i servizi basati su un'immagine sorgente. Per informazioni sui servizi basati sul codice sorgente, vedereServizio basato su codice.

Un'immagine sorgente è un'immagine contenitore pubblica o privata memorizzata in un archivio di immagini. Indirizzi App Runner a un'immagine e avvia un servizio che esegue un contenitore basato su tale immagine. Non è necessaria alcuna fase di costruzione. Piuttosto, fornisci un' ready-to-deployimmagine.

> **Note**
>
> Quando fornisci immagini dei contenitori, sei responsabile dell'aggiornamento e della correzione regolari di tali immagini. Sebbene App Runner gestisca l'infrastruttura, è necessario garantire la sicurezza e up-to-date lo stato delle immagini del contenitore fornite. Per ulteriori informazioni, consulta la documentazione di AWS App Runner

Fornitori di repository di immagini

App Runner supporta i seguenti provider di repository di immagini:

- Amazon Elastic Container Registry (Amazon ECR): archivia immagini private di un. Account AWS
- Amazon Elastic Container Registry Public (Amazon ECR Public): archivia immagini leggibili pubblicamente.

Casi d'uso del provider

- Utilizzo di un'immagine memorizzata in Amazon ECR nel tuo account AWS
- Utilizzo di un'immagine archiviata in Amazon ECR in un altro account AWS

- Utilizzo di un'immagine archiviata in Amazon ECR Public

Utilizzo di un'immagine memorizzata in Amazon ECR nel tuo account AWS

Amazon ECR archivia le immagini in repository. Esistono repository privati e pubblici. Per distribuire l'immagine su un servizio App Runner da un repository privato, App Runner necessita dell'autorizzazione per leggere l'immagine da Amazon ECR. Per concedere tale autorizzazione ad App Runner, devi fornire ad App Runner un ruolo di accesso. Si tratta di un ruolo AWS Identity and Access Management (IAM) che dispone delle autorizzazioni di azione Amazon ECR necessarie. Quando utilizzi la console App Runner per creare il servizio, puoi scegliere un ruolo esistente nel tuo account. In alternativa, puoi utilizzare la console IAM per creare un nuovo ruolo personalizzato. In alternativa, puoi scegliere che la console App Runner crei un ruolo per te in base a politiche gestite.

Quando utilizzi l'API App Runner o la AWS CLI, completi un processo in due fasi. Innanzitutto, utilizzi la console IAM per creare un ruolo di accesso. Puoi utilizzare una policy gestita fornita da App Runner o inserire le tue autorizzazioni personalizzate. Quindi, fornisci il ruolo di accesso durante la creazione del servizio utilizzando l'azione CreateServiceAPI.

Per informazioni sulla creazione del servizio App Runner, consultathe section called “Creazione”.

Utilizzo di un'immagine archiviata in Amazon ECR in un altro account AWS

Quando crei un servizio App Runner, puoi utilizzare un'immagine archiviata in un repository Amazon ECR che appartiene a un AWS account diverso da quello in cui si trova il servizio. Ci sono alcune considerazioni aggiuntive da tenere a mente quando si utilizza un'immagine per più account, oltre a quelle elencate nella sezione precedente sull'immagine dello stesso account.

- Al repository tra account diversi dovrebbe essere associata una policy. La policy del repository fornisce al ruolo di accesso le autorizzazioni per leggere le immagini nel repository. Utilizzate la seguente politica per questo scopo. Sostituiscilo *`access-role-arn`* con l'Amazon Resource Name (ARN) del tuo ruolo di accesso.

  JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Principal": {
        "AWS": "access-role-arn"
      },
      "Action": [
        "ecr:BatchGetImage",
        "ecr:DescribeImages",
        "ecr:GetDownloadUrlForLayer"
      ]
    }
  ]
}
```

Per informazioni su come allegare una policy di repository a un repository Amazon ECR, consulta Setting a repository policy nella Amazon Elastic Container Registry User Guide.

- App Runner non supporta la distribuzione automatica di immagini Amazon ECR in un account diverso da quello in cui si trova il servizio.

Utilizzo di un'immagine archiviata in Amazon ECR Public

Amazon ECR Public archivia immagini leggibili pubblicamente. Queste sono le principali differenze tra Amazon ECR e Amazon ECR Public di cui dovresti essere a conoscenza nel contesto dei servizi App Runner:

- Le immagini pubbliche di Amazon ECR sono leggibili pubblicamente. Non è necessario fornire un ruolo di accesso quando si crea un servizio basato su un'immagine pubblica di Amazon ECR. Il repository non necessita di alcuna policy allegata.
- App Runner non supporta la distribuzione automatica (continua) per le immagini pubbliche di Amazon ECR.

Avvia un servizio direttamente da Amazon ECR Public

Puoi avviare direttamente immagini di container di applicazioni Web compatibili ospitate su Amazon ECR Public Gallery come servizi Web in esecuzione su App Runner. Quando sfogli la galleria, cerca Launch with App Runner nella pagina della galleria per un'immagine. Un'immagine con questa opzione è compatibile con App Runner. Per ulteriori informazioni sulla galleria, consulta Using the Amazon ECR Public Gallery nella guida per l'utente di Amazon ECR Public.

Per avviare un'immagine della galleria come servizio App Runner

1. Nella pagina della galleria di un'immagine, scegli Avvia con App Runner.

   Risultato: la console App Runner si apre in una nuova scheda del browser. La console visualizza la procedura guidata di creazione del servizio, con la maggior parte dei dettagli richiesti del nuovo servizio precompilati.
2. Se desideri creare il servizio in una AWS regione diversa da quella mostrata dalla console, scegli la regione visualizzata nell'intestazione della console. Quindi, seleziona un'altra regione.
3. In Porta, immettete il numero di porta su cui l'applicazione di immagine è in ascolto. In genere puoi trovarlo nella pagina della galleria dell'immagine.
4. Facoltativamente, puoi modificare qualsiasi altro dettaglio di configurazione.
5. Scegli Avanti, rivedi le impostazioni, quindi scegli Crea e distribuisci.

Esempio di immagine

Il team di App Runner mantiene l'immagine di hello-app-runneresempio in una galleria pubblica di Amazon ECR. Puoi usare questo esempio per iniziare a creare un servizio App Runner basato su immagini. Per ulteriori informazioni, consulta hello-app-runner.

Servizio App Runner basato su codice sorgente

Puoi utilizzarli AWS App Runner per creare e gestire servizi basati su due tipi di fonti di servizio fondamentalmente diversi: codice sorgente e immagine sorgente. Indipendentemente dal tipo di sorgente, App Runner si occupa dell'avvio, dell'esecuzione, della scalabilità e del bilanciamento del carico del servizio. Puoi utilizzare la funzionalità CI/CD di App Runner per tenere traccia delle modifiche all'immagine o al codice sorgente. Quando App Runner rileva una modifica, crea automaticamente (per il codice sorgente) e distribuisce la nuova versione nel servizio App Runner.

Questo capitolo descrive i servizi basati sul codice sorgente. Per informazioni sui servizi basati su un'immagine sorgente, vedereServizio basato su immagini.

Il codice sorgente è il codice applicativo che App Runner crea e distribuisce per te. Indirizzi App Runner a una directory sorgente in un repository di codice e scegli un runtime adatto che corrisponda a una versione della piattaforma di programmazione. App Runner crea un'immagine basata sull'immagine di base del runtime e del codice dell'applicazione. Quindi avvia un servizio che esegue un contenitore basato su questa immagine.

App Runner offre comodi runtime gestiti specifici della piattaforma. Ciascuno di questi runtime crea un'immagine del contenitore a partire dal codice sorgente e aggiunge dipendenze di runtime del linguaggio all'immagine. Non è necessario fornire istruzioni di configurazione del contenitore e di compilazione come un Dockerfile.

I sottoargomenti di questo capitolo illustrano le varie piattaforme supportate da App Runner, piattaforme gestite che forniscono runtime gestiti per diversi ambienti e versioni di programmazione.

Argomenti

- Utilizzo della piattaforma PHP di
- Utilizzo della piattaforma Ruby di
- Utilizzo della piattaforma Go di

Fornitori di repository di codice sorgente

App Runner distribuisce il codice sorgente leggendolo da un repository di codice sorgente. App Runner supporta due provider di repository di codice sorgente: e Bitbucket. GitHub

Distribuzione dal tuo provider di repository di codice sorgente

Per distribuire il codice sorgente su un servizio App Runner da un repository di codice sorgente, App Runner stabilisce una connessione ad esso. Quando si utilizza la console App Runner per creare un servizio, si forniscono i dettagli di connessione e una directory sorgente per App Runner per distribuire il codice sorgente.

Connessioni

I dettagli di connessione vengono forniti come parte della procedura di creazione del servizio. Quando si utilizza l'API App Runner o AWS CLI, una connessione è una risorsa separata. Innanzitutto, crei la connessione utilizzando l'azione CreateConnectionAPI. Quindi, fornisci l'ARN della connessione durante la creazione del servizio utilizzando l'azione CreateServiceAPI.

Directory di origine

Quando si crea un servizio, si fornisce anche una directory dei sorgenti. Per impostazione predefinita, App Runner utilizza la directory principale del repository come directory di origine. La directory dei sorgenti è la posizione nel repository del codice sorgente che memorizza il codice sorgente e i file di configurazione dell'applicazione. I comandi build e start vengono eseguiti anche dalla directory dei sorgenti. Quando utilizzi l'API App Runner o AWS CLI per creare o aggiornare un servizio, fornisci la directory di origine nelle azioni CreateServicee UpdateServiceAPI. Per ulteriori informazioni, consultare la sezione seguente Directory dei sorgenti.

Per ulteriori informazioni sulla creazione del servizio App Runner, consulta. the section called “Creazione” Per ulteriori informazioni sulle connessioni App Runner, vedere. the section called “Connessioni”

Directory dei sorgenti

Quando crei un servizio App Runner puoi fornire la directory dei sorgenti, insieme al repository e al ramo. Imposta il valore del campo Directory di origine sul percorso della directory del repository che memorizza il codice sorgente e i file di configurazione dell'applicazione. App Runner esegue i comandi di build e start dal percorso della directory di origine fornito dall'utente.

Immettete il valore assoluto per il percorso della directory di origine dalla directory principale del repository. Se non specificate un valore, il valore predefinito è la directory di primo livello del repository, nota anche come directory principale del repository.

È inoltre possibile fornire diversi percorsi di directory di origine oltre alla directory del repository di primo livello. Ciò supporta un'architettura di repository monorepo, il che significa che il codice sorgente per più applicazioni è archiviato in un unico repository. Per creare e supportare più servizi App Runner da un singolo monorepo, specifica diverse directory di origine quando crei ciascun servizio.

> **Note**
>
> Se si specifica la stessa directory di origine per più servizi App Runner, entrambi i servizi verranno distribuiti e funzioneranno singolarmente.

Se scegli di utilizzare un file di `apprunner.yaml` configurazione per definire i parametri del servizio, inseriscilo nella cartella della directory di origine del repository.

Se l'opzione Deployment trigger è impostata su Automatico, le modifiche effettuate nella directory di origine attiveranno una distribuzione automatica. Solo le modifiche nel percorso della directory di origine attiveranno una distribuzione automatica. È importante capire in che modo la posizione della directory di origine influisce sull'ambito di una distribuzione automatica. Per ulteriori informazioni, consulta le distribuzioni automatizzate in. Metodi di distribuzione

> **Note**
>
> Se il servizio App Runner utilizza i runtime gestiti da PHP e desideri designare una directory di origine diversa dall'archivio principale predefinito, è importante utilizzare la versione di runtime PHP corretta. Per ulteriori informazioni, consulta Utilizzo della piattaforma PHP di .

Piattaforme gestite da App Runner

Le piattaforme gestite di App Runner forniscono runtime gestiti per vari ambienti di programmazione. Ogni runtime gestito semplifica la creazione e l'esecuzione di contenitori basati su una versione di un linguaggio di programmazione o di un ambiente di runtime. Quando si utilizza un runtime gestito, App Runner inizia con un'immagine di runtime gestita. Questa immagine è basata sull'immagine Docker di Amazon Linux e contiene un pacchetto Language Runtime oltre ad alcuni strumenti e pacchetti di dipendenze popolari. App Runner utilizza questa immagine di runtime gestita come immagine di base e aggiunge il codice dell'applicazione per creare un'immagine Docker. Quindi distribuisce questa immagine per eseguire il servizio Web in un contenitore.

Si specifica un runtime per il servizio App Runner quando si crea un servizio utilizzando la console App Runner o l'CreateServiceoperazione API. Puoi anche specificare un runtime come parte del codice sorgente. Usa la `runtime` parola chiave in un file di configurazione di App Runner che includi nel tuo repository di codice. La convenzione di denominazione di un runtime gestito è. *`<language-name><major-version>`*

App Runner aggiorna il runtime del servizio alla versione più recente a ogni distribuzione o aggiornamento del servizio. Se l'applicazione richiede una versione specifica di un runtime gestito, è possibile specificarla utilizzando la `runtime-version` parola chiave nel file di configurazione di App Runner. È possibile utilizzare qualsiasi livello di versione, inclusa una versione principale o secondaria. App Runner effettua solo aggiornamenti di livello inferiore al runtime del servizio.

Versioni di runtime gestite e build di App Runner

App Runner offre un processo di compilazione aggiornato per le applicazioni eseguite sui runtime delle versioni principali più recenti. Questo processo di compilazione rivisto è più veloce ed efficiente. Inoltre, crea un'immagine finale con un ingombro ridotto che contiene solo il codice sorgente, gli artefatti di build e i runtime necessari per eseguire l'applicazione.

Ci riferiamo al processo di compilazione più recente come build di App Runner rivista e al processo di compilazione originale come build originale di App Runner. Per evitare di interrompere le modifiche alle versioni precedenti delle piattaforme di runtime, App Runner applica la build rivista solo a versioni di runtime specifiche, in genere versioni principali appena rilasciate.

Abbiamo introdotto un nuovo componente nel file di `apprunner.yaml` configurazione per rendere la build rivista retrocompatibile per un caso d'uso molto specifico e per offrire anche una maggiore

flessibilità nella configurazione della build dell'applicazione. Questo è il parametro opzionale `pre-run`. Nelle sezioni seguenti spieghiamo quando utilizzare questo parametro insieme ad altre informazioni utili sulle build.

La tabella seguente indica quale versione della build di App Runner si applica a specifiche versioni di runtime gestito. Continueremo ad aggiornare questo documento per tenerti informato sui nostri runtime attuali.

Platform (Piattaforma)	Build originale	Costruzione rivista
Python – Informazioni sul rilascio	• Python 3.8 • Python 3.7	• Python 3.11 (!)
Node.js: Informazioni sulla versione	• Node.js 16 • Node.js 14 • Node.js 12	• Node.js 22 • Node.js 18
Corretto — Informazioni sul rilascio	• Corretto 11 • Corretto 8	
.NET: Informazioni sul rilascio	• .NET 6	
PHP: Informazioni sul rilascio	• PHP 8.1	
Ruby: Informazioni sul rilascio	• Ruby 3.1	
Go: Informazioni sulla versione	• Go 1	

Important

Python 3.11 — Abbiamo raccomandazioni specifiche per la configurazione di build dei servizi che utilizzano il runtime gestito di Python 3.11. Per ulteriori informazioni, consultate l'Callout per versioni di runtime specificheargomento relativo alla piattaforma Python.

Ulteriori informazioni sulle build e sulla migrazione di App Runner

Quando esegui la migrazione dell'applicazione a un runtime più recente che utilizza la build rivista, potrebbe essere necessario modificare leggermente la configurazione della build.

Per fornire un contesto alle considerazioni sulla migrazione, descriveremo innanzitutto i processi di alto livello sia per la build originale di App Runner che per la build rivista. Seguirà una sezione che descrive gli attributi specifici del servizio che potrebbero richiedere alcuni aggiornamenti della configurazione.

La build originale di App Runner

Il processo di creazione dell'applicazione App Runner originale sfrutta il servizio. AWS CodeBuild I passaggi iniziali si basano su immagini curate dal servizio. CodeBuild Segue un processo di compilazione di Docker che utilizza l'immagine di runtime gestita di App Runner applicabile come immagine di base.

I passaggi generali sono i seguenti:

1. Esegui `pre-build` i comandi in un' CodeBuildimmagine curata.

   I `pre-build` comandi sono opzionali. Possono essere specificati solo nel file `apprunner.yaml` di configurazione.
2. Esegui i `build` CodeBuild comandi utilizzando la stessa immagine del passaggio precedente.

   I `build` comandi sono obbligatori. Possono essere specificati nella console App Runner, nell'API App Runner o nel `apprunner.yaml` file di configurazione.
3. Esegui una build Docker per generare un'immagine basata sull'immagine di runtime gestita da App Runner per la tua piattaforma e versione di runtime specifiche.
4. Copia la `/app` directory dall'immagine che abbiamo generato nel passaggio 2. La destinazione è l'immagine basata sull'immagine di runtime gestita da App Runner, che abbiamo generato nel passaggio 3.
5. Esegui nuovamente i `build` comandi sull'immagine di runtime gestita da App Runner generata. Eseguiamo nuovamente i comandi build per generare artefatti di compilazione dal codice sorgente nella **`/app`** directory che abbiamo copiato nella Fase 4. Questa immagine verrà successivamente distribuita da App Runner per eseguire il servizio web in un contenitore.

   I `build` comandi sono obbligatori. Possono essere specificati nella console App Runner, nell'API App Runner o nel `apprunner.yaml` file di configurazione.

6. Esegui `post-build` i comandi nell' CodeBuild immagine del passaggio 2.

   I `post-build` comandi sono opzionali. Possono essere specificati solo nel file `apprunner.yaml` di configurazione.

Una volta completata la build, App Runner distribuisce l'immagine di runtime gestita da App Runner generata dalla Fase 5 per eseguire il servizio Web in un contenitore.

La build di App Runner rivista

Il processo di compilazione rivisto è più veloce ed efficiente rispetto al processo di compilazione originale descritto nella sezione precedente. Elimina la duplicazione dei comandi di compilazione che si verifica nella build della versione precedente. Crea inoltre un'immagine finale con un ingombro ridotto che contiene solo il codice sorgente, gli artefatti di build e i runtime necessari per eseguire l'applicazione.

Questo processo di compilazione utilizza una build Docker in più fasi. Le fasi generali del processo sono le seguenti:

1. Fase di compilazione: avvia un processo di compilazione docker che esegue `pre-build` e `build` comandi sulle immagini di build di App Runner.
   a. Copia il codice sorgente dell'applicazione nella directory. `/app`

      > **Note**
      >
      > Questa `/app` directory è designata come directory di lavoro in ogni fase della build di Docker.

   b. Tramite i comandi `pre-build`.

      I `pre-build` comandi sono opzionali. Possono essere specificati solo nel file `apprunner.yaml` di configurazione.

   c. Esegui i `build` comandi.

      I `build` comandi sono obbligatori. Possono essere specificati nella console App Runner, nell'API App Runner o nel `apprunner.yaml` file di configurazione.
2. Fase di confezionamento: genera l'immagine del contenitore del cliente finale, anch'essa basata sull'immagine di esecuzione di App Runner.

a. Copia la `/app` directory dalla fase di compilazione precedente alla nuova immagine Run. Ciò include il codice sorgente dell'applicazione e gli elementi di compilazione della fase precedente.

b. Esegui i comandi. `pre-run` Se è necessario modificare l'immagine di runtime all'esterno della `/app` directory utilizzando i `build` comandi, aggiungete gli stessi comandi o quelli necessari a questo segmento del file di `apprunner.yaml` configurazione.

   Questo è un nuovo parametro che è stato introdotto per supportare la versione rivista di App Runner.

   I `pre-run` comandi sono opzionali. Possono essere specificati solo nel file `apprunner.yaml` di configurazione.

   > **Note**
   >
   > - I `pre-run` comandi sono supportati solo dalla build rivista. Non aggiungeteli al file di configurazione se il servizio utilizza versioni di runtime che utilizzano la build originale.
   > - Se non è necessario modificare nulla all'esterno della `/app` directory con i `build` comandi, non è necessario specificare `pre-run` i comandi.

3. Fase successiva alla compilazione: questa fase riprende dalla fase di compilazione ed esegue i comandi. `post-build`

   a. Esegui i `post-build` comandi all'interno della directory. `/app`

      I `post-build` comandi sono opzionali. Possono essere specificati solo nel file `apprunner.yaml` di configurazione.

Al termine della build, App Runner distribuisce quindi l'immagine Run per eseguire il servizio Web in un contenitore.

> **Note**
>
> Non fatevi ingannare dalle `env` voci nella sezione Esegui di `apprunner.yaml` quando configuri il processo di compilazione. Anche se il parametro `pre-run` command, a cui si fa riferimento nel passaggio 2 (b), si trova nella sezione Esegui, non utilizzare il `env` parametro nella sezione Run per configurare la build. I `pre-run` comandi fanno riferimento solo alle

`env` variabili definite nella sezione Build del file di configurazione. Per ulteriori informazioni, consultate il Sezione Esegui capitolo sul file di configurazione di App Runner.

Considerazione dei requisiti di servizio per la migrazione

Se il tuo ambiente applicativo presenta uno di questi due requisiti, dovrai rivedere la configurazione della build aggiungendo `pre-run` comandi.

- Se è necessario modificare qualcosa al di fuori della `/app` directory con i `build` comandi.
- Se è necessario eseguire i `build` comandi due volte per creare l'ambiente richiesto. Si tratta di un requisito molto insolito. La stragrande maggioranza delle build non lo farà.

Modifiche al di fuori della directory **`/app`**

- La build rivista di App Runner presuppone che l'applicazione non abbia dipendenze al di fuori della directory. `/app`
- I comandi forniti con il `apprunner.yaml` file, l'API App Runner o la console App Runner devono generare artefatti di build nella directory. `/app`
- È possibile modificare i `post-build` comandi `pre-buildbuild`, e per garantire che tutti gli artefatti di build siano presenti nella directory. `/app`
- Se l'applicazione richiede la build per modificare ulteriormente l'immagine generata per il servizio, al di fuori della `/app` directory, è possibile utilizzare i nuovi `pre-run` comandi in. `apprunner.yaml` Per ulteriori informazioni, consulta Impostazione delle opzioni del servizio App Runner utilizzando un file di configurazione.

Esecuzione dei **`build`** comandi due volte

- La build originale di App Runner esegue i `build` comandi due volte, prima nel passaggio 2, poi di nuovo nel passaggio 5. La build rivista di App Runner pone rimedio a questa ridondanza ed esegue i comandi solo una volta. `build` Se l'applicazione dovesse richiedere un'esecuzione insolita dei `build` comandi due volte, la build aggiornata di App Runner offre la possibilità di specificare ed eseguire nuovamente gli stessi comandi utilizzando il parametro. `pre-run` In questo modo si mantiene lo stesso comportamento di doppia build.

Utilizzo della piattaforma Python di

La piattaforma AWS App Runner Python fornisce runtime gestiti. Ogni runtime semplifica la creazione e l'esecuzione di contenitori con applicazioni Web basate su una versione Python. Quando si utilizza un runtime Python, App Runner inizia con un'immagine di runtime Python gestita. Questa immagine è basata sull'immagine Docker di Amazon Linux e contiene il pacchetto runtime per una versione di Python e alcuni strumenti e pacchetti di dipendenze popolari. App Runner utilizza questa immagine di runtime gestita come immagine di base e aggiunge il codice dell'applicazione per creare un'immagine Docker. Quindi distribuisce questa immagine per eseguire il servizio Web in un contenitore.

Si specifica un runtime per il servizio App Runner quando si crea un servizio utilizzando la console App Runner o l'CreateServiceoperazione API. Puoi anche specificare un runtime come parte del codice sorgente. Usa la `runtime` parola chiave in un file di configurazione di App Runner che includi nel tuo repository di codice. La convenzione di denominazione di un runtime gestito è. *`<language-name><major-version>`*

Per i nomi e le versioni di runtime Python validi, vedere. the section called “Informazioni sul rilascio”

App Runner aggiorna il runtime del servizio alla versione più recente a ogni distribuzione o aggiornamento del servizio. Se l'applicazione richiede una versione specifica di un runtime gestito, è possibile specificarla utilizzando la `runtime-version` parola chiave nel file di configurazione di App Runner. È possibile utilizzare qualsiasi livello di versione, inclusa una versione principale o secondaria. App Runner effettua solo aggiornamenti di livello inferiore al runtime del servizio.

Sintassi della versione per i runtime di Python: *`major`*`[.`*`minor`*`[.`*`patch`*`]]`

Ad esempio: `3.8.5`

Gli esempi seguenti mostrano il blocco della versione:

- `3.8`— Blocca le versioni principali e secondarie. App Runner aggiorna solo le versioni patch.
- `3.8.5`— Blocca a una versione di patch specifica. App Runner non aggiorna la tua versione di runtime.

Argomenti

- Configurazione del runtime di Python
- Callout per versioni di runtime specifiche
- Esempi di runtime in Python

- Informazioni sul rilascio del runtime di Python

Configurazione del runtime di Python

Quando scegli un runtime gestito, devi anche configurare, come minimo, creare ed eseguire comandi. Li configuri durante la creazione o l'aggiornamento del servizio App Runner. Puoi farlo utilizzando uno dei seguenti metodi:

- Utilizzo della console App Runner: specifica i comandi nella sezione Configure build del processo di creazione o della scheda di configurazione.
- Utilizzo dell'API App Runner: richiama l'operazione CreateServiceor UpdateServiceAPI. Specificate i comandi utilizzando i `StartCommand` membri `BuildCommand` e del tipo di CodeConfigurationValuesdati.
- Utilizzo di un file di configurazione: specifica uno o più comandi di compilazione in un massimo di tre fasi di compilazione e un singolo comando di esecuzione che serve per avviare l'applicazione. Sono disponibili impostazioni di configurazione opzionali aggiuntive.

Fornire un file di configurazione è facoltativo. Quando crei un servizio App Runner utilizzando la console o l'API, specifichi se App Runner ottiene le impostazioni di configurazione direttamente al momento della creazione o da un file di configurazione.

Callout per versioni di runtime specifiche

> **Note**
>
> App Runner ora esegue un processo di compilazione aggiornato per le applicazioni basate sulle seguenti versioni di runtime: Python 3.11, Node.js 22 e Node.js 18. Se l'applicazione viene eseguita su una di queste versioni di runtime, consulta Versioni di runtime gestite e build di App Runner per ulteriori informazioni sul processo di compilazione rivisto. Le applicazioni che utilizzano tutte le altre versioni di runtime non sono interessate e continuano a utilizzare il processo di compilazione originale.

Python 3.11 (build App Runner rivista)

Usa le seguenti impostazioni in apprunner.yaml per il runtime Python 3.11 gestito.

- Imposta la chiave nella sezione Top su `runtime` `python311`

  Example

  ```
  runtime: python311
  ```

- Usa `pip3` invece di `pip` per installare le dipendenze.
- Usa l'`python3`interprete invece di. `python`
- Esegui il `pip3` programma di installazione come comando. `pre-run` Python installa le dipendenze all'esterno della directory. `/app` Poiché App Runner esegue la build aggiornata di App Runner per Python 3.11, tutto ciò che è installato all'esterno della `/app` directory tramite i comandi nella sezione Build del file andrà perso. `apprunner.yaml` Per ulteriori informazioni, consulta La build di App Runner rivista.

  Example

  ```
  run:
    runtime-version: 3.11
    pre-run:
      - pip3 install pipenv
      - pipenv install
      - python3 copy-global-files.py
    command: pipenv run gunicorn django_apprunner.wsgi --log-file -
  ```

Per ulteriori informazioni, vedi anche l'esempio di un file di configurazione esteso per Python 3.11 più avanti in questo argomento.

Esempi di runtime in Python

Gli esempi seguenti mostrano i file di configurazione di App Runner per la creazione e l'esecuzione di un servizio Python. L'ultimo esempio è il codice sorgente per un'applicazione Python completa che puoi distribuire su un servizio di runtime Python.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è and. `3.7.7` `3.11` Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime di Python supportata, vedere. the section called “Informazioni sul rilascio”

File di configurazione Python minimo

Questo esempio mostra un file di configurazione minimo che puoi usare con un runtime gestito da Python. Per le ipotesi che App Runner fa con un file di configurazione minimo, vedi. the section called “Esempi di file di configurazione”

Python 3.11 utilizza i comandi and. `pip3` `python3` Per ulteriori informazioni, vedete l'esempio di un file di configurazione esteso per Python 3.11 più avanti in questo argomento.

Example apprunner.yaml

```
version: 1.0
runtime: python3
build:
  commands:
    build:
      - pip install pipenv
      - pipenv install
run:
  command: python app.py
```

File di configurazione Python esteso

Questo esempio mostra l'uso di tutte le chiavi di configurazione con un runtime gestito da Python.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è*3.7.7*. Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime di Python supportata, vedere. the section called “Informazioni sul rilascio”
> Python 3.11 utilizza i comandi and. `pip3` `python3` Per ulteriori informazioni, vedete l'esempio di un file di configurazione esteso per Python 3.11 più avanti in questo argomento.

Example apprunner.yaml

```
version: 1.0
runtime: python3
build:
  commands:
```

```
    pre-build:
      - wget -c https://s3.amazonaws.com/amzn-s3-demo-bucket/test-lib.tar.gz -O - | tar
 -xz
    build:
      - pip install pipenv
      - pipenv install
    post-build:
      - python manage.py test
  env:
    - name: DJANGO_SETTINGS_MODULE
      value: "django_apprunner.settings"
    - name: MY_VAR_EXAMPLE
      value: "example"
run:
  runtime-version: 3.7.7
  command: pipenv run gunicorn django_apprunner.wsgi --log-file -
  network:
    port: 8000
    env: MY_APP_PORT
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
  secrets:
    - name: my-secret
      value-from: "arn:aws:secretsmanager:us-
east-1:123456789012:secret:testingstackAppRunnerConstr-kJFXde2ULKbT-S7t8xR:username::"
    - name: my-parameter
      value-from: "arn:aws:ssm:us-east-1:123456789012:parameter/parameter-name"
    - name: my-parameter-only-name
      value-from: "parameter-name"
```

File di configurazione Python esteso — Python 3.11 (utilizza la build rivista)

Questo esempio mostra l'uso di tutte le chiavi di configurazione con un runtime gestito da Python 3.11 in. `apprunner.yaml` Questo esempio include una `pre-run` sezione, poiché questa versione di Python utilizza la build di App Runner rivista.

Il `pre-run` parametro è supportato solo dalla build aggiornata di App Runner. Non inserire questo parametro nel file di configurazione se l'applicazione utilizza versioni di runtime supportate dalla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è*3.11*. Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime di Python supportata, vedere. the section called “Informazioni sul rilascio”

Example apprunner.yaml

```
version: 1.0
runtime: python311
build:
  commands:
    pre-build:
      - wget -c https://s3.amazonaws.com/amzn-s3-demo-bucket/test-lib.tar.gz -O - | tar
 -xz
    build:
      - pip3 install pipenv
      - pipenv install
    post-build:
      - python3 manage.py test
  env:
    - name: DJANGO_SETTINGS_MODULE
      value: "django_apprunner.settings"
    - name: MY_VAR_EXAMPLE
      value: "example"
run:
  runtime-version: 3.11
  pre-run:
    - pip3 install pipenv
    - pipenv install
    - python3 copy-global-files.py
  command: pipenv run gunicorn django_apprunner.wsgi --log-file -
  network:
    port: 8000
    env: MY_APP_PORT
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
  secrets:
    - name: my-secret
```

```
      value-from: "arn:aws:secretsmanager:us-
east-1:123456789012:secret:testingstackAppRunnerConstr-kJFXde2ULKbT-S7t8xR:username::"
    - name: my-parameter
      value-from: "arn:aws:ssm:us-east-1:123456789012:parameter/parameter-name"
    - name: my-parameter-only-name
      value-from: "parameter-name"
```

Sorgente completo dell'applicazione Python

Questo esempio mostra il codice sorgente per un'applicazione Python completa che puoi distribuire su un servizio di runtime Python.

Example requirements.txt

```
pyramid==2.0
```

Example server.py

```
from wsgiref.simple_server import make_server
from pyramid.config import Configurator
from pyramid.response import Response
import os

def hello_world(request):
    name = os.environ.get('NAME')
    if name == None or len(name) == 0:
        name = "world"
    message = "Hello, " + name + "!\n"
    return Response(message)

if __name__ == '__main__':
    port = int(os.environ.get("PORT"))
    with Configurator() as config:
        config.add_route('hello', '/')
        config.add_view(hello_world, route_name='hello')
        app = config.make_wsgi_app()
    server = make_server('0.0.0.0', port, app)
    server.serve_forever()
```

Example apprunner.yaml

```
version: 1.0
```

```
runtime: python3
build:
  commands:
    build:
      - pip install -r requirements.txt
run:
  command: python server.py
```

Informazioni sul rilascio del runtime di Python

Questo argomento elenca i dettagli completi per le versioni di runtime di Python supportate da App Runner.

Versioni di runtime supportate: build di App Runner rivista

Nome runtime	Versioni minori	Pacchetti inclusi
Python 3.11 (python311)	3.11.13	SQLite 3,50,1
	3,11,12	SQLite 3,50,0
	3,11,11	SQLite 3,49,1
	3,11,10	SQLite 3,46,1
	3,11,9	SQLite 3,46,1
	3,11,8	SQLite 3,445,2
	3,11,7	SQLite 3,44,2

Note

- Python 3.11 — Abbiamo raccomandazioni specifiche per la configurazione di build dei servizi che utilizzano il runtime gestito di Python 3.11. Per ulteriori informazioni, consultate l'Callout per versioni di runtime specificheargomento relativo alla piattaforma Python.
- App Runner fornisce un processo di compilazione rivisto per i principali runtime specifici che sono stati rilasciati più di recente. Per questo motivo, in alcune sezioni di questo documento vedrai riferimenti alla build di App Runner rivista e alla build originale di App

Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

Versioni di runtime supportate: build originale di App Runner

Nome runtime	Versioni minori	Pacchetti inclusi
Python 3 (python3)	3.8.20	SQLite 3,50,1
	3,8,20	SQLite 3,50,0
	3,8,16	SQLite 3,46,1
	3,8,15	SQLite 3,4,0
	3,7,16	SQLite 3,50,0
	3,7,15	SQLite 3,4,0
	3,7,10	SQLite 3,4,0

Note

App Runner fornisce un processo di compilazione rivisto per i principali runtime specifici che sono stati rilasciati più di recente. Per questo motivo, in alcune sezioni di questo documento vedrai riferimenti alla build di App Runner rivista e alla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

Utilizzo della piattaforma Node.js di

La piattaforma AWS App Runner Node.js fornisce runtime gestiti. Ogni runtime semplifica la creazione e l'esecuzione di contenitori con applicazioni Web basate su una versione Node.js. Quando si utilizza un runtime Node.js, App Runner si avvia con un'immagine di runtime Node.js gestita. Questa immagine è basata sull'immagine Docker di Amazon Linux e contiene il pacchetto di runtime per una versione di Node.js e alcuni strumenti. App Runner utilizza questa immagine di runtime

gestita come immagine di base e aggiunge il codice dell'applicazione per creare un'immagine Docker. Quindi distribuisce questa immagine per eseguire il servizio Web in un contenitore.

Si specifica un runtime per il servizio App Runner quando si crea un servizio utilizzando la console App Runner o l'CreateServiceoperazione API. Puoi anche specificare un runtime come parte del codice sorgente. Usa la `runtime` parola chiave in un file di configurazione di App Runner che includi nel tuo repository di codice. La convenzione di denominazione di un runtime gestito è. *`<language-name><major-version>`*

Per i nomi e le versioni di runtime di Node.js validi, vederethe section called “Informazioni sulla versione”.

App Runner aggiorna il runtime del servizio alla versione più recente a ogni distribuzione o aggiornamento del servizio. Se l'applicazione richiede una versione specifica di un runtime gestito, è possibile specificarla utilizzando la `runtime-version` parola chiave nel file di configurazione di App Runner. È possibile utilizzare qualsiasi livello di versione, inclusa una versione principale o secondaria. App Runner effettua solo aggiornamenti di livello inferiore al runtime del servizio.

Sintassi della versione per i runtime di Node.js: *`major`*`[.`*`minor`*`[.`*`patch`*`]]`

Ad esempio: `22.14.0`

Gli esempi seguenti mostrano il blocco della versione:

- `22.14`— Blocca le versioni principali e secondarie. App Runner aggiorna solo le versioni patch.
- `22.14.0`— Blocca a una versione di patch specifica. App Runner non aggiorna la versione di runtime.

Argomenti

- Configurazione del runtime di Node.js
- Callout per versioni di runtime specifiche
- Esempi di runtime di Node.js
- Informazioni sulla versione di runtime di Node.js

Configurazione del runtime di Node.js

Quando si sceglie un runtime gestito, è inoltre necessario configurare, come minimo, creare ed eseguire comandi. Li configuri durante la creazione o l'aggiornamento del servizio App Runner. Puoi farlo utilizzando uno dei seguenti metodi:

- Utilizzo della console App Runner: specifica i comandi nella sezione Configure build del processo di creazione o della scheda di configurazione.
- Utilizzo dell'API App Runner: richiama l'operazione CreateServiceor UpdateServiceAPI. Specificate i comandi utilizzando i `StartCommand` membri `BuildCommand` e del tipo di CodeConfigurationValuesdati.
- Utilizzo di un file di configurazione: specifica uno o più comandi di compilazione in un massimo di tre fasi di compilazione e un singolo comando di esecuzione che serve per avviare l'applicazione. Sono disponibili impostazioni di configurazione opzionali aggiuntive.

Fornire un file di configurazione è facoltativo. Quando crei un servizio App Runner utilizzando la console o l'API, specifichi se App Runner ottiene le impostazioni di configurazione direttamente al momento della creazione o da un file di configurazione.

In particolare, con i runtime di Node.js, puoi anche configurare la build e il runtime utilizzando un file JSON denominato `package.json` nella radice del tuo repository di origine. Utilizzando questo file, è possibile configurare la versione del motore Node.js, i pacchetti di dipendenze e vari comandi (applicazioni a riga di comando). I gestori di pacchetti come npm o yarn interpretano questo file come input per i loro comandi.

Per esempio:

- npm installinstalla i pacchetti definiti dal nodo `dependencies` and `devDependencies` in. `package.json`
- npm starto npm run start esegue il comando definito dal `scripts/start` nodo in`package.json`.

Di seguito è riportato un esempio del file `package.json`.

package.json

```
{
  "name": "node-js-getting-started",
  "version": "0.3.0",
```

```
  "description": "A sample Node.js app using Express 4",
  "engines": {
    "node": "22.14.0"
  },
  "scripts": {
    "start": "node index.js",
    "test": "node test.js"
  },
  "dependencies": {
    "cool-ascii-faces": "^1.3.4",
    "ejs": "^2.5.6",
    "express": "^4.15.2"
  },
  "devDependencies": {
    "got": "^11.3.0",
    "tape": "^4.7.0"
  }
}
```

Per ulteriori informazioni su*`package.json`*, vedere Creazione di un file package.json sul sito Web di npm Docs.

Suggerimenti

- Se il `package.json` file definisce un start comando, è possibile utilizzarlo come comando nel file di configurazione di App Runner, come illustrato nell'esempio run seguente.

  Example

  pacchetto.json

  ```
  {
    "scripts": {
      "start": "node index.js"
    }
  }
  ```

  apprunner.yaml

  ```
  run:
    command: npm start
  ```

- Quando esegui npm install nel tuo ambiente di sviluppo, npm crea il file. `package-lock.json` Questo file contiene un'istantanea delle versioni del pacchetto appena installate da npm. Successivamente, quando npm installa le dipendenze, utilizza queste versioni esatte. Se installi yarn, crea un file. `yarn.lock` Salva questi file nel tuo archivio del codice sorgente per assicurarti che l'applicazione sia installata con le versioni delle dipendenze che hai sviluppato e con cui l'hai testata.
- È inoltre possibile utilizzare un file di configurazione di App Runner per configurare la versione di Node.js e il comando start. Quando si esegue questa operazione, queste definizioni sostituiscono quelle incluse. `package.json` Un conflitto tra la `node` versione in `package.json` e il `runtime-version` valore nel file di configurazione di App Runner causa il fallimento della fase di compilazione di App Runner.

Callout per versioni di runtime specifiche

Node.js 22 e Node.js 18 (build App Runner rivista)

App Runner ora esegue un processo di compilazione aggiornato per le applicazioni basate sulle seguenti versioni di runtime: Python 3.11, Node.js 22 e Node.js 18. Se l'applicazione viene eseguita su una di queste versioni di runtime, consulta Versioni di runtime gestite e build di App Runner per ulteriori informazioni sul processo di compilazione rivisto. Le applicazioni che utilizzano tutte le altre versioni di runtime non sono interessate e continuano a utilizzare il processo di compilazione originale.

Esempi di runtime di Node.js

Gli esempi seguenti mostrano i file di configurazione di App Runner per la creazione e l'esecuzione di un servizio Node.js.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è*`22.14.0`*. Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime supportata di Node.js, consultathe section called “Informazioni sulla versione”.

File di configurazione minimo Node.js

Questo esempio mostra un file di configurazione minimo che è possibile utilizzare con un runtime gestito da Node.js. Per le ipotesi che App Runner fa con un file di configurazione minimo, vedi. the section called “Esempi di file di configurazione”

Example apprunner.yaml

```
version: 1.0
runtime: nodejs22
build:
  commands:
    build:
      - npm install --production
run:
  command: node app.js
```

File di configurazione Node.js esteso

Questo esempio mostra l'uso di tutte le chiavi di configurazione con un runtime gestito da Node.js.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è*22.14.0*. Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime supportata di Node.js, consultathe section called “Informazioni sulla versione”.

Example apprunner.yaml

```
version: 1.0
runtime: nodejs22
build:
  commands:
    pre-build:
      - npm install --only=dev
      - node test.js
    build:
      - npm install --production
    post-build:
      - node node_modules/ejs/postinstall.js
```

```
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
run:
  runtime-version: 22.14.0
  command: node app.js
  network:
    port: 8000
    env: APP_PORT
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
```

File di configurazione Node.js esteso — Node.js 22 (utilizza la build rivista)

Questo esempio mostra l'uso di tutte le chiavi di configurazione con un runtime gestito da Node.js in. `apprunner.yaml` Questo esempio include una `pre-run` sezione, poiché questa versione di Node.js utilizza la build aggiornata di App Runner.

Il `pre-run` parametro è supportato solo dalla build aggiornata di App Runner. Non inserire questo parametro nel file di configurazione se l'applicazione utilizza versioni di runtime supportate dalla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è`22.14.0`. Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime supportata di Node.js, consultathe section called “Informazioni sulla versione”.

Example apprunner.yaml

```
version: 1.0
runtime: nodejs22
build:
  commands:
    pre-build:
      - npm install --only=dev
      - node test.js
    build:
```

```
      - npm install --production
    post-build:
      - node node_modules/ejs/postinstall.js
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
run:
  runtime-version: 22.14.0
  pre-run:
    - node copy-global-files.js
  command: node app.js
  network:
    port: 8000
    env: APP_PORT
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
```

App Node.js con Grunt

Questo esempio mostra come configurare un'applicazione Node.js sviluppata con Grunt. Grunt è un JavaScript task runner da riga di comando. Esegue attività ripetitive e gestisce l'automazione dei processi per ridurre l'errore umano. I plugin Grunt e Grunt vengono installati e gestiti utilizzando npm. Si configura Grunt includendo il `Gruntfile.js` file nella radice del repository di origine.

Example package.json

```
{
  "scripts": {
    "build": "grunt uglify",
    "start": "node app.js"
  },
  "devDependencies": {
    "grunt": "~0.4.5",
    "grunt-contrib-jshint": "~0.10.0",
    "grunt-contrib-nodeunit": "~0.4.1",
    "grunt-contrib-uglify": "~0.5.0"
  },
  "dependencies": {
    "express": "^4.15.2"
  },
}
```

Example Gruntfile.js

```
module.exports = function(grunt) {

  // Project configuration.
  grunt.initConfig({
    pkg: grunt.file.readJSON('package.json'),
    uglify: {
      options: {
        banner: '/*! <%= pkg.name %> <%= grunt.template.today("yyyy-mm-dd") %> */\n'
      },
      build: {
        src: 'src/<%= pkg.name %>.js',
        dest: 'build/<%= pkg.name %>.min.js'
      }
    }
  });

  // Load the plugin that provides the "uglify" task.
  grunt.loadNpmTasks('grunt-contrib-uglify');

  // Default task(s).
  grunt.registerTask('default', ['uglify']);

};
```

Example apprunner.yaml

> **Note**
>
> La versione di runtime utilizzata in questi esempi è. *22.14.0* Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime supportata di Node.js, consultathe section called “Informazioni sulla versione”.

```
version: 1.0
runtime: nodejs22
build:
  commands:
    pre-build:
      - npm install grunt grunt-cli
      - npm install --only=dev
```

```
      - npm run build
    build:
      - npm install --production
run:
  runtime-version: 22.14.0
  command: node app.js
  network:
    port: 8000
    env: APP_PORT
```

Informazioni sulla versione di runtime di Node.js

> **Note**
>
> La politica di deprecazione standard di App Runner prevede di rendere obsoleto un runtime quando uno dei componenti principali del runtime raggiunge la fine del supporto a lungo termine della community (LTS) e gli aggiornamenti di sicurezza non sono più disponibili. In alcuni casi, App Runner può ritardare l'obsolescenza di un runtime per un periodo limitato, oltre la data della versione linguistica supportata dal runtime. end-of-support Un esempio di questo caso potrebbe essere l'estensione del supporto per un runtime per consentire ai clienti il tempo necessario per la migrazione.

Questo argomento elenca i dettagli completi per le versioni di runtime di Node.js supportate da App Runner.

Versioni di runtime supportate: build di App Runner rivista

Nome runtime	Versioni minori	Pacchetti inclusi
Node.js 22 (nodejs22)	22.17.0	npm 10.9.2, filato 1.22.22
	22.16.0	npm 10.9.2, filato 1.22.22
	22.14.0	npm 10.9.2, filato 1.22.22

> **Note**
>
> App Runner fornisce un processo di compilazione rivisto per i principali runtime specifici che sono stati rilasciati più di recente. Per questo motivo, in alcune sezioni di questo documento vedrai riferimenti alla build di App Runner rivista e alla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

Versioni di runtime supportate: build di App Runner rivista

Nome runtime	Versioni minori	Pacchetti inclusi
Node.js 18 (nodejs18)	18.20.8	npm 10.8.2, filato 1.22.22
	18.20.7	npm 10.8.2, filato 1.22.22
	18.20.6	npm 10.8.2, filato 1.22.22
	18.20.5	npm 10.8.2, filato 1.22.22
	18.20.4	npm 10.7.0, filato 1.22.22
	18.20.3	npm 10.7.0, filato 1.22.22
	18.20.2	npm 10, filato*
	18.19.1	npm 10, filato*
	18.19,0	npm 10, filato*

Versioni di runtime supportate: build originale di App Runner

Nome runtime	Versioni minori	Pacchetti inclusi
Node.js 16 (nodejs16)	16.20.2	npm 8.19.4, filato 1.22.22
	16.20.1	npm 8.19.4, filato*
	16.20,0	npm 8.19.4, filato *

Nome runtime	Versioni minori	Pacchetti inclusi
	16.19.1	npm 8.19.4, filato*
	16.19,0	npm 8.19.4, filato *
	16.18.1	npm 8.19.4, filato*
	16.17.1	npm 8.19.4, filato*
	16.17,0	npm 8.19.4, filato *
Node.js 14 (node.js14)	14.21.3	npm 6.14.18, filato 1.22.22
	14.21.2	npm 6.14.18, filato *
	14.21.1	npm 6.14.18, filato *
	14.20.1	npm 6.14.18, filato *
	14.19,0	npm 6.14.18, filato *
Node.js 12 (nodejs12)	22.12.12	npm 6.14.16, filato 1.22.22
	12.21.0	npm 6.14.16, filato *

Utilizzo della piattaforma Java

La piattaforma AWS App Runner Java fornisce runtime gestiti. Ogni runtime semplifica la creazione e l'esecuzione di contenitori con applicazioni Web basate su una versione Java. Quando si utilizza un runtime Java, App Runner inizia con un'immagine di runtime Java gestita. Questa immagine è basata sull'immagine Docker di Amazon Linux e contiene il pacchetto di runtime per una versione di Java e alcuni strumenti. App Runner utilizza questa immagine di runtime gestita come immagine di base e aggiunge il codice dell'applicazione per creare un'immagine Docker. Quindi distribuisce questa immagine per eseguire il servizio Web in un contenitore.

Si specifica un runtime per il servizio App Runner quando si crea un servizio utilizzando la console App Runner o l'CreateServiceoperazione API. Puoi anche specificare un runtime come parte del codice sorgente. Usa la `runtime` parola chiave in un file di configurazione di App Runner che includi

nel tuo repository di codice. La convenzione di denominazione di un runtime gestito è. *<language-name><major-version>*

Al momento, tutti i runtime Java supportati sono basati su Amazon Corretto. Per nomi e versioni di runtime Java validi, consultathe section called “Informazioni sul rilascio”.

App Runner aggiorna il runtime del servizio alla versione più recente a ogni distribuzione o aggiornamento del servizio. Se l'applicazione richiede una versione specifica di un runtime gestito, è possibile specificarla utilizzando la `runtime-version` parola chiave nel file di configurazione di App Runner. È possibile utilizzare qualsiasi livello di versione, inclusa una versione principale o secondaria. App Runner effettua solo aggiornamenti di livello inferiore al runtime del servizio.

Sintassi della versione per i runtime di Amazon Corretto:

Runtime	Sintassi	Esempio
corretto11	`11.0[.`*`openjdk-update`* `[.`*`openjdk-build`* `[.`*`corretto-specific-revision`* `]]]`	`11.0.13.08.1`
corretto8	`8[.`*`openjdk-update`* `[.`*`openjdk-build`* `[.`*`corretto-specific-revision`* `]]]`	`8.312.07.1`

Gli esempi seguenti mostrano il blocco della versione:

- `11.0.13`— Blocca la versione di aggiornamento di Open JDK. App Runner aggiorna solo le build di livello inferiore di Open JDK e Amazon Corretto.
- `11.0.13.08.1`— Blocca a una versione specifica. App Runner non aggiorna la tua versione di runtime.

Argomenti

- Configurazione del runtime Java
- Esempi di runtime Java
- Informazioni sulla release di Java Runtime

Configurazione del runtime Java

Quando scegli un runtime gestito, devi anche configurare, come minimo, creare ed eseguire comandi. Li configuri durante la creazione o l'aggiornamento del servizio App Runner. Puoi farlo utilizzando uno dei seguenti metodi:

- Utilizzo della console App Runner: specifica i comandi nella sezione Configure build del processo di creazione o della scheda di configurazione.
- Utilizzo dell'API App Runner: richiama l'operazione CreateServiceor UpdateServiceAPI. Specificate i comandi utilizzando i `StartCommand` membri `BuildCommand` e del tipo di CodeConfigurationValuesdati.
- Utilizzo di un file di configurazione: specifica uno o più comandi di compilazione in un massimo di tre fasi di compilazione e un singolo comando di esecuzione che serve per avviare l'applicazione. Sono disponibili impostazioni di configurazione opzionali aggiuntive.

Fornire un file di configurazione è facoltativo. Quando crei un servizio App Runner utilizzando la console o l'API, specifichi se App Runner ottiene le impostazioni di configurazione direttamente al momento della creazione o da un file di configurazione.

Esempi di runtime Java

Gli esempi seguenti mostrano i file di configurazione di App Runner per la creazione e l'esecuzione di un servizio Java. L'ultimo esempio è il codice sorgente di un'applicazione Java completa che è possibile distribuire su un servizio di runtime Corretto 11.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è*`11.0.13.08.1`*. Puoi sostituirla con una versione che desideri utilizzare. Per la versione di runtime Java più recente supportata, vederethe section called “Informazioni sul rilascio”.

File di configurazione Minimal Corretto 11

Questo esempio mostra un file di configurazione minimo che è possibile utilizzare con un runtime gestito di Corretto 11. Per le ipotesi che App Runner fa con un file di configurazione minimo, vedi.

Example apprunner.yaml

```
version: 1.0
runtime: corretto11
build:
  commands:
    build:
      - mvn clean package
run:
  command: java -Xms256m -jar target/MyApp-1.0-SNAPSHOT.jar .
```

File di configurazione Corretto 11 esteso

Questo esempio mostra come utilizzare tutte le chiavi di configurazione con un runtime gestito di Corretto 11.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è*11.0.13.08.1*. Puoi sostituirla con una versione che desideri utilizzare. Per la versione di runtime Java più recente supportata, vederethe section called “Informazioni sul rilascio”.

Example apprunner.yaml

```
version: 1.0
runtime: corretto11
build:
  commands:
    pre-build:
      - yum install some-package
      - scripts/prebuild.sh
    build:
      - mvn clean package
    post-build:
      - mvn clean test
  env:
    - name: M2
      value: "/usr/local/apache-maven/bin"
    - name: M2_HOME
      value: "/usr/local/apache-maven/bin"
run:
```

```
  runtime-version: 11.0.13.08.1
  command: java -Xms256m -jar target/MyApp-1.0-SNAPSHOT.jar .
  network:
    port: 8000
    env: APP_PORT
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
```

Sorgente completo dell'applicazione Corretto 11

Questo esempio mostra il codice sorgente di un'applicazione Java completa che è possibile distribuire su un servizio di runtime Corretto 11.

Example src/main/java/com/HelloWorld/HelloWorld.java

```
package com.HelloWorld;

import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HelloWorld {

    @RequestMapping("/")
    public String index(){
        String s = "Hello World";
        return s;
    }
}
```

Example src/main/java/com/HelloWorld/Main.java

```
package com.HelloWorld;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class Main {

    public static void main(String[] args) {
```

```
        SpringApplication.run(Main.class, args);
    }
}
```

Example apprunner.yaml

```
version: 1.0
runtime: corretto11
build:
  commands:
    build:
      - mvn clean package
run:
  command: java -Xms256m -jar target/HelloWorldJavaApp-1.0-SNAPSHOT.jar .
  network:
    port: 8080
```

Example pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/
maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.3.1.RELEASE</version>
    <relativePath/>
  </parent>
  <groupId>com.HelloWorld</groupId>
  <artifactId>HelloWorldJavaApp</artifactId>
  <version>1.0-SNAPSHOT</version>

  <properties>
    <java.version>11</java.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-rest</artifactId>
```

```
    </dependency>

    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-test</artifactId>
      <scope>test</scope>
      <exclusions>
        <exclusion>
          <groupId>org.junit.vintage</groupId>
          <artifactId>junit-vintage-engine</artifactId>
        </exclusion>
      </exclusions>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.0</version>
        <configuration>
          <release>11</release>
        </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```

Informazioni sulla release di Java Runtime

Questo argomento elenca i dettagli completi per le versioni di runtime Java supportate da App Runner.

Versioni di runtime supportate: build originale di App Runner

Nome runtime	Versioni minori	Pacchetti inclusi
Corretto 11 (corretto 11)	11.0.27.6.1	Maven 3.9.10, Gradle 6.9.4

Nome runtime	Versioni minori	Pacchetti inclusi
	11.0.27.6.1	Maven 3.9.9, Gradle 6.9.4
	11.0.26.4.1	Maven 3.9.9, Gradle 6.9.4
	11.0.259.1	Maven 3.9.9, Gradle 6.9.4
	11.0.24.8.1	Maven 3.9.9, Gradle 6.9.4
	11.0.23.9.1	Maven 3.9.8, Gradle 6.9.4
	11.0.22.7.1	Maven 3.9.6, Gradle 6.9.4
	11.0.21.9.1	Maven 3.9.6, Gradle 6.9.4
	11.0.21.9.1	Maven 3.9.5, Gradle 6.9.4
	11.0.20.8.1	Maven 3.9.3, Gradle 6.9.4
	11.0.19.7.1	Maven 3.9.3, Gradle 6.9.4
	11.0.18.10.1	Maven 3.9.1, Gradle 6.9.4
	11.0.17.8.1	Maven 3.8.6, Gradle 6.9.3
	11.0.16.9.1	Maven 3.8.6, Gradle 6.9.2
	11.0.13.08.1	Maven 3.6.3, Gradle 6.5
Corretto 8 (corretto 8)	8,452,09.2	Maven 3.9.10, Gradle 6.9.4
	8,452,09,2	Maven 3.9.9, Gradle 6.9.4
	8,452,09,1	Maven 3.9.9, Gradle 6.9.4
	8,442,06.1	Maven 3.9.9, Gradle 6.9.4
	8,432,06.1	Maven 3.9.9, Gradle 6.9.4
	8,422,05,1	Maven 3.9.9, Gradle 6.9.4

Nome runtime	Versioni minori	Pacchetti inclusi
	8,412,08.1	Maven 3.9.8, Gradle 6.9.4
	8,402,08.1	Maven 3.9.6, Gradle 6.9.4
	8,392,08.1	Maven 3.9.6, Gradle 6.9.4
	8,382,05,1	Maven 3.9.4, Gradle 6.9.4
	8,372,07,1	Maven 3.9.3, Gradle 6.9.4
	8.362,08.1	Maven 3.9.1, Gradle 6.9.4
	8,352,08.1	Maven 3.8.6, Gradle 6.9.3
	8,342,07,4	Maven 3.8.6, Gradle 6.9.2
	8.31207.1	Maven 3.6.3, Gradle 6.5

> **Note**
>
> App Runner fornisce un processo di compilazione rivisto per i principali runtime specifici che sono stati rilasciati più di recente. Per questo motivo, in alcune sezioni di questo documento vedrai riferimenti alla build di App Runner rivista e alla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

Utilizzo della piattaforma .NET di

La piattaforma AWS App Runner .NET fornisce runtime gestiti. Ogni runtime semplifica la creazione e l'esecuzione di contenitori con applicazioni Web basate su una versione.NET. Quando si utilizza un runtime .NET, App Runner si avvia con un'immagine di runtime .NET gestita. Questa immagine è basata sull'immagine Docker di Amazon Linux e contiene il pacchetto di runtime per una versione di.NET e alcuni strumenti e pacchetti di dipendenze popolari. App Runner utilizza questa immagine di runtime gestita come immagine di base e aggiunge il codice dell'applicazione per creare un'immagine Docker. Quindi distribuisce questa immagine per eseguire il servizio Web in un contenitore.

Si specifica un runtime per il servizio App Runner quando si crea un servizio utilizzando la console App Runner o l'CreateServiceoperazione API. Puoi anche specificare un runtime come parte del codice sorgente. Usa la `runtime` parola chiave in un file di configurazione di App Runner che includi nel tuo repository di codice. La convenzione di denominazione di un runtime gestito è. *`<language-name><major-version>`*

Per i nomi e le versioni di runtime di.NET validi, vederethe section called “Informazioni sul rilascio”.

App Runner aggiorna il runtime del servizio alla versione più recente a ogni distribuzione o aggiornamento del servizio. Se l'applicazione richiede una versione specifica di un runtime gestito, è possibile specificarla utilizzando la `runtime-version` parola chiave nel file di configurazione di App Runner. È possibile utilizzare qualsiasi livello di versione, inclusa una versione principale o secondaria. App Runner effettua solo aggiornamenti di livello inferiore al runtime del servizio.

Sintassi della versione per i runtime.NET: *`major`*`[.`*`minor`*`[.`*`patch`*`]]`

Ad esempio: `6.0.9`

Gli esempi seguenti mostrano il blocco della versione:

- `6.0`— Blocca le versioni principali e secondarie. App Runner aggiorna solo le versioni patch.
- `6.0.9`— Blocca a una versione di patch specifica. App Runner non aggiorna la versione di runtime.

Argomenti

- Configurazione del runtime di.NET
- Esempi di runtime di.NET
- Informazioni sulla release del runtime di.NET

Configurazione del runtime di.NET

Quando scegli un runtime gestito, devi anche configurare, come minimo, creare ed eseguire comandi. Li configuri durante la creazione o l'aggiornamento del servizio App Runner. Puoi farlo utilizzando uno dei seguenti metodi:

- Utilizzo della console App Runner: specifica i comandi nella sezione Configure build del processo di creazione o della scheda di configurazione.

- Utilizzo dell'API App Runner: richiama l'operazione CreateServiceor UpdateServiceAPI. Specificate i comandi utilizzando i `StartCommand` membri `BuildCommand` e del tipo di CodeConfigurationValuesdati.
- Utilizzo di un file di configurazione: specifica uno o più comandi di compilazione in un massimo di tre fasi di compilazione e un singolo comando di esecuzione che serve per avviare l'applicazione. Sono disponibili impostazioni di configurazione opzionali aggiuntive.

Fornire un file di configurazione è facoltativo. Quando crei un servizio App Runner utilizzando la console o l'API, specifichi se App Runner ottiene le impostazioni di configurazione direttamente al momento della creazione o da un file di configurazione.

Esempi di runtime di.NET

Gli esempi seguenti mostrano i file di configurazione di App Runner per la creazione e l'esecuzione di un servizio.NET. L'ultimo esempio è il codice sorgente di un'applicazione.NET completa che è possibile distribuire su un servizio di runtime.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è`6.0.9`. Puoi sostituirla con una versione che desideri utilizzare. Per la versione di runtime di.NET più recente supportata, vedithe section called “Informazioni sul rilascio”.

File di configurazione.NET minimo

Questo esempio mostra un file di configurazione minimo che è possibile utilizzare con un runtime gestito.NET. Per le ipotesi che App Runner fa con un file di configurazione minimo, vedi. the section called “Esempi di file di configurazione”

Example apprunner.yaml

```
version: 1.0
runtime: dotnet6
build:
  commands:
    build:
      - dotnet publish -c Release -o out
run:
```

```
  command: dotnet out/HelloWorldDotNetApp.dll
```

File di configurazione.NET esteso

Questo esempio mostra l'uso di tutte le chiavi di configurazione con un runtime gestito.NET.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è*6.0.9*. Puoi sostituirla con una versione che desideri utilizzare. Per la versione di runtime di.NET più recente supportata, vedithe section called “Informazioni sul rilascio”.

Example apprunner.yaml

```
version: 1.0
runtime: dotnet6
build:
  commands:
    pre-build:
      - scripts/prebuild.sh
    build:
      - dotnet publish -c Release -o out
    post-build:
      - scripts/postbuild.sh
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
run:
  runtime-version: 6.0.9
  command: dotnet out/HelloWorldDotNetApp.dll
  network:
    port: 5000
    env: APP_PORT
  env:
    - name: ASPNETCORE_URLS
      value: "http://*:5000"
```

Fonte completa dell'applicazione.NET

Questo esempio mostra il codice sorgente di un'applicazione.NET completa che è possibile distribuire su un servizio di runtime.

> **Note**
>
> - Esegui il comando seguente per creare una semplice app web.NET 6: `dotnet new web --name HelloWorldDotNetApp -f net6.0`
> - Aggiungi il `apprunner.yaml` all'app web.NET 6 creata.

Example HelloWorldDotNetApp

```
version: 1.0
runtime: dotnet6
build:
  commands:
    build:
      - dotnet publish -c Release -o out
run:
  command: dotnet out/HelloWorldDotNetApp.dll
  network:
    port: 5000
    env: APP_PORT
  env:
    - name: ASPNETCORE_URLS
      value: "http://*:5000"
```

Informazioni sulla release del runtime di.NET

In questo argomento sono elencati i dettagli completi per le versioni di runtime di.NET supportate da App Runner.

Versioni di runtime supportate: build originale di App Runner

Nome runtime	Versioni minori	Pacchetti inclusi
.NET 6 (dotnet6)	6,0,36	.NET SDK 6.0.428
	6.0.33	.NET SDK 6.0.425
	6.0.32	.NET SDK 6.0.424
	6,0,31	.NET SDK 6.0.423

Nome runtime	Versioni minori	Pacchetti inclusi
	6,0,30	.NET SDK 6.0.422
	6,0,29	.NET SDK 6.0.421
	6,0,28	.NET SDK 6.0.420
	6,0,26	.NET SDK 6.0.418
	6,0,25	.NET SDK 6.0.417
	6,0,24	.NET SDK 6.0.416
	6.0.22	.NET SDK 6.0.414
	6,0,21	.NET SDK 6.0.413
	6.0,20	.NET SDK 6.0.412
	6.0.19	.NET SDK 6.0.411
	6,0,16	.NET SDK 6.0.408
	6,0,15	.NET SDK 6.0.407
	6.0.14	.NET SDK 6.0.406
	6.0.13	.NET SDK 6.0.405
	6.0.12	.NET SDK 6.0.404
	6.0.11	.NET SDK 6.0.403
	6.0.10	.NET SDK 6.0.402
	6.0.9	.NET SDK 6.0.401

> **Note**
>
> App Runner fornisce un processo di compilazione rivisto per i principali runtime specifici che sono stati rilasciati più di recente. Per questo motivo, in alcune sezioni di questo documento vedrai riferimenti alla build di App Runner rivista e alla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

Utilizzo della piattaforma PHP di

La piattaforma AWS App Runner PHP fornisce runtime gestiti. Puoi utilizzare ogni runtime per creare ed eseguire contenitori con applicazioni Web basate su una versione PHP. Quando si utilizza un runtime PHP, App Runner inizia con un'immagine di runtime PHP gestita. Questa immagine è basata sull'immagine Docker di Amazon Linux e contiene il pacchetto di runtime per una versione di PHP e alcuni strumenti. App Runner utilizza questa immagine di runtime gestita come immagine di base e aggiunge il codice dell'applicazione per creare un'immagine Docker. Quindi distribuisce questa immagine per eseguire il servizio Web in un contenitore.

Si specifica un runtime per il servizio App Runner quando si crea un servizio utilizzando la console App Runner o l'CreateServiceoperazione API. Puoi anche specificare un runtime come parte del codice sorgente. Usa la `runtime` parola chiave in un file di configurazione di App Runner che includi nel tuo repository di codice. La convenzione di denominazione di un runtime gestito è. *`<language-name><major-version>`*

Per i nomi e le versioni di runtime PHP validi, vedere. the section called “Informazioni sul rilascio”

App Runner aggiorna il runtime del servizio alla versione più recente a ogni distribuzione o aggiornamento del servizio. Se l'applicazione richiede una versione specifica di un runtime gestito, è possibile specificarla utilizzando la `runtime-version` parola chiave nel file di configurazione di App Runner. È possibile utilizzare qualsiasi livello di versione, inclusa una versione principale o secondaria. App Runner effettua solo aggiornamenti di livello inferiore al runtime del servizio.

Sintassi della versione per i runtime PHP: *`major`*`[.`*`minor`*`[.`*`patch`*`]]`

Ad esempio: `8.1.10`

Di seguito sono riportati alcuni esempi di blocco delle versioni:

- `8.1`— Blocca le versioni principali e secondarie. App Runner aggiorna solo le versioni patch.

- `8.1.10`— Blocca a una versione di patch specifica. App Runner non aggiorna la tua versione di runtime.

> **Important**
>
> Se desideri specificare la directory sorgente del repository di codice per il tuo servizio App Runner in una posizione diversa dalla directory principale del repository predefinita, la versione del runtime gestito da PHP deve essere PHP o successiva. `8.1.22` Le versioni di runtime PHP precedenti `8.1.22` possono utilizzare solo la directory principale dei sorgenti predefinita.

Argomenti

Configurazione del runtime PHP

Quando scegli un runtime gestito, devi anche configurare, come minimo, creare ed eseguire comandi. Li configuri durante la creazione o l'aggiornamento del servizio App Runner. Puoi farlo utilizzando uno dei seguenti metodi:

- Utilizzo della console App Runner: specifica i comandi nella sezione Configure build del processo di creazione o della scheda di configurazione.
- Utilizzo dell'API App Runner: richiama l'operazione CreateServiceor UpdateServiceAPI. Specificate i comandi utilizzando i `StartCommand` membri `BuildCommand` e del tipo di CodeConfigurationValuesdati.
- Utilizzo di un file di configurazione: specifica uno o più comandi di compilazione in un massimo di tre fasi di compilazione e un singolo comando di esecuzione che serve per avviare l'applicazione. Sono disponibili impostazioni di configurazione opzionali aggiuntive.

Fornire un file di configurazione è facoltativo. Quando crei un servizio App Runner utilizzando la console o l'API, specifichi se App Runner ottiene le impostazioni di configurazione direttamente al momento della creazione o da un file di configurazione.

Compatibilità

È possibile eseguire i servizi App Runner sulla piattaforma PHP utilizzando uno dei seguenti server Web:

- Apache HTTP Server
- NGINX

Apache HTTP Server e NGINX sono compatibili con PHP-FPM. È possibile avviare Apache HTTP Server e NGINXutilizzando uno dei seguenti:

- Supervisord - Per ulteriori informazioni sull'esecuzione di un supervisord, vedi Running supervisord.
- Script di avvio

Per esempi su come configurare il servizio App Runner con la piattaforma PHP utilizzando Apache HTTP Server o NGINX, vedi. the section called “Fonte completa dell'applicazione PHP”

Struttura dei file

`index.php`Deve essere installato nella `public` cartella sotto la `root` directory del server web.

> **Note**
>
> Si consiglia di archiviare `supervisord.conf` i file `startup.sh` or nella directory principale del server Web. Assicuratevi che il `start` comando punti alla posizione in cui sono archiviati `supervisord.conf` i file `startup.sh` or.

Di seguito è riportato un esempio della struttura del file se si utilizza supervisord.

```
/
## public/
```

```
#  ## index.php
## apprunner.yaml
## supervisord.conf
```

Di seguito è riportato un esempio della struttura del file se si utilizza uno script di avvio.

```
/
## public/
#  ## index.php
## apprunner.yaml
## startup.sh
```

Si consiglia di archiviare queste strutture di file nella directory dei sorgenti del repository di codice designata per il servizio App Runner.

```
/<sourceDirectory>/
## public/
#  ## index.php
## apprunner.yaml
## startup.sh
```

> **Important**
>
> Se desideri specificare la directory sorgente del repository di codice per il servizio App Runner in una posizione diversa dalla directory principale del repository predefinita, la versione del runtime gestito da PHP deve essere PHP o successiva. `8.1.22` Le versioni di runtime PHP precedenti `8.1.22` possono utilizzare solo la directory principale dei sorgenti predefinita.
>
> App Runner aggiorna il runtime del servizio alla versione più recente a ogni distribuzione o aggiornamento del servizio. Il servizio utilizzerà i runtime più recenti per impostazione predefinita, a meno che non sia stato specificato il blocco della versione utilizzando la `runtime-version` parola chiave nel file di configurazione di App Runner.

Esempi di runtime PHP

Di seguito sono riportati alcuni esempi di file di configurazione di App Runner utilizzati per creare ed eseguire un servizio PHP.

File di configurazione PHP minimo

L'esempio seguente è un file di configurazione minimo che è possibile utilizzare con un runtime gestito da PHP. Per ulteriori informazioni su un file di configurazione minimo, vederethe section called "Esempi di file di configurazione".

Example apprunner.yaml

```
version: 1.0
runtime: php81
build:
  commands:
    build:
      - echo example build command for PHP
run:
  command: ./startup.sh
```

File di configurazione PHP esteso

L'esempio seguente utilizza tutte le chiavi di configurazione con un runtime gestito da PHP.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è*8.1.10*. Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime PHP supportata, vedithe section called "Informazioni sul rilascio".

Example apprunner.yaml

```
version: 1.0
runtime: php81
build:
  commands:
     pre-build:
      - scripts/prebuild.sh
    build:
      - echo example build command for PHP
    post-build:
      - scripts/postbuild.sh
  env:
    - name: MY_VAR_EXAMPLE
```

```
      value: "example"
run:
  runtime-version: 8.1.10
  command: ./startup.sh
  network:
    port: 5000
    env: APP_PORT
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
```

Fonte completa dell'applicazione PHP

I seguenti esempi sono del codice sorgente di un'applicazione PHP che è possibile utilizzare per la distribuzione su un servizio di runtime PHP utilizzando Apache HTTP Server o NGINX. Questi esempi presuppongono che si utilizzi la struttura di file predefinita.

Esecuzione della piattaforma PHP con Apache HTTP Server utilizzo di supervisord

Example Struttura dei file

> **Note**
>
> - Il `supervisord.conf` file può essere archiviato ovunque nel repository. Assicurati che il `start` comando punti alla posizione in cui è archiviato il `supervisord.conf` file.
> - `index.php`Deve essere installato nella `public` cartella sotto la `root` directory.

```
/
## public/
#  ## index.php
## apprunner.yaml
## supervisord.conf
```

Example supervisord.conf

```
[supervisord]
nodaemon=true

[program:httpd]
command=httpd -DFOREGROUND
```

```
autostart=true
autorestart=true
stdout_logfile=/dev/stdout
stdout_logfile_maxbytes=0
stderr_logfile=/dev/stderr
stderr_logfile_maxbytes=0

[program:php-fpm]
command=php-fpm -F
autostart=true
autorestart=true
stdout_logfile=/dev/stdout
stdout_logfile_maxbytes=0
stderr_logfile=/dev/stderr
stderr_logfile_maxbytes=0
```

Example apprunner.yaml

```
version: 1.0
runtime: php81
build:
  commands:
    build:
      - PYTHON=python2 amazon-linux-extras install epel
      - yum -y install supervisor
run:
  command: supervisord
  network:
    port: 8080
    env: APP_PORT
```

Example index.php

```
<html>
<head> <title>First PHP App</title> </head>
<body>
<?php
    print("Hello World!");
    print("<br>");
?>
</body>
</html>
```

Piattaforma PHP in esecuzione con Apache HTTP Server utilizzo di startup script

Example Struttura dei file

> **Note**
>
> - Il `startup.sh` file può essere archiviato ovunque nel repository. Assicurati che il `start` comando punti alla posizione in cui è archiviato il `startup.sh` file.
> - `index.php`Deve essere installato nella `public` cartella sotto la `root` directory.

```
/
## public/
#  ## index.php
## apprunner.yaml
## startup.sh
```

Example startup.sh

```
#!/bin/bash

set -o monitor

trap exit SIGCHLD

# Start apache
httpd -DFOREGROUND &

# Start php-fpm
php-fpm -F &

wait
```

> **Note**
>
> - Assicurati di salvare il `startup.sh` file come eseguibile prima di salvarlo in un repository Git. Utilizzalo `chmod +x startup.sh` per impostare l'autorizzazione di esecuzione sul tuo `startup.sh` file.

- Se non salvate il `startup.sh` file come eseguibile, `chmod +x startup.sh` immettetelo come `build` comando nel `apprunner.yaml` file.

Example apprunner.yaml

```
version: 1.0
runtime: php81
build:
  commands:
    build:
      - echo example build command for PHP
run:
  command: ./startup.sh
  network:
    port: 8080
    env: APP_PORT
```

Example index.php

```
<html>
<head> <title>First PHP App</title> </head>
<body>
<?php
    print("Hello World!");
    print("<br>");
?>
</body>
</html>
```

Piattaforma PHP in esecuzione con NGINX utilizzo di supervisord

Example Struttura dei file

Note

- Il `supervisord.conf` file può essere archiviato ovunque nel repository. Assicurati che il `start` comando punti alla posizione in cui è archiviato il `supervisord.conf` file.
- `index.php`Deve essere installato nella `public` cartella sotto la `root` directory.

```
/
## public/
#  ## index.php
## apprunner.yaml
## supervisord.conf
```

Example supervisord.conf

```
[supervisord]
nodaemon=true

[program:nginx]
command=nginx -g "daemon off;"
autostart=true
autorestart=true
stdout_logfile=/dev/stdout
stdout_logfile_maxbytes=0
stderr_logfile=/dev/stderr
stderr_logfile_maxbytes=0

[program:php-fpm]
command=php-fpm -F
autostart=true
autorestart=true
stdout_logfile=/dev/stdout
stdout_logfile_maxbytes=0
stderr_logfile=/dev/stderr
stderr_logfile_maxbytes=0
```

Example apprunner.yaml

```
version: 1.0
runtime: php81
build:
  commands:
    build:
      - PYTHON=python2 amazon-linux-extras install epel
      - yum -y install supervisor
run:
  command: supervisord
  network:
    port: 8080
```

```
    env: APP_PORT
```

Example index.php

```
<html>
<head> <title>First PHP App</title> </head>
<body>
<?php
    print("Hello World!");
    print("<br>");
?>
</body>
</html>
```

Piattaforma PHP in esecuzione con NGINX utilizzo di startup script

Example Struttura dei file

> **Note**
>
> - Il `startup.sh` file può essere archiviato ovunque nel repository. Assicurati che il `start` comando punti alla posizione in cui è archiviato il `startup.sh` file.
> - `index.php`Deve essere installato nella `public` cartella sotto la `root` directory.

```
/
## public/
#  ## index.php
## apprunner.yaml
## startup.sh
```

Example startup.sh

```
#!/bin/bash

set -o monitor

trap exit SIGCHLD

# Start nginx
```

```
nginx -g 'daemon off;' &

# Start php-fpm
php-fpm -F &

wait
```

> **Note**
>
> - Assicurati di salvare il `startup.sh` file come eseguibile prima di salvarlo in un repository Git. Utilizzalo `chmod +x startup.sh` per impostare l'autorizzazione di esecuzione sul tuo `startup.sh` file.
> - Se non salvate il `startup.sh` file come eseguibile, `chmod +x startup.sh` immettetelo come `build` comando nel `apprunner.yaml` file.

Example apprunner.yaml

```
version: 1.0
runtime: php81
build:
  commands:
    build:
      - echo example build command for PHP
run:
  command: ./startup.sh
  network:
    port: 8080
    env: APP_PORT
```

Example index.php

```
<html>
<head> <title>First PHP App</title> </head>
<body>
<?php
    print("Hello World!");
    print("<br>");
?>
</body>
```

```
</html>
```

Informazioni sul rilascio del runtime di PHP

Questo argomento elenca i dettagli completi per le versioni di runtime PHP supportate da App Runner.

Versioni di runtime supportate: build originale di App Runner

Nome runtime	Versioni minori	Pacchetti inclusi
PHP 8.1 (php81)	8.1.32	
	8,1,31	
	8,1,29	
	8,1,28	
	8,1,27	
	8,1,26	
	8,1,24	
	81,22	
	8,1,21	
	81,20	
	8,1,19	
	8,1,17	
	8,1,16	
	8,1,14	
	8,1,13	
	8,1,12	

Nome runtime	Versioni minori	Pacchetti inclusi
	81,10	

> **Note**
>
> App Runner fornisce un processo di compilazione rivisto per i principali runtime specifici che sono stati rilasciati più di recente. Per questo motivo, in alcune sezioni di questo documento vedrai riferimenti alla build di App Runner rivista e alla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

Utilizzo della piattaforma Ruby di

La piattaforma AWS App Runner Ruby offre runtime gestiti. Ogni runtime semplifica la creazione e l'esecuzione di contenitori con applicazioni Web basate su una versione di Ruby. Quando si utilizza un runtime Ruby, App Runner si avvia con un'immagine di runtime Ruby gestita. Questa immagine è basata sull'immagine Docker di Amazon Linux e contiene il pacchetto di runtime per una versione di Ruby e alcuni strumenti. App Runner utilizza questa immagine di runtime gestita come immagine di base e aggiunge il codice dell'applicazione per creare un'immagine Docker. Quindi distribuisce questa immagine per eseguire il servizio Web in un contenitore.

Si specifica un runtime per il servizio App Runner quando si crea un servizio utilizzando la console App Runner o l'CreateServiceoperazione API. Puoi anche specificare un runtime come parte del codice sorgente. Usa la `runtime` parola chiave in un file di configurazione di App Runner che includi nel tuo repository di codice. La convenzione di denominazione di un runtime gestito è. *`<language-name><major-version>`*

Per i nomi e le versioni di runtime di Ruby validi, vedere. the section called “Informazioni sul rilascio”

App Runner aggiorna il runtime del servizio alla versione più recente a ogni distribuzione o aggiornamento del servizio. Se l'applicazione richiede una versione specifica di un runtime gestito, è possibile specificarla utilizzando la `runtime-version` parola chiave nel file di configurazione di App Runner. È possibile utilizzare qualsiasi livello di versione, inclusa una versione principale o secondaria. App Runner effettua solo aggiornamenti di livello inferiore al runtime del servizio.

Sintassi della versione per i runtime di Ruby: *`major`*`[.`*`minor`*`[.`*`patch`*`]]`

Ad esempio: `3.1.2`

I seguenti esempi mostrano il blocco della versione:

- `3.1`— Blocca le versioni principali e secondarie. App Runner aggiorna solo le versioni patch.
- `3.1.2`— Blocca a una versione di patch specifica. App Runner non aggiorna la versione di runtime.

Argomenti

Configurazione del runtime di Ruby

Quando scegli un runtime gestito, devi anche configurare, come minimo, creare ed eseguire comandi. Li configuri durante la creazione o l'aggiornamento del servizio App Runner. Puoi farlo utilizzando uno dei seguenti metodi:

- Utilizzo della console App Runner: specifica i comandi nella sezione Configure build del processo di creazione o della scheda di configurazione.
- Utilizzo dell'API App Runner: richiama l'operazione CreateServiceor UpdateServiceAPI. Specificate i comandi utilizzando i `StartCommand` membri `BuildCommand` e del tipo di CodeConfigurationValuesdati.
- Utilizzo di un file di configurazione: specifica uno o più comandi di compilazione in un massimo di tre fasi di compilazione e un singolo comando di esecuzione che serve per avviare l'applicazione. Sono disponibili impostazioni di configurazione opzionali aggiuntive.

Fornire un file di configurazione è facoltativo. Quando crei un servizio App Runner utilizzando la console o l'API, specifichi se App Runner ottiene le impostazioni di configurazione direttamente al momento della creazione o da un file di configurazione.

Esempi di runtime in Ruby

Gli esempi seguenti mostrano i file di configurazione di App Runner per la creazione e l'esecuzione di un servizio Ruby.

File di configurazione minimo di Ruby

Questo esempio mostra un file di configurazione minimo che puoi usare con un runtime gestito da Ruby. Per le ipotesi che App Runner fa con un file di configurazione minimo, vedi. the section called “Esempi di file di configurazione”

Example apprunner.yaml

```
version: 1.0
runtime: ruby31
build:
  commands:
    build:
      - bundle install
run:
  command: bundle exec rackup --host 0.0.0.0 -p 8080
```

File di configurazione Ruby esteso

Questo esempio mostra l'uso di tutte le chiavi di configurazione con un runtime gestito da Ruby.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è`3.1.2`. Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime di Ruby supportata, vedithe section called “Informazioni sul rilascio”.

Example apprunner.yaml

```
version: 1.0
runtime: ruby31
build:
  commands:
     pre-build:
      - scripts/prebuild.sh
    build:
      - bundle install
    post-build:
      - scripts/postbuild.sh
  env:
```

```
    - name: MY_VAR_EXAMPLE
      value: "example"
run:
  runtime-version: 3.1.2
  command: bundle exec rackup --host 0.0.0.0 -p 4567
  network:
    port: 4567
    env: APP_PORT
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
```

Fonte completa dell'applicazione Ruby

Questi esempi mostrano il codice sorgente di un'applicazione Ruby completa che è possibile distribuire su un servizio di runtime Ruby.

Example server.rb

```
# server.rb
require 'sinatra'

get '/' do
  'Hello World!'
end
```

Example config.ru

```
# config.ru

require './server'

run Sinatra::Application
```

Example Gemfile

```
# Gemfile
source 'https://rubygems.org (https://rubygems.org/)'

gem 'sinatra'
gem 'puma'
```

Example apprunner.yaml

```
version: 1.0
runtime: ruby31
build:
  commands:
    build:
      - bundle install
run:
  command: bundle exec rackup --host 0.0.0.0 -p 4567
  network:
    port: 4567
    env: APP_PORT
```

Informazioni sul rilascio del runtime di Ruby

Questo argomento elenca i dettagli completi per le versioni di runtime di Ruby supportate da App Runner.

Versioni di runtime supportate: build originale di App Runner

Nome runtime	Versioni minori	Pacchetti inclusi
Ruby 3.1 (ruby31)	3.1.7	SQLite 3.50.1
	3.1.7	SQLite 3,50,0
	3.1.6	SQLite 3,49,1
	3.1.4	SQLite 3,46,0
	3.1.3	SQLite 3,41,0
	3.1.2	SQLite 3,339,4

> **Note**
>
> App Runner fornisce un processo di compilazione rivisto per i principali runtime specifici che sono stati rilasciati più di recente. Per questo motivo, in alcune sezioni di questo documento

vedrai riferimenti alla build di App Runner rivista e alla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

Utilizzo della piattaforma Go di

La piattaforma AWS App Runner Go offre tempi di esecuzione gestiti. Ogni runtime semplifica la creazione e l'esecuzione di contenitori con applicazioni Web basate su una versione Go. Quando si utilizza un runtime Go, App Runner si avvia con un'immagine di runtime Go gestita. Questa immagine è basata sull'immagine Docker di Amazon Linux e contiene il pacchetto di runtime per una versione di Go e alcuni strumenti. App Runner utilizza questa immagine di runtime gestita come immagine di base e aggiunge il codice dell'applicazione per creare un'immagine Docker. Quindi distribuisce questa immagine per eseguire il servizio Web in un contenitore.

Si specifica un runtime per il servizio App Runner quando si crea un servizio utilizzando la console App Runner o l'CreateServiceoperazione API. Puoi anche specificare un runtime come parte del codice sorgente. Usa la `runtime` parola chiave in un file di configurazione di App Runner che includi nel tuo repository di codice. La convenzione di denominazione di un runtime gestito è. *`<language-name><major-version>`*

Per i nomi e le versioni di runtime Go validi, vedithe section called “Informazioni sulla versione”.

App Runner aggiorna il runtime del servizio alla versione più recente a ogni distribuzione o aggiornamento del servizio. Se l'applicazione richiede una versione specifica di un runtime gestito, è possibile specificarla utilizzando la `runtime-version` parola chiave nel file di configurazione di App Runner. È possibile utilizzare qualsiasi livello di versione, inclusa una versione principale o secondaria. App Runner effettua solo aggiornamenti di livello inferiore al runtime del servizio.

Sintassi della versione per i runtime Go: *`major`*`[.`*`minor`*`[.`*`patch`*`]]`

Ad esempio: `1.18.7`

I seguenti esempi mostrano il blocco della versione:

- `1.18`— Blocca le versioni principali e secondarie. App Runner aggiorna solo le versioni patch.
- `1.18.7`— Blocca a una versione di patch specifica. App Runner non aggiorna la versione di runtime.

Argomenti

- Configurazione Go runtime
- Esempi di runtime Go
- Informazioni sulla versione di Go Runtime

Configurazione Go runtime

Quando scegli un runtime gestito, devi anche configurare, come minimo, creare ed eseguire comandi. Li configuri durante la creazione o l'aggiornamento del servizio App Runner. Puoi farlo utilizzando uno dei seguenti metodi:

- Utilizzo della console App Runner: specifica i comandi nella sezione Configure build del processo di creazione o della scheda di configurazione.
- Utilizzo dell'API App Runner: richiama l'operazione CreateServiceor UpdateServiceAPI. Specificate i comandi utilizzando i `StartCommand` membri `BuildCommand` e del tipo di CodeConfigurationValuesdati.
- Utilizzo di un file di configurazione: specifica uno o più comandi di compilazione in un massimo di tre fasi di compilazione e un singolo comando di esecuzione che serve per avviare l'applicazione. Sono disponibili impostazioni di configurazione opzionali aggiuntive.

Fornire un file di configurazione è facoltativo. Quando crei un servizio App Runner utilizzando la console o l'API, specifichi se App Runner ottiene le impostazioni di configurazione direttamente al momento della creazione o da un file di configurazione.

Esempi di runtime Go

Gli esempi seguenti mostrano i file di configurazione di App Runner per la creazione e l'esecuzione di un servizio Go.

File di configurazione Minimal Go

Questo esempio mostra un file di configurazione minimo che è possibile utilizzare con un runtime gestito da Go. Per le ipotesi che App Runner fa con un file di configurazione minimo, vedi. the section called “Esempi di file di configurazione”

Example apprunner.yaml

```
version: 1.0
```

```
runtime: go1
build:
  commands:
    build:
      - go build main.go
run:
  command: ./main
```

File di configurazione Extended Go

Questo esempio mostra l'uso di tutte le chiavi di configurazione con un runtime Go gestito.

> **Note**
>
> La versione di runtime utilizzata in questi esempi è*1.18.7*. Puoi sostituirla con una versione che desideri utilizzare. Per l'ultima versione di runtime Go supportata, consultathe section called “Informazioni sulla versione”.

Example apprunner.yaml

```
version: 1.0
runtime: go1
build:
  commands:
     pre-build:
      - scripts/prebuild.sh
    build:
      - go build main.go
    post-build:
      - scripts/postbuild.sh
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
run:
  runtime-version: 1.18.7
  command: ./main
  network:
    port: 3000
    env: APP_PORT
  env:
    - name: MY_VAR_EXAMPLE
```

```
      value: "example"
```

Fonte completa dell'applicazione Go

Questi esempi mostrano il codice sorgente di un'applicazione Go completa che puoi distribuire su un servizio di runtime Go.

Example main.go

```
package main
import (
    "fmt"
    "net/http"
)

func main() {
    http.HandleFunc("/", func(w http.ResponseWriter, r *http.Request) {
        fmt.Fprint(w, "<h1>Welcome to App Runner</h1>")
    })
    fmt.Println("Starting the server on :3000...")
    http.ListenAndServe(":3000", nil)
}
```

Example apprunner.yaml

```
version: 1.0
runtime: go1
build:
  commands:
    build:
      - go build main.go
run:
  command: ./main
  network:
    port: 3000
    env: APP_PORT
```

Informazioni sulla versione di Go Runtime

Questo argomento elenca i dettagli completi per le versioni di runtime Go supportate da App Runner.

Versioni di runtime supportate: build originale di App Runner

Nome runtime	Versioni minori	Pacchetti inclusi
Vai 1 (go1)	1.18.10	
	1.18,9	
	1.18,8	
	1.18,7	

> **Note**
>
> App Runner fornisce un processo di compilazione rivisto per i principali runtime specifici che sono stati rilasciati più di recente. Per questo motivo, in alcune sezioni di questo documento vedrai riferimenti alla build di App Runner rivista e alla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

Sviluppo di codice applicativo per App Runner

Questo capitolo illustra le informazioni sul runtime e le linee guida di sviluppo da prendere in considerazione quando si sviluppa o si migra il codice dell'applicazione per la distribuzione. AWS App Runner

Informazioni sul runtime

Sia che tu fornisca un'immagine del contenitore o che App Runner ne crei una per te, App Runner esegue il codice dell'applicazione in un'istanza del contenitore. Ecco alcuni aspetti chiave dell'ambiente di runtime dell'istanza del contenitore.

- Supporto del framework: App Runner supporta qualsiasi immagine che implementa un'applicazione Web. È indipendente dal linguaggio di programmazione scelto e dal server o framework di applicazioni Web che utilizzi, se ne utilizzi uno. Per tua comodità, forniamo runtime gestiti specifici della piattaforma per varie piattaforme di programmazione, per semplificare il processo di creazione delle applicazioni e la creazione di immagini astratte.
- Richieste Web: App Runner fornisce supporto per HTTP 1.0 e HTTP 1.1 alle istanze del contenitore. Per ulteriori informazioni sulla configurazione del servizio, consulta. the section called “Configurazione” Non è necessario implementare la gestione del traffico sicuro HTTPS. App Runner reindirizza tutte le richieste HTTP in entrata agli endpoint HTTPS corrispondenti. Non è necessario configurare alcuna impostazione per abilitare il reindirizzamento delle richieste Web HTTP. App Runner termina il TLS prima di passare le richieste all'istanza del contenitore dell'applicazione.

> **Note**
>
> - Esiste un limite di timeout di richiesta totale di 120 secondi per le richieste HTTP. I 120 secondi includono il tempo impiegato dall'applicazione per leggere la richiesta, incluso il corpo, e completare la scrittura della risposta HTTP.
> - Il limite di timeout di lettura e risposta della richiesta dipende dalle applicazioni utilizzate. Queste applicazioni possono avere i propri timeout interni, ad esempio il server HTTP per Python, Gunicorn, ha un limite di timeout predefinito di 30 secondi. In questi casi, il limite di timeout dell'applicazione ha la precedenza sul limite di timeout di 120 secondi di App Runner.

- Non è necessario configurare le suite di crittografia TLS o altri parametri poiché App Runner, essendo un servizio completamente gestito, gestisce la terminazione TLS al posto tuo.

- App stateless: attualmente App Runner non supporta un'app stateful. Pertanto, App Runner non garantisce la persistenza dello stato oltre la durata dell'elaborazione di una singola richiesta web in entrata.
- Archiviazione: App Runner aumenta o riduce automaticamente le istanze dell'applicazione App Runner in base al volume di traffico in entrata. Puoi configurare le opzioni di ridimensionamento automatico per la tua applicazione App Runner. Poiché il numero di istanze attualmente attive che elaborano le richieste Web si basa sul volume di traffico in entrata, App Runner non può garantire che i file possano persistere oltre l'elaborazione di una singola richiesta. Pertanto, App Runner implementa il file system nell'istanza del contenitore come archiviazione temporanea, il che implica che i file siano transitori. Ad esempio, i file non persistono quando metti in pausa e riprendi il servizio App Runner.

  App Runner offre 3 GB di spazio di archiviazione temporaneo e utilizza una parte di tale spazio per l'immagine del contenitore estratta, compressa e non compressa sull'istanza. Lo spazio di archiviazione temporaneo rimanente può essere utilizzato dal servizio App Runner. Tuttavia, questa non è una memoria permanente a causa della sua natura stateless.

  > **Note**
  >
  > Potrebbero verificarsi scenari in cui i file di archiviazione persistono tra le richieste. Ad esempio, se la richiesta successiva arriva sulla stessa istanza, i file di archiviazione persisteranno. La persistenza dei file di archiviazione tra le richieste può essere utile in determinate situazioni. Ad esempio, quando gestisci una richiesta, puoi memorizzare nella cache i file scaricati dall'applicazione se le richieste future potrebbero averne bisogno. Ciò potrebbe velocizzare la gestione delle richieste future, ma non può garantire i guadagni di velocità. Il codice non deve presupporre che esista ancora un file scaricato in una richiesta precedente.
  > Per garantire la memorizzazione nella cache utilizzando un archivio dati in memoria ad alta velocità e bassa latenza, utilizza un servizio come Amazon. ElastiCache

- Variabili di ambiente: per impostazione predefinita, App Runner rende disponibile la variabile di `PORT` ambiente nell'istanza del contenitore. È possibile configurare il valore della variabile con informazioni sulla porta e aggiungere variabili e valori di ambiente personalizzati. È inoltre

possibile fare riferimento ai dati sensibili archiviati in AWS Secrets ManagerAWS Systems Manager Parameter Store come variabili di ambiente. Per ulteriori informazioni sulla creazione di variabili di ambiente, vedereVariabili di ambiente di riferimento.

- Ruolo dell'istanza: se il codice dell'applicazione effettua chiamate a qualsiasi AWS servizio, utilizzando il servizio APIs o uno dei due AWS SDKs, crea un ruolo di istanza utilizzando AWS Identity and Access Management (IAM). Quindi, collegalo al servizio App Runner quando lo crei. AWS Includi tutte le autorizzazioni di azione di servizio richieste dal codice nel tuo ruolo di istanza. Per ulteriori informazioni, consulta the section called “Ruolo dell'istanza”.

Linee guida di sviluppo del codice

Prendi in considerazione queste linee guida quando sviluppi codice per un'applicazione web App Runner.

- Applicazione di patch alle immagini dei contenitori: quando fornisci immagini dei contenitori, sei responsabile dell'aggiornamento e dell'applicazione di patch regolari a tali immagini. Sebbene App Runner gestisca l'infrastruttura, è necessario garantire la sicurezza e up-to-date lo stato delle immagini del contenitore fornite. Per ulteriori informazioni, consulta la documentazione di AWS App Runner
- Progetta codice stateless: progetta l'applicazione web che distribuisci sul tuo servizio App Runner in modo che sia stateless. Il codice dovrebbe presupporre che nessuno stato persista oltre la durata dell'elaborazione di una singola richiesta web in entrata.
- Eliminazione di file temporanei: quando crei file, questi vengono archiviati in un file system e occupano parte dell'allocazione dello spazio di archiviazione del tuo servizio. Per evitare out-of-storage errori, non conservate i file temporanei per periodi prolungati. Quando prendi decisioni sulla memorizzazione dei file nella cache, bilancia le dimensioni dello storage con la velocità di gestione delle richieste.
- Avvio dell'istanza: App Runner fornisce cinque minuti di avvio dell'istanza. L'istanza deve ascoltare le richieste sulle porte di ascolto configurate ed essere integra entro cinque minuti dall'avvio. Durante l'avvio, alle istanze di App Runner viene allocata una CPU virtuale (vCPU) in base alla configurazione della vCPU. Per ulteriori informazioni sulla configurazione delle vCPU disponibili, vedere. the section called “Configurazioni supportate da App Runner”

  Una volta avviata correttamente, l'istanza entra in uno stato di inattività e attende le richieste. Il pagamento viene effettuato in base alla durata di avvio dell'istanza, con un addebito minimo di un minuto per avvio dell'istanza. Per informazioni sui prezzi, consultare Prezzi di AWS App Runner.

Utilizzo della console App Runner

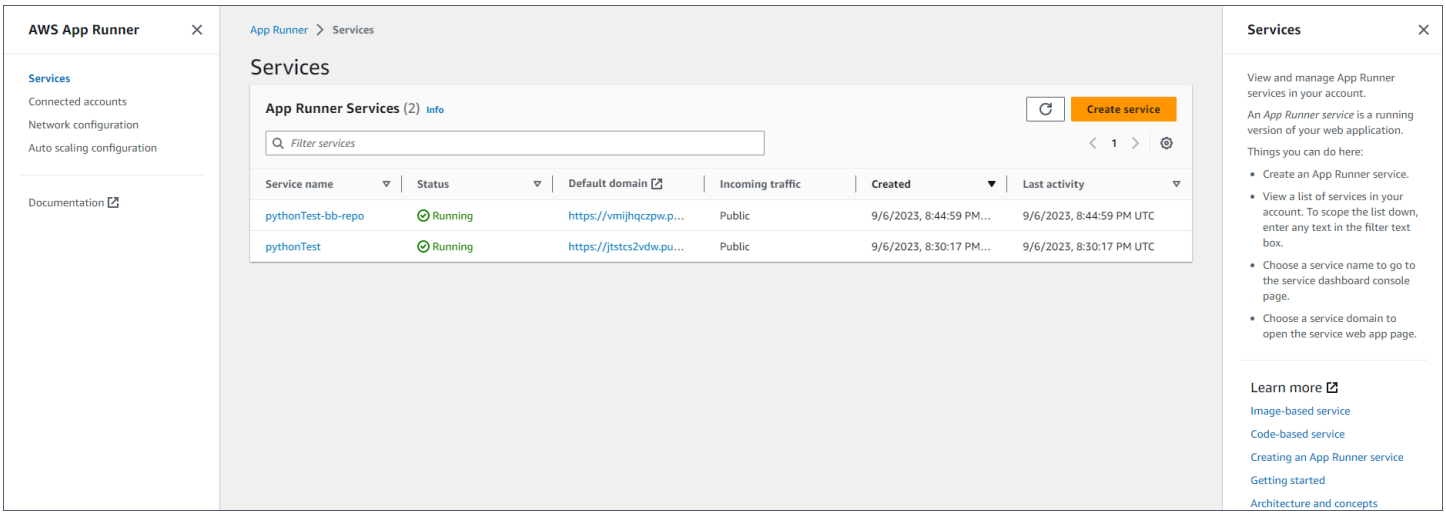
Usa la AWS App Runner console per creare, gestire e monitorare i servizi App Runner e le risorse correlate, come gli account connessi. È possibile visualizzare i servizi esistenti, crearne di nuovi e configurare un servizio. È possibile visualizzare lo stato di un servizio App Runner, visualizzare i registri, monitorare l'attività e tenere traccia delle metriche. Puoi anche accedere al sito Web del servizio o all'archivio di origine.

Le sezioni seguenti descrivono il layout e la funzionalità della console e forniscono informazioni correlate.

Layout generale della console

La console App Runner ha tre aree. Da sinistra a destra:

- Pannello di navigazione: un riquadro laterale che può essere compresso o espanso. Utilizzalo per scegliere la pagina della console di primo livello che desideri utilizzare.
- Riquadro dei contenuti: la parte principale della pagina della console. Utilizzatelo per visualizzare informazioni ed eseguire le vostre attività.
- Riquadro Aiuto: un riquadro laterale per ulteriori informazioni. Espandilo per ricevere assistenza sulla pagina in cui ti trovi. Oppure scegli un link Informazioni su una pagina della console per ottenere assistenza contestuale.



La pagina Servizi

La pagina Servizi elenca i servizi App Runner presenti nell'account. È possibile restringere l'elenco utilizzando la casella di testo del filtro.

Per accedere alla pagina Servizi

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua Regione AWS.
2. Nel riquadro di navigazione, scegli Servizi.

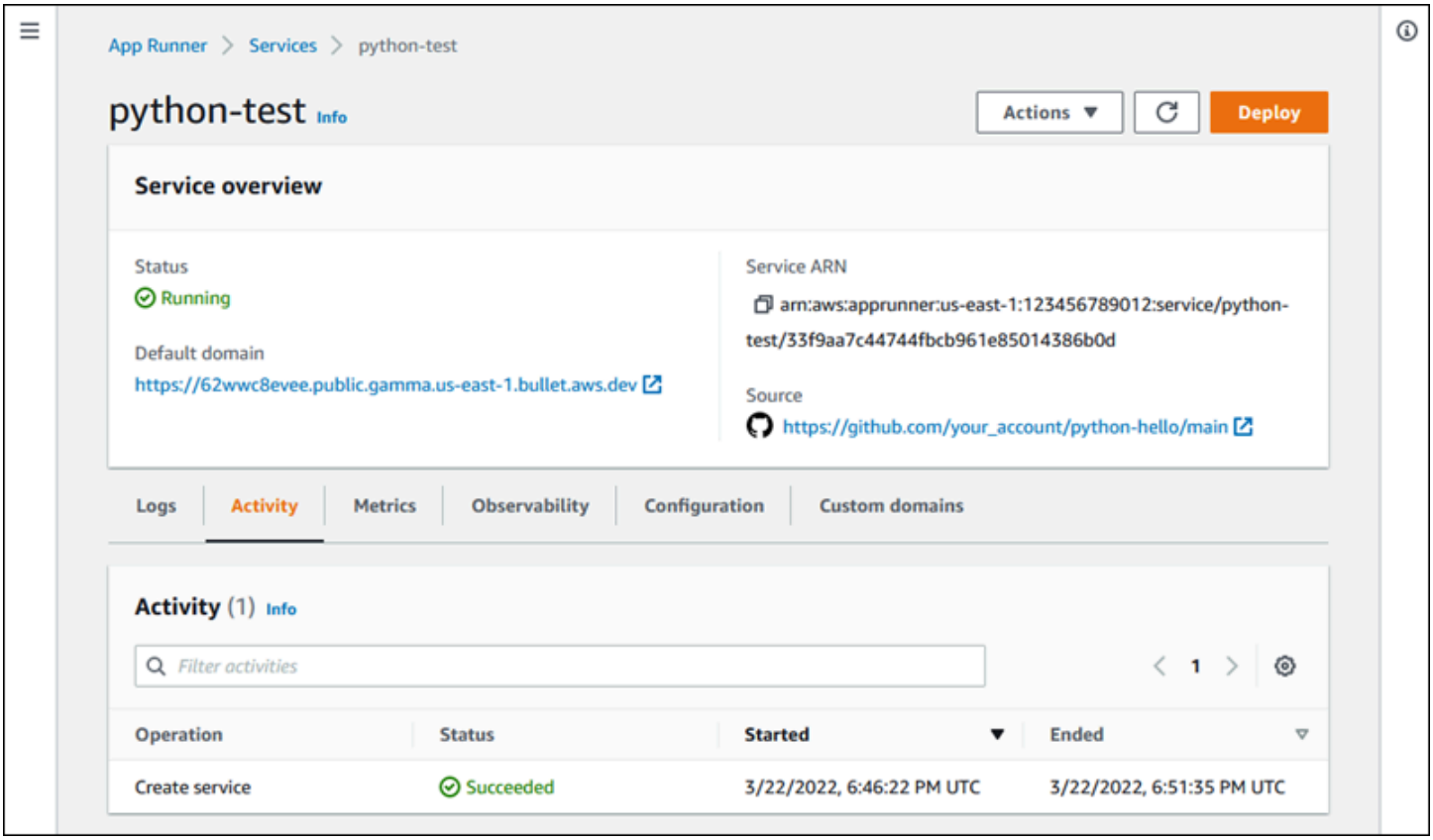
Cose che puoi fare qui:

- Crea un servizio App Runner. Per ulteriori informazioni, consulta the section called “Creazione”.
- Scegli un nome di servizio per accedere alla pagina della console del dashboard del servizio.
- Scegli un dominio di servizio per aprire la pagina dell'app Web del servizio.

La pagina del pannello di controllo del servizio

È possibile visualizzare le informazioni su un servizio App Runner e gestirle dalla pagina del pannello di controllo del servizio. Nella parte superiore della pagina, puoi vedere il nome del servizio.

Per accedere alla dashboard del servizio, vai alla pagina Servizi (vedi sezione precedente), quindi scegli il servizio App Runner.



La sezione Panoramica del servizio fornisce dettagli di base sul servizio App Runner e sull'applicazione. Cose che puoi fare qui:

- Visualizza i dettagli del servizio come lo stato, lo stato e l'ARN.
- Passa al dominio predefinito, il dominio fornito da App Runner per l'applicazione Web in esecuzione nel tuo servizio. Si tratta di un sottodominio nel dominio di proprietà di App `awsapprunner.com` Runner.
- Passa al repository di origine distribuito nel servizio.
- Avvia l'implementazione di un repository di origine nel tuo servizio.
- Metti in pausa, riprendi ed elimina il servizio.

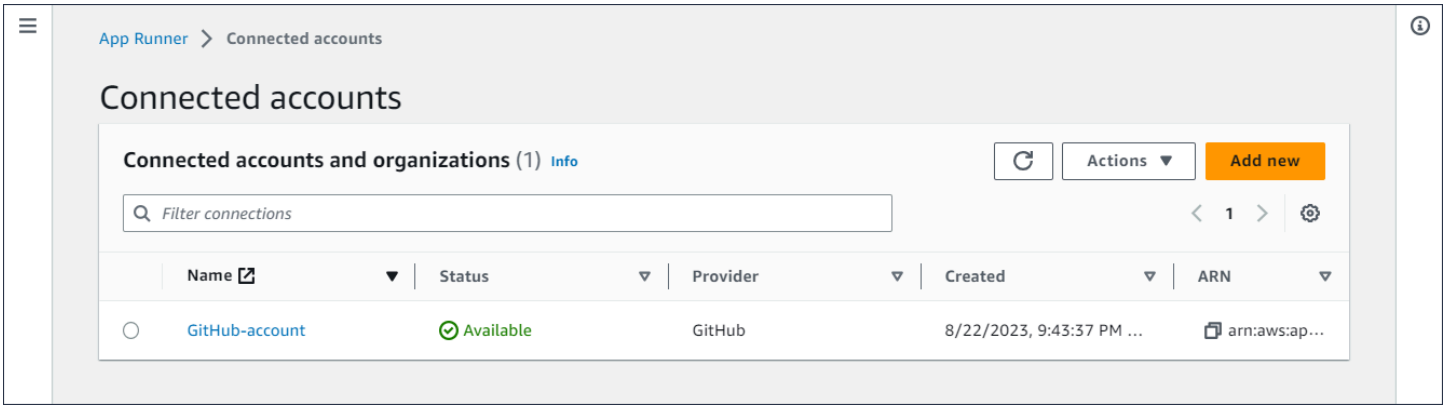
Le schede sotto la panoramica del servizio riguardano la gestione e l'osservabilità del servizio.

La pagina Account connessi

La pagina Account connessi elenca le connessioni di App Runner ai provider di repository di codice sorgente presenti nell'account. È possibile restringere l'elenco utilizzando la casella di testo del filtro. Per ulteriori informazioni sugli account connessi, vederethe section called “Connessioni”.

Per accedere alla pagina Account connessi

1. Apri la console App Runner e, nell'elenco Regioni, seleziona le tue Regione AWS.
2. Nel riquadro di navigazione, scegli Account connessi.



Cose che puoi fare qui:

- Visualizza un elenco di connessioni ai provider di repository presenti nel tuo account. Per restringere l'elenco, inserisci qualsiasi testo nella casella di testo del filtro.
- Scegli un nome di connessione per accedere all'account o all'organizzazione del provider correlato.
- Seleziona una connessione per completare l'handshake per una connessione appena stabilita (come parte della creazione di un servizio) o per eliminare la connessione.

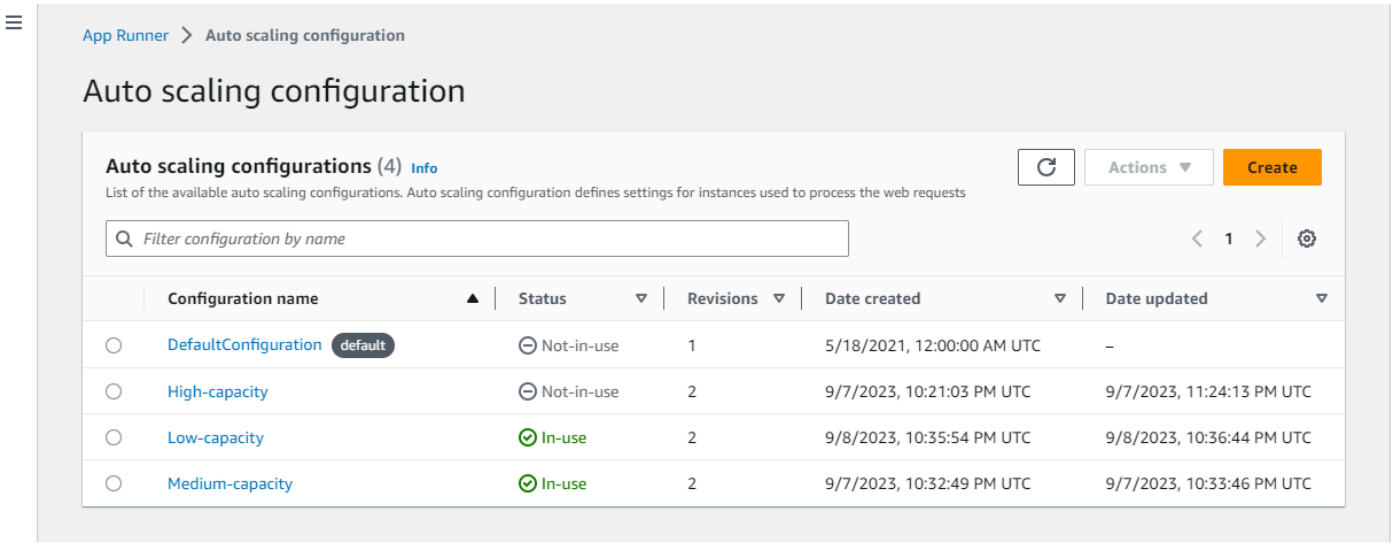
La pagina delle configurazioni di Auto Scaling

La pagina Configurazioni di ridimensionamento automatico elenca le configurazioni di ridimensionamento automatico che hai impostato nel tuo account. È possibile configurare alcuni parametri per regolare il comportamento del ridimensionamento automatico e salvarli in diverse configurazioni che è possibile assegnare successivamente a uno o più servizi App Runner. È possibile restringere l'elenco utilizzando la casella di testo del filtro. Per ulteriori informazioni sulle

configurazioni di ridimensionamento automatico, vedere. Gestire la scalabilità automatica per un servizio

Per accedere alla pagina di configurazione della scalabilità automatica

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Nel riquadro di navigazione, scegli Configurazione della scalabilità automatica.



Cose che puoi fare qui:

- Visualizza l'elenco delle configurazioni di autoscaling esistenti nel tuo account.
- Crea una nuova configurazione di scalabilità automatica o una revisione per una configurazione esistente.
- Imposta una configurazione di ridimensionamento automatico come predefinita per i nuovi servizi che crei.
- Eliminare una configurazione.
- Seleziona il nome di una configurazione per accedere al pannello Auto scaling revisioni e gestire le revisioni.

Gestione del servizio App Runner

Questo capitolo descrive come gestire il AWS App Runner servizio. In questo capitolo imparerai a gestire il ciclo di vita del servizio: creare, configurare ed eliminare un servizio, distribuire nuove versioni dell'applicazione al servizio e controllare la disponibilità del servizio Web sospendendo e riavviando il servizio. Imparerai anche a gestire altri aspetti del tuo servizio, come le connessioni e la scalabilità automatica.

Argomenti

Creazione di un servizio App Runner

AWS App Runner automatizza la transizione da un'immagine del contenitore o da un archivio di codice sorgente a un servizio Web in esecuzione con scalabilità automatica. Indirizzi App Runner all'immagine o al codice sorgente, specificando solo un numero limitato di impostazioni richieste. App Runner crea l'applicazione se necessario, fornisce risorse di calcolo e distribuisce l'applicazione per eseguirla su di esse.

Quando si crea un servizio, App Runner crea una risorsa di servizio. In alcuni casi, potrebbe essere necessario fornire una risorsa di connessione. Se si utilizza la console App Runner, la console crea implicitamente la risorsa di connessione. Per ulteriori informazioni sui tipi di risorse App Runner, consulta. the section called “Risorse App Runner” Per ognuno di questi tipi di risorse sono associate delle quote associate al tuo account. Regione AWS Per ulteriori informazioni, consulta the section called “Quote di risorse di App Runner”.

Esistono sottili differenze nella procedura di creazione di un servizio a seconda del tipo di fonte e del provider. Questo argomento descrive diverse procedure per la creazione di questi tipi di fonti in modo da poter seguire quella più adatta alla propria situazione. Per iniziare una procedura di base con un esempio di codice, vedereNozioni di base.

Prerequisiti

Prima di creare il servizio App Runner, assicurati di completare le seguenti azioni:

- Completa i passaggi di configurazione inConfigurazione.
- Assicurati che il codice sorgente dell'applicazione sia pronto. Puoi utilizzare un repository di codice in Bitbucket o un'immagine contenitore in GitHubAmazon Elastic Container Registry (Amazon ECR) Elastic Container Registry (Amazon ECR) per creare un servizio App Runner.

Creazione di un servizio

Questa sezione illustra il processo di creazione per i due tipi di servizio App Runner: basato sul codice sorgente e basato sull'immagine di un contenitore.

> **Note**
>
> Se crei un connettore VPC per il traffico in uscita per un servizio, il processo di avvio del servizio che segue presenterà una latenza unica. È possibile impostare questa configurazione per un nuovo servizio al momento della creazione o in seguito, con un aggiornamento del servizio. Per ulteriori informazioni, consulta il Latenza una tantum capitolo Networking with App Runner di questa guida.

Crea un servizio da un repository di codice

Le sezioni seguenti mostrano come creare un servizio App Runner quando la fonte è un repository di codice in GitHubo Bitbucket. Quando si utilizza un repository di codice, App Runner deve connettersi all'organizzazione o all'account del provider. Pertanto, è necessario contribuire a stabilire questa connessione. Per ulteriori informazioni sulle connessioni App Runner, vederethe section called “Connessioni”.

Quando crei il servizio, App Runner crea un'immagine Docker che contiene il codice dell'applicazione e le dipendenze. Quindi avvia un servizio che esegue un'istanza contenitore di questa immagine.

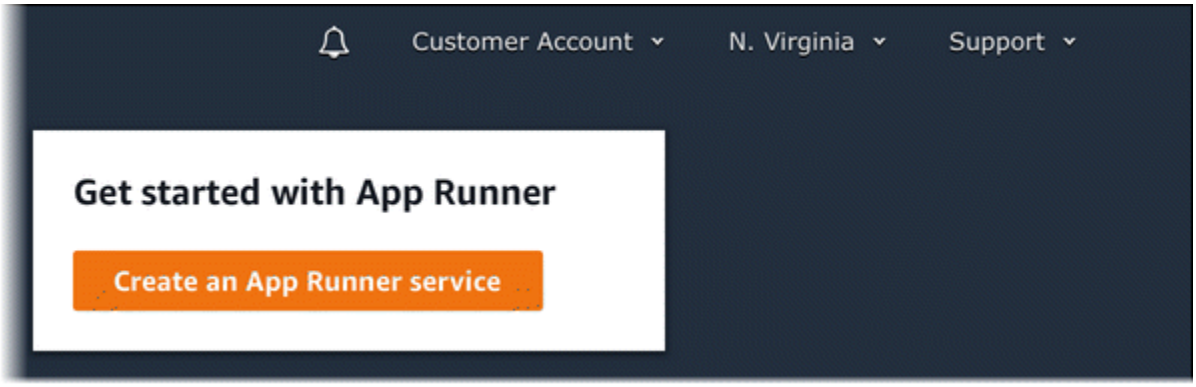
Creazione di un servizio dal codice utilizzando la console App Runner

Per creare un servizio App Runner utilizzando la console

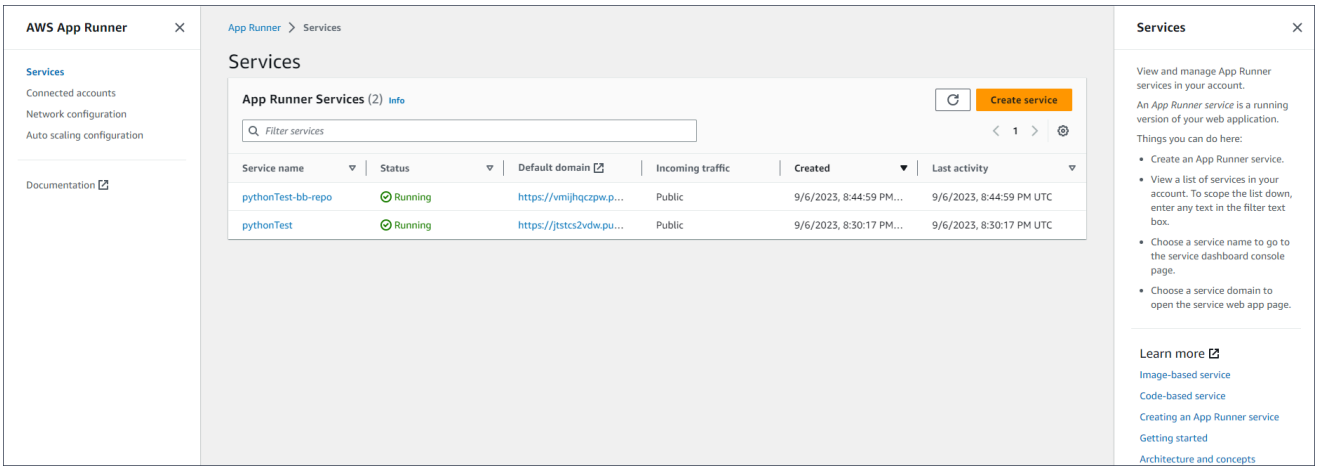
1. Configura il tuo codice sorgente.

   a. Apri la console App Runner e, nell'elenco Regioni, seleziona il tuo Regione AWS.

   b. Se Account AWS non dispone ancora di alcun servizio App Runner, viene visualizzata la home page della console. Scegli Crea un servizio App Runner.



   Se Account AWS dispone di servizi esistenti, viene visualizzata la pagina Servizi con un elenco dei servizi. Selezionare Create service (Crea servizio).



   c. Nella pagina Origine e distribuzione, nella sezione Sorgente, per Tipo di archivio, scegli Archivio del codice sorgente.

   d. Seleziona un tipo di provider. Scegli uno dei due GitHubo Bitbucket.

   e. Quindi seleziona un account o un'organizzazione per il provider che hai utilizzato in precedenza o scegli Aggiungi nuovo. Quindi, esegui la procedura di fornitura delle

credenziali del repository del codice e scegli un account o un'organizzazione a cui connetterti.

f. Per Repository, seleziona il repository che contiene il codice dell'applicazione.

g. Per Branch, seleziona il ramo che desideri distribuire.

h. Per la directory di origine, inserite la directory nel repository di origine che memorizza il codice dell'applicazione e i file di configurazione.

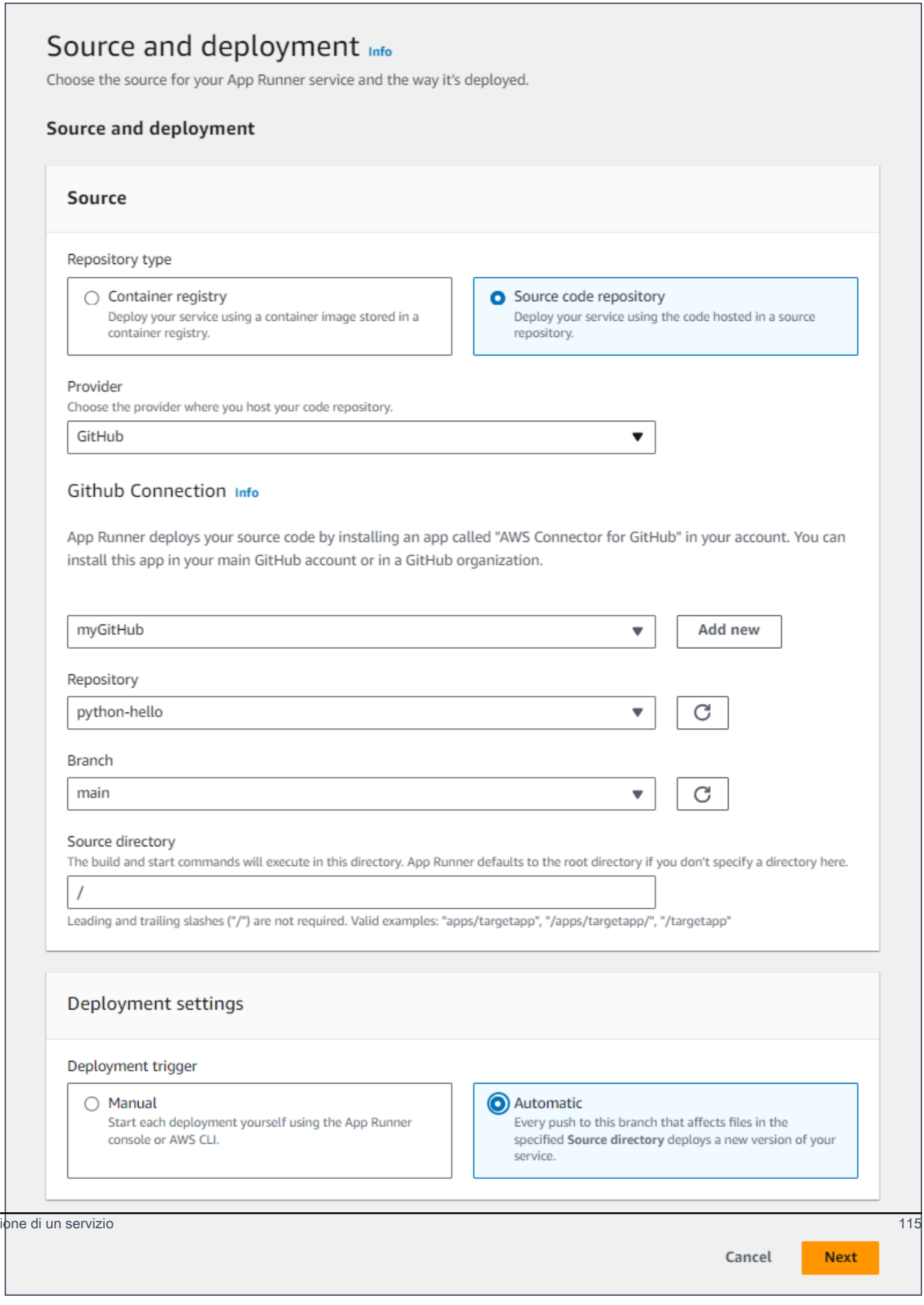
> **Note**
>
> I comandi build e start vengono eseguiti dalla directory di origine specificata. App Runner gestisce il percorso come assoluto a partire da root. Se non si specifica un valore qui, la directory predefinita è la radice del repository.

2. Configura le tue distribuzioni.

   a. Nella sezione Impostazioni di distribuzione, scegli Manuale o Automatico.

   Per ulteriori informazioni sui metodi di distribuzione, consultathe section called “Metodi di distribuzione”.

   b. Scegli Next (Successivo).


Source and deployment Info
Choose the source for your App Runner service and the way it's deployed.
Source and deployment
Source
Repository type
Container registry
Deploy your service using a container image stored in a container registry.
Source code repository
Deploy your service using the code hosted in a source repository.
Provider
Choose the provider where you host your code repository.
GitHub
Github Connection Info
App Runner deploys your source code by installing an app called "AWS Connector for GitHub" in your account. You can install this app in your main GitHub account or in a GitHub organization.
myGitHub
Add new
Repository
python-hello
Branch
main
Source directory
The build and start commands will execute in this directory. App Runner defaults to the root directory if you don't specify a directory here.
/
Leading and trailing slashes ("/") are not required. Valid examples: "apps/targetapp", "/apps/targetapp/", "/targetapp"
Deployment settings
Deployment trigger
Manual
Start each deployment yourself using the App Runner console or AWS CLI.
Automatic
Every push to this branch that affects files in the specified Source directory deploys a new version of your service.
Cancel
Next

3. Configura la build dell'applicazione.

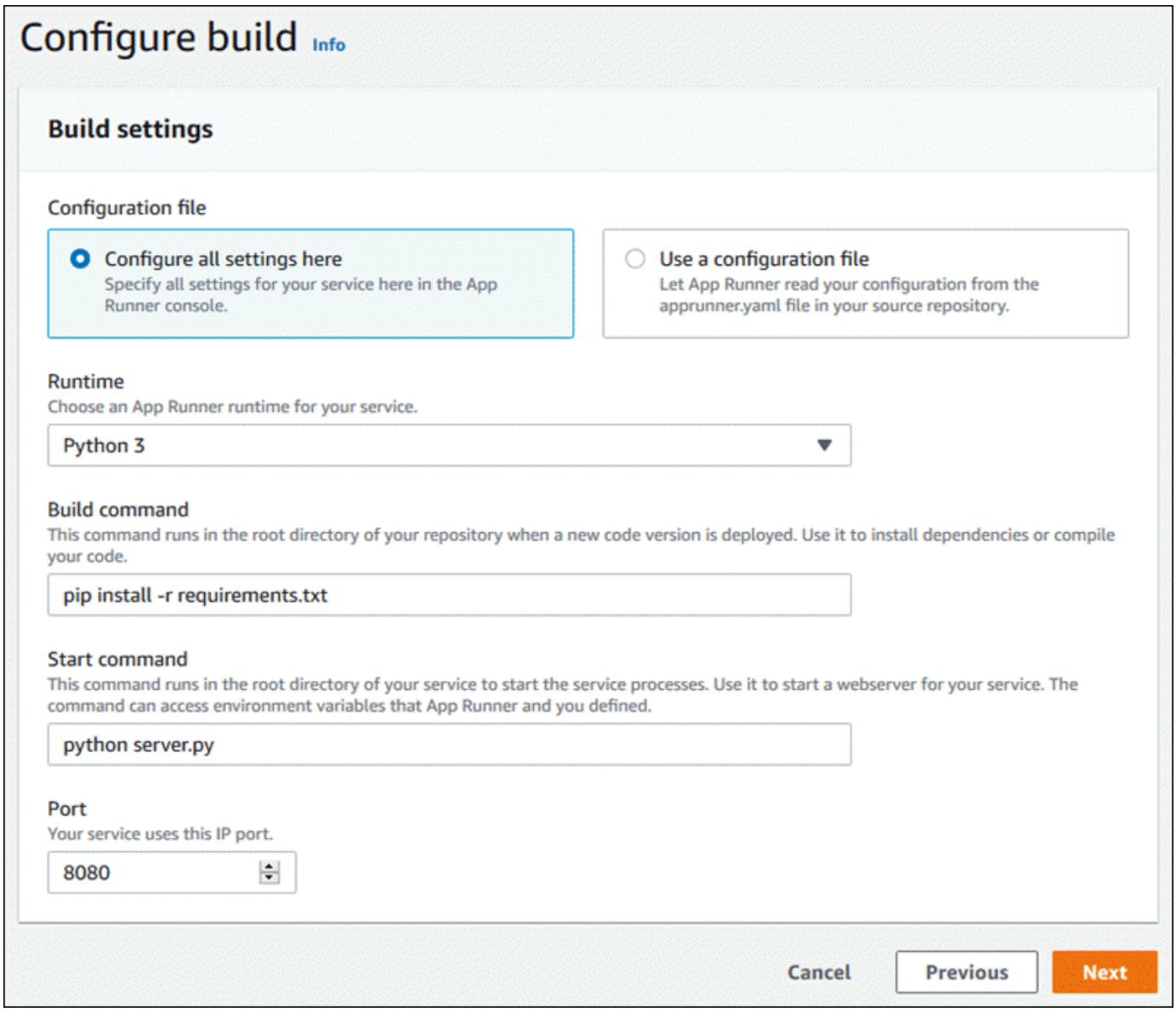
   a. Nella pagina Configura build, per File di configurazione, scegli Configura tutte le impostazioni qui se il tuo repository non contiene un file di configurazione di App Runner, oppure Usa un file di configurazione in caso affermativo.

      > **Note**
      >
      > Un file di configurazione di App Runner è un modo per mantenere la configurazione della build come parte dell'origine dell'applicazione. Quando ne fornisci uno, App Runner legge alcuni valori dal file e non ti consente di impostarli nella console.

   b. Fornisci le seguenti impostazioni di build:

      - Runtime: scegli un runtime gestito specifico per la tua applicazione.
      - Comando di compilazione: immetti un comando che crea l'applicazione dal relativo codice sorgente. Potrebbe trattarsi di uno strumento specifico della lingua o di uno script fornito con il codice.
      - Comando di avvio: immetti il comando che avvia il tuo servizio web.
      - Porta: inserisci la porta IP che il tuo servizio web ascolta.

   c. Scegli Next (Successivo).



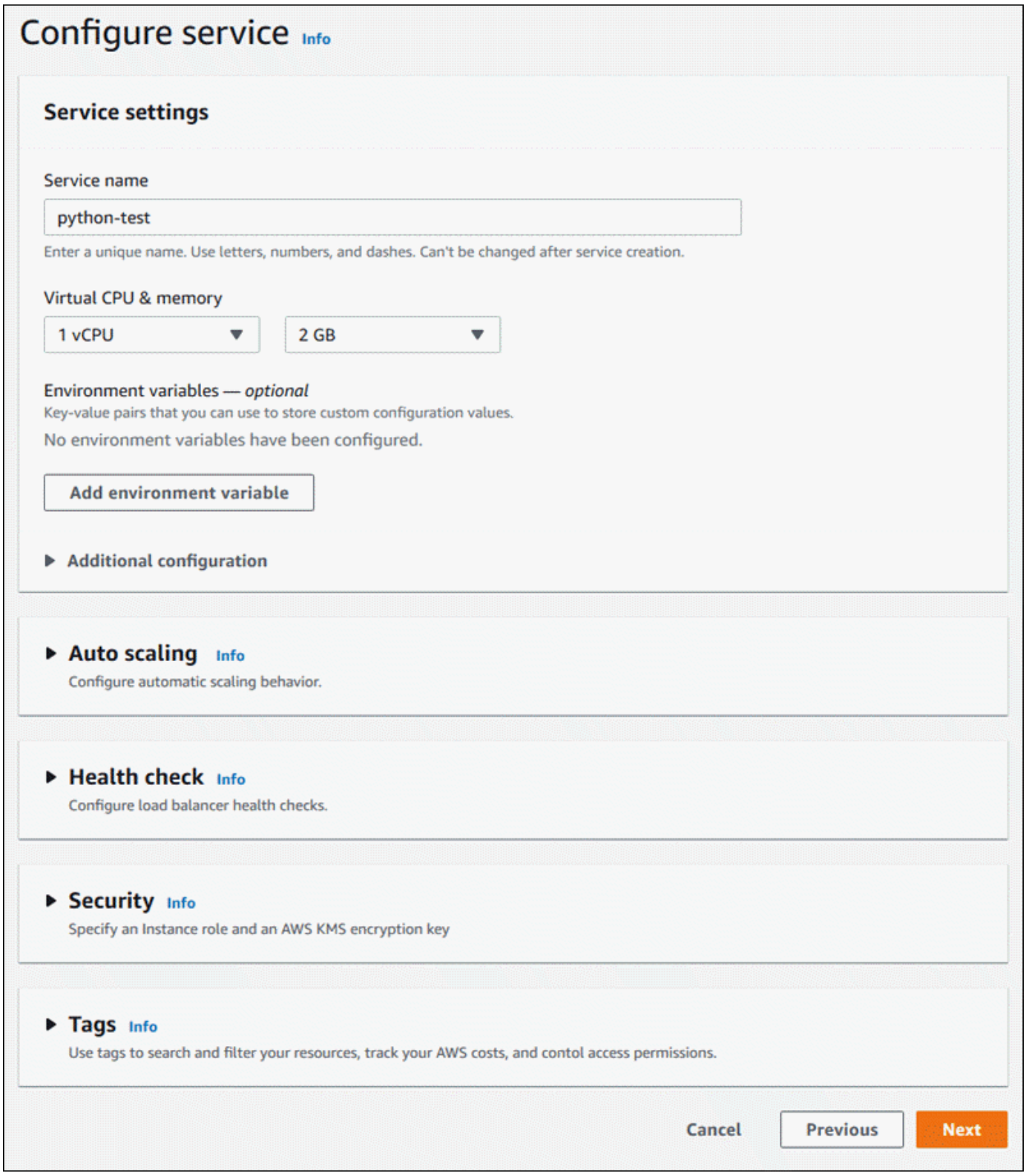
4. Configura il tuo servizio.

   a. Nella pagina Configura servizio, nella sezione Impostazioni del servizio, inserisci un nome di servizio.

> **Note**
>
> Tutte le altre impostazioni del servizio sono opzionali o hanno impostazioni predefinite fornite dalla console.

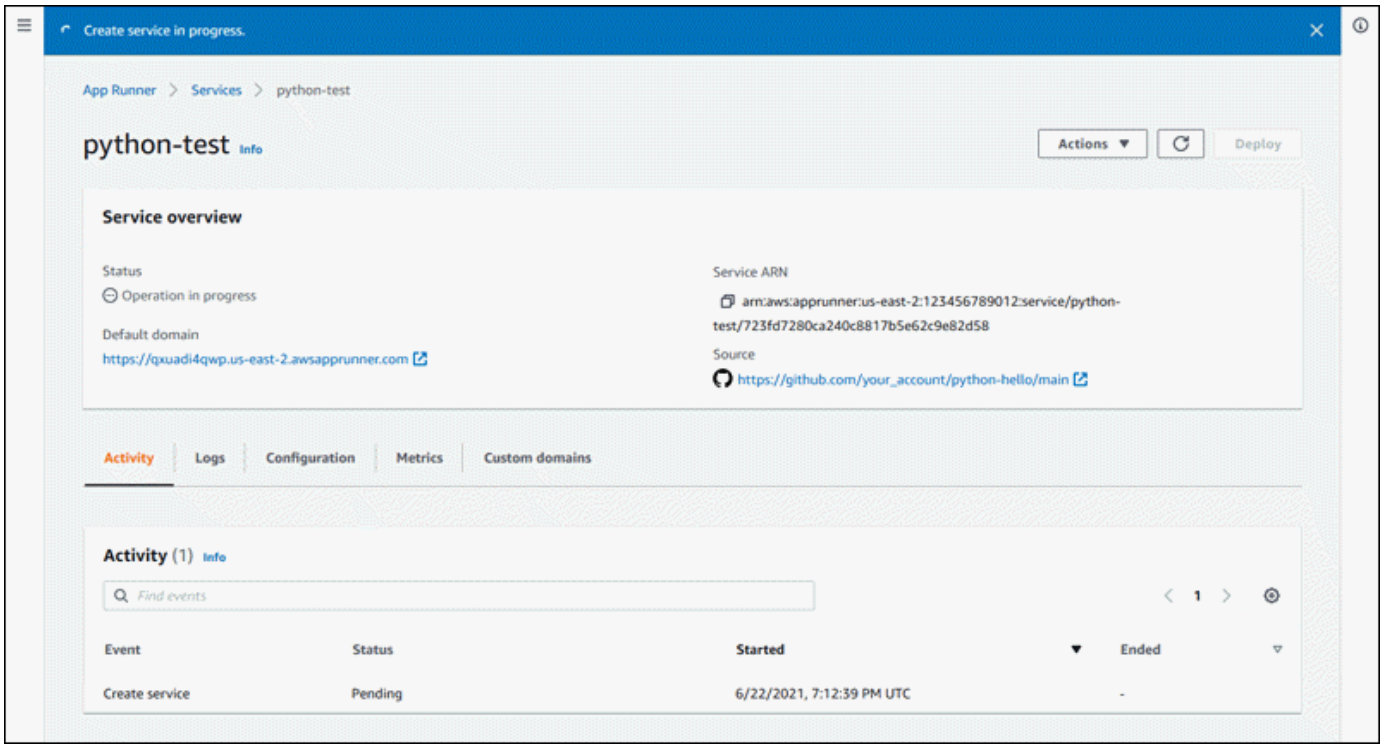
b. È possibile modificare o aggiungere altre impostazioni per soddisfare i requisiti dell'applicazione.

c. Scegli Next (Successivo).



5. Nella pagina Rivedi e crea, verifica tutti i dettagli che hai inserito, quindi scegli Crea e distribuisci.

Risultato: se il servizio viene creato correttamente, la console visualizza la dashboard del servizio con una panoramica del servizio del nuovo servizio.



6. Verifica che il servizio sia in esecuzione.

   a. Nella pagina del pannello di controllo del servizio, attendi che lo stato del servizio sia in esecuzione.

   b. Scegli il valore di dominio predefinito. È l'URL del sito Web del tuo servizio.

   c. Usa il tuo sito Web e verifica che funzioni correttamente.

Creazione di un servizio dal codice utilizzando l'API App Runner o AWS CLI

Per creare un servizio utilizzando l'API App Runner oppure AWS CLI, richiama l'azione `CreateService` API. Per ulteriori informazioni e un esempio, consulta CreateService. Se è la prima volta che crei un servizio utilizzando un'organizzazione o un account specifico per un repository di codice sorgente (GitHub o Bitbucket), inizia chiamando. CreateConnection Ciò stabilisce una connessione tra App Runner e l'organizzazione o l'account del fornitore del repository. Per ulteriori informazioni sulle connessioni App Runner, vedere. the section called “Connessioni”

Se la chiamata restituisce una risposta corretta con un oggetto Service visualizzato`"Status": "CREATING"`, il servizio inizia la creazione.

Per una chiamata di esempio, consulta Creare un servizio di repository del codice sorgente nell'AWS App Runner API Reference

Creare un servizio da un'immagine Amazon ECR

Le seguenti sezioni mostrano come creare un servizio App Runner quando l'origine è un'immagine del contenitore archiviata in Amazon ECR. Amazon ECR è un Servizio AWS. Pertanto, per creare un servizio basato su un'immagine Amazon ECR, fornisci ad App Runner un ruolo di accesso contenente le autorizzazioni di azione Amazon ECR necessarie.

> **Note**
>
> Le immagini archiviate in Amazon ECR Public sono disponibili pubblicamente. Pertanto, se l'immagine è archiviata in Amazon ECR Public, non è richiesto un ruolo di accesso.

Quando il servizio viene creato, App Runner avvia un servizio che esegue un'istanza contenitore dell'immagine fornita. In questo caso non è prevista alcuna fase di compilazione.

Per ulteriori informazioni, consulta Servizio basato su immagini.

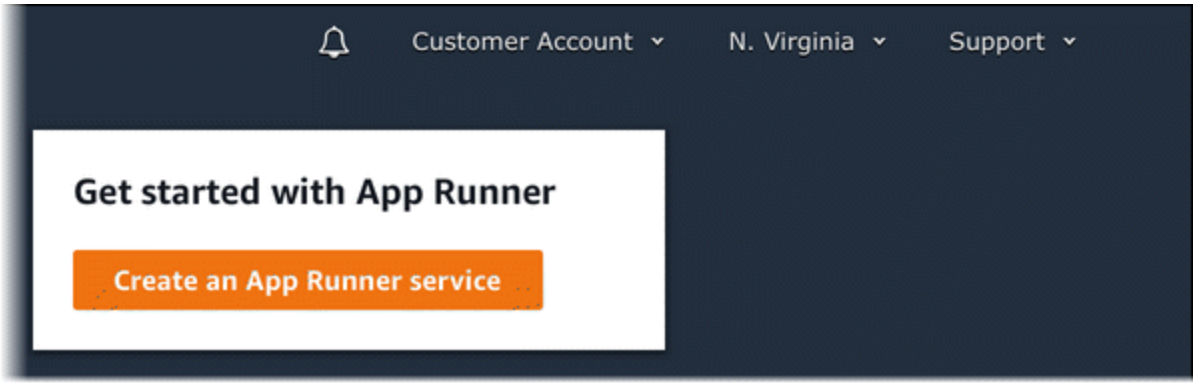
Creazione di un servizio da un'immagine utilizzando la console App Runner

Per creare un servizio App Runner utilizzando la console

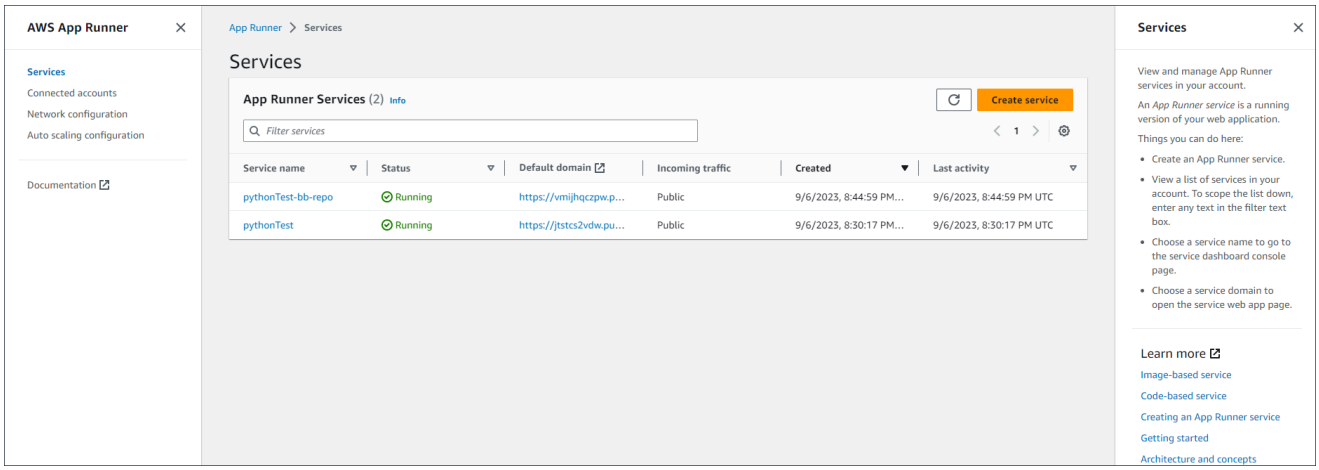
1. Configura il tuo codice sorgente.

   a. Apri la console App Runner e, nell'elenco Regioni, seleziona il tuo Regione AWS.

   b. Se Account AWS non dispone ancora di alcun servizio App Runner, viene visualizzata la home page della console. Scegli Crea un servizio App Runner.



Se Account AWS dispone di servizi esistenti, viene visualizzata la pagina Servizi con un elenco dei servizi. Selezionare Create service (Crea servizio).



c. Nella pagina Origine e distribuzione, nella sezione Origine, per Tipo di repository, scegli Container registry.

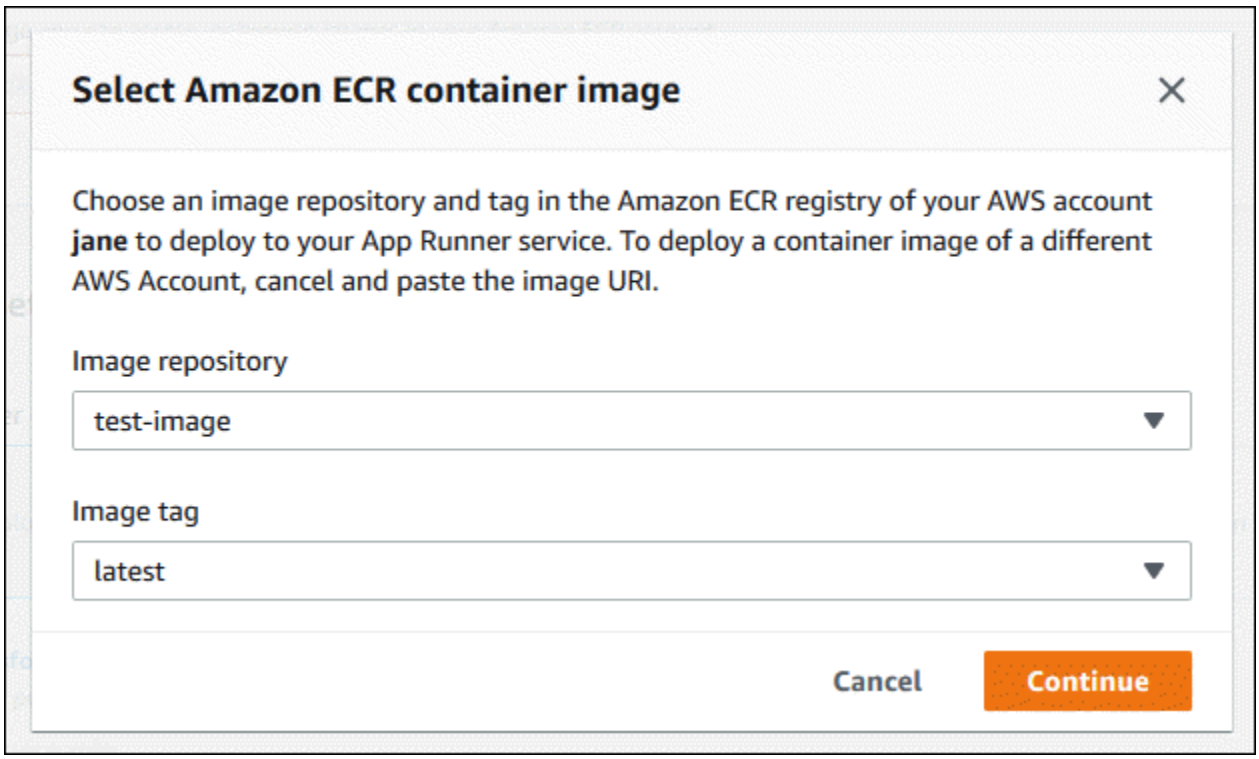
d. Per Provider, scegli il provider in cui è archiviata l'immagine:

- Amazon ECR: un'immagine privata archiviata in Amazon ECR.
- Amazon ECR Public: un'immagine leggibile pubblicamente archiviata in Amazon ECR Public.

e. Per l'URI dell'immagine del contenitore, scegli Sfoglia.

f. Nella finestra di dialogo Seleziona l'immagine del contenitore Amazon ECR, per Image repository, seleziona il repository che contiene l'immagine.

g. Per il tag Image, seleziona il tag di immagine specifico che desideri distribuire (ad esempio, il più recente), quindi scegli Continua.



2. Configura le tue distribuzioni.

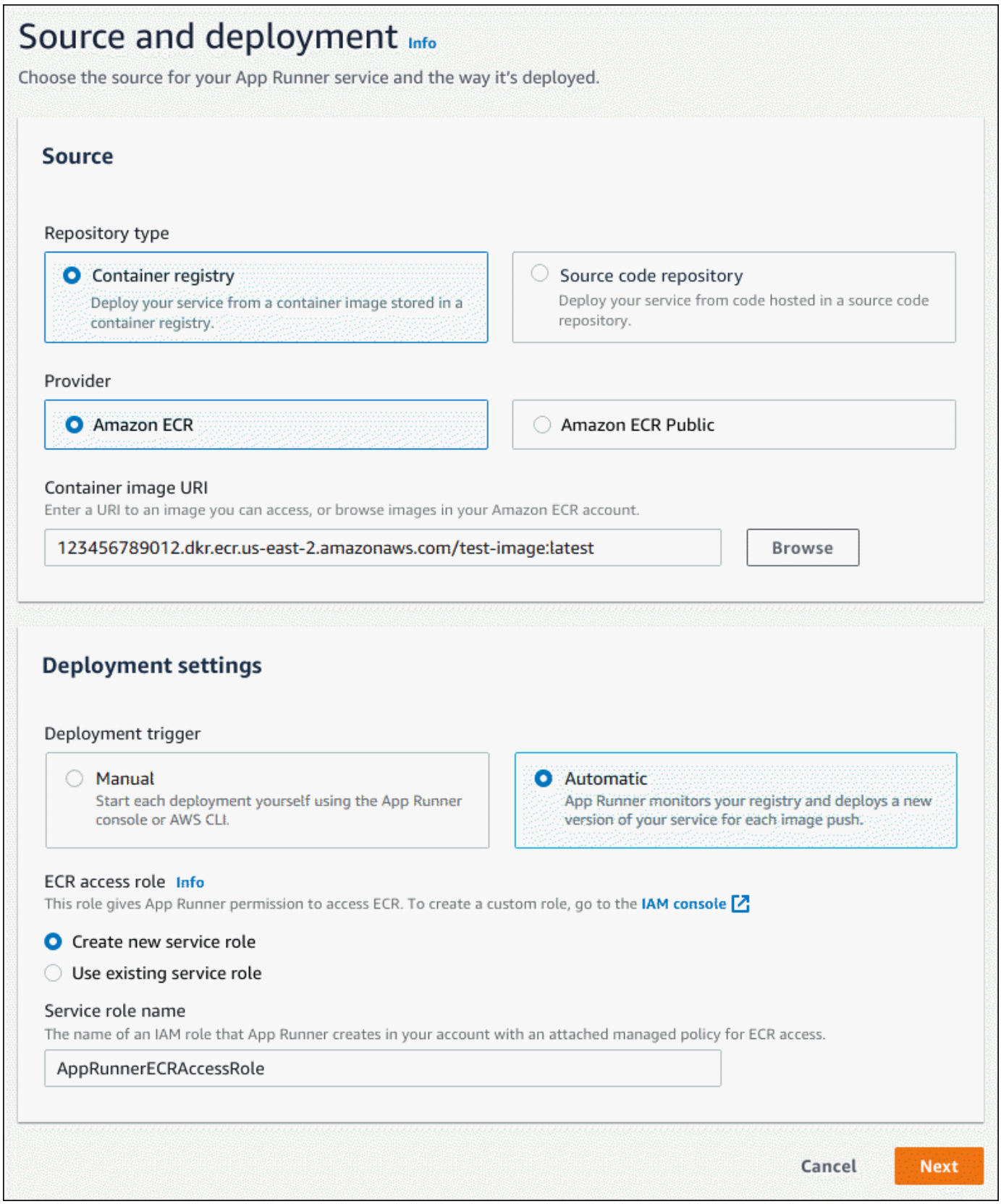
   a. Nella sezione Impostazioni di distribuzione, scegli Manuale o Automatico.

   > **Note**
   >
   > App Runner non supporta la distribuzione automatica per le immagini pubbliche di Amazon ECR e per le immagini in un repository Amazon ECR che appartiene a un AWS account diverso da quello in cui si trova il servizio.

   Per ulteriori informazioni sui metodi di distribuzione, consulta. the section called “Metodi di distribuzione”

   b. [Provider Amazon ECR] Per il ruolo di accesso ECR, scegli un ruolo di servizio esistente nel tuo account o scegli di crearne uno nuovo. Se utilizzi la distribuzione manuale, puoi anche scegliere di utilizzare il ruolo utente IAM al momento della distribuzione.

   c. Scegli Next (Successivo).


Source and deployment Info
Choose the source for your App Runner service and the way it's deployed.
Source
Repository type
Container registry
Deploy your service from a container image stored in a container registry.
Source code repository
Deploy your service from code hosted in a source code repository.
Provider
Amazon ECR
Amazon ECR Public
Container image URI
Enter a URI to an image you can access, or browse images in your Amazon ECR account.
123456789012.dkr.ecr.us-east-2.amazonaws.com/test-image:latest
Browse
Deployment settings
Deployment trigger
Manual
Start each deployment yourself using the App Runner console or AWS CLI.
Automatic
App Runner monitors your registry and deploys a new version of your service for each image push.
ECR access role Info
This role gives App Runner permission to access ECR. To create a custom role, go to the IAM console
Create new service role
Use existing service role
Service role name
The name of an IAM role that App Runner creates in your account with an attached managed policy for ECR access.
AppRunnerECRAccessRole
Cancel
Next

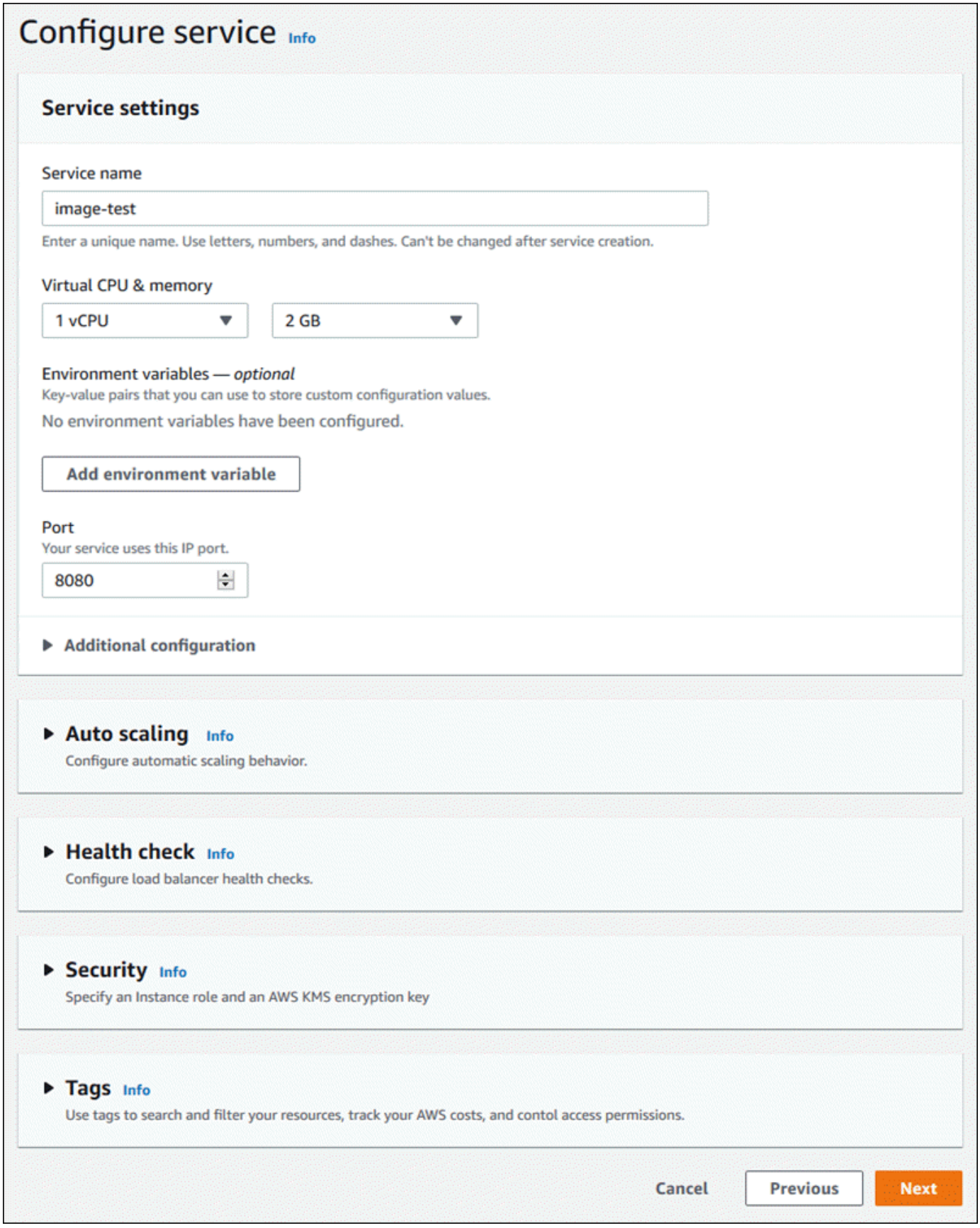
3. Configura il tuo servizio.

   a. Nella pagina Configura servizio, nella sezione Impostazioni del servizio, inserisci un nome di servizio e la porta IP che il sito Web del servizio ascolta.

      > **Note**
      >
      > Tutte le altre impostazioni del servizio sono opzionali o hanno impostazioni predefinite fornite dalla console.

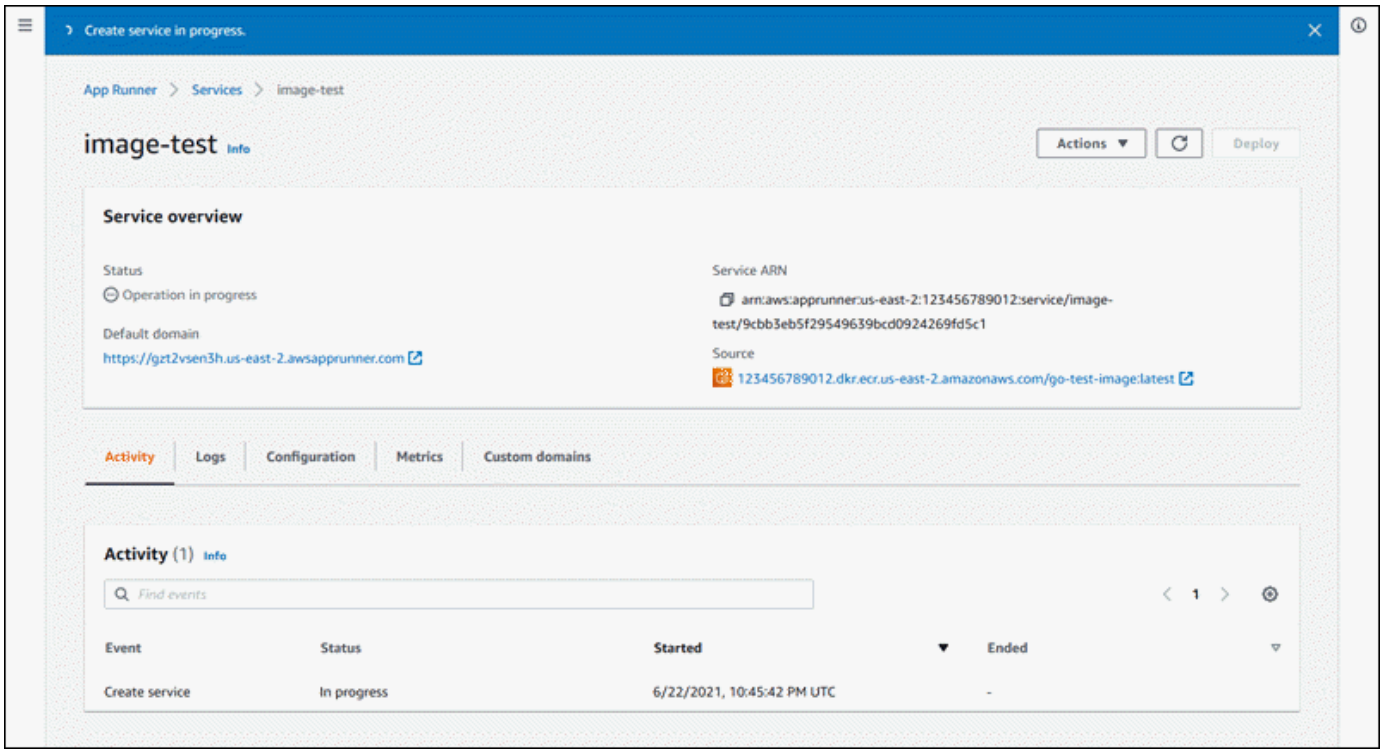
   b. (Facoltativo) Modifica o aggiungi altre impostazioni in base alle esigenze dell'applicazione.

   c. Scegli Next (Successivo).


Configure service Info
Service settings
Service name
image-test
Enter a unique name. Use letters, numbers, and dashes. Can't be changed after service creation.
Virtual CPU & memory
1 vCPU
2 GB
Environment variables — optional
Key-value pairs that you can use to store custom configuration values.
No environment variables have been configured.
Add environment variable
Port
Your service uses this IP port.
8080
Additional configuration
Auto scaling Info
Configure automatic scaling behavior.
Health check Info
Configure load balancer health checks.
Security Info
Specify an Instance role and an AWS KMS encryption key
Tags Info
Use tags to search and filter your resources, track your AWS costs, and contol access permissions.
Cancel
Previous
Next

4. Nella pagina Rivedi e crea, verifica tutti i dettagli che hai inserito, quindi scegli Crea e distribuisci.

   Risultato: se il servizio viene creato correttamente, la console mostra la dashboard del servizio, con una panoramica del servizio del nuovo servizio.



5. Verifica che il servizio sia in esecuzione.

   a. Nella pagina del pannello di controllo del servizio, attendi che lo stato del servizio sia in esecuzione.

   b. Scegli il valore di dominio predefinito. È l'URL del sito Web del tuo servizio.

   c. Usa il tuo sito Web e verifica che funzioni correttamente.

Creazione di un servizio da un'immagine utilizzando l'API App Runner o AWS CLI

Per creare un servizio utilizzando l'API App Runner oppure AWS CLI, richiama l'azione CreateServiceAPI.

La creazione del servizio inizia se la chiamata restituisce una risposta corretta con la visualizzazione `"Status": "CREATING"` di un oggetto Service.

Per una chiamata di esempio, consulta Creare un servizio di repository di immagini di origine nell'AWS App Runner API Reference

Ricostruzione di un servizio App Runner fallito

Se ricevi un errore Failed to create durante la creazione di un servizio App Runner, puoi effettuare una delle seguenti operazioni.

- Segui i passaggi indicati the section called “Impossibile creare il servizio” per identificare la causa dell'errore.
- Se trovi un errore nell'origine o nella configurazione, apporta le modifiche necessarie e poi ricostruisci il servizio.
- Se un problema temporaneo con App Runner ha causato il fallimento del servizio, ricostruisci il servizio guasto senza apportare modifiche all'origine o alla configurazione.

Puoi ricostruire il servizio guasto tramite la console App Runner o l'API App Runner oppure. AWS CLI

Ricostruzione di un servizio App Runner fallito utilizzando la console App Runner

Rebuild with updates

La creazione di un servizio può fallire per diversi motivi. Quando ciò accade, è importante identificare e correggere la causa principale del problema prima di ricostruire il servizio. Per ulteriori informazioni, consulta the section called “Impossibile creare il servizio”.

Per ricostruire un servizio guasto con aggiornamenti

1. Vai alla scheda Configurazioni nella pagina del servizio e scegli Modifica.

   La pagina apre un pannello di riepilogo che mostra un elenco di tutti gli aggiornamenti.
2. Apporta le modifiche richieste e rivedile nel pannello di riepilogo.
3. Scegli Salva e ricostruisci.

   Puoi monitorare i progressi nella scheda Registri della pagina del servizio.

Rebuild without updates

Se un problema temporaneo causa un errore nella creazione del servizio, puoi ricostruirlo senza modificarne l'origine o le impostazioni di configurazione.

Per ricostruire un servizio guasto senza aggiornamenti

- Scegli Rebuild nell'angolo in alto a destra della pagina del servizio.

  Puoi monitorare i progressi nella scheda Registri della pagina di servizio.
- Se il servizio non riesce a creare nuovamente, segui le istruzioni per la risoluzione dei problemi riportate inthe section called “Impossibile creare il servizio”. Apporta le modifiche necessarie e quindi ricostruisci il servizio.

Ricostruzione del servizio App Runner fallito utilizzando l'API App Runner o AWS CLI

Rebuild with updates

Per ricostruire un servizio guasto:

1. Segui le istruzioni riportate the section called “Impossibile creare il servizio” per trovare la causa dell'errore.
2. Apporta le modifiche necessarie al ramo o all'immagine del repository di origine o alla configurazione che ha causato l'errore.
3. Ricostruisci richiamando l'azione dell'UpdateServiceAPI con il nuovo repository di codice sorgente o i parametri del repository di immagini di origine. App Runner recupera il commit più recente dal repository del codice sorgente.

Example Ricostruzione con aggiornamenti

Nell'esempio seguente viene aggiornata la configurazione di origine di un servizio basato su immagini. Il valore di `Port` viene modificato in. `80`

Aggiornamento del `input.json` file per il servizio App Runner basato su immagini

```
{
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
  "SourceConfiguration": {
    "ImageRepository": {
      "ImageConfiguration": {
        "Port": "80"
```

```
      }
    }
  }
}
```

Chiamata dell'azione `UpdateService` API.

```
aws apprunner update-service
--cli-input-json file://input.json
```

Rebuild without updates

Per ricostruire il servizio guasto utilizzando l'API App Runner oppure AWS CLI, richiama l'azione UpdateServiceAPI senza apportare modifiche all'origine o alla configurazione del servizio. Scegli di ricostruire senza effettuare aggiornamenti solo se la creazione del servizio non è riuscita a causa di un problema temporaneo con App Runner.

Distribuzione di una nuova versione dell'applicazione su App Runner

Quando crei un servizio in AWS App Runner, configuri un'origine dell'applicazione, un'immagine del contenitore o un repository di origine. App Runner fornisce risorse per eseguire il servizio e distribuisce su di esse l'applicazione.

Questo argomento descrive i modi per ridistribuire l'origine dell'applicazione sul servizio App Runner quando diventa disponibile una nuova versione. Può trattarsi di una nuova versione dell'immagine nell'archivio delle immagini o di un nuovo commit nell'archivio del codice. App Runner offre due metodi per l'implementazione su un servizio: automatico e manuale.

Metodi di distribuzione

App Runner offre i seguenti metodi per controllare come vengono avviate le distribuzioni delle applicazioni.

Distribuzione automatica

Utilizza la distribuzione automatica quando desideri un comportamento di integrazione e distribuzione continue (CI/CD) per il tuo servizio. App Runner monitora l'archivio di immagini o codice per rilevare eventuali modifiche.

Archivio di immagini: ogni volta che inserisci una nuova versione di immagine nel tuo archivio di immagini o un nuovo commit nel tuo repository di codice, App Runner la distribuisce automaticamente al tuo servizio senza ulteriori azioni da parte tua.

Archivio di codice: ogni volta che inserisci un nuovo commit nel tuo repository di codice che apporta modifiche nella directory dei sorgenti, App Runner distribuisce l'intero repository. Poiché solo le modifiche nella directory di origine attivano una distribuzione automatica, è importante capire in che modo la posizione della directory di origine influisce sull'ambito di una distribuzione automatizzata.

- Directory di primo livello (radice del repository): questo è il valore predefinito impostato per la directory di origine quando si crea un servizio. Se la directory di origine è impostata su questo valore, significa che l'intero repository si trova all'interno della directory dei sorgenti. Quindi, in questo caso, tutti i commit che invii al repository di origine attiveranno una distribuzione.
- Qualsiasi percorso di directory che non sia la radice del repository (impostazione non predefinita): poiché solo le modifiche inserite nella directory di origine attivano una distribuzione automatica, qualsiasi modifica inserita nel repository che non si trova nella directory di origine non attiverà una distribuzione automatica. Pertanto, è necessario utilizzare una distribuzione manuale per distribuire le modifiche che si inseriscono all'esterno della directory di origine.

> **Note**
>
> App Runner non supporta la distribuzione automatica per le immagini pubbliche di Amazon ECR e per le immagini in un repository Amazon ECR che appartiene a un AWS account diverso da quello in cui si trova il servizio.

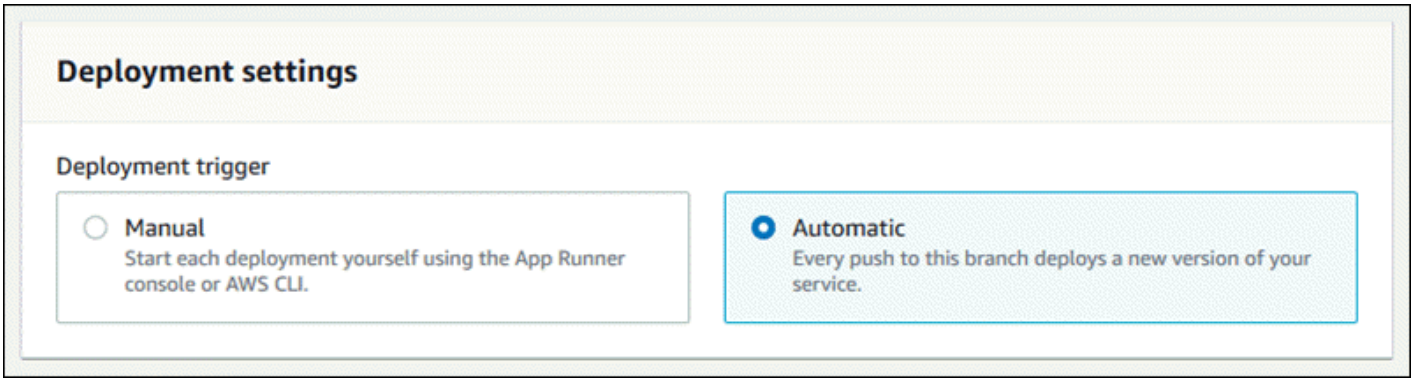
Distribuzione manuale

Utilizza la distribuzione manuale quando desideri avviare in modo esplicito ogni distribuzione del tuo servizio. Si avvia una distribuzione se il repository configurato per il servizio ha una nuova versione che si desidera distribuire. Per ulteriori informazioni, consulta the section called “Distribuzione manuale”.

> **Note**
>
> Quando esegui una distribuzione manuale, App Runner distribuisce il codice sorgente dall'archivio completo.

È possibile configurare il metodo di distribuzione per il servizio nei seguenti modi:

- Console: per un nuovo servizio che stai creando o per un servizio esistente, nella sezione Impostazioni di distribuzione della pagina di configurazione di origine e distribuzione, scegli Manuale o Automatico.



- API o AWS CLI: in una chiamata all'UpdateServiceazione CreateServiceo, imposta il `AutoDeploymentsEnabled` membro del SourceConfigurationparametro per la distribuzione manuale o `False` `True` per la distribuzione automatica.

> Confronto tra distribuzioni automatiche e manuali
>
> Sia le implementazioni automatiche che quelle manuali producono lo stesso risultato: entrambi i metodi distribuiscono l'intero repository.
> La differenza tra i due metodi è il meccanismo di attivazione:
>
> - Le distribuzioni manuali vengono attivate da una distribuzione dalla console, da una chiamata all'API App Runner o da una chiamata all' AWS CLI API App Runner. La Distribuzione manuale sezione che segue fornisce le relative procedure.
> - Le distribuzioni automatiche vengono attivate da una modifica del contenuto della directory di origine.

Distribuzione manuale

Con la distribuzione manuale, è necessario avviare esplicitamente ogni implementazione del servizio. Quando hai una nuova versione dell'immagine o del codice dell'applicazione pronta per la

distribuzione, puoi fare riferimento alle seguenti sezioni per scoprire come eseguire una distribuzione utilizzando la console e l'API.

> **Note**
>
> Quando esegui una distribuzione manuale, App Runner distribuisce il codice sorgente dall'archivio completo.

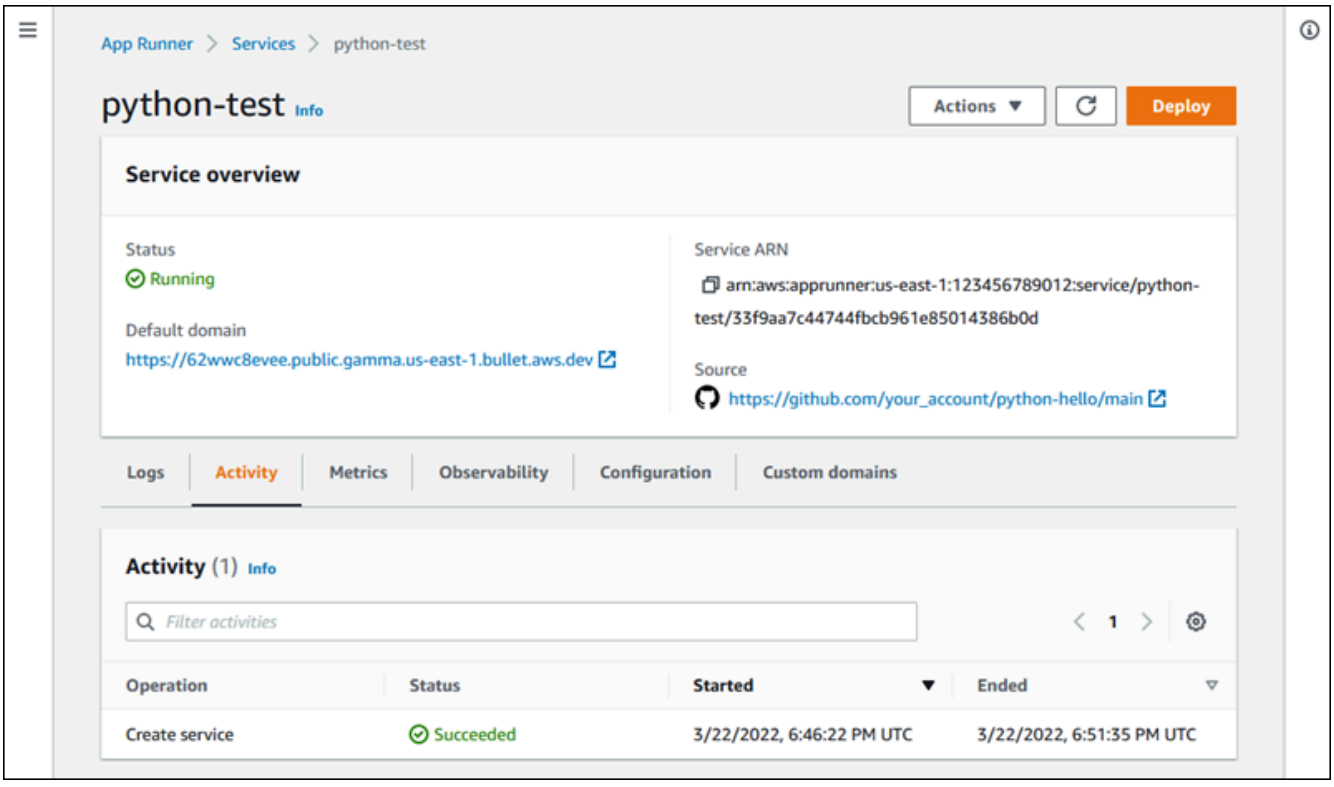
Distribuisci una versione dell'applicazione utilizzando uno dei seguenti metodi:

App Runner console

Per eseguire la distribuzione utilizzando la console App Runner

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Nel pannello di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

   La console mostra la dashboard del servizio con una panoramica del servizio.



3. Seleziona Deploy (Implementa).

Risultato: inizia la distribuzione della nuova versione. Nella pagina del dashboard del servizio, lo stato del servizio cambia in Operazione in corso.

4. Attendi il termine della distribuzione. Nella pagina del dashboard del servizio, lo stato del servizio dovrebbe tornare in esecuzione.
5. Per verificare che l'implementazione abbia esito positivo, nella pagina del pannello di controllo del servizio, scegli il valore di dominio predefinito, che è l'URL del sito Web del servizio. Ispeziona o interagisci con la tua applicazione web e verifica la modifica della versione.

> **Note**
>
> Per aumentare la sicurezza delle applicazioni App Runner, il dominio*.awsapprunner.com è registrato nella Public Suffix List (PSL). Per una maggiore sicurezza, ti consigliamo di utilizzare i cookie con un `__Host-` prefisso se hai bisogno di impostare cookie sensibili nel nome di dominio predefinito per le tue applicazioni App Runner. Questa pratica ti aiuterà a difendere il tuo dominio dai tentativi CSRF (cross-site request forgery). Per ulteriori informazioni, consulta la pagina Impostazione cookie nella pagina Mozilla Developer Network.

App Runner API or AWS CLI

Per eseguire la distribuzione utilizzando l'API App Runner oppure AWS CLI, richiama l'azione API. StartDeployment L'unico parametro da passare è l'ARN del servizio. Hai già configurato la posizione di origine dell'applicazione quando hai creato il servizio e App Runner può trovare la nuova versione. La distribuzione inizia se la chiamata restituisce una risposta corretta.

Configurazione di un servizio App Runner

Quando si crea un AWS App Runner servizio, si impostano diversi valori di configurazione. È possibile modificare alcune di queste impostazioni di configurazione dopo aver creato il servizio. Altre impostazioni possono essere applicate solo durante la creazione del servizio e non possono essere modificate in seguito. Questo argomento descrive la configurazione del servizio utilizzando l'API App Runner, la console App Runner e un file di configurazione App Runner.

Argomenti

- Configura il tuo servizio utilizzando l'API App Runner o AWS CLI
- Configura il tuo servizio utilizzando la console App Runner
- Configura il tuo servizio utilizzando un file di configurazione di App Runner
- Configurazione dell'osservabilità per il servizio
- Configurazione delle impostazioni del servizio utilizzando risorse condivisibili
- Configurazione dei controlli sanitari per il servizio

Configura il tuo servizio utilizzando l'API App Runner o AWS CLI

L'API definisce quali impostazioni possono essere modificate dopo la creazione del servizio. L'elenco seguente illustra le azioni, i tipi e le limitazioni pertinenti.

- UpdateServiceazione: può essere richiamato dopo la creazione per aggiornare alcune impostazioni di configurazione.
  - Può essere aggiornato: è possibile aggiornare le impostazioni nei `HealthCheckConfiguration` parametri `SourceConfiguration`, e. Tuttavia, in`InstanceConfiguration`, non è possibile cambiare il tipo di sorgente da codice a immagine o viceversa. È necessario fornire lo stesso parametro di repository fornito al momento della creazione del servizio. O è o. `CodeRepository` `ImageRepository`

    È inoltre possibile aggiornare le seguenti risorse ARNs di configurazione separate associate al servizio:

    - `AutoScalingConfigurationArn`
    - `VpcConnectorArn`
  - Non può essere aggiornato: non è possibile modificare i `EncryptionConfiguration` parametri `ServiceName` and disponibili nell'CreateServiceazione. Non possono essere modificati dopo la creazione. L'UpdateServiceazione non include questi parametri.
  - API o file: puoi impostare il `ConfigurationSource` parametro del CodeConfigurationtipo (utilizzato per gli archivi di codice sorgente come parte di`SourceConfiguration`) su. `Repository` In questo caso, App Runner ignora le impostazioni di configurazione presenti nel repository `CodeConfigurationValues` e le legge da un file di configurazione presente nel repository. Se si imposta su `ConfigurationSource`API, App Runner ottiene tutte le impostazioni di configurazione dalla chiamata API e ignora il file di configurazione, anche se ne esiste uno.

- TagResourceazione: può essere richiamata dopo la creazione del servizio per aggiungere tag al servizio o aggiornare i valori dei tag esistenti.
- UntagResourceazione: può essere richiamato dopo la creazione del servizio per rimuovere i tag dal servizio.

> **Note**
>
> Se crei un connettore VPC per il traffico in uscita per un servizio, il processo di avvio del servizio che segue presenterà una latenza unica. È possibile impostare questa configurazione per un nuovo servizio al momento della creazione o in seguito, con un aggiornamento del servizio. Per ulteriori informazioni, consulta il Latenza una tantum capitolo Networking with App Runner di questa guida.

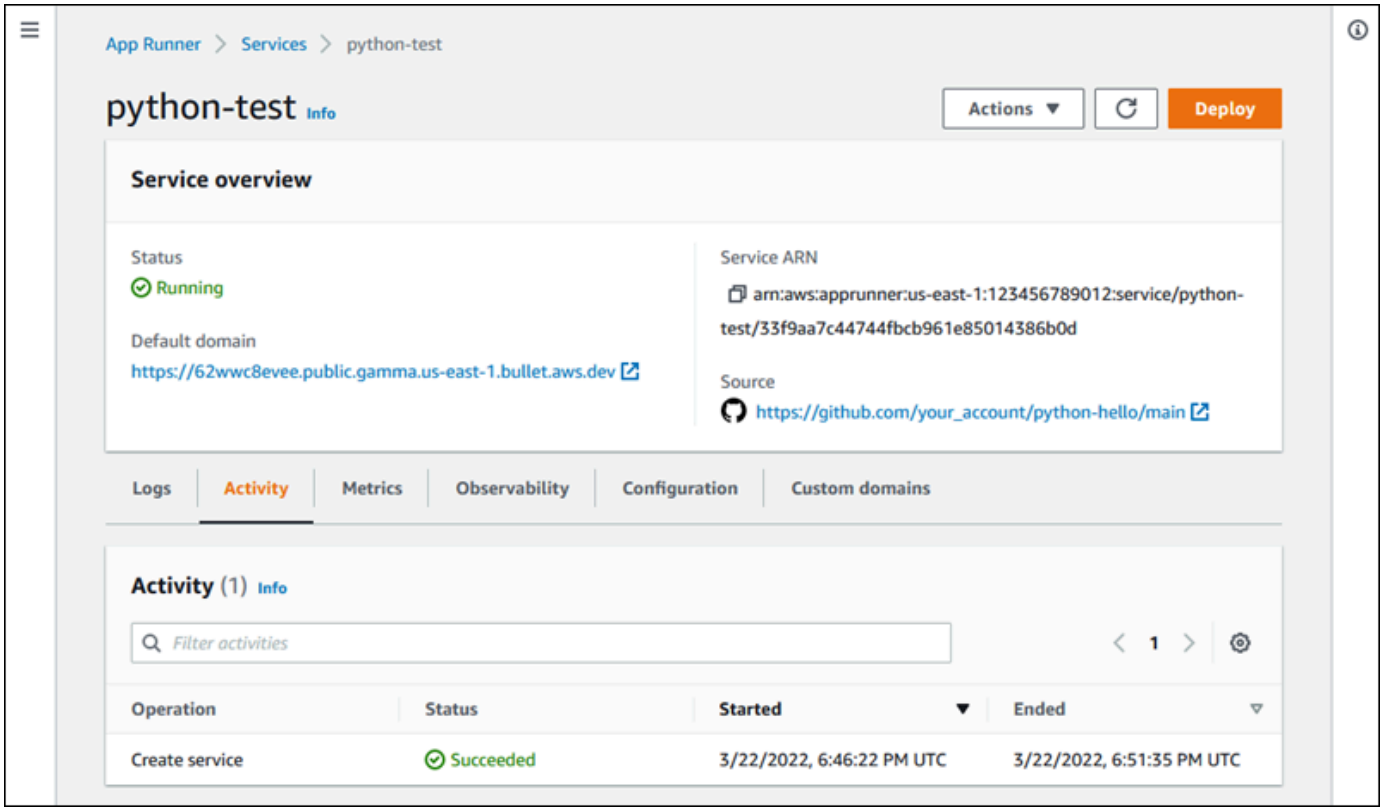
Configura il tuo servizio utilizzando la console App Runner

La console utilizza l'API App Runner per applicare gli aggiornamenti di configurazione. Le regole di aggiornamento imposte dall'API, come definito nella sezione precedente, determinano cosa è possibile configurare utilizzando la console. Alcune impostazioni che erano disponibili durante la creazione del servizio non possono essere modificate in seguito. Inoltre, se decidi di utilizzare un file di configurazione, le impostazioni aggiuntive vengono nascoste nella console e App Runner le legge dal file.

Per configurare il servizio

1. Apri la console App Runner e, nell'elenco Regioni, seleziona il tuo Regione AWS.
2. Nel pannello di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

   La console mostra la dashboard del servizio con una panoramica del servizio.



3. Nella pagina del dashboard del servizio, scegli la scheda Configurazione.

   Risultato: la console mostra le impostazioni di configurazione correnti del servizio in diverse sezioni: Origine e distribuzione, Configure build e Configure service.

4. Per aggiornare le impostazioni in qualsiasi categoria, scegli Modifica.

5. Nella pagina di modifica della configurazione, apporta le modifiche desiderate, quindi scegli Salva modifiche.

> **Note**
>
> Se crei un connettore VPC per il traffico in uscita per un servizio, il processo di avvio del servizio che segue presenterà una latenza unica. È possibile impostare questa configurazione per un nuovo servizio al momento della creazione o in seguito, con un aggiornamento del servizio. Per ulteriori informazioni, consulta il Latenza una tantum capitolo Networking with App Runner di questa guida.

Configura il tuo servizio utilizzando un file di configurazione di App Runner

Quando crei o aggiorni un servizio App Runner, puoi indicare ad App Runner di leggere alcune impostazioni di configurazione da un file di configurazione fornito come parte del tuo repository di origine. In questo modo, puoi gestire le impostazioni relative al codice sorgente sotto il controllo del codice sorgente, insieme al codice stesso. Il file di configurazione fornisce anche alcune impostazioni avanzate che non è possibile configurare utilizzando la console o l'API. Per ulteriori informazioni, consulta File di configurazione di App Runner.

> **Note**
>
> Se crei un connettore VPC per il traffico in uscita per un servizio, il processo di avvio del servizio che segue presenterà una latenza unica. È possibile impostare questa configurazione per un nuovo servizio al momento della creazione o in seguito, con un aggiornamento del servizio. Per ulteriori informazioni, consulta il Latenza una tantum capitolo Networking with App Runner di questa guida.

Configurazione dell'osservabilità per il servizio

AWS App Runner si integra con diversi AWS servizi per fornirti un'ampia suite di strumenti di osservabilità per il tuo servizio App Runner. Per ulteriori informazioni, consulta Osservabilità.

App Runner supporta l'attivazione di alcune funzionalità di osservabilità e la configurazione del loro comportamento utilizzando una risorsa condivisibile chiamata. ObservabilityConfiguration È possibile fornire una risorsa di configurazione dell'osservabilità quando si crea o si aggiorna un servizio. La console App Runner ne crea una per te quando crei un nuovo servizio App Runner. La fornitura di una configurazione di osservabilità è facoltativa. Se non ne fornisci una, App Runner fornisce una configurazione di osservabilità predefinita.

È possibile condividere una singola configurazione di osservabilità tra più servizi App Runner per garantire che abbiano lo stesso comportamento di osservabilità. Per ulteriori informazioni, consulta the section called “Risorse di configurazione”.

È possibile configurare le seguenti funzionalità di osservabilità utilizzando configurazioni di osservabilità:

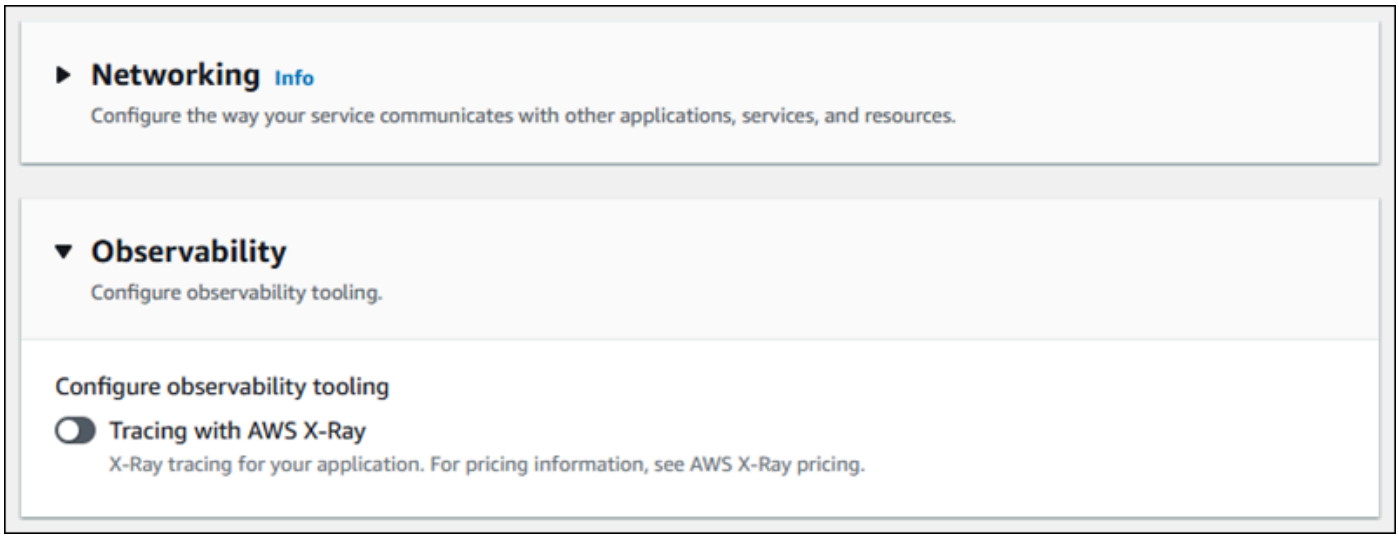
- Configurazione di tracciamento: impostazioni per tracciare le richieste servite dall'applicazione e le chiamate a valle effettuate dall'applicazione. Per ulteriori informazioni sul tracciamento, consulta the section called “Tracciamento (X-Ray)”.

Gestisci l'osservabilità

Gestisci l'osservabilità dei tuoi servizi App Runner utilizzando uno dei seguenti metodi:

App Runner console

Quando crei un servizio utilizzando la console App Runner o quando ne aggiorni la configurazione in un secondo momento, puoi configurare le funzionalità di osservabilità per il tuo servizio. Cerca la sezione sulla configurazione dell'osservabilità nella pagina della console.



App Runner API or AWS CLI

Quando si richiamano le azioni dell'API CreateServiceo UpdateServicedell'App Runner, è possibile utilizzare l'oggetto `ObservabilityConfiguration` parametro per abilitare le funzionalità di osservabilità e specificare una risorsa di configurazione dell'osservabilità per il servizio.

Utilizza le seguenti azioni dell'API App Runner per gestire le risorse di configurazione dell'osservabilità.

- CreateObservabilityConfiguration— Crea una nuova configurazione di osservabilità o una revisione di una configurazione esistente.
- ListObservabilityConfigurations— Restituisce un elenco delle configurazioni di osservabilità associate all'utente Account AWS, con informazioni di riepilogo.

- DescribeObservabilityConfiguration— Restituisce una descrizione completa di una configurazione di osservabilità.
- DeleteObservabilityConfiguration— Elimina una configurazione di osservabilità. È possibile eliminare una revisione specifica o l'ultima revisione attiva. Potrebbe essere necessario eliminare le configurazioni di osservabilità non necessarie se si raggiunge la quota di configurazione di osservabilità prevista per la propria. Account AWS

Configurazione delle impostazioni del servizio utilizzando risorse condivisibili

Per alcune funzionalità, è opportuno condividere la configurazione tra AWS App Runner i servizi. Ad esempio, potresti volere che un set di servizi abbia lo stesso comportamento di ridimensionamento automatico. Oppure potresti volere impostazioni di osservabilità identiche per tutti i tuoi servizi. App Runner consente di condividere le impostazioni utilizzando risorse condivisibili separate. Crei una risorsa che definisce un set di impostazioni di configurazione per una funzionalità, quindi fornisci l'Amazon Resource Name (ARN) di questa risorsa di configurazione a uno o più servizi App Runner.

App Runner implementa risorse di configurazione condivisibili per le seguenti funzionalità:

- Dimensionamento automatico
- Osservabilità
- Accesso VPC

La pagina del documento per ciascuna di queste funzionalità fornisce informazioni sulle impostazioni disponibili e sulle procedure di gestione.

Le funzionalità che utilizzano risorse di configurazione separate condividono alcune caratteristiche e considerazioni di progettazione.

- Revisioni: alcune risorse di configurazione possono avere delle revisioni. La scalabilità automatica e l'osservabilità sono esempi di due risorse di configurazione che utilizzano le revisioni. In questi casi, ogni configurazione ha un nome e una revisione numerica. Più revisioni di una configurazione hanno lo stesso nome e numeri di revisione diversi. È possibile utilizzare nomi di configurazione diversi per scenari diversi. Per ogni nome, puoi aggiungere più revisioni per ottimizzare le impostazioni per uno scenario specifico.

La prima configurazione creata con un nome ottiene il numero di revisione 1. Le configurazioni successive con lo stesso nome ottengono numeri di revisione consecutivi (a partire da 2). È possibile associare il servizio App Runner a una revisione di configurazione specifica o all'ultima revisione della configurazione.

- Condiviso: puoi condividere una singola risorsa di configurazione tra più servizi App Runner. Ciò è utile se si desidera mantenere configurazioni identiche tra questi servizi. In particolare, se le tue risorse supportano le revisioni, puoi configurare più servizi per utilizzare la revisione più recente di una configurazione. È possibile farlo specificando solo il nome della configurazione, ma non una revisione. Tutti i servizi configurati in questo modo ricevono aggiornamenti di configurazione quando si aggiorna il servizio. Per ulteriori informazioni sulle modifiche alla configurazione, vederethe section called “Configurazione”.
- Gestione delle risorse: puoi utilizzare App Runner per creare ed eliminare configurazioni. Non puoi aggiornare direttamente una configurazione. Invece, per le risorse che supportano le revisioni, puoi creare una nuova revisione di un nome di configurazione esistente per aggiornare efficacemente la configurazione.

> **Note**
>
> Per la scalabilità automatica, puoi creare configurazioni e revisioni multiple sia con la console App Runner che con l'API App Runner. Sia la console App Runner che l'API App Runner possono anche eliminare configurazioni e revisioni. Per ulteriori dettagli, consulta Gestione del ridimensionamento automatico di App Runner.
> Per altri tipi di configurazione, come le configurazioni di osservabilità, puoi creare una configurazione con una sola revisione solo con la console App Runner. Per creare più revisioni ed eliminare le configurazioni, devi utilizzare l'API App Runner.

- Quota di risorse: sono previste quote prestabilite per il numero di nomi di configurazione e di revisioni univoci che è possibile avere per le risorse di configurazione in ciascuna di esse. Regione AWS Se raggiungi queste quote, devi eliminare un nome di configurazione o almeno alcune delle relative revisioni prima di poterne creare altre. Per le revisioni delle configurazioni con ridimensionamento automatico, puoi utilizzare la console App Runner o l'API App Runner per eliminarle. Per ulteriori dettagli, consulta Gestione del ridimensionamento automatico di App Runner. È necessario utilizzare l'API App Runner per eliminare altre risorse. Per ulteriori informazioni sulle quote, consulta the section called “Quote di risorse di App Runner”.
- Nessun costo in termini di risorse: non sono previsti costi aggiuntivi per la creazione di una risorsa di configurazione. Potresti incorrere in costi per la funzionalità stessa (ad esempio, ti vengono

addebitati i AWS X-Ray costi normali quando attivi il tracciamento X-Ray), ma non per la risorsa di configurazione App Runner che configura la funzionalità per il tuo servizio App Runner.

Configurazione dei controlli sanitari per il servizio

AWS App Runner monitora lo stato del servizio eseguendo controlli sanitari. Il protocollo predefinito per il controllo dello stato di salute è TCP. App Runner esegue il ping del dominio assegnato al servizio. In alternativa, puoi impostare il protocollo di controllo dello stato di salute su HTTP. App Runner invia richieste HTTP per il controllo dello stato di salute all'applicazione web.

È possibile configurare alcune impostazioni relative ai controlli sanitari. La tabella seguente descrive le impostazioni dei controlli sanitari e i relativi valori predefiniti.

Impostazione	Descrizione	Impostazione predefinita
Protocollo	Il protocollo IP utilizzato da App Runner per eseguire i controlli dell'integrità per il servizio. Se imposti il protocollo su`TCP`, App Runner esegue il ping del dominio predefinito assegnato al servizio sulla porta che l'applicazione sta ascoltando. Se si imposta il protocollo su`HTTP`, App Runner invia le richieste di controllo dello stato di salute al percorso configurato.	`TCP`
Path	L'URL a cui App Runner invia le richieste di controllo dello stato HTTP. Applicabile solo ai controlli HTTP.	`/`
Interval	L'intervallo di tempo, in secondi, tra i controlli dell'integrità.	5
Timeout	Il tempo di attesa, in secondi, per una risposta del controllo dell'integrità prima di decidere che il controllo dell'integrità non è riuscito.	2

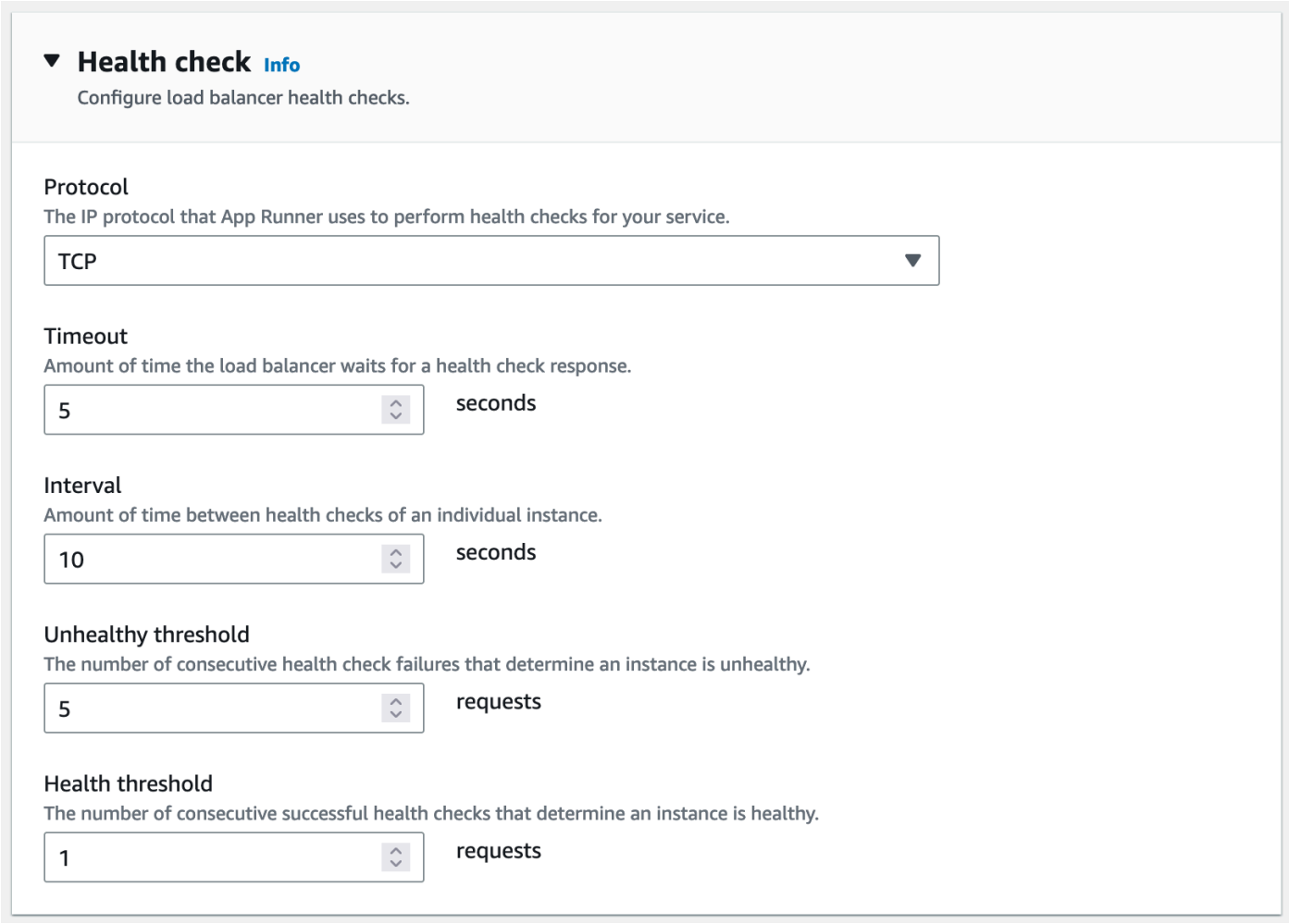
Impostazione	Descrizione	Impostazione predefinita
Soglia salutare	Il numero di controlli consecutivi che devono essere completati correttamente prima che App Runner decida che il servizio è integro.	1
Soglia non salutare	Il numero di controlli consecutivi non riusciti prima che App Runner decida che il servizio non è integro.	5

Configurazione dei controlli dello stato

Configura i controlli di integrità per il tuo servizio App Runner utilizzando uno dei seguenti metodi:

App Runner console

Quando crei il servizio App Runner utilizzando la console App Runner o quando ne aggiorni la configurazione in un secondo momento, puoi configurare le impostazioni del controllo dello stato. Per le procedure complete della console, consulta the section called “Creazione” e. the section called “Configurazione” In entrambi i casi, cerca la sezione di configurazione Health check nella pagina della console.



App Runner API or AWS CLI

Quando richiami le azioni CreateServiceo UpdateServiceAPI, puoi utilizzare il `HealthCheckConfiguration` parametro per specificare le impostazioni del controllo dello stato.

Per informazioni sulla struttura del parametro, consulta HealthCheckConfigurationl'AWS App Runner API Reference.

Gestione delle connessioni App Runner

Quando crei un servizio in AWS App Runner, configuri un'origine dell'applicazione, un'immagine del contenitore o un repository di origine archiviato presso un provider. App Runner deve stabilire una connessione autenticata e autorizzata con il provider. Quindi, App Runner può leggere il tuo repository e distribuirlo al tuo servizio. App Runner non richiede la creazione di una connessione quando crei un servizio che accede al codice memorizzato nel tuo. Account AWS

App Runner conserva le informazioni di connessione in una risorsa chiamata connessione. La console App Runner e questa guida si riferiscono anche alle connessioni come account connessi. App Runner richiede una risorsa di connessione quando si crea un servizio che richiede informazioni di connessione di terze parti. Di seguito sono riportate alcune informazioni importanti sulle connessioni:

- Provider: App Runner attualmente richiede risorse di connessione con GitHubo Bitbucket.
- Condiviso: puoi utilizzare una risorsa di connessione per creare più servizi App Runner che utilizzano lo stesso account del provider di repository.
- Gestione delle risorse: in App Runner, puoi creare ed eliminare connessioni. Tuttavia, non è possibile modificare una connessione esistente.
- Quota di risorse: le risorse di connessione hanno una quota prestabilita associata Account AWS a ciascuna di esse Regione AWS. Se raggiungi questa quota, potresti dover eliminare una connessione prima di poterti connettere a un nuovo account del provider. È possibile eliminare una connessione utilizzando la console o l'API App Runner come descritto nella sezione seguente,the section called “Gestire le connessioni”. Per ulteriori informazioni, consulta the section called “Quote di risorse di App Runner”.

Gestire le connessioni

Gestisci le tue connessioni App Runner utilizzando uno dei seguenti metodi:

App Runner console

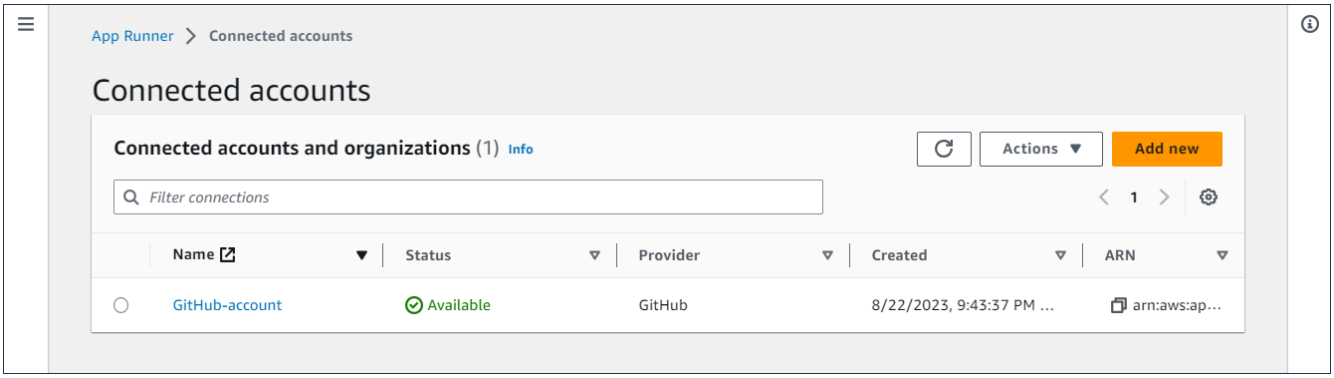
Quando utilizzi la console App Runner per creare un servizio, fornisci i dettagli di connessione. Non è necessario creare esplicitamente una risorsa di connessione. Nella console, puoi scegliere di connetterti a un GitHub account Bitbucket a cui ti sei connesso in precedenza oppure connetterti a un nuovo account. Se necessario, App Runner crea una risorsa di connessione per te. Per una nuova connessione, alcuni provider richiedono il completamento di un handshake di autenticazione prima di poter utilizzare la connessione. La console ti guida attraverso questo processo.

La console dispone anche di una pagina per la gestione delle connessioni esistenti. Puoi completare l'handshake di autenticazione per una connessione se non l'hai fatto quando hai creato il servizio. Puoi anche eliminare le connessioni che non utilizzi più. La procedura seguente mostra come gestire le connessioni dei provider di repository.

Per gestire le connessioni nel tuo account

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua Regione AWS.
2. Nel riquadro di navigazione, scegli Account connessi.

   La console visualizza quindi un elenco di connessioni ai provider di repository presenti nel tuo account.



3. Ora puoi eseguire una delle seguenti azioni con qualsiasi connessione nell'elenco:

   - Apri GitHub/Bitbucket un account o un'organizzazione: scegli il nome della connessione.
   - Handshake di autenticazione completo: seleziona la connessione, quindi dal menu Azioni scegli Completa handshake. La console ti guida attraverso il processo di handshake di autenticazione.
   - Elimina connessione: seleziona la connessione, quindi dal menu Azioni scegli Elimina. Segui le istruzioni sulla richiesta di eliminazione.

App Runner API or AWS CLI

Puoi utilizzare le seguenti azioni dell'API App Runner per gestire le tue connessioni.

- CreateConnection— Crea una connessione a un account del fornitore di repository. Dopo aver creato la connessione, è necessario completare manualmente l'handshake di autenticazione utilizzando la console App Runner. Questo processo è spiegato nella sezione precedente.
- ListConnections— Restituisce un elenco di connessioni App Runner associate al tuo Account AWS.
- DeleteConnection— Elimina una connessione. Potrebbe essere necessario eliminare le connessioni non necessarie se si raggiunge la quota di connessione prevista per Account AWS.

Gestione del ridimensionamento automatico di App Runner

AWS App Runner aumenta o riduce automaticamente le risorse di calcolo, in particolare le istanze, per l'applicazione App Runner. Il ridimensionamento automatico fornisce una gestione adeguata delle richieste quando il traffico è intenso e riduce i costi quando il traffico rallenta.

Configurazione della scalabilità automatica

Puoi configurare alcuni parametri per regolare il comportamento di auto scaling per il tuo servizio. App Runner mantiene le impostazioni di ridimensionamento automatico in una risorsa condivisibile chiamata. AutoScalingConfiguration È possibile creare e gestire configurazioni autonome di scalabilità automatica, prima di assegnarle ai servizi. Dopo averle associate a un servizio, puoi continuare a gestire le configurazioni. Puoi anche scegliere di creare una nuova configurazione di scalabilità automatica durante la creazione di un nuovo servizio o la configurazione di uno esistente. Una volta creata la nuova configurazione di autoscaling, è possibile associarla al servizio e continuare con il processo di creazione o configurazione del servizio.

Denominazione e revisioni

Una configurazione di ridimensionamento automatico ha un nome e una revisione numerica. Più revisioni di una configurazione hanno lo stesso nome e numeri di revisione diversi. È possibile utilizzare nomi di configurazione diversi per diversi scenari di scalabilità automatica, ad esempio alta disponibilità o basso costo. Per ogni nome, è possibile aggiungere più revisioni per ottimizzare le impostazioni per uno scenario specifico. È possibile avere fino a 10 nomi di configurazione con scalabilità automatica univoci e fino a 5 revisioni per ogni configurazione. Se raggiungi il limite e devi crearne altri, puoi eliminarne uno e crearne un altro. App Runner non ti consentirà di eliminare una configurazione impostata come predefinita o utilizzata da un servizio attivo. Per ulteriori informazioni sulle quote, consulta the section called “Quote di risorse di App Runner”.

Impostazione di una configurazione predefinita

Quando crei o aggiorni un servizio App Runner, puoi fornire una risorsa di configurazione con scalabilità automatica. La fornitura di una configurazione di scalabilità automatica è facoltativa. Se non ne fornisci uno, App Runner fornisce una configurazione di ridimensionamento automatico predefinita con valori consigliati. La funzione di configurazione della scalabilità automatica offre la possibilità di impostare la propria configurazione di ridimensionamento automatico predefinita invece di utilizzare l'impostazione predefinita fornita da App Runner. Una volta specificata un'altra configurazione di autoscaling come impostazione predefinita, tale configurazione viene

automaticamente assegnata come predefinita ai nuovi servizi che creerai in futuro. La nuova designazione predefinita non influisce sulle associazioni precedentemente impostate per i servizi esistenti.

Configurazione dei servizi con scalabilità automatica

È possibile condividere una singola configurazione di scalabilità automatica tra più servizi App Runner per garantire che i servizi abbiano lo stesso comportamento di ridimensionamento automatico. Per ulteriori informazioni sulla configurazione delle configurazioni di auto scaling con la console App Runner o l'API App Runner, consulta le sezioni che seguono in questo argomento. Per informazioni più generali sulle risorse condivisibili, vedere. the section called “Risorse di configurazione”

Impostazioni configurabili

È possibile configurare le seguenti impostazioni di ridimensionamento automatico:

- Concorrenza massima: il numero massimo di richieste simultanee elaborate da un'istanza. Quando il numero di richieste simultanee supera questa quota, App Runner aumenta il servizio.
- Dimensione massima: il numero massimo di istanze fino a cui il servizio può scalare. Si tratta del numero massimo di istanze in grado di gestire contemporaneamente il traffico del servizio.
- Dimensione minima: il numero minimo di istanze che App Runner può fornire per il tuo servizio. Il servizio ha sempre almeno questo numero di istanze fornite. Alcune di queste istanze gestiscono attivamente il traffico. Le altre fanno parte della riserva di capacità di calcolo economica, pronta per essere attivata rapidamente. Paghi per l'utilizzo della memoria di tutte le istanze fornite. Paghi solo per l'utilizzo della CPU del sottoinsieme attivo.

> **Note**
>
> Il conteggio delle risorse vCPU determina il numero di istanze che App Runner può fornire al servizio. Si tratta di un valore di quota regolabile per il numero di risorse vCPU Fargate On-Demand che risiede nel servizio. AWS Fargate Per visualizzare le impostazioni della quota vCPU per il tuo account o per richiedere un aumento della quota, utilizza la console Service Quotas in. AWS Management Console Per ulteriori informazioni, consulta le quote AWS Fargate di servizio nella Amazon Elastic Container Service Developer Guide.

Gestire la scalabilità automatica per un servizio

Gestisci la scalabilità automatica per i tuoi servizi App Runner utilizzando uno dei seguenti metodi:

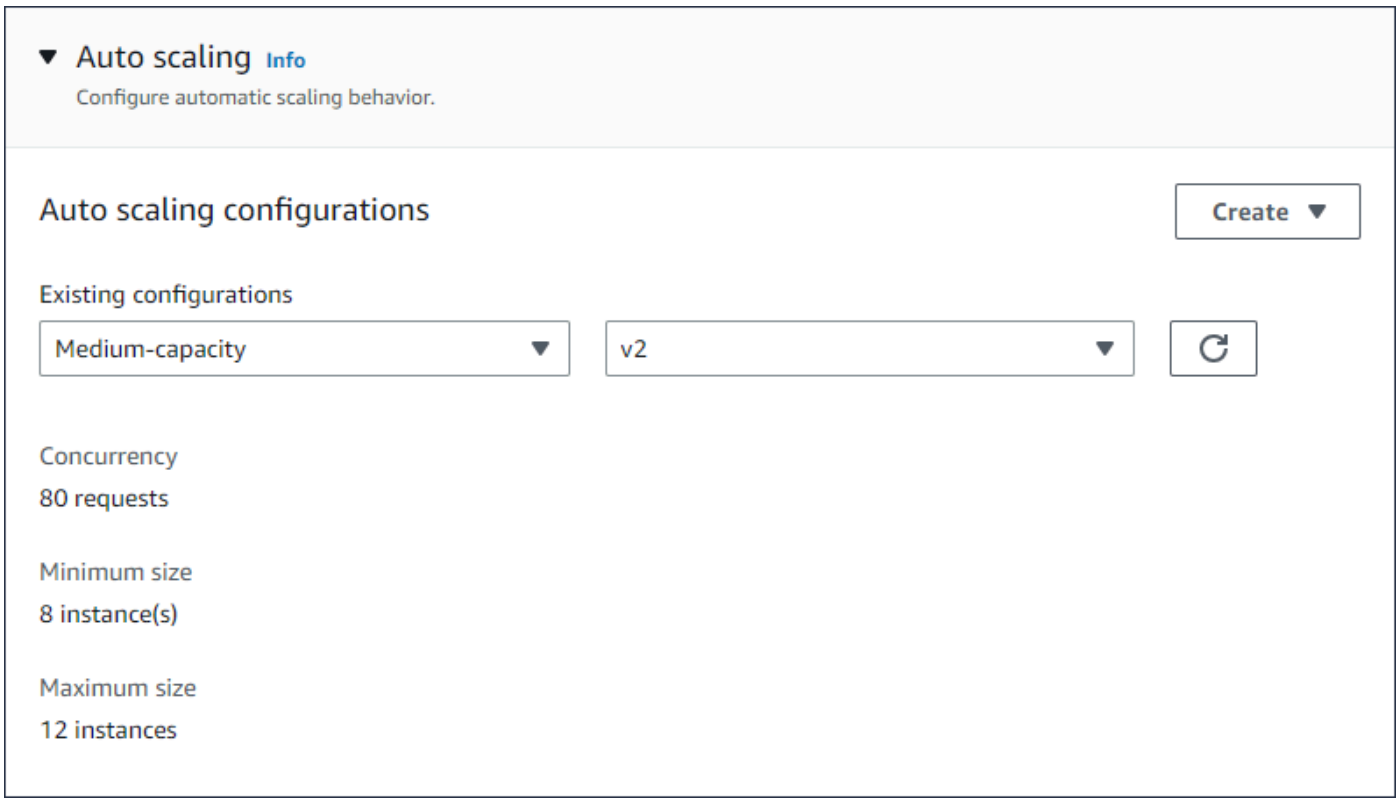
App Runner console

Quando si crea un servizio utilizzando la console App Runner o si aggiorna una configurazione del servizio, è possibile specificare una configurazione di ridimensionamento automatico.

> **Note**
>
> Quando si modifica la configurazione o la revisione della scalabilità automatica associata a un servizio, il servizio viene ridistribuito.

La pagina di configurazione della scalabilità automatica offre diverse opzioni per configurare la scalabilità automatica per il servizio.

- Per assegnare una configurazione e una revisione esistenti: scegli un valore dal menu a discesa Configurazioni esistenti. La versione di revisione più recente verrà visualizzata come predefinita nel menu a discesa adiacente. Se esiste una revisione diversa che preferisci selezionare, fallo dal menu a discesa della revisione. Vengono visualizzati i valori di configurazione per la versione di revisione.
- Per creare e assegnare una nuova configurazione di scalabilità automatica: seleziona Crea nuovo ASC dal menu Crea. Verrà avviata la pagina Aggiungi configurazione di autoscaling personalizzato. Immettete un nome e dei valori di configurazione per i parametri di autoscaling. Quindi seleziona Aggiungi. App Runner crea per te la nuova risorsa di configurazione della scalabilità automatica e ti riporta alla sezione Auto scaling con la nuova configurazione selezionata e visualizzata.
- Per creare e assegnare una nuova revisione: seleziona innanzitutto il nome della configurazione dal menu a discesa Configurazioni esistenti. Quindi seleziona Crea revisione ASC dal menu Crea. Verrà avviata la pagina Aggiungi configurazione di autoscaling personalizzato. Immettete i valori per i parametri di ridimensionamento automatico. Quindi seleziona Aggiungi. App Runner crea per te una nuova revisione della configurazione di ridimensionamento automatico e ti riporta alla sezione Auto scaling con la nuova revisione selezionata e visualizzata.



App Runner API or AWS CLI

Quando richiami le azioni API CreateServiceo UpdateServiceApp Runner, puoi utilizzare il `AutoScalingConfigurationArn` parametro per specificare una risorsa di configurazione con scalabilità automatica per il tuo servizio.

La sezione successiva fornisce indicazioni per gestire le risorse di configurazione con scalabilità automatica.

Gestisci le risorse relative alle configurazioni con scalabilità automatica

Gestisci le configurazioni e le revisioni del ridimensionamento automatico di App Runner per il tuo account utilizzando uno dei seguenti metodi:

App Runner console

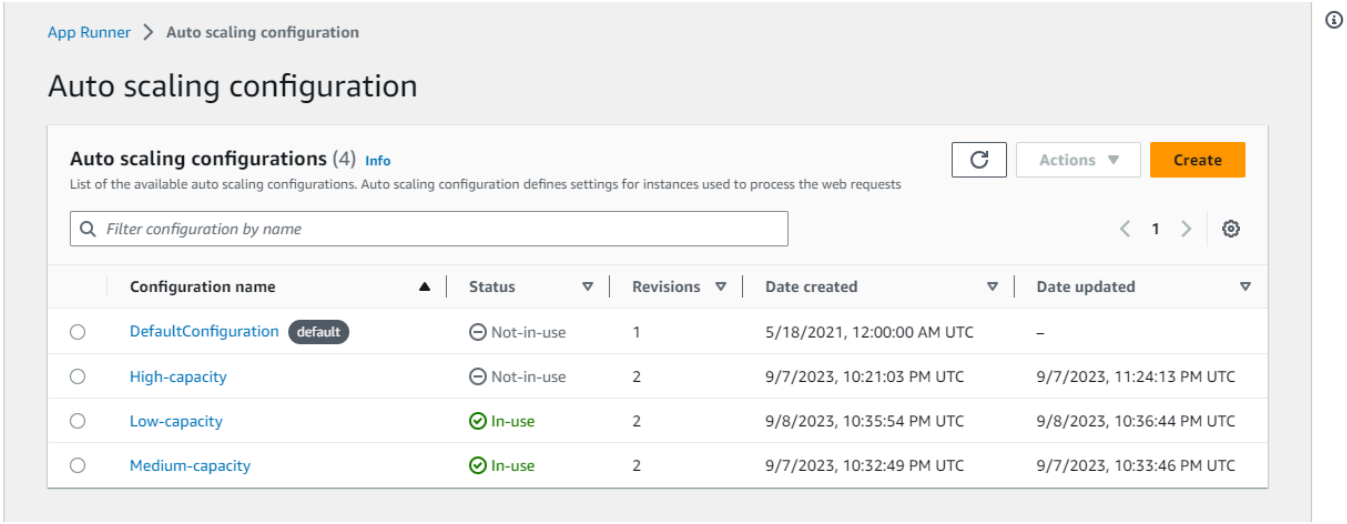
Gestione delle configurazioni di scalabilità automatica

La pagina Configurazioni di ridimensionamento automatico elenca le configurazioni di ridimensionamento automatico che hai impostato nel tuo account. È possibile creare e gestire le

configurazioni di autoscaling in questa pagina e successivamente assegnarle a uno o più servizi App Runner.

Da questa pagina è possibile effettuare una delle seguenti operazioni:

- Crea una nuova configurazione di scalabilità automatica.
- Crea una nuova revisione per una configurazione di auto scaling esistente.
- Eliminare una configurazione di scalabilità automatica.
- Imposta una configurazione di ridimensionamento automatico come predefinita.



Per gestire le configurazioni di scalabilità automatica nel tuo account

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Nel riquadro di navigazione, scegli Configurazioni di ridimensionamento automatico. La console visualizza un elenco di configurazioni di ridimensionamento automatico nel tuo account.

   Ora puoi eseguire una delle seguenti operazioni.

   - Per creare una nuova configurazione di scalabilità automatica, segui questi passaggi.

     a. Nella pagina Configurazioni di ridimensionamento automatico, seleziona Crea.

        Viene visualizzata la pagina Crea configurazione auto scaling.

     b. Immettete i valori per il nome della configurazione, la concorrenza, la dimensione minima e la dimensione massima.

c. (Facoltativo) Se desideri aggiungere tag, seleziona Nuovo tag automatico. Quindi, nei campi visualizzati, inserisci un nome e un valore (opzionale).

d. Seleziona Crea.

- Per creare una nuova revisione per una configurazione di auto scaling esistente, segui questi passaggi.

  a. Nella pagina Configurazioni di ridimensionamento automatico, seleziona il pulsante di opzione accanto alla configurazione che richiede la nuova revisione. Quindi seleziona Crea revisione dal menu Azioni.

     Viene visualizzata la pagina Crea revisione.

  b. Attiva, inserisci i valori per Concorrenza, Dimensione minima e Dimensione massima.

  c. (Facoltativo) Se desideri aggiungere tag, seleziona Nuovo tag automatico. Quindi, nei campi visualizzati, inserisci un nome e un valore (opzionale).

  d. Seleziona Crea.

- Per eliminare una configurazione di ridimensionamento automatico, segui questi passaggi.

  a. Nella pagina Configurazioni di ridimensionamento automatico, seleziona il pulsante di opzione accanto alla configurazione da eliminare.

  b. Seleziona Elimina dal menu Azioni.

  c. Per procedere con l'eliminazione, seleziona Elimina nella finestra di dialogo di conferma. Altrimenti, seleziona Annulla.

  > **Note**
  >
  > App Runner verifica che la scelta di eliminazione non sia impostata come predefinita o sia attualmente utilizzata da qualsiasi servizio attivo.

- Per impostare una configurazione di ridimensionamento automatico come predefinita, segui questi passaggi.

  a. Nella pagina Configurazioni di ridimensionamento automatico, seleziona il pulsante di opzione accanto alla configurazione che devi impostare come predefinita.

  b. Seleziona Imposta come predefinito dal menu Azioni.

c. Viene visualizzata una finestra di dialogo che ti informa che App Runner utilizzerà l'ultima revisione come configurazione predefinita per tutti i nuovi servizi che crei. Seleziona Conferma per procedere. Altrimenti seleziona Annulla.

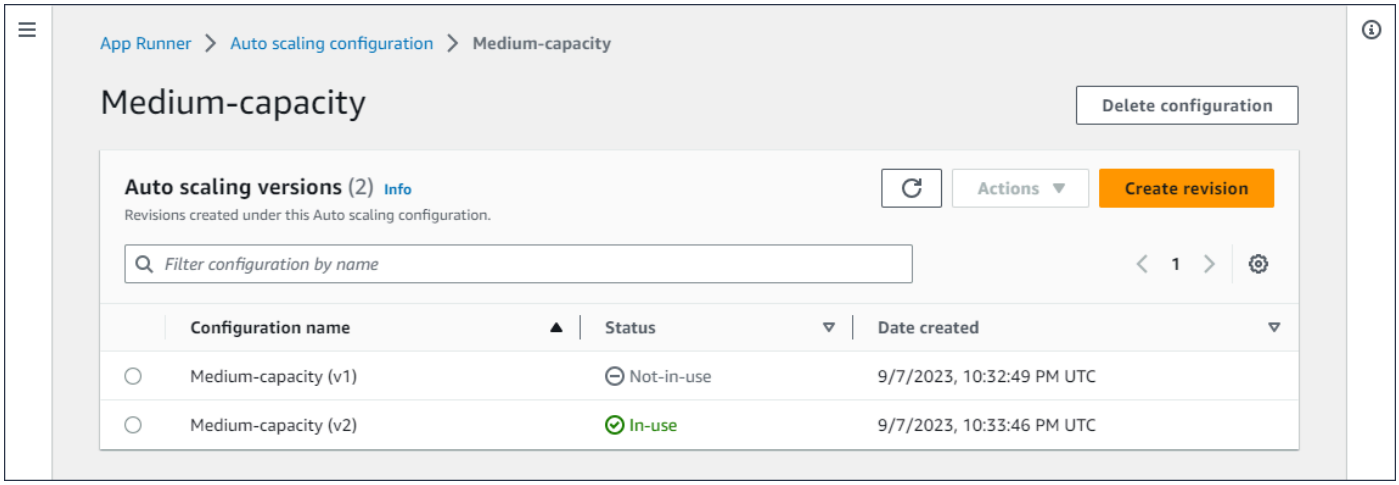
> **Note**
>
> - Quando imposti una configurazione di scalabilità automatica come predefinita, questa viene automaticamente assegnata come configurazione predefinita ai nuovi servizi che creerai in futuro.
> - La nuova designazione predefinita non influisce sulle associazioni precedentemente impostate per i servizi esistenti.
> - Se la configurazione di ridimensionamento automatico predefinita designata presenta delle revisioni, App Runner assegna la revisione più recente come predefinita.

Gestisci le revisioni

La console dispone anche di una pagina per la creazione e la gestione delle revisioni di auto scaling esistenti denominata Revisioni di auto scaling. Accedi a questa pagina selezionando il nome di una configurazione nella pagina Configurazioni di Auto scaling.

È possibile effettuare una delle seguenti operazioni dalla pagina Revisioni di Auto scaling:

- Crea una nuova revisione con ridimensionamento automatico.
- Imposta una revisione della configurazione con scalabilità automatica come predefinita.
- Eliminare una revisione.
- Eliminare l'intera configurazione di auto scaling, incluse tutte le revisioni associate.
- Visualizza i dettagli di configurazione per una revisione.
- Visualizza un elenco dei servizi associati a una revisione.
- Modifica la revisione di un servizio elencato.



Per gestire le revisioni della scalabilità automatica nel tuo account

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Nel riquadro di navigazione, scegli Configurazioni di ridimensionamento automatico. La console visualizza un elenco di configurazioni di ridimensionamento automatico nel tuo account. La serie precedente di procedure illustrata nella Gestione delle configurazioni di scalabilità automatica sezione include un'immagine della schermata di questa pagina.
3. Ora puoi approfondire una specifica configurazione di auto scaling per visualizzarne e gestirne tutte le revisioni. Nel riquadro Configurazioni di ridimensionamento automatico, nella colonna Nome configurazione, scegli un nome per le configurazioni di ridimensionamento automatico. Seleziona il nome effettivo, anziché il pulsante di opzione. In questo modo verrai indirizzato a un elenco di tutte le revisioni per quella configurazione nella pagina Revisioni di Auto Scaling.
4. Ora puoi eseguire una delle seguenti operazioni.
   - Per creare una nuova revisione per una configurazione di auto scaling esistente, segui questi passaggi.
     a. Nella pagina Revisioni con ridimensionamento automatico, seleziona Crea revisione.

        Viene visualizzata la pagina Crea revisione.
     b. Immettete i valori per Concorrenza, Dimensione minima e Dimensione massima.
     c. (Facoltativo) Se desideri aggiungere tag, seleziona Nuovo tag automatico. Quindi, nei campi visualizzati, inserisci un nome e un valore (opzionale).
     d. Seleziona Crea.

- Per eliminare l'intera configurazione di auto scaling, incluse tutte le revisioni associate, segui questi passaggi.

  a. Seleziona Elimina configurazione in alto a destra nella pagina.

  b. Per procedere con l'eliminazione, seleziona Elimina nella finestra di dialogo di conferma. Altrimenti, seleziona Annulla.

  > **Note**
  >
  > App Runner verifica che la scelta di eliminazione non sia impostata come predefinita o sia attualmente utilizzata da qualsiasi servizio attivo.

- Per impostare una revisione con ridimensionamento automatico come predefinita, segui questi passaggi.

  a. Seleziona il pulsante di opzione accanto alla revisione che desideri impostare come predefinita.

  b. Seleziona Imposta come predefinito dal menu Azioni.

  > **Note**
  >
  > - Quando imposti una configurazione di scalabilità automatica come predefinita, questa viene automaticamente assegnata come configurazione predefinita ai nuovi servizi che creerai in futuro.
  > - La nuova designazione predefinita non influisce sulle associazioni precedentemente impostate per i servizi esistenti.

- Per visualizzare i dettagli di configurazione per una revisione, segui questi passaggi.

  - Seleziona il pulsante di opzione accanto alla revisione.

    I dettagli di configurazione per la revisione, incluso l'ARN, vengono visualizzati nel pannello diviso inferiore. Fate riferimento all'immagine sullo schermo al termine di questa procedura.

- Per visualizzare un elenco dei servizi associati a una revisione, procedi nel seguente modo.

  - Seleziona il pulsante di opzione accanto alla revisione.

Il pannello Servizi, visualizzato nel pannello diviso inferiore, sotto i dettagli della configurazione della revisione. Il pannello elenca tutti i servizi che utilizzano questa revisione della configurazione con scalabilità automatica. Fate riferimento all'immagine sullo schermo al termine di questa procedura.

- Per modificare la revisione di un servizio elencato, segui questi passaggi.

  a. Seleziona il pulsante di opzione accanto alla revisione, se non l'hai già fatto.

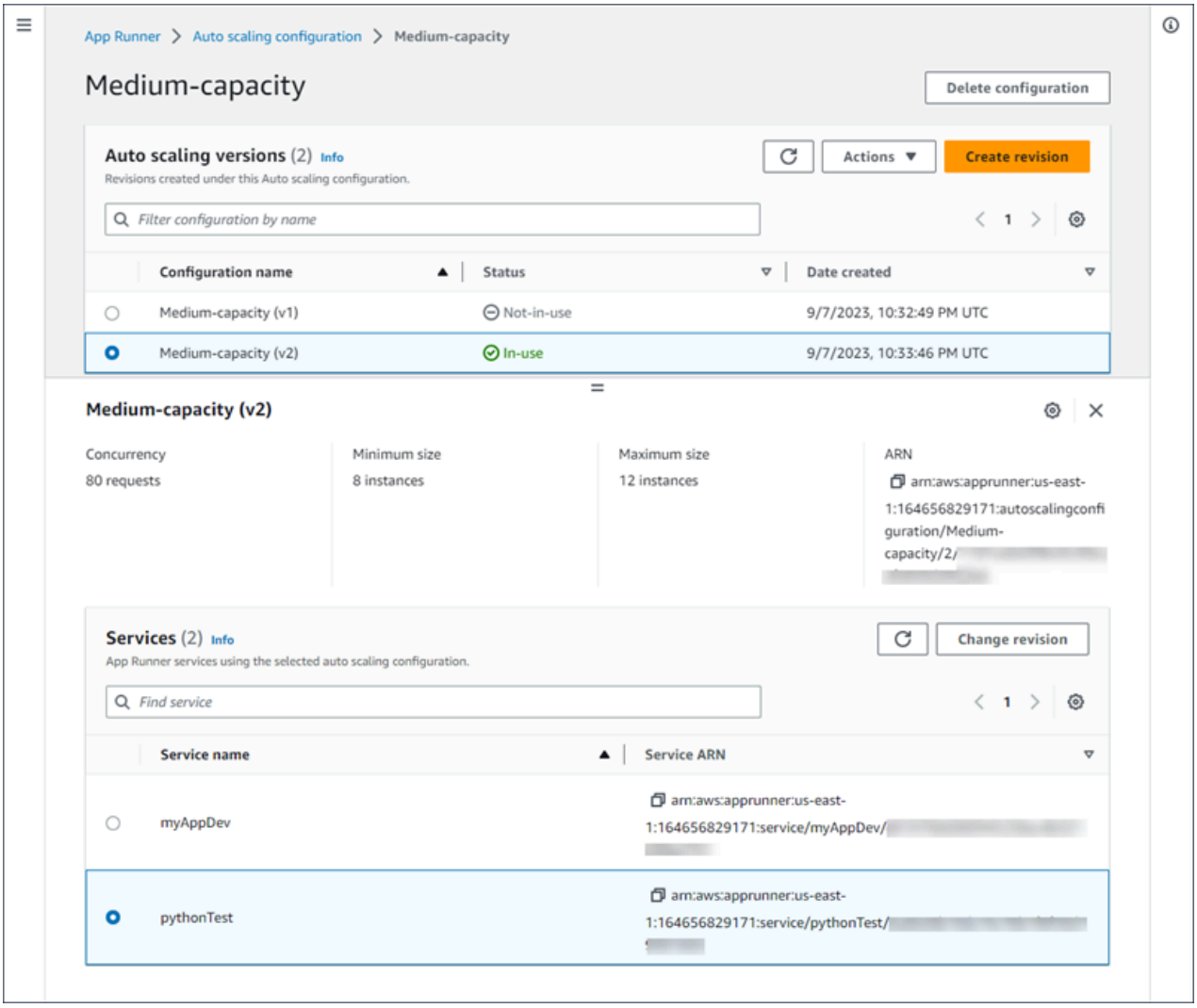
     Il pannello Servizi viene visualizzato nel pannello diviso inferiore, sotto i dettagli della configurazione della revisione. Il pannello elenca tutti i servizi che utilizzano questa revisione della configurazione con scalabilità automatica. Fate riferimento all'immagine sullo schermo al termine di questa procedura.

  b. Nel pannello Servizi, selezionate il pulsante di opzione accanto al servizio che desiderate modificare. Quindi seleziona Modifica revisioni.

  c. Viene visualizzato il pannello di revisione Change ASC. Scegli tra le revisioni disponibili nel menu a discesa. Sono disponibili solo le revisioni della configurazione di autoscaling scelta in precedenza. Se è necessario passare a una configurazione di autoscaling diverso, seguire le procedure riportate nella sezione the section called “Gestire la scalabilità automatica per un servizio” precedente.

     Seleziona Aggiorna per procedere con la modifica. Altrimenti seleziona Annulla.

     > **Note**
     >
     > Quando modifichi una revisione associata a un servizio, il servizio viene ridistribuito.
     > È necessario selezionare Aggiorna in questo pannello per visualizzare le associazioni aggiornate.
     > Per visualizzare l'attività in corso e lo stato della ridistribuzione del servizio, utilizza il pannello breadcrumbs per accedere a App Runner > Servizi, seleziona il servizio, quindi visualizza la scheda Registri dal pannello Panoramica dei servizi.



App Runner API or AWS CLI

Utilizza le seguenti azioni dell'API App Runner per gestire le risorse di configurazione con scalabilità automatica.

- CreateAutoScalingConfiguration— Crea una nuova configurazione di ridimensionamento automatico o una revisione di una configurazione esistente.
- UpdateDefaultAutoScalingConfiguration—Imposta una configurazione di ridimensionamento automatico come predefinita. La configurazione di ridimensionamento automatico predefinita esistente verrà impostata automaticamente su non predefinita.
- ListAutoScalingConfigurations— Restituisce un elenco delle configurazioni di autoscaling associate all'utente Account AWS, con informazioni di riepilogo.

- ListServicesForAutoScalingConfiguration— Restituisce un elenco dei servizi App Runner associati utilizzando una configurazione di ridimensionamento automatico.
- DescribeAutoScalingConfiguration— Restituisce una descrizione completa di una configurazione di autoscaling.
- DeleteAutoScalingConfiguration— Elimina una configurazione di ridimensionamento automatico. È possibile eliminare una configurazione di ridimensionamento automatico di primo livello, una revisione specifica di una o tutte le revisioni associate alla configurazione di livello superiore. Utilizzate il `DeleteAllRevisions` parametro opzionale per eliminare tutte le revisioni. Se raggiungi la quota di risorse di configurazione auto scaling per il tuo Account AWS, potrebbe essere necessario eliminare le configurazioni di auto scaling non necessarie.

Gestione di nomi di dominio personalizzati per un servizio App Runner

Quando crei un AWS App Runner servizio, App Runner gli assegna un nome di dominio. Si tratta di un sottodominio del `awsapprunner.com` dominio di proprietà di App Runner. Puoi utilizzare il nome di dominio per accedere all'applicazione web in esecuzione nel tuo servizio.

> **Note**
>
> Per aumentare la sicurezza delle applicazioni App Runner, il dominio*.awsapprunner.com è registrato nella Public Suffix List (PSL). Per una maggiore sicurezza, ti consigliamo di utilizzare i cookie con un `__Host-` prefisso se hai bisogno di impostare cookie sensibili nel nome di dominio predefinito per le tue applicazioni App Runner. Questa pratica ti aiuterà a difendere il tuo dominio dai tentativi CSRF (cross-site request forgery). Per ulteriori informazioni, consulta la pagina Impostazione cookie nella pagina Mozilla Developer Network.

Se possiedi un nome di dominio, puoi associarlo al tuo servizio App Runner. Dopo che App Runner ha convalidato il nuovo dominio, puoi utilizzare il dominio per accedere all'applicazione oltre al dominio App Runner. Puoi associare fino a cinque domini personalizzati.

> **Note**
>
> Facoltativamente, puoi includere il `www` sottodominio del tuo dominio. Tuttavia, al momento questo è supportato solo dall'API. La console App Runner non supporta l'inclusione del `www` sottodominio del dominio.

> **Note**
>
> AWS App Runner non supporta l'utilizzo di zone ospitate private di Route 53. Le zone private ospitate personalizzano la risoluzione dei nomi di dominio per il traffico Amazon VPC. Per ulteriori informazioni sulle zone ospitate private, consulta Lavorare con le zone ospitate private nella documentazione di Route 53.

Associa (collega) un dominio personalizzato al tuo servizio

Quando associ un dominio personalizzato al tuo servizio, devi aggiungere i record CNAME e i record di destinazione DNS al tuo server DNS. Le seguenti sezioni forniscono informazioni sui record CNAME e sui record di destinazione DNS e su come utilizzarli.

> **Note**
>
> Se utilizzi Amazon Route 53 come provider DNS, App Runner configura automaticamente il tuo dominio personalizzato con la convalida del certificato e i record DNS richiesti da collegare alla tua applicazione web App Runner. Questo accade quando usi la console App Runner per collegare il tuo dominio personalizzato al tuo servizio. L'Gestisci domini personalizzatiargomento che segue fornisce ulteriori informazioni.

Registri CNAME

Quando associ un dominio personalizzato al tuo servizio, App Runner ti fornisce una serie di record di convalida dei certificati per la convalida dei certificati. È necessario aggiungere questi record di convalida dei certificati al server DNS (Domain Name System). Aggiungi i record di convalida dei certificati, forniti da App Runner, al tuo server DNS. In questo modo, App Runner può confermare che tu possiedi o controlli il dominio.

> **Note**
>
> Per rinnovare automaticamente i certificati di dominio personalizzati, assicurati di non eliminare i record di convalida dei certificati dal tuo server DNS. Per informazioni su come risolvere i problemi relativi al rinnovo del certificato, consulta. the section called “Rinnovo del certificato di dominio personalizzato”

App Runner utilizza ACM per verificare il dominio. Se utilizzi record CAA nei tuoi record DNS, assicurati che almeno un record CAA faccia riferimento. `amazon.com` Altrimenti, ACM non potrà verificare il dominio e crearlo con successo.

Se ricevi errori relativi alla CAA, consulta i seguenti link per scoprire come risolverli:

- Problemi di autorizzazione dell'Autorità di certificazione (CAA)
- Come posso risolvere gli errori CAA per l'emissione o il rinnovo di un certificato ACM?
- Nomi di dominio personalizzati

> **Note**
>
> Se utilizzi Amazon Route 53 come provider DNS, App Runner configura automaticamente il tuo dominio personalizzato con la convalida del certificato e i record DNS richiesti da collegare alla tua applicazione web App Runner. Questo accade quando usi la console App Runner per collegare il tuo dominio personalizzato al tuo servizio. L'Gestisci domini personalizzatiargomento che segue fornisce ulteriori informazioni.

Record di destinazione DNS

Aggiungi i record di destinazione DNS al tuo server DNS per indirizzare il dominio App Runner. Aggiungi un record per il dominio personalizzato e un altro per il `www` sottodominio, se hai scelto questa opzione. Quindi, attendi che lo stato del dominio personalizzato diventi attivo nella console App Runner. Questa operazione richiede in genere alcuni minuti, ma potrebbe richiedere fino a 24-48 ore (1-2 giorni). Quando il dominio personalizzato viene convalidato, App Runner inizia a indirizzare il traffico da questo dominio alla tua applicazione web.

> **Note**
>
> Per una migliore compatibilità con i servizi App Runner, ti consigliamo di utilizzare Amazon Route 53 come provider DNS. Se non usi Amazon Route 53 per gestire i tuoi record DNS pubblici, contatta il tuo provider DNS per scoprire come aggiungere record.
> Se utilizzi Amazon Route 53 come provider DNS, puoi aggiungere CNAME o record di alias per il sottodominio. Per il dominio root, assicurati di utilizzare il record di alias.

Puoi acquistare un nome di dominio da Amazon Route 53 o da un altro provider. Per acquistare un nome di dominio con Amazon Route 53, consulta Registrazione di un nuovo dominio nella Amazon Route 53 Developer Guide.

Per istruzioni su come configurare una destinazione DNS in Route 53, consulta Routing del traffico verso le tue risorse, nella Amazon Route 53 Developer Guide.

Per istruzioni su come configurare un target DNS su altri registrar, come GoDaddy Shopify, Hover e così via, consulta la loro documentazione specifica sull'aggiunta di record DNS Target.

Specificate un dominio da associare al servizio App Runner

È possibile specificare un dominio da associare al servizio App Runner nei seguenti modi:

- Un dominio radice: il DNS presenta alcune limitazioni intrinseche che potrebbero impedire la creazione di record CNAME per il nome di dominio radice. Ad esempio, se il tuo nome di dominio è`example.com`, puoi creare un record CNAME che indirizza il traffico verso il servizio App `acme.example.com` Runner. Tuttavia, non puoi creare un record CNAME che indirizzi il traffico verso il servizio App `example.com` Runner. Per creare un dominio root, assicurati di aggiungere un record di alias.

  Un record di alias è specifico di Route 53 e presenta i seguenti vantaggi rispetto ai record CNAME:

  - Route 53 offre una maggiore flessibilità in quanto è possibile creare record di alias per il dominio principale o il sottodominio. Ad esempio, se il tuo nome di dominio è`example.com`, puoi creare un record che indirizza le richieste `acme.example.com` da `example.com` o verso il servizio App Runner.
  - È più efficiente in termini di costi. Questo perché Route 53 non addebita alcun costo per le richieste che utilizzano un record di alias per indirizzare il traffico.

- Un sottodominio, `login.example.com` ad esempio o. `admin.login.example.com` Facoltativamente, puoi anche associare il `www` sottodominio come parte della stessa operazione. Puoi aggiungere CNAME o un record di alias per il sottodominio.
- Un jolly: ad esempio,. `*.example.com` In questo caso non è possibile utilizzare l'`www`opzione. È possibile specificare un carattere jolly solo come sottodominio immediato di un dominio radice e solo singolarmente. Queste non sono specifiche valide:`login*.example.com`,. `*.login.example.com` Questa specifica con caratteri jolly associa tutti i sottodomini immediati e non associa il dominio principale stesso. Il dominio radice deve essere associato in un'operazione separata.

Un'associazione di dominio più specifica sostituisce un'associazione di dominio meno specifica. Ad esempio, `login.example.com` le sostituzioni. `*.example.com` Vengono utilizzati il certificato e il CNAME dell'associazione più specifica.

L'esempio seguente mostra come utilizzare più associazioni di dominio personalizzate:

1. `example.com`Associalo alla home page del tuo servizio. Abilita `www` l'associazione`www.example.com`.
2. `login.example.com`Associati alla pagina di accesso del tuo servizio.
3. `*.example.com`Associalo a una pagina personalizzata «non trovata».

Dissocia (scollega) un dominio personalizzato

Puoi dissociare (scollegare) un dominio personalizzato dal tuo servizio App Runner. Quando scolleghi un dominio, App Runner interrompe il routing del traffico da questo dominio alla tua applicazione web.

> **Note**
>
> È necessario eliminare i record relativi al dominio dissociato dal server DNS.

App Runner crea internamente certificati che tengono traccia della validità del dominio. Questi certificati sono archiviati in AWS Certificate Manager (ACM). App Runner non elimina questi certificati per 7 giorni dopo la dissociazione di un dominio dal servizio o dopo l'eliminazione del servizio.

Gestisci domini personalizzati

Gestisci i domini personalizzati per il tuo servizio App Runner utilizzando uno dei seguenti metodi:

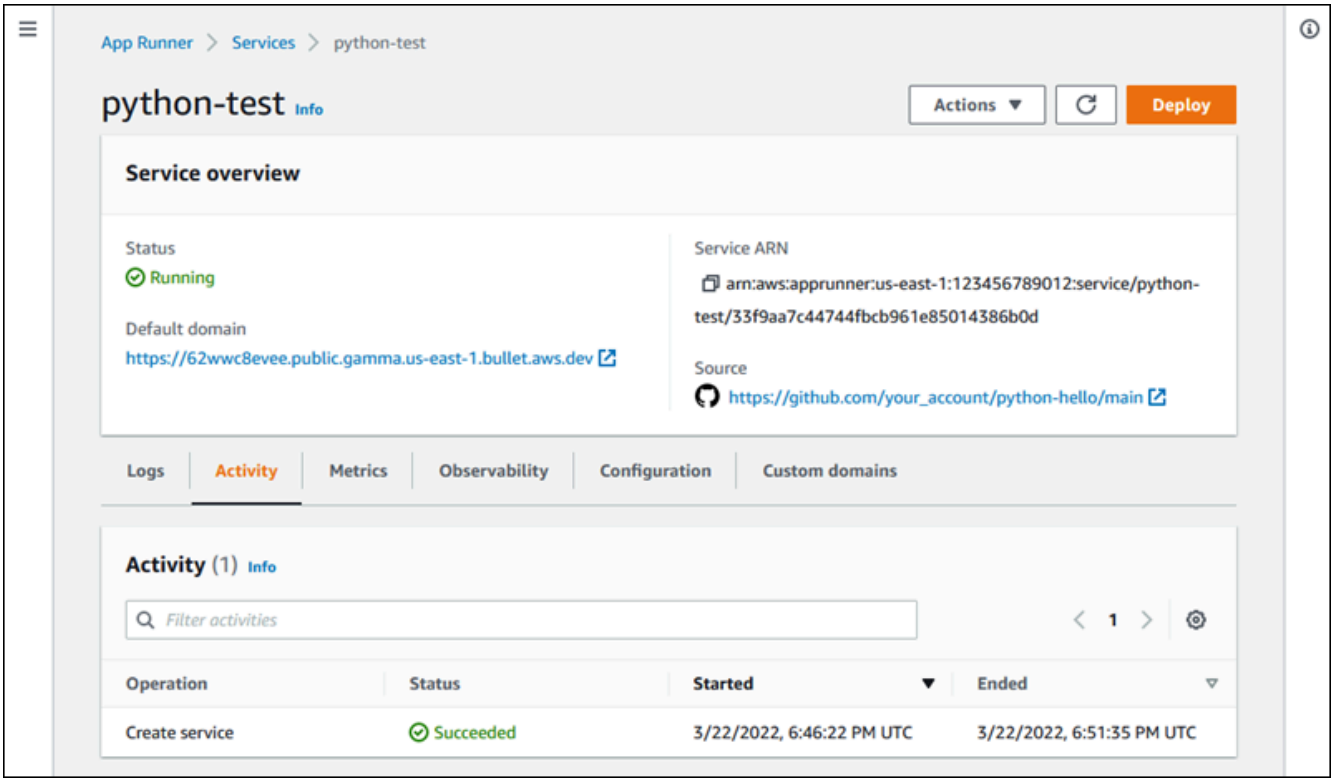
> **Note**
>
> Per una migliore compatibilità con i servizi App Runner, ti consigliamo di utilizzare Amazon Route 53 come provider DNS. Se non usi Amazon Route 53 per gestire i tuoi record DNS pubblici, contatta il tuo provider DNS per scoprire come aggiungere record.
> Se utilizzi Amazon Route 53 come provider DNS, puoi aggiungere CNAME o record di alias per il sottodominio. Per il dominio root, assicurati di utilizzare il record di alias.

App Runner console

Per associare (collegare) un dominio personalizzato utilizzando la console App Runner

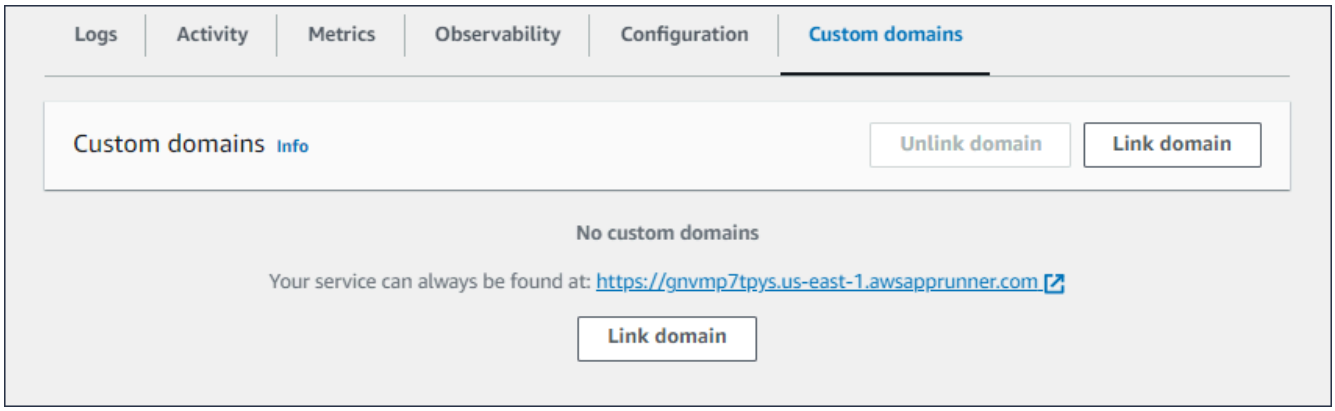
1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Nel pannello di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

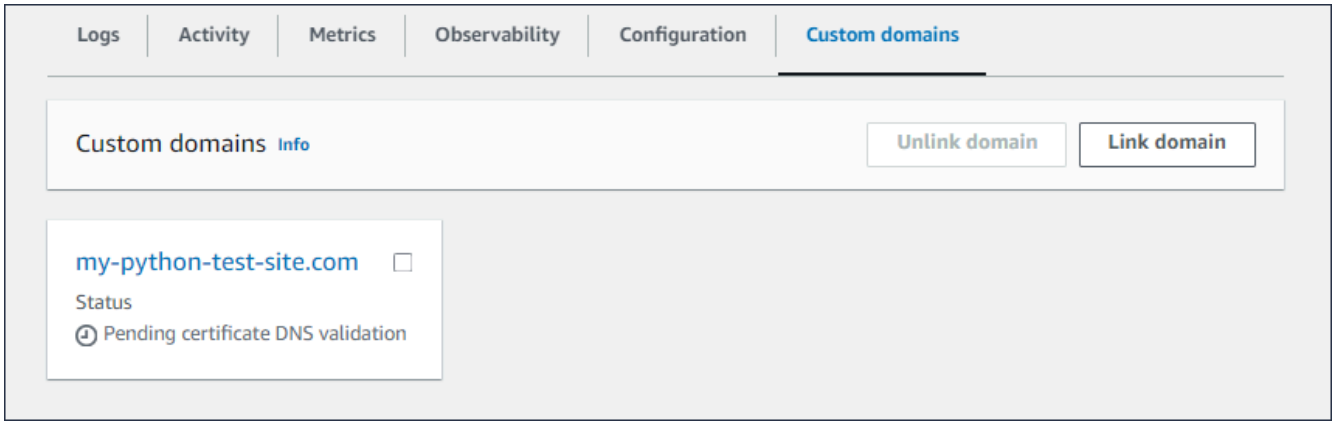
   La console mostra la dashboard del servizio con una panoramica del servizio.



3. Nella pagina della dashboard del servizio, scegli la scheda Domini personalizzati.

   La console mostra i domini personalizzati associati al servizio o Nessun dominio personalizzato.



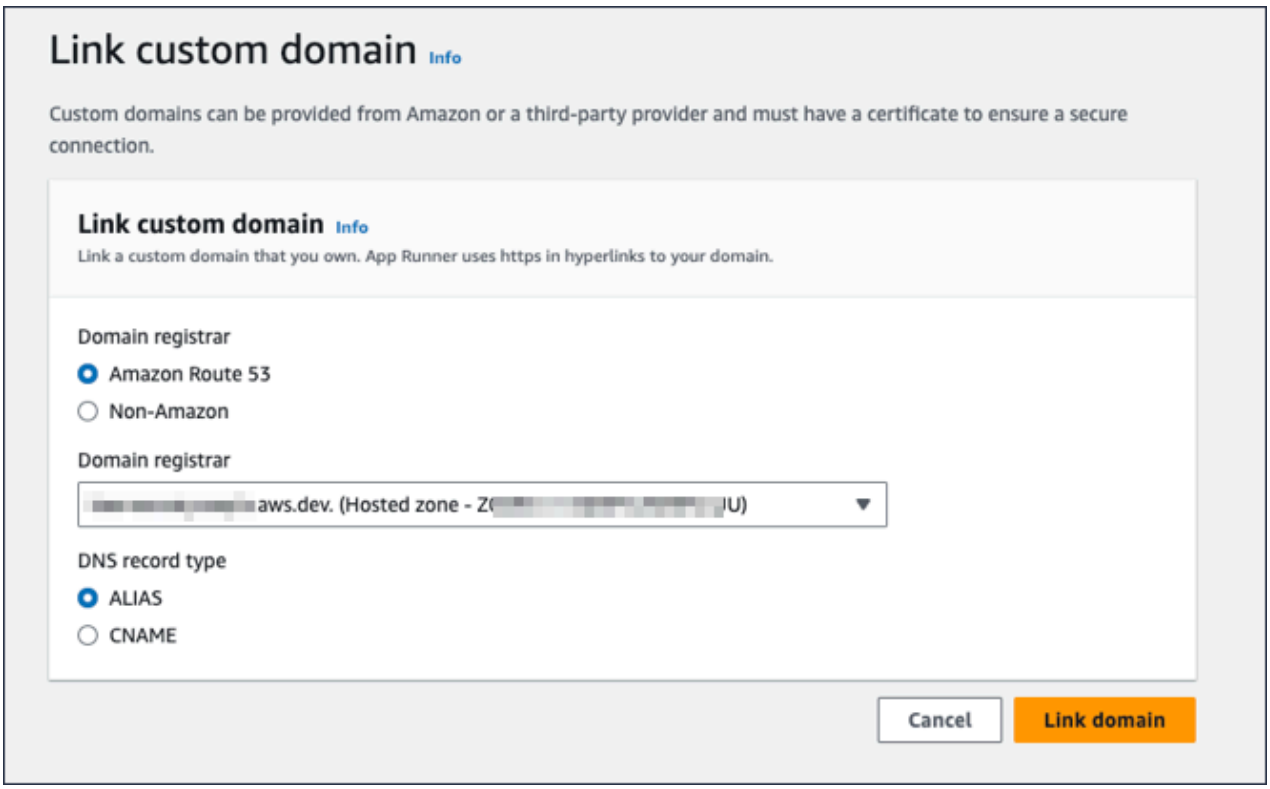


4. Nella scheda Domini personalizzati, scegli Collega dominio.
5. Viene visualizzata la pagina Collega dominio personalizzato.
   - Se il tuo dominio personalizzato è registrato con Amazon Route 53, seleziona Amazon Route 53 for Domain registrar.
     a. Seleziona il nome di dominio dall'elenco a discesa. Questo elenco mostra il nome dei nomi di dominio Route 53 e l'ID della zona ospitata.

        > **Note**
        >
        > Devi prima creare un dominio Route 53 utilizzando il servizio Amazon Route 53 dallo stesso AWS account che usi per gestire le altre risorse di App Runner.

     b. Seleziona il tipo di record DNS.

c. Scegli il dominio Link.

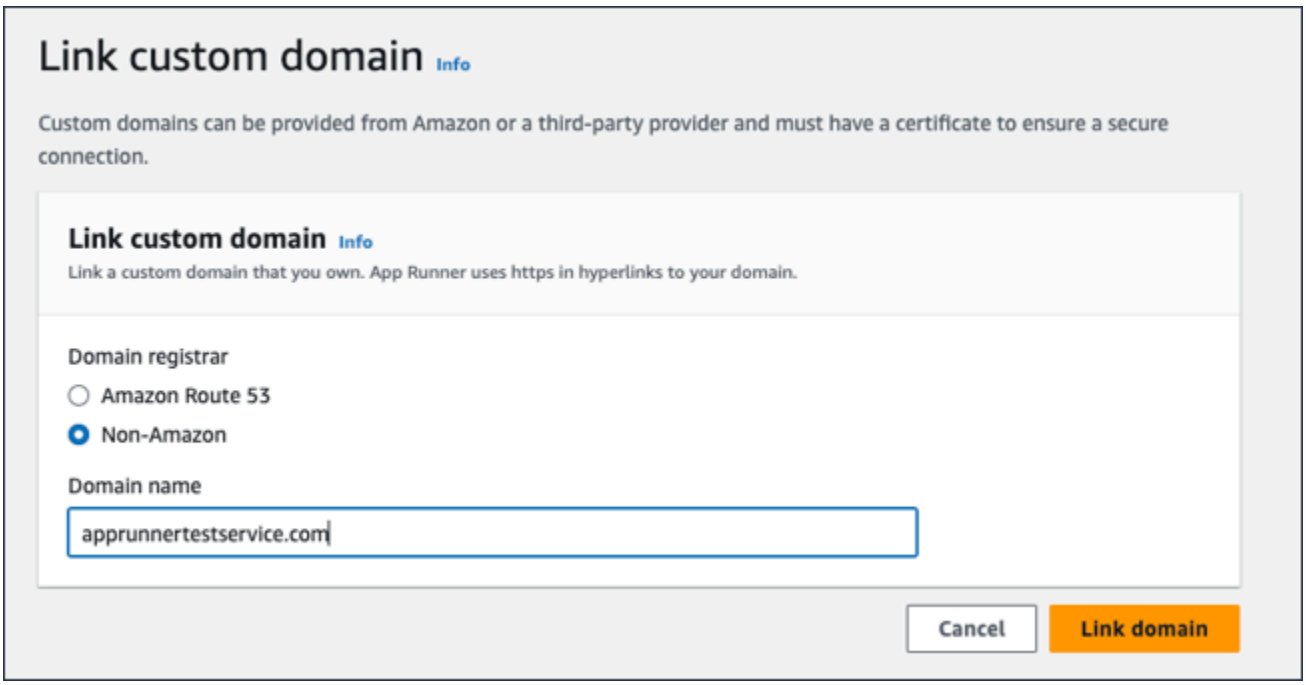


> **Note**
>
> Se App Runner visualizza un messaggio di errore che indica che il tentativo di configurazione automatica non è riuscito, puoi procedere configurando i record DNS manualmente. Questo problema può verificarsi se lo stesso nome di dominio è stato precedentemente scollegato da un servizio, senza che i record del provider DNS che rimandano al servizio vengano successivamente eliminati. In questo caso, ad App Runner viene impedito di sovrascrivere automaticamente questi record. Per completare la configurazione DNS, salta il resto dei passaggi di questa procedura e segui le istruzioni riportate in. Configurazione di un record di alias Amazon Route 53

- Se il tuo dominio personalizzato è registrato presso un altro registrar di domini, seleziona Non—Amazon for Domain registrar.

  a. Inserisci il nome del dominio.

  b. Scegli il dominio Link.



6. Viene visualizzata la pagina Configura DNS.

   - Se Amazon Route 53 è il tuo provider DNS, questo passaggio è facoltativo.

     A questo punto App Runner ha configurato automaticamente il dominio Route 53 con la convalida del certificato e i record DNS richiesti.

     > **Note**
     >
     > Se lo stesso nome di dominio era stato precedentemente scollegato da un servizio, senza i record del provider DNS che fanno riferimento alla successiva eliminazione del servizio, la configurazione automatica tentata da App Runner avrebbe potuto fallire. Per risolvere questo problema e completare l'associazione DNS, procedi con i passaggi (1) e (2) nella pagina Configura DNS per copiare i record di destinazione e certificato correnti sul provider DNS.

   - Copia i record di convalida dei certificati e i record di destinazione DNS e aggiungili al tuo server DNS. App Runner può quindi confermare che tu possiedi o controlli il dominio.

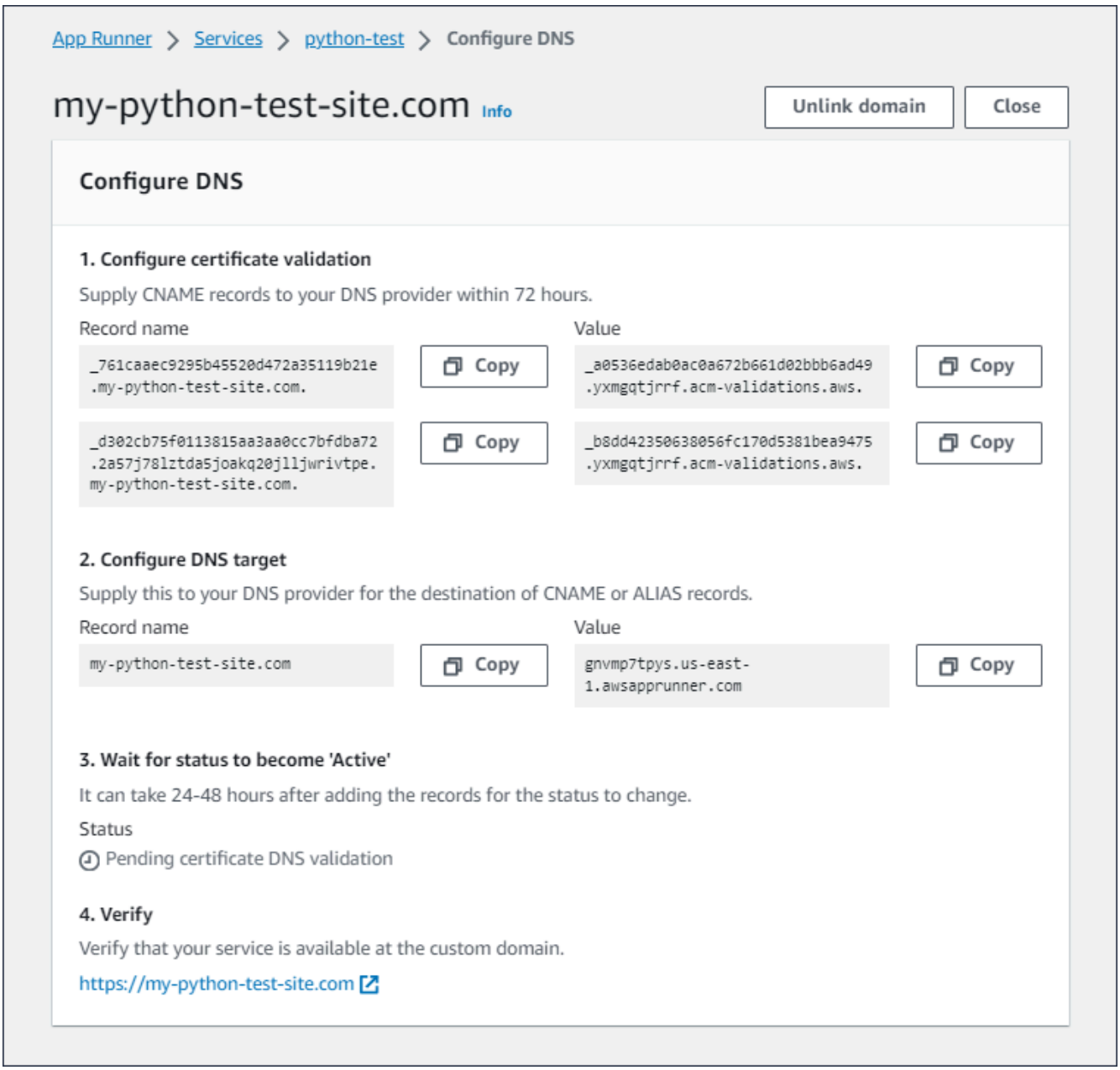
> **Note**
>
> Per rinnovare automaticamente i certificati di dominio personalizzati, assicurati di non eliminare i record di convalida dei certificati dal tuo server DNS.

- Per ulteriori informazioni sulla configurazione della convalida dei certificati, consulta la sezione ValidazioneDNS nella Guida per l'utente.AWS Certificate Manager
- Per informazioni su come configurare la destinazione DNS con il record di alias Amazon Route 53, consulta. the section called “Configurazione di un record di alias Amazon Route 53”

- Se utilizzi un provider DNS diverso da Amazon Route 53, segui questi passaggi.
  - Copia i record di convalida dei certificati e i record di destinazione DNS e aggiungili al tuo server DNS. App Runner può quindi confermare che tu possiedi o controlli il dominio.

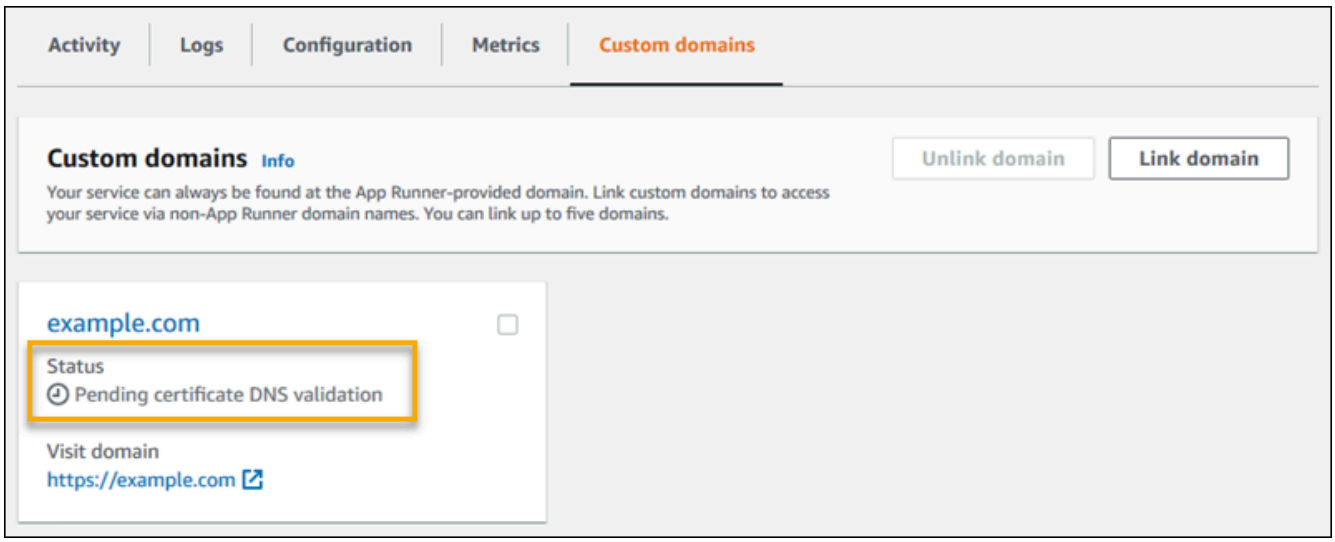
> **Note**
>
> Per rinnovare automaticamente i certificati di dominio personalizzati, assicurati di non eliminare i record di convalida dei certificati dal tuo server DNS.

- Per ulteriori informazioni sulla configurazione della convalida dei certificati, consulta la sezione ValidazioneDNS nella Guida per l'utente.AWS Certificate Manager
- Per istruzioni su come configurare un target DNS su altri registrar, come Shopify, Hover e così via GoDaddy, consulta la loro documentazione specifica sull'aggiunta di DNS Target.

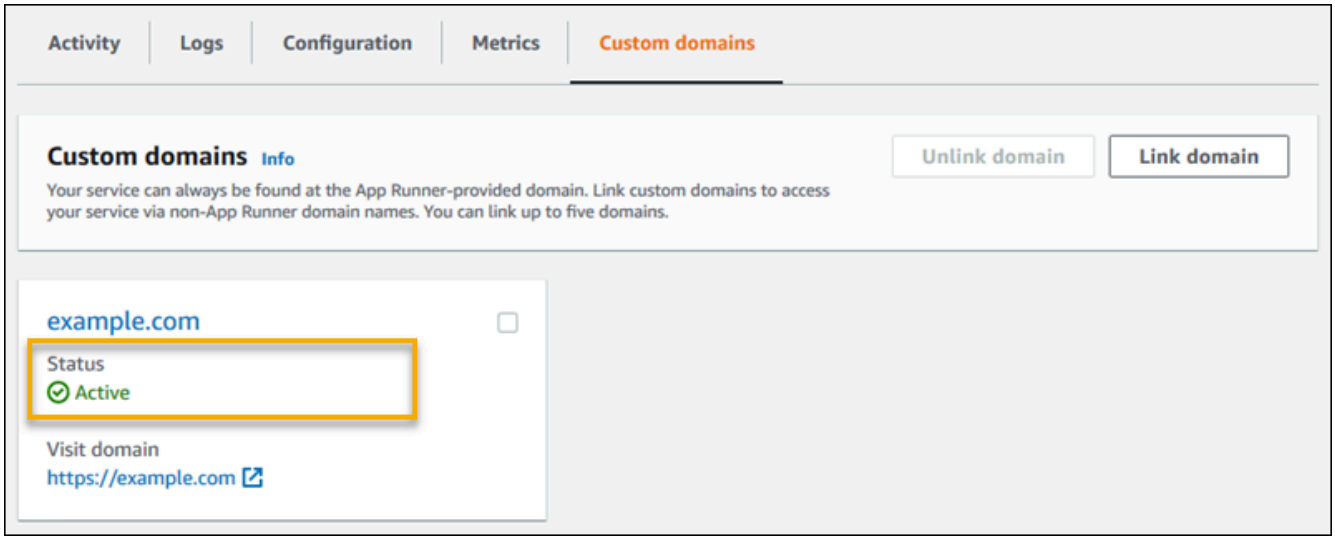


7. Scegli Chiudi

La console mostra nuovamente la dashboard. La scheda Domini personalizzati presenta un nuovo riquadro che mostra il dominio che hai appena collegato nello stato di convalida DNS del certificato in sospeso.



8. Quando lo stato del dominio diventa Attivo, verifica che il dominio funzioni per instradare il traffico accedendo al dominio.



> **Note**
>
> Per istruzioni su come risolvere gli errori relativi al dominio personalizzato, consulta. the section called “Nomi di dominio personalizzati”

Per dissociare (scollegare) un dominio personalizzato utilizzando la console App Runner

1. Nella scheda Domini personalizzati, seleziona il riquadro relativo al dominio da dissociare, quindi scegli Scollega dominio.

2. Nella finestra di dialogo Scollega il dominio, verifica l'azione scegliendo Scollega dominio.

> **Note**
>
> È necessario eliminare i record del dominio che si è dissociato dal server DNS.

App Runner API or AWS CLI

Per associare un dominio personalizzato al tuo servizio utilizzando l'API App Runner oppure chiama AWS CLI l'AssociateCustomDomainazione API. Quando la chiamata ha esito positivo, viene restituito un CustomDomainoggetto che descrive il dominio personalizzato associato al servizio. L'oggetto mostra uno `CREATING` stato e contiene un elenco di CertificateValidationRecordoggetti. La chiamata restituisce anche l'alias di destinazione che è possibile utilizzare per configurare la destinazione DNS. Si tratta di record che puoi aggiungere al tuo DNS.

Per dissociare un dominio personalizzato dal tuo servizio utilizzando l'API App Runner oppure AWS CLI, richiama l'DisassociateCustomDomainazione API. Quando la chiamata ha esito positivo, viene restituito un CustomDomainoggetto che descrive il dominio personalizzato che viene dissociato dal servizio. L'oggetto mostra uno `DELETING` stato.

Argomenti

- Configura il record di alias Amazon Route 53 per il tuo DNS di destinazione

Configura il record di alias Amazon Route 53 per il tuo DNS di destinazione

> **Note**
>
> Non è necessario seguire questa procedura se Amazon Route 53 è il tuo provider DNS. In questo caso App Runner configura automaticamente il tuo dominio Route 53 con la convalida del certificato e i record DNS richiesti da collegare alla tua applicazione web App Runner. Se il tentativo di configurazione automatica di App Runner non è riuscito, segui questa procedura per completare la configurazione DNS. Se lo stesso nome di dominio era stato precedentemente scollegato da un servizio, senza i record del provider DNS che fanno riferimento alla successiva eliminazione del servizio, App Runner non può sovrascrivere

automaticamente questi record. Questa procedura spiega come copiarli manualmente sul DNS di Route 53.

Puoi usare Amazon Route 53 come provider DNS per indirizzare il traffico verso il tuo servizio App Runner. È un servizio web DNS (Domain Name System) altamente disponibile e scalabile. Il record Amazon Route 53 contiene le impostazioni che controllano il modo in cui il traffico viene indirizzato al servizio App Runner. Puoi creare un record CNAME o un record ALIAS. Per un confronto tra CNAME e record di alias, consulta Scelta tra record alias e non alias, nella Amazon Route 53 Developer Guide.

> **Note**
>
> Amazon Route 53 attualmente supporta il record di alias per i servizi creati dopo il 1° agosto 2022.

Amazon Route 53 console

Per configurare il record di alias di Amazon Route 53

1. Accedi AWS Management Console e apri la console Route 53.
2. Nel pannello di navigazione, scegli Zone ospitate.
3. Scegli il nome della zona ospitata che desideri utilizzare per indirizzare il traffico verso il tuo servizio App Runner.
4. Scegli Crea record.
5. Specifica i seguenti valori:
   - Politica di routing: scegli la politica di routing applicabile. Per ulteriori informazioni, vedere Scelta di una politica di routing.
   - Nome del record: inserisci il nome di dominio che desideri utilizzare per indirizzare il traffico verso il tuo servizio App Runner. Il valore predefinito è il nome della hosted zone. Ad esempio, se il nome della zona ospitata è `example.com` e desideri utilizzare `acme.example.com` per indirizzare il traffico verso il tuo ambiente, inserisci`acme`.
   - Valorizza/Indirizza il traffico verso: scegli Alias per l'applicazione App Runner, quindi scegli la regione da cui proviene l'endpoint. Scegli il nome di dominio dell'applicazione verso cui indirizzare il traffico.

- Tipo di record: accetta l' IPv4 indirizzo predefinito, A —.
- Valuta lo stato dell'obiettivo: accetta il valore predefinito, Sì.

6. Scegli Crea record.

Il record di alias Route 53 che hai creato viene propagato su tutti i server Route 53 entro 60 secondi. Quando i server Route 53 vengono propagati con il tuo record di alias, puoi indirizzare il traffico verso il servizio App Runner utilizzando il nome del record di alias che hai creato.

Per informazioni su come risolvere i problemi relativi alla propagazione delle modifiche DNS, vedi Perché le mie modifiche DNS impiegano così tanto tempo a propagarsi nella Route 53 e nei resolver pubblici? .

Amazon Route 53 API or AWS CLI

Per configurare il record di alias di Amazon Route 53 utilizzando l'API Amazon Route 53 o AWS CLI chiama l'azione ChangeResourceRecordSetsAPI. Per ulteriori informazioni sull'ID della zona ospitata di destinazione di Route 53, consulta Service endpoints.

Sospensione e ripresa di un servizio App Runner

Se è necessario disattivare temporaneamente l'applicazione Web e interrompere l'esecuzione del codice, è possibile sospendere il servizio. AWS App Runner App Runner riduce a zero la capacità di calcolo del servizio.

Quando sei pronto per eseguire nuovamente l'applicazione, puoi riprendere il servizio App Runner. App Runner fornisce una nuova capacità di calcolo, implementa l'applicazione su di essa ed esegue l'applicazione. L'origine dell'applicazione non viene ridistribuita e non è necessaria alcuna build. Piuttosto, App Runner riprende con la versione attualmente distribuita. L'applicazione mantiene il dominio App Runner.

⚠ Important

- Quando metti in pausa il servizio, l'applicazione perde lo stato. Ad esempio, qualsiasi spazio di archiviazione temporaneo utilizzato dal codice viene perso. Per il codice, mettere in pausa e riprendere il servizio equivale a passare a un nuovo servizio.

- Se metti in pausa un servizio a causa di un difetto nel codice (ad esempio, un bug scoperto o un problema di sicurezza), non puoi distribuire una nuova versione prima di riprendere il servizio.

  Pertanto, ti consigliamo di mantenere il servizio in esecuzione e di ripristinare invece l'ultima versione stabile dell'applicazione.
- Quando riprendi il servizio, App Runner distribuisce l'ultima versione dell'applicazione utilizzata prima della sospensione del servizio. Se hai aggiunto nuove versioni di origine dopo aver sospeso il servizio, App Runner non le distribuisce automaticamente anche se è selezionata la distribuzione automatica. Ad esempio, supponiamo di avere nuove versioni delle immagini nell'archivio delle immagini o nuovi commit nell'archivio del codice. Queste versioni non vengono distribuite automaticamente.

  Per distribuire una versione più recente, esegui una distribuzione manuale o aggiungi un'altra versione al tuo repository di origine dopo aver ripreso il servizio App Runner.

Sospensione ed eliminazione a confronto

Metti in pausa il servizio App Runner per disabilitarlo temporaneamente. Vengono terminate solo le risorse di elaborazione e i dati memorizzati (ad esempio, l'immagine del contenitore con la versione dell'applicazione) rimangono intatti. La ripresa del servizio è rapida: l'applicazione è pronta per essere distribuita su nuove risorse di elaborazione. Il dominio App Runner rimane lo stesso.

Elimina il servizio App Runner per rimuoverlo definitivamente. I dati memorizzati vengono eliminati. Se devi ricreare il servizio, App Runner deve recuperare nuovamente il codice sorgente e crearlo se si tratta di un repository di codice. La tua applicazione Web ottiene un nuovo dominio App Runner.

Quando il servizio è in pausa

Quando metti in pausa il servizio e si trova nello stato In pausa, risponde in modo diverso alle richieste di azione, incluse le chiamate API o le operazioni della console. Quando un servizio è in pausa, puoi comunque eseguire azioni di App Runner che non modificano la definizione o la configurazione del servizio in modo tale da influire sulla sua durata. In altre parole, se un'azione modifica il comportamento, la scala o altre caratteristiche di un servizio in esecuzione, non è possibile eseguire tale azione su un servizio in pausa.

Gli elenchi seguenti forniscono informazioni sulle azioni API che è possibile e non è possibile eseguire su un servizio in pausa. Analogamente, le operazioni equivalenti della console sono consentite o negate.

Azioni che è possibile eseguire su un servizio in pausa

- *`List*`*e *`Describe*`* azioni: azioni che leggono solo informazioni.
- *`DeleteService`*— È sempre possibile eliminare un servizio.
- *`TagResource`*, *`UntagResource`* — I tag sono associati a un servizio, ma non fanno parte della sua definizione e non influiscono sul suo comportamento in fase di esecuzione.

Azioni che non è possibile eseguire su un servizio in pausa

- *`StartDeployment`*azioni (o una distribuzione manuale tramite la console)
- *`UpdateService`*(o una modifica della configurazione tramite la console, ad eccezione delle modifiche ai tag)
- *`CreateCustomDomainAssociations`*, *`DeleteCustomDomainAssociations`*
- *`CreateConnection`*, *`DeleteConnection`*

Metti in pausa e riprendi il servizio

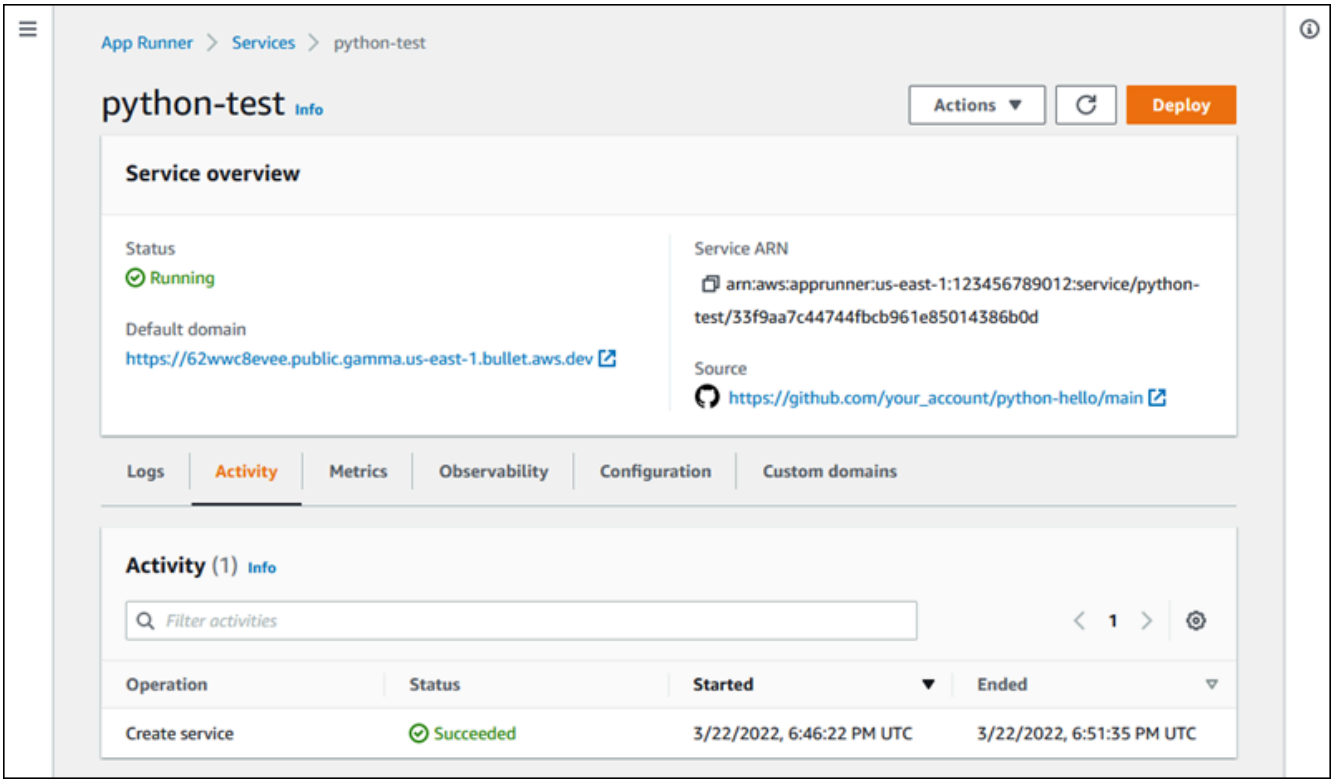
Metti in pausa e riprendi il servizio App Runner utilizzando uno dei seguenti metodi:

App Runner console

Per mettere in pausa il servizio utilizzando la console App Runner

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Nel pannello di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

   La console mostra la dashboard del servizio con una panoramica del servizio.



3. Scegli Azioni, quindi scegli Pausa.

   Nella pagina del pannello di controllo del servizio, lo stato del servizio cambia in Operazione in corso, quindi cambia in Sospeso. Il servizio è ora in pausa.

Per riprendere il servizio utilizzando la console App Runner

1. Scegli Azioni, quindi scegli Riprendi.

   Nella pagina del dashboard del servizio, lo stato del servizio cambia in Operazione in corso.

2. Attendi la ripresa del servizio. Nella pagina del dashboard del servizio, lo stato del servizio torna a Esecuzione.
3. Per verificare che la ripresa del servizio abbia esito positivo, nella pagina del dashboard del servizio, scegli il valore del dominio App Runner. È l'URL del sito Web del tuo servizio. Verifica che l'applicazione web funzioni correttamente.

App Runner API or AWS CLI

Per mettere in pausa il servizio utilizzando l'API App Runner oppure AWS CLI, richiama l'azione PauseServiceAPI. Se la chiamata restituisce una risposta corretta con un oggetto Service

visualizzato`"Status": "OPERATION_IN_PROGRESS"`, App Runner inizia a sospendere il servizio.

Per riprendere il servizio utilizzando l'API App Runner oppure AWS CLI, richiama l'azione API. ResumeService Se la chiamata restituisce una risposta corretta con un oggetto Service visualizzato`"Status": "OPERATION_IN_PROGRESS"`, App Runner inizia a riprendere il servizio.

Eliminazione di un servizio App Runner

Se desideri terminare l'applicazione Web in esecuzione nel tuo AWS App Runner servizio, puoi eliminare il servizio. L'eliminazione di un servizio interrompe il servizio Web in esecuzione, rimuove le risorse sottostanti ed elimina i dati associati.

Potresti voler eliminare un servizio App Runner per uno o più dei seguenti motivi:

- L'applicazione Web non è più necessaria: ad esempio, è stata ritirata o è una versione di sviluppo che hai finito di utilizzare.
- Hai raggiunto la quota di servizi App Runner: desideri creare un nuovo servizio nello stesso servizio Regione AWS e hai raggiunto la quota associata al tuo account. Per ulteriori informazioni, consulta the section called “Quote di risorse di App Runner”.
- Considerazioni sulla sicurezza o sulla privacy: desideri che App Runner elimini i dati che archivia per il tuo servizio.

Pausa ed eliminazione a confronto

Metti in pausa il servizio App Runner per disabilitarlo temporaneamente. Vengono terminate solo le risorse di elaborazione e i dati memorizzati (ad esempio, l'immagine del contenitore con la versione dell'applicazione) rimangono intatti. La ripresa del servizio è rapida: l'applicazione è pronta per essere distribuita su nuove risorse di elaborazione. Il dominio App Runner rimane lo stesso.

Elimina il servizio App Runner per rimuoverlo definitivamente. I dati memorizzati vengono eliminati. Se devi ricreare il servizio, App Runner deve recuperare nuovamente il codice sorgente e crearlo se si tratta di un repository di codice. La tua applicazione Web ottiene un nuovo dominio App Runner.

Cosa elimina App Runner?

Quando elimini il servizio, App Runner elimina alcuni elementi associati e non ne elimina altri. I seguenti elenchi forniscono i dettagli.

Elementi eliminati da App Runner:

- Immagine del contenitore: una copia dell'immagine che hai distribuito o dell'immagine creata da App Runner a partire dal tuo codice sorgente. È archiviato in Amazon Elastic Container Registry (Amazon ECR) utilizzando componenti Account AWS interni di proprietà di App Runner.
- Configurazione del servizio: le impostazioni di configurazione associate al servizio App Runner. Sono archiviati in Amazon DynamoDB utilizzando sistemi Account AWS interni di proprietà di App Runner.

Elementi che App Runner non elimina:

- Connessione: potresti avere una connessione associata al tuo servizio. Una connessione App Runner è una risorsa separata che può essere condivisa tra diversi servizi App Runner. Se non hai più bisogno della connessione, puoi eliminarla esplicitamente. Per ulteriori informazioni, consulta the section called “Connessioni”.
- Certificati di dominio personalizzati: se colleghi domini personalizzati a un servizio App Runner, App Runner crea internamente certificati che tengono traccia della validità del dominio. Sono archiviati in (ACM). AWS Certificate Manager App Runner non elimina il certificato per sette giorni dopo lo scollegamento di un dominio dal servizio o dopo l'eliminazione del servizio. Per ulteriori informazioni, consulta the section called “Nomi di dominio personalizzati”.

Elimina il tuo servizio

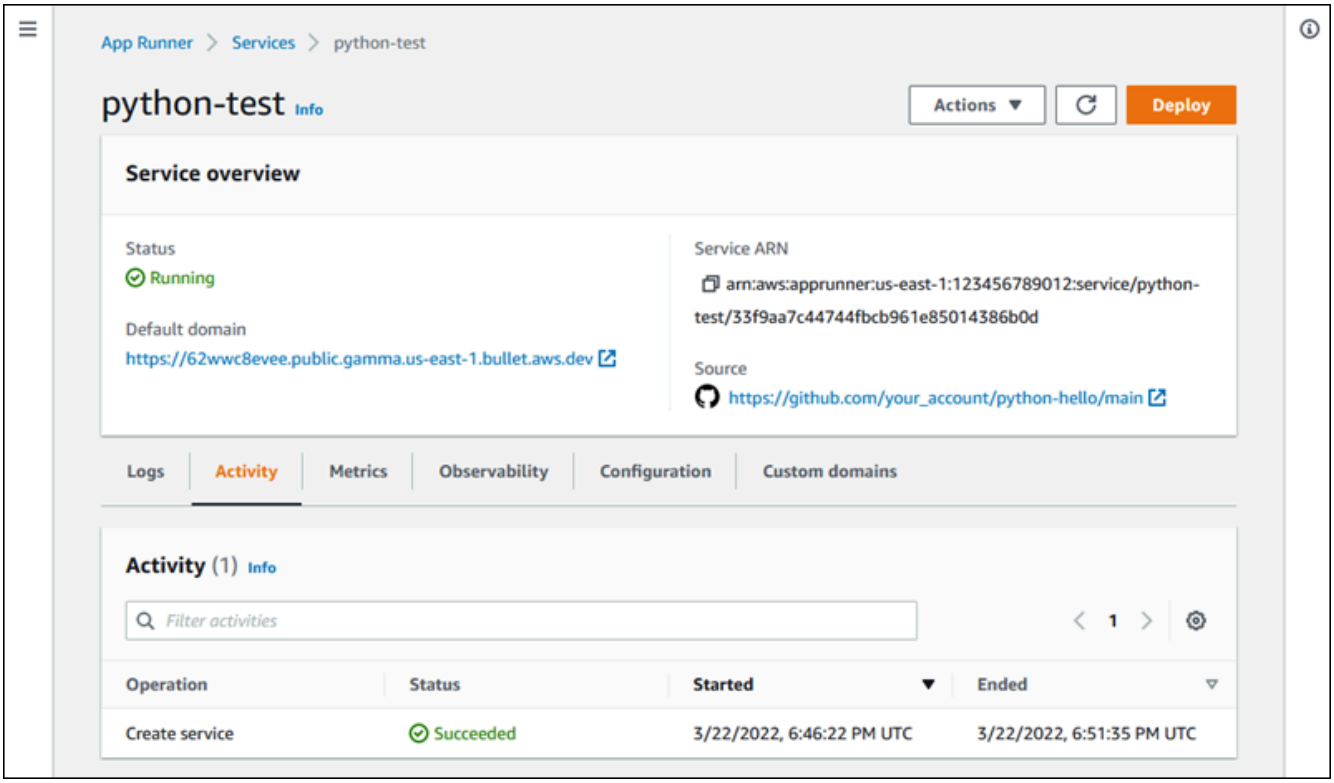
Elimina il servizio App Runner utilizzando uno dei seguenti metodi:

App Runner console

Per eliminare il servizio utilizzando la console App Runner

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Nel pannello di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

   La console mostra la dashboard del servizio con una panoramica del servizio.



3. Scegli Azioni, quindi Elimina.

   La console ti porta alla pagina Servizi. Il servizio eliminato si troverà nello stato Operazione in corso, poi scomparirà dall'elenco. Il servizio è ora eliminato.

App Runner API or AWS CLI

Per eliminare il servizio utilizzando l'API App Runner oppure AWS CLI, richiama l'azione DeleteServiceAPI. Se la chiamata restituisce una risposta corretta con un oggetto Service visualizzato`"Status": "OPERATION_IN_PROGRESS"`, App Runner inizia a eliminare il servizio.

Riferimento alle variabili di ambiente

Con App Runner, puoi fare riferimento a segreti e configurazioni come variabili di ambiente nel tuo servizio quando crei un servizio o aggiorni un servizio.

È possibile fare riferimento a dati di configurazione non sensibili come i timeout e il conteggio dei tentativi in testo normale come coppie chiave-valore. I dati di configurazione a cui fai riferimento in Plain Text non sono crittografati e sono visibili agli altri nella configurazione del servizio App Runner e nei registri delle applicazioni.

> **Note**
>
> Per motivi di sicurezza, non fate riferimento ai dati sensibili in Plain Text nel servizio App Runner.

Riferimento ai dati sensibili come variabili di ambiente

App Runner supporta il riferimento sicuro ai dati sensibili come variabili di ambiente nel servizio. Valuta la possibilità di archiviare i dati sensibili a cui desideri fare riferimento nel AWS Secrets Managernostro Parameter Store.AWS Systems Manager Quindi, puoi farvi riferimento in modo sicuro nel tuo servizio come variabili di ambiente dalla console App Runner o chiamando l'API. Ciò separa efficacemente la gestione dei segreti e dei parametri dalla configurazione del codice dell'applicazione e del servizio, migliorando la sicurezza complessiva delle applicazioni in esecuzione su App Runner.

> **Note**
>
> App Runner non ti addebita alcun costo per fare riferimento a Secrets Manager e SSM Parameter Store come variabili di ambiente. Tuttavia, paghi un prezzo standard per l'utilizzo di Secrets Manager e SSM Parameter Store.
> Per ulteriori informazioni sui prezzi, vedi i seguenti argomenti:
>
> - AWS Prezzi di Secrets Manager
> - AWS Prezzi di SSM Parameter Store

Di seguito è riportato il processo per fare riferimento ai dati sensibili come variabili di ambiente:

1. Archivia dati sensibili, come chiavi API, credenziali del database, parametri di connessione al database o versioni delle applicazioni come segreti o parametri in uno AWS Secrets Manager o più AWS Systems Manager Parameter Store.
2. Aggiorna la policy IAM del tuo ruolo di istanza in modo che App Runner possa accedere ai segreti e ai parametri archiviati in Secrets Manager e SSM Parameter Store. Per ulteriori informazioni, consulta l'argomento relativo alle autorizzazioni .
3. Fai riferimento in modo sicuro ai segreti e ai parametri come variabili di ambiente assegnando un nome e fornendo il relativo Amazon Resource Name (ARN). Puoi aggiungere variabili di ambiente quando crei un servizio o aggiorni la configurazione di un servizio. È possibile utilizzare una delle seguenti opzioni per aggiungere variabili di ambiente:
   - Console App Runner
   - API App Runner
   - `apprunner.yaml`file di configurazione

   > **Note**
   >
   > Non è `PORT` possibile assegnare un nome a una variabile di ambiente durante la creazione o l'aggiornamento del servizio App Runner. È una variabile di ambiente riservata per il servizio App Runner.

   Per ulteriori informazioni su come fare riferimento a segreti e parametri, consulta Gestione delle variabili di ambiente.

> **Note**
>
> Poiché App Runner memorizza solo il riferimento al segreto e al parametro ARNs, i dati sensibili non sono visibili agli altri nella configurazione del servizio App Runner e nei registri delle applicazioni.

Considerazioni

- Assicurati di aggiornare il ruolo dell'istanza con le autorizzazioni appropriate per accedere ai segreti e ai parametri in AWS Secrets Manager o in Parameter Store. AWS Systems Manager Per ulteriori informazioni, consulta l'argomento relativo alle autorizzazioni .
- Assicurati che AWS Systems Manager Parameter Store sia lo Account AWS stesso del servizio che desideri avviare o aggiornare. Al momento, non è possibile fare riferimento ai parametri di SSM Parameter Store tra gli account.
- Quando i segreti e i valori dei parametri vengono ruotati o modificati, non vengono aggiornati automaticamente nel servizio App Runner. Ridistribuisci il servizio App Runner poiché App Runner recupera segreti e parametri solo durante la distribuzione.
- Hai anche la possibilità di chiamare AWS Secrets Manager direttamente AWS Systems Manager Parameter Store tramite l'SDK del tuo servizio App Runner.
- Per evitare errori, assicuratevi di quanto segue quando li referenziate come variabili di ambiente:
  - Specificate l'ARN corretto del segreto.
  - È necessario specificare il nome o l'ARN corretto del parametro.

Autorizzazioni

Per abilitare il riferimento a segreti e parametri archiviati nel AWS Secrets Manager o SSM Parameter Store, aggiungi le autorizzazioni appropriate alla policy IAM del tuo ruolo di istanza per accedere a Secrets Manager e SSM Parameter Store.

> **Note**
>
> App Runner non può accedere alle risorse del tuo account senza la tua autorizzazione. Fornisci l'autorizzazione aggiornando la tua policy IAM.

Puoi utilizzare i seguenti modelli di policy per aggiornare il ruolo dell'istanza nella console IAM. Puoi modificare questi modelli di policy per soddisfare i tuoi requisiti specifici. Per ulteriori informazioni sull'aggiornamento di un ruolo di istanza, consulta Modifying a role nella IAM User Guide.

> **Note**
>
> Puoi anche copiare i seguenti modelli dalla console App Runner durante la creazione delle variabili di ambiente.

Copia il seguente modello nel tuo ruolo di istanza da cui aggiungere l'autorizzazione a fare riferimento ai segreti. AWS Secrets Manager

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "kms:Decrypt*"
            ],
            "Resource": [

 "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:<secret_name>",
                "arn:aws:kms:<region>:<aws_account_id>:key/<key_id>"
            ]
        }
    ]
}
```

Copia il seguente modello nel tuo ruolo di istanza per aggiungere l'autorizzazione ai parametri di riferimento da AWS Systems ManagerParameter Store.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
                "arn:aws:ssm:<region>:<aws_account_id>:parameter/
<parameter_name>"
            ]
        }
    ]
}
```

Gestione delle variabili di ambiente

Gestisci le variabili di ambiente per il tuo servizio App Runner utilizzando uno dei seguenti metodi:

- the section called “Console App Runner”
- the section called “API App Runner o AWS CLI”

Console App Runner

Quando crei un servizio o aggiorni un servizio sulla console App Runner, puoi aggiungere variabili di ambiente.

Aggiungere variabili di ambiente

Per aggiungere una variabile di ambiente

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua Regione AWS.
2. A seconda che tu stia creando o aggiornando un servizio, esegui una delle seguenti operazioni:
   - Se stai creando un nuovo servizio, scegli Crea un servizio App Runner e vai a Configura servizio.
   - Se stai aggiornando un servizio esistente, seleziona il servizio che desideri aggiornare e vai alla scheda Configurazione del servizio.
3. Vai a Variabili di ambiente (facoltativo) in Impostazioni del servizio.
4. Scegliete una delle seguenti opzioni in base alle vostre esigenze:

- Scegliete Testo normale dall'origine della variabile di ambiente e inserite le relative coppie chiave-valore rispettivamente in Nome della variabile di ambiente e Valore della variabile di ambiente.

  > **Note**
  >
  > Scegliete Testo normale se desiderate fare riferimento a dati non sensibili. Questi dati non sono crittografati e sono visibili agli altri nella configurazione del servizio App Runner e nei registri delle applicazioni.

- Scegli Secrets Manager dall'origine della variabile di ambiente per fare riferimento al segreto archiviato AWS Secrets Manager come variabile di ambiente nel tuo servizio. Fornisci il nome della variabile di ambiente e Amazon Resource Name (ARN) del segreto a cui fai riferimento rispettivamente in Environment variable name e Environment variable value.
- Scegli SSM Parameter Store dall'origine della variabile di ambiente per fare riferimento al parametro memorizzato in SSM Parameter Store come variabile di ambiente nel tuo servizio. Fornisci il nome della variabile di ambiente e l'ARN del parametro a cui fai riferimento rispettivamente in Nome variabile di ambiente e Valore variabile di ambiente.

  > **Note**
  >
  > - Non è `PORT` possibile assegnare un nome a una variabile di ambiente durante la creazione o l'aggiornamento del servizio App Runner. È una variabile di ambiente riservata per il servizio App Runner.
  > - Se il parametro SSM Parameter Store è lo Regione AWS stesso del servizio che desideri avviare, puoi specificare l'Amazon Resource Name (ARN) completo o il nome del parametro. Se il parametro si trova in una regione diversa, è necessario specificare l'ARN completo.
  > - Assicurati che il parametro a cui fai riferimento si trovi nello stesso account del servizio che stai avviando o aggiornando. Al momento, non puoi fare riferimento al parametro SSM Parameter Store tra gli account.

5. Scegli Aggiungi variabile di ambiente per fare riferimento a un'altra variabile di ambiente.
6. Espandi i modelli di policy IAM per visualizzare e copiare i modelli di policy IAM forniti per SSM Parameter Store. AWS Secrets Manager Devi farlo solo se non hai ancora aggiornato la policy

IAM del tuo ruolo di istanza con le autorizzazioni richieste. Per ulteriori informazioni, consulta l'argomento relativo alle autorizzazioni .

Rimozione della variabile di ambiente

Prima di eliminare una variabile di ambiente, assicuratevi che il codice dell'applicazione sia aggiornato in modo da riflettere lo stesso. Se il codice dell'applicazione non viene aggiornato, il servizio App Runner potrebbe non funzionare.

Per rimuovere le variabili di ambiente

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua Regione AWS.
2. Vai alla scheda Configurazione del servizio che desideri aggiornare.
3. Vai a Variabili di ambiente (facoltativo) in Impostazioni del servizio.
4. Scegli Rimuovi accanto alla variabile di ambiente che desideri rimuovere. Riceverai un messaggio per confermare l'eliminazione.
5. Scegliere Delete (Elimina).

API App Runner o AWS CLI

Puoi fare riferimento ai dati sensibili archiviati in Secrets Manager e SSM Parameter Store aggiungendoli come variabili di ambiente nel tuo servizio.

> **Note**
>
> Aggiorna la policy IAM del tuo ruolo di istanza in modo che App Runner possa accedere a segreti e parametri archiviati in Secrets Manager e SSM Parameter Store. Per ulteriori informazioni, consulta l'argomento relativo alle autorizzazioni .

Per fare riferimento a segreti e configurazioni come variabili di ambiente

1. Crea un segreto o una configurazione in Secrets Manager o SSM Parameter Store.

   Gli esempi seguenti mostrano come creare un segreto e un parametro utilizzando SSM Parameter Store.

Example Creazione di un segreto - Richiesta

L'esempio seguente mostra come creare un segreto che rappresenti la credenziale del database.

```
aws secretsmanager create-secret \
–name DevRdsCredentials \
–description "Rds credentials for development account." \
–secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
```

Example Creazione di un segreto - Risposta

```
arn:aws:secretsmanager:<region>:<aws_account_id>:secret:DevRdsCredentials
```

Example Creazione di una configurazione - Richiesta

L'esempio seguente mostra come creare un parametro che rappresenti la stringa di connessione RDS.

```
aws systemsmanager put-parameter \
–name DevRdsConnectionString \
–value "mysql2://dev-mysqlcluster-rds.com:3306/diegor" \
–type "String" \
–description "Rds connection string for development account."
```

Example Creazione di una configurazione - Risposta

```
arn:aws:ssm:<region>:<aws_account_id>:parameter/DevRdsConnectionString
```

2. Fai riferimento ai segreti e alle configurazioni archiviati in Secrets Manager e SSM Parameter Store aggiungendoli come variabili di ambiente. È possibile aggiungere variabili di ambiente quando si crea o si aggiorna il servizio App Runner.

   Gli esempi seguenti mostrano come fare riferimento a segreti e configurazioni come variabili di ambiente in un servizio App Runner basato su codice e su immagini.

   Example File.json di input per il servizio App Runner basato su immagini

```
{
  "ServiceName": "example-secrets",
```

```
  "SourceConfiguration": {
    "ImageRepository": {
      "ImageIdentifier": "<image-identifier>",
      "ImageConfiguration": {
        "Port": "<port>",
        "RuntimeEnvironmentSecrets": {

 "Credential1":"arn:aws:secretsmanager:<region>:<aws_account_id>:secret:XXXXXXXXXXXX",
          "Credential2":"arn:aws:ssm:<region>:<aws_account_id>:parameter/
<parameter-name>"
        }
      },
      "ImageRepositoryType": "ECR_PUBLIC"
    }
  },
  "InstanceConfiguration": {
    "Cpu": "1 vCPU",
    "Memory": "3 GB",
    "InstanceRoleArn": "<instance-role-arn>"
  }
}
```

Example Servizio App Runner basato su immagini: richiesta

```
aws apprunner create-service \
--cli-input-json file://input.json
```

Example Servizio App Runner basato su immagini — Risposta

```
{
...
      "ImageRepository": {
         "ImageIdentifier":"<image-identifier>",
         "ImageConfiguration":{
            "Port": "<port>",
            "RuntimeEnvironmentSecrets":{
               "Credential1":
 "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:XXXXXXXXXXXX",
               "Credential2": "arn:aws:ssm:<region>:<aws_account_id>:parameter/
<parameter-name>"
            },
            "ImageRepositoryType":"ECR"
```

```
      }
   },
   "InstanceConfiguration": {
        "CPU": "1 vCPU",
        "Memory": "3 GB",
        "InstanceRoleArn: "<instance-role-arn>"
   }
...
}
```

Example File.json di input per il servizio App Runner basato su codice

```
{
  "ServiceName": "example-secrets",
  "SourceConfiguration": {
    "AuthenticationConfiguration": {
      "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/my-
github-connection/XXXXXXXXXX"
    },
    "AutoDeploymentsEnabled": false,
    "CodeRepository": {
      "RepositoryUrl": "<repository-url>",
      "SourceCodeVersion": {
        "Type": "BRANCH",
        "Value": "main"
      },
      "CodeConfiguration": {
        "ConfigurationSource": "API",
        "CodeConfigurationValues": {
          "Runtime": "<runtime>",
          "BuildCommand": "<build-command>",
          "StartCommand": "<start-command>",
          "Port": "<port>",
          "RuntimeEnvironmentSecrets": {

 "Credential1":"arn:aws:secretsmanager:<region>:<aws_account_id>:secret:XXXXXXXXXXXX",
            "Credential2":"arn:aws:ssm:<region>:<aws_account_id>:parameter/
<parameter-name>"
          }
        }
      }
    }
  },
```

```
  "InstanceConfiguration": {
    "Cpu": "1 vCPU",
    "Memory": "3 GB",
    "InstanceRoleArn": "<instance-role-arn>"
  }
}
```

Example Servizio App Runner basato su codice: richiesta

```
aws apprunner create-service \
--cli-input-json file://input.json
```

Example Servizio App Runner basato su codice — Risposta

```
{
...
   "SourceConfiguration":{
      "CodeRepository":{
         "RepositoryUrl":"<repository-url>",
         "SourceCodeVersion":{
            "Type":"Branch",
            "Value":"main"
         },
         "CodeConfiguration":{
            "ConfigurationSource":"API",
            "CodeConfigurationValues":{
               "Runtime":"<runtime>",
               "BuildCommand":"<build-command>",
               "StartCommand":"<start-command>",
               "Port":"<port>",
               "RuntimeEnvironmentSecrets":{
                  "Credential1" :
 "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:XXXXXXXX",
                  "Credential2" : "arn:aws:ssm:<region>:<aws_account_id>:parameter/
<parameter-name>"
               }
            }
         }
      },
      "InstanceConfiguration": {
        "CPU": "1 vCPU",
        "Memory": "3 GB",
```

```
        "InstanceRoleArn: "<instance-role-arn>"
   }
...
}
```

3. Il `apprunner.yaml` modello viene aggiornato per riflettere i segreti aggiunti.

   Di seguito è riportato un esempio del `apprunner.yaml` modello aggiornato.

   Example **`apprunner.yaml`**

```
version: 1.0
runtime: python3
build:
  commands:
    build:
      - python -m pip install flask
run:
  command: python app.py
  network:
    port: 8080
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
  secrets:
    - name: my-secret
      value-from:
 "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:XXXXXXXXXXXX"
    - name: my-parameter
      value-from: "arn:aws:ssm:<region>:<aws_account_id>:parameter/<parameter-
name>"
    - name: my-parameter-only-name
      value-from: "parameter-name"
```

Rete con App Runner

Questo capitolo descrive le configurazioni di rete per i tuoi AWS App Runner servizi.

Da questo capitolo imparerai quanto segue:

- Come configurare il traffico in entrata per endpoint privati e pubblici. Per ulteriori informazioni, consulta Configurazione delle configurazioni di rete per il traffico in entrata.
- Come configurare il traffico in uscita per accedere ad altre applicazioni in esecuzione in un Amazon VPC. Per ulteriori informazioni, consulta Abilitazione dell'accesso VPC per il traffico in uscita.

> **Note**
>
> App Runner attualmente supporta il tipo di indirizzo dual-stack (IPv4 e IPv6) solo per il traffico pubblico in entrata. È supportato solo il traffico in uscita e il traffico privato in entrata. IPv4

Argomenti

Terminology

Per sapere come personalizzare il traffico di rete in base alle vostre esigenze, comprendiamo i seguenti termini utilizzati in questo capitolo.

Condizioni generali

Per sapere cosa è necessario associare a un Amazon Virtual Private Cloud (VPC), comprendiamo i seguenti termini:

- VPC: Amazon VPC è una rete virtuale logicamente isolata che ti offre il controllo completo del tuo ambiente di rete virtuale, inclusi posizionamento delle risorse, connettività e sicurezza. È una rete virtuale molto simile a una rete tradizionale che gestiresti nel tuo data center.

- Endpoint di interfaccia VPC: l'endpoint di interfaccia VPC, una risorsa AWS PrivateLink , collega un VPC a un servizio endpoint. Crea un endpoint con interfaccia VPC per inviare traffico ai servizi endpoint che utilizzano un Network Load Balancer per distribuire il traffico. Il traffico destinato al servizio endpoint viene risolto utilizzando il DNS.
- Regioni: ogni regione è un'area geografica separata in cui è possibile ospitare un servizio App Runner.
- Zone di disponibilità: una zona di disponibilità è una posizione isolata all'interno di una AWS regione. Si tratta di uno o più data center discreti con alimentazione, rete e connettività ridondanti. Le zone di disponibilità consentono di rendere le applicazioni di produzione altamente disponibili, tolleranti ai guasti e scalabili.
- Sottoreti: una sottorete è un intervallo di indirizzi IP nel tuo VPC. Una sottorete deve risiedere in una singola zona di disponibilità. È possibile avviare una AWS risorsa in una sottorete specificata. Utilizza una sottorete pubblica per le risorse che devono essere connesse a Internet e una sottorete privata per le risorse da non connettere a Internet.
- Gruppi di sicurezza: un gruppo di sicurezza controlla il traffico che può raggiungere e uscire dalle risorse a cui è associato. I gruppi di sicurezza forniscono un ulteriore livello di sicurezza per proteggere le AWS risorse in ogni sottorete, offrendovi un maggiore controllo sul traffico di rete. Al momento della creazione di un VPC, questo include un gruppo di sicurezza di default. È possibile creare gruppi di sicurezza aggiuntivi per ogni VPC. È possibile associare un gruppo di sicurezza solo alle risorse all'interno del VPC per cui è stato creato.
- Dual stack: un dual stack è un tipo di indirizzo che supporta il traffico di rete proveniente sia IPv4 dagli endpoint che dagli endpoint. IPv6

Termine specifico per la configurazione del traffico in uscita

Connettore VPC

Un connettore VPC è una risorsa App Runner che consente al servizio App Runner di accedere alle applicazioni eseguite in un Amazon VPC privato.

Termini specifici per la configurazione del traffico in entrata

Per sapere come puoi rendere i tuoi servizi accessibili privatamente solo dall'interno di un Amazon VPC, comprendiamo i seguenti termini:

- Connessione VPC Ingress: VPC Ingress Connection è una risorsa App Runner che fornisce un endpoint App Runner per il traffico in entrata. App Runner assegna la risorsa VPC Ingress

Connection dietro le quinte quando scegli Endpoint privato sulla console App Runner per il traffico in entrata. La risorsa VPC Ingress Connection collega il servizio App Runner all'endpoint dell'interfaccia VPC di Amazon VPC.

> **Note**
>
> Se utilizzi l'API App Runner, la risorsa VPC Ingress Connection non viene creata automaticamente.

- Endpoint privato: l'endpoint privato è un'opzione della console App Runner selezionata per configurare il traffico di rete in entrata in modo che sia accessibile solo all'interno di un Amazon VPC.

Configurazione delle configurazioni di rete per il traffico in entrata

Puoi configurare il tuo servizio per ricevere traffico in entrata da endpoint privati o pubblici.

Un endpoint pubblico è la configurazione predefinita. Apre il servizio a qualsiasi traffico in entrata proveniente dalla rete Internet pubblica. Inoltre, offre la flessibilità di scegliere tra il protocollo Internet versione 4 (IPv4) o il tipo di indirizzo dual-stack (IPv4 and IPv6) per il servizio.

Un endpoint privato consente solo al traffico proveniente da un Amazon VPC di accedere al servizio App Runner. Ciò si ottiene configurando un endpoint di interfaccia VPC, una AWS PrivateLink risorsa, per il servizio App Runner. In tal modo, si crea una connessione privata tra Amazon VPC e il servizio App Runner.

> **Note**
>
> App Runner attualmente supporta il tipo di indirizzo dual-stack (IPv4 e IPv6) solo per gli endpoint pubblici. Per gli endpoint privati, è supportato solo. IPv4

Di seguito sono riportati gli argomenti trattati nell'ambito della configurazione delle configurazioni di rete per il traffico in entrata:

- Come configurare il traffico in entrata per rendere il servizio disponibile privatamente solo all'interno di un Amazon VPC. Per ulteriori informazioni, consulta Enabling Private Endpoint for Incoming Traffic.

- Come configurare il servizio per ricevere traffico Internet dal tipo di indirizzo dual-stack. Per ulteriori informazioni, consulta Abilitazione del dual stack per il traffico pubblico in entrata.

Headers

Con App Runner puoi accedere alla fonte IPv4 e IPv6 agli indirizzi originali del traffico che entra nella tua applicazione. Gli indirizzi IP di origine originali vengono conservati assegnando loro l'intestazione della `X-Forwarded-For` richiesta. Ciò consente alle applicazioni di recuperare gli indirizzi IP di origine originali quando necessario.

> **Note**
>
> Se il servizio è configurato per utilizzare un endpoint privato, l'intestazione della `X-Forwarded-For` richiesta non può essere utilizzata per accedere agli indirizzi IP di origine originali. Se utilizzato, recupera valori falsi.

Abilitazione dell'endpoint privato per il traffico in entrata

Per impostazione predefinita, quando crei un AWS App Runner servizio, il servizio è accessibile tramite Internet. Tuttavia, puoi anche rendere il tuo servizio App Runner privato e accessibile solo all'interno di un Amazon Virtual Private Cloud (Amazon VPC).

Con il servizio App Runner privato, hai il controllo completo sul traffico in entrata, aggiungendo un ulteriore livello di sicurezza. Ciò è utile in una varietà di casi d'uso, tra cui l'esecuzione di applicazioni Web aziendali interne APIs o di applicazioni ancora in fase di sviluppo che richiedono un livello maggiore di privacy e sicurezza o che devono soddisfare requisiti di conformità specifici.

> **Note**
>
> Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché il Web WAF. ACLs Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Di conseguenza, le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP.
> Per ulteriori informazioni sulla sicurezza dell'infrastruttura e sui gruppi di sicurezza, comprese le best practice, consulta i seguenti argomenti nella Guida per l'utente di Amazon VPC:

Controllo del traffico di rete e Controllo del traffico verso le risorse AWS utilizzando gruppi di sicurezza.

Quando il servizio App Runner è privato, puoi accedervi da un Amazon VPC. Non è richiesto un gateway Internet, un dispositivo NAT o una connessione VPN.

> **Note**
>
> App Runner attualmente supporta il tipo di indirizzo dual-stack (IPv4 e IPv6) solo per il traffico pubblico in entrata. È supportato solo il traffico in uscita e il traffico privato in entrata. IPv4

Considerazioni

- Prima di configurare un endpoint di interfaccia VPC per App Runner, consulta le considerazioni nella Guida.AWS PrivateLink
- Le policy degli endpoint VPC non sono supportate per App Runner. Per impostazione predefinita, l'accesso completo ad App Runner è consentito tramite l'endpoint dell'interfaccia VPC. In alternativa, puoi associare un gruppo di sicurezza alle interfacce di rete degli endpoint per controllare il traffico verso App Runner attraverso l'endpoint dell'interfaccia VPC.
- Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché il Web WAF. ACLs Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Di conseguenza, le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP.
- Dopo aver abilitato un endpoint privato, il servizio è accessibile solo dal tuo VPC e non da Internet.
- Per una maggiore disponibilità, si consiglia di selezionare almeno due sottoreti nella zona di disponibilità diverse per l'endpoint dell'interfaccia VPC. Non è consigliabile utilizzare una sola sottorete.
- Puoi utilizzare lo stesso endpoint di interfaccia VPC per accedere a più servizi App Runner in un VPC.

Per informazioni sui termini utilizzati in questa sezione, vedere Terminologia.

Autorizzazioni

Di seguito è riportato l'elenco delle autorizzazioni necessarie per abilitare Private Endpoint:

- ec2: CreateTags
- ec2: CreateVpcEndpoint
- ec2: ModifyVpcEndpoint
- ec2: DeleteVpcEndpoints
- ec2: DescribeSubnets
- ec2: DescribeVpcEndpoints
- ec2: DescribeVpcs

Endpoint dell'interfaccia VPC

Un endpoint di interfaccia VPC è una AWS PrivateLinkrisorsa che collega un Amazon VPC a un servizio endpoint. Puoi specificare in quale Amazon VPC desideri rendere accessibile il tuo servizio App Runner passando un endpoint di interfaccia VPC. Per creare un endpoint di interfaccia VPC, specificare quanto segue:

- Amazon VPC per abilitare la connettività.
- Aggiungi gruppi di sicurezza. Per impostazione predefinita, un gruppo di sicurezza viene assegnato all'endpoint dell'interfaccia VPC. Puoi scegliere di associare un gruppo di sicurezza personalizzato per aumentare il controllo del traffico di rete in entrata.
- Aggiungi sottoreti. Per garantire una maggiore disponibilità, si consiglia di selezionare almeno due sottoreti per ogni zona di disponibilità da cui accedere al servizio App Runner. Un endpoint di interfaccia di rete viene creato in ogni sottorete abilitata per l'endpoint dell'interfaccia VPC. Si tratta di interfacce di rete gestite dai richiedenti che fungono da punto di ingresso per il traffico destinato ad App Runner. Un'interfaccia di rete gestita dal richiedente è un'interfaccia di rete che un servizio AWS crea nel VPC per conto dell'utente.
- Se utilizzi l'API, aggiungi l'endpoint dell'interfaccia VPC App Runner. `Servicename` Ad esempio,

```
com.amazonaws.region.apprunner.requests
```

È possibile creare un endpoint di interfaccia VPC utilizzando uno dei seguenti servizi: AWS

- Console App Runner. Per ulteriori informazioni, consulta Manage Private Endpoint.
- Console o API Amazon VPC e AWS Command Line Interface ()AWS CLI. Per ulteriori informazioni, consulta Access Servizi AWS through AWS PrivateLink nella AWS PrivateLink Guida.

> **Note**
>
> Ti viene addebitato un costo per ogni endpoint di interfaccia VPC che utilizzi in base ai prezzi.AWS PrivateLink Pertanto, per una migliore efficienza in termini di costi, puoi utilizzare lo stesso endpoint di interfaccia VPC per accedere a più servizi App Runner all'interno di un VPC. Tuttavia, per un migliore isolamento, prendi in considerazione l'associazione di un endpoint di interfaccia VPC diverso per ciascuno dei tuoi servizi App Runner.

Connessione in ingresso del VPC

Una connessione di ingresso VPC è una risorsa App Runner che specifica un endpoint App Runner per il traffico in entrata. App Runner assegna la risorsa VPC Ingress Connection dietro le quinte quando scegli Endpoint privato sulla console App Runner per il traffico in entrata. Scegli questa opzione per consentire solo al traffico proveniente da un Amazon VPC di accedere al tuo servizio App Runner. La risorsa VPC Ingress Connection collega il servizio App Runner all'endpoint dell'interfaccia VPC di Amazon VPC. Puoi creare una risorsa VPC Ingress Connection solo se utilizzi le operazioni API per configurare le impostazioni di rete per il traffico in entrata. Per ulteriori informazioni su come creare la risorsa VPC Ingress Connection, consulta l'AWS App Runner API CreateVpcIngressConnectionReference.

> **Note**
>
> Una risorsa VPC Ingress Connection di App Runner può connettersi a un endpoint di interfaccia VPC di Amazon VPC. Inoltre, puoi creare una sola risorsa VPC Ingress Connection per ogni servizio App Runner.

Endpoint privato

L'endpoint privato è un'opzione della console App Runner che puoi scegliere se desideri ricevere solo traffico in entrata da un Amazon VPC. La scelta dell'opzione Endpoint privato sulla console App Runner offre la possibilità di connettere il servizio a un VPC configurando l'endpoint dell'interfaccia

VPC. Dietro le quinte, App Runner assegna una risorsa VPC Ingress Connection all'endpoint dell'interfaccia VPC che configuri.

> **Note**
>
> Per Private Endpoint è IPv4 supportato solo il traffico di rete.

Riepilogo

Rendi privato il tuo servizio consentendo solo al traffico proveniente da un Amazon VPC di accedere al tuo servizio App Runner. A tal fine, crei un endpoint di interfaccia VPC per l'Amazon VPC selezionato utilizzando App Runner o Amazon VPC. Sulla console App Runner, crei un endpoint di interfaccia VPC quando abiliti l'endpoint privato per il traffico in entrata. App Runner crea quindi automaticamente una risorsa VPC Ingress Connection e si connette all'endpoint dell'interfaccia VPC e al servizio App Runner. In questo modo viene creata una connessione di servizio privata che garantisce che solo il traffico proveniente dal VPC selezionato possa accedere al servizio App Runner.

Gestione dell'endpoint privato

Gestisci l'endpoint privato per il traffico in entrata utilizzando uno dei seguenti metodi:

- the section called “Console App Runner”
- the section called “API App Runner o AWS CLI”

> **Note**
>
> Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché il Web WAF. ACLs Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Di conseguenza, le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP.
> Per ulteriori informazioni sulla sicurezza dell'infrastruttura e sui gruppi di sicurezza, comprese le best practice, consulta i seguenti argomenti nella Guida per l'utente di Amazon VPC:

Controllo del traffico di rete e Controllo del traffico verso le risorse AWS utilizzando gruppi di sicurezza.

Console App Runner

Quando crei un servizio utilizzando la console App Runner o quando ne aggiorni la configurazione in un secondo momento, puoi scegliere di configurare il traffico in entrata.

Per configurare il traffico in entrata, scegli una delle seguenti opzioni.

- Endpoint pubblico: per rendere il servizio accessibile a tutti i servizi su Internet. Per impostazione predefinita, è selezionato Public endpoint.
- Endpoint privato: per rendere il servizio App Runner accessibile solo all'interno di un Amazon VPC.

> **Note**
>
> Attualmente, App Runner supporta IPv6 solo gli endpoint pubblici. IPv6 gli endpoint non sono supportati per i servizi App Runner ospitati in un Amazon Virtual Private Cloud (Amazon VPC). Se aggiorni un servizio che utilizza un endpoint pubblico dual-stack a un endpoint privato, per impostazione predefinita il servizio App Runner supporta il traffico proveniente solo IPv4 dagli endpoint e non riceve traffico dagli endpoint. IPv6

Abilita l'endpoint privato

Abilita un endpoint privato associandolo all'endpoint dell'interfaccia VPC dell'Amazon VPC a cui desideri accedere. Puoi creare un nuovo endpoint di interfaccia VPC o sceglierne uno esistente.

Per creare un endpoint di interfaccia VPC

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Vai alla sezione Rete in Configura servizio.
3. Scegli Endpoint privato, per il traffico di rete in entrata. Si aprono le opzioni per connettersi a un VCP utilizzando l'endpoint dell'interfaccia VPC.
4. Scegli Crea nuovo endpoint. Viene visualizzata la finestra di dialogo Crea nuovo endpoint dell'interfaccia VPC.
5. Inserisci un nome per l'endpoint dell'interfaccia VPC.

6. Scegli l'endpoint dell'interfaccia VPC richiesto dall'elenco a discesa disponibile.
7. Scegli il gruppo di sicurezza dall'elenco a discesa. L'aggiunta di gruppi di sicurezza fornisce un ulteriore livello di sicurezza all'endpoint dell'interfaccia VPC. Si consiglia di scegliere due o più gruppi di sicurezza. Se non scegli un gruppo di sicurezza, App Runner assegna un gruppo di sicurezza predefinito all'endpoint dell'interfaccia VPC. Assicurati che le regole del gruppo di sicurezza non blocchino le risorse che desiderano comunicare con il tuo servizio App Runner. Le regole del gruppo di sicurezza devono consentire alle risorse di interagire con il servizio App Runner.

   > **Note**
   >
   > Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché il Web WAF. ACLs Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Di conseguenza, le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP.
   > Per ulteriori informazioni sulla sicurezza dell'infrastruttura e sui gruppi di sicurezza, comprese le best practice, consulta i seguenti argomenti nella Guida per l'utente di Amazon VPC: Controllo del traffico di rete e Controllo del traffico verso le risorse AWS utilizzando gruppi di sicurezza.

8. Scegli le sottoreti richieste dall'elenco a discesa. Si consiglia di selezionare almeno due sottoreti per ogni zona di disponibilità da cui accedere al servizio App Runner.
9. (Facoltativo) Scegli Aggiungi nuovo tag e inserisci la chiave del tag e il valore del tag.
10. Scegli Create (Crea) . Si apre la pagina Configura servizio che mostra il messaggio di corretta creazione dell'endpoint dell'interfaccia VPC sulla barra in alto.

Per scegliere un endpoint di interfaccia VPC esistente

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Vai alla sezione Rete in Configura servizio.
3. Scegli Endpoint privato, per il traffico di rete in entrata. Si aprono le opzioni per connettersi a un VPC utilizzando l'endpoint dell'interfaccia VPC. Viene visualizzato un elenco degli endpoint dell'interfaccia VPC disponibili.
4. Scegli l'endpoint di interfaccia VPC richiesto elencato in Endpoint dell'interfaccia VPC.

5. Scegli Avanti per creare il tuo servizio. App Runner abilita l'endpoint privato.

> **Note**
>
> Dopo aver creato il servizio, puoi scegliere di modificare i gruppi di sicurezza e le sottoreti associati all'endpoint dell'interfaccia VPC, se necessario.

Per controllare i dettagli dell'endpoint privato, vai al tuo servizio ed espandi la sezione Rete nella scheda Configurazione. Mostra i dettagli del VPC e dell'endpoint dell'interfaccia VPC associati all'endpoint privato.

Aggiorna l'endpoint dell'interfaccia VPC

Dopo aver creato il servizio App Runner, puoi modificare l'endpoint dell'interfaccia VPC associato all'endpoint privato.

> **Note**
>
> Non è possibile aggiornare il nome dell'endpoint e i campi VPC.

Per aggiornare l'endpoint dell'interfaccia VPC

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Vai al tuo servizio e scegli Configurazioni di rete nel pannello di sinistra.
3. Scegli Traffico in entrata per visualizzare gli endpoint dell'interfaccia VPC associati ai rispettivi servizi.
4. Scegli l'endpoint dell'interfaccia VPC che desideri modificare.
5. Scegli Modifica. Si apre la finestra di dialogo per modificare l'endpoint dell'interfaccia VPC.
6. Scegli i gruppi di sicurezza e le sottoreti richiesti e fai clic su Aggiorna. La pagina che mostra i dettagli dell'endpoint dell'interfaccia VPC si apre con il messaggio di aggiornamento riuscito dell'endpoint dell'interfaccia VPC sulla barra in alto.

> **Note**
>
> Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché il Web WAF. ACLs Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Di conseguenza, le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP.
> Per ulteriori informazioni sulla sicurezza dell'infrastruttura e sui gruppi di sicurezza, comprese le best practice, consulta i seguenti argomenti nella Guida per l'utente di Amazon VPC: Controllo del traffico di rete e Controllo del traffico verso le risorse AWS utilizzando gruppi di sicurezza.

Eliminare l'endpoint dell'interfaccia VPC

Se non desideri che il servizio App Runner sia accessibile privatamente, puoi impostare il traffico in entrata su Pubblico. La modifica a Pubblico rimuove l'endpoint privato, ma non elimina l'endpoint dell'interfaccia VPC

Per eliminare l'endpoint dell'interfaccia VPC

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Vai al tuo servizio e scegli Configurazioni di rete nel pannello di sinistra.
3. Scegli Traffico in entrata per visualizzare gli endpoint dell'interfaccia VPC associati ai rispettivi servizi.

   > **Note**
   >
   > Prima di eliminare un endpoint di interfaccia VPC, rimuovilo da tutti i servizi a cui è connesso aggiornando il servizio.

4. Scegli Elimina.

   Se ci sono servizi collegati all'endpoint dell'interfaccia VPC, riceverai il messaggio Impossibile eliminare l'endpoint dell'interfaccia VPC. Se non ci sono servizi collegati all'endpoint dell'interfaccia VPC, riceverai un messaggio per confermare l'eliminazione.

5. Scegli Elimina. Si apre la pagina delle configurazioni di rete per il traffico in entrata con il messaggio di avvenuta eliminazione dell'endpoint dell'interfaccia VPC nella barra superiore.

API App Runner o AWS CLI

Puoi distribuire un'applicazione su App Runner accessibile solo dall'interno di un Amazon VPC.

Per informazioni sulle autorizzazioni necessarie per rendere privato il servizio, consulta. the section called “Autorizzazioni”

> **Note**
>
> Attualmente, App Runner supporta IPv6 solo gli endpoint pubblici. IPv6 gli endpoint non sono supportati per i servizi App Runner ospitati in un Amazon Virtual Private Cloud (Amazon VPC). Se aggiorni un servizio che utilizza un endpoint pubblico dual-stack a un endpoint privato, per impostazione predefinita il servizio App Runner supporta il traffico proveniente solo IPv4 dagli endpoint e non riceve traffico dagli endpoint. IPv6

Per creare una connessione di servizio privata ad Amazon VPC

1. Crea un endpoint di interfaccia VPC, una AWS PrivateLink risorsa, per connetterti ad App Runner. Per fare ciò, specifica le sottoreti e i gruppi di sicurezza da associare all'applicazione. Di seguito è riportato un esempio di creazione di un endpoint di interfaccia VPC.

   > **Note**
   >
   > Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché le regole del gruppo di sicurezza per gli endpoint privati anziché WAF Web. ACLs Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Di conseguenza, le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP.
   > Per ulteriori informazioni sulla sicurezza dell'infrastruttura e sui gruppi di sicurezza, comprese le best practice, consulta i seguenti argomenti nella Guida per l'utente di Amazon VPC: Controllo del traffico di rete e Controllo del traffico verso le risorse AWS utilizzando gruppi di sicurezza.

Example

```
aws ec2 create-vpc-endpoint
 --vpc-endpoint-type: Interface
 --service-name: com.amazonaws.us-east-1.apprunner.requests
 --subnets: subnet1, subnet2
 --security-groups: sg1
```

2. Fai riferimento all'endpoint dell'interfaccia VPC utilizzando le azioni dell'API CreateServiceo UpdateServiceApp Runner tramite la CLI. Configura il tuo servizio in modo che non sia accessibile al pubblico. `IsPubliclyAccessible`Imposta su `False` nel `IngressConfiguration` membro del `NetworkConfiguration` parametro. Di seguito è riportato un esempio di riferimento all'endpoint dell'interfaccia VPC.

   Example

   ```
   aws apprunner create-service
    --network-configuration: ingress-configuration=<ingress_configuration>
    --service-name: com.amazonaws.us-east-1.apprunner.requests
    --source-configuration: <source_configuration>
    # Ingress Configuration
    {
    "IsPubliclyAccessible": False
    }
   ```

3. Richiama l'azione `create-vpc-ingress-connection` API per creare la risorsa VPC Ingress Connection per App Runner e associarla all'endpoint dell'interfaccia VPC creato nel passaggio precedente. Restituisce un nome di dominio utilizzato per accedere al servizio nel VPC specificato. Di seguito è riportato un esempio di creazione di una risorsa VPC Ingress Connection.

   Example Richiesta

   ```
   aws apprunner create-vpc-ingress-connection
    --service-arn: <apprunner_service_arn>
    --ingress-vpc-configuration: {"VpcId":<vpc_id>, "VpceId": <vpce_id>}
    --vpc-ingress-connection-name: <vic_connection_name>
   ```

Example Risposta

```
{
   "VpcIngressConnectionArn": <vpc_ingress_connection_arn>,
   "VpcIngressConnectionName": <vic_connection_name>,
   "ServiceArn": <apprunner_service_arn>,
   "Status": "PENDING_CREATION",
   "AccountId": <connection_owner_id>,
   "DomainName": <domain_name_associated_with_vpce>,
   "IngressVpcConfiguration": {"VpcId":<vpc_id>, "VpceId":<vpce_id>},
   "CreatedAt": <date_created>
}
```

Aggiorna la connessione VPC Ingress

È possibile aggiornare la risorsa VPC Ingress Connection. La connessione di ingresso VPC deve trovarsi in uno dei seguenti stati per essere aggiornata:

- DISPONIBILE
- CREAZIONE FALLITA
- AGGIORNAMENTO_FALLITO

Di seguito è riportato un esempio di aggiornamento di una risorsa VPC Ingress Connection.

Example Richiesta

```
aws apprunner update-vpc-ingress-connection
      --vpc-ingress-connection-arn: <vpc_ingress_connection_arn>
```

Example Risposta

```
{
    "VpcIngressConnectionArn":  <vpc_ingress_connection_arn>,
    "VpcIngressConnectionName":  <vic_connection_name>,
    "ServiceArn":  <apprunner_service_arn>,
    "Status": "FAILED_UPDATE",
    "AccountId":  <connection_owner_id>,
    "DomainName":  <domain_name_associated_with_vpce>,
```

```
    "IngressVpcConfiguration": {"VpcId":<vpc_id>, "VpceId":<vpce_id>},
    "CreatedAt":  <date_created>
}
```

Eliminare la connessione di ingresso VPC

Puoi eliminare la risorsa VPC Ingress Connection se non hai più bisogno della connessione privata ad Amazon VPC.

La connessione di ingresso VPC deve trovarsi in uno dei seguenti stati per essere eliminata:

- DISPONIBILE
- CREAZIONE NON RIUSCITA
- AGGIORNAMENTO NON RIUSCITO
- ELIMINAZIONE NON RIUSCITA

Di seguito è riportato un esempio di eliminazione di una connessione di ingresso VPC

Example Richiesta

```
aws apprunner delete-vpc-ingress-connection
      --vpc-ingress-connection-arn: <vpc_ingress_connection_arn>
```

Example Risposta

```
{
   "VpcIngressConnectionArn": <vpc_ingress_connection_arn>,
   "VpcIngressConnectionName": <vic_connection_name>,
   "ServiceArn": <apprunner_service_arn>,
   "Status": "PENDING_DELETION",
   "AccountId": <connection_owner_id>,
   "DomainName": <domain_name_associated_with_vpce>,
   "IngressVpcConfiguration": {"VpcId":<vpc_id>, "VpceId":<vpce_id>},
   "CreatedAt": <date_created>,
   "DeletedAt": <date_deleted>
}
```

Utilizza le seguenti azioni dell'API App Runner per gestire il traffico privato in entrata per il tuo servizio.

- CreateVpcIngressConnection— Crea una nuova risorsa VPC Ingress Connection. App Runner richiede questa risorsa per associare il servizio App Runner a un endpoint VPC Amazon.
- ListVpcIngressConnections— Restituisci un elenco di endpoint AWS App Runner VPC Ingress Connection associati al tuo account. AWS
- DescribeVpcIngressConnection— Restituisce una descrizione completa della risorsa AWS App Runner VPC Ingress Connection.
- UpdateVpcIngressConnection— Aggiorna la risorsa AWS App Runner VPC Ingress Connection.
- DeleteVpcIngressConnection— Eliminare una risorsa App Runner VPC Ingress Connection associata al servizio App Runner.

Per ulteriori informazioni sull'utilizzo dell'API App Runner, consulta la guida di riferimento dell'API App Runner.

IPv6 Attivazione del traffico pubblico in entrata

Se desideri che il servizio riceva traffico di rete in entrata dagli IPv6 indirizzi o da entrambi IPv4 gli IPv6 indirizzi, scegli il tipo di indirizzo Dual-stack per l'endpoint pubblico. Quando crei una nuova applicazione, puoi trovare questa impostazione nella sezione Configura servizio > Rete. Per ulteriori informazioni su come abilitare l' IPv6 utilizzo della console App Runner o dell'API App Runner, consulta. the section called “Gestisci il dual stack per endpoint pubblici”

Per ulteriori informazioni sull'adozione di IPv6 on AWS, consulta on. IPv6 AWS

App Runner supporta il dual stack solo per gli endpoint pubblici del servizio App Runner. È supportato solo per tutti i servizi privati di App Runner. IPv4

> **Note**
>
> Quando il tipo di indirizzo IP è impostato su Dual-stack e si modifica la configurazione di rete da endpoint pubblico a privato, App Runner cambierà automaticamente il tipo di indirizzo in. IPv4 Questo perché App Runner supporta solo gli endpoint pubblici. IPv6

Scopri le informazioni di base su vs. IPv4 IPv6

Il livello IPv4 di rete, comunemente usato per indirizzare il traffico di rete su Internet, utilizza uno schema di indirizzi a 32 bit. Questo spazio di indirizzi è limitato e può essere esaurito con un numero

elevato di dispositivi di rete. Per questo motivo, il Network Address Translation (NAT) viene in genere utilizzato per indirizzare più IPv4 indirizzi attraverso un unico indirizzo di rete pubblico.

IPv6, una versione più recente dell'Internet Protocol, sviluppa IPv4 ed espande lo spazio degli indirizzi con uno schema di indirizzamento a 128 bit. Con IPv6, è possibile creare una rete con un numero quasi illimitato di dispositivi connessi. A causa della grande quantità di indirizzi di rete, il NAT non è necessario. IPv6

IPv4 e gli IPv6 endpoint non sono compatibili tra loro perché gli IPv4 endpoint non possono ricevere IPv6 traffico in entrata e viceversa. Il dual stack offre una soluzione conveniente, in cui è possibile supportare contemporaneamente sia IPv4 il traffico di IPv6 rete che il traffico di rete.

Gestione del dual stack per il traffico pubblico in entrata

Gestisci il tipo di indirizzo dual-stack per il traffico pubblico in entrata utilizzando uno dei seguenti metodi:

- the section called “Console App Runner”
- the section called “API App Runner o AWS CLI”

Console App Runner

Puoi scegliere il tipo di indirizzo dual-stack per il traffico Internet in entrata, quando crei un servizio utilizzando la console App Runner o quando ne aggiorni la configurazione in un secondo momento.

Per abilitare il tipo di indirizzo dual-stack

1. Quando crei o aggiorni un servizio, espandi la sezione Rete in Configura servizio.
2. Scegli Endpoint pubblico, per il traffico di rete in entrata. Si apre l'opzione del tipo di indirizzo IP dell'endpoint pubblico.
3. Espandi il tipo di indirizzo IP dell'endpoint pubblico per visualizzare i seguenti tipi di indirizzi IP.
    - IPv4
    - Dual-stack (e) IPv4 IPv6

> **Note**
>
> Se non espandi il tipo di indirizzo IP dell'endpoint pubblico per effettuare una selezione, App Runner lo IPv4 assegna come configurazione predefinita.

4. Scegli Dual-stack (e). IPv4 IPv6
5. Scegli Avanti e poi Crea e distribuisci se stai creando un servizio. Altrimenti, scegli Salva modifiche se stai aggiornando un servizio.

   Quando il servizio viene distribuito, l'applicazione inizia a ricevere il traffico di rete da entrambi IPv4 gli IPv6 endpoint.

> **Note**
>
> Attualmente, App Runner supporta IPv6 solo gli endpoint pubblici. IPv6 gli endpoint non sono supportati per i servizi App Runner ospitati in un Amazon Virtual Private Cloud (Amazon VPC). Se aggiorni un servizio che utilizza un endpoint pubblico dual-stack a un endpoint privato, per impostazione predefinita il servizio App Runner supporta il traffico proveniente solo IPv4 dagli endpoint e non riceve traffico dagli endpoint. IPv6

Per modificare il tipo di indirizzo

1. Segui i passaggi per aggiornare un servizio e vai a Rete.
2. Passa al tipo di indirizzo IP dell'endpoint pubblico in Traffico di rete in entrata e seleziona il tipo di indirizzo richiesto.
3. Scegli Save changes (Salva modifiche). Il servizio viene aggiornato in base alla selezione effettuata.

API App Runner o AWS CLI

Quando richiami le azioni dell'API CreateServiceo UpdateServicedell'App Runner, utilizza il `IpAddressType` membro del `NetworkConfiguration` parametro per specificare il tipo di indirizzo. I valori supportati che è possibile specificare sono `IPv4` e`DUAL_STACK`. Specificate

`DUAL_STACK` se desiderate che il servizio riceva traffico Internet da IPv4 ed IPv6 endpoint. Se non si specifica alcun valore per`IpAddressType`, IPv4 viene applicato per impostazione predefinita.

Di seguito è riportato l'esempio di creazione di un servizio con il dual stack come indirizzo IP. Questo esempio chiama un `input.json` file.

Example Richiesta di creazione di un servizio con supporto dual stack

```
aws apprunner create-service \
 --cli-input-json file://input.json
```

Example Contenuto di **`input.json`**

```
{
  "ServiceName": "example-service",
  "SourceConfiguration": {
    "ImageRepository": {
      "ImageIdentifier": "public.ecr.aws/aws-containers/hello-app-runner:latest",
      "ImageConfiguration": {
        "Port": "8000"
      },
      "ImageRepositoryType": "ECR_PUBLIC"
    },
    "NetworkConfiguration": {
      "IpAddressType": "DUAL_STACK"
    }
  }
}
```

Example Risposta

```
{
  "Service": {
    "ServiceName": "example-service",
    "ServiceId": "<service-id>",
    "ServiceArn": "arn:aws:apprunner:us-east-2:123456789012:service/example-
service/<service-id>",
    "ServiceUrl": "1234567890.us-east-2.awsapprunner.com",
    "CreatedAt": "2023-10-16T12:30:51.724000-04:00",
    "UpdatedAt": "2023-10-16T12:30:51.724000-04:00",
    "Status": "OPERATION_IN_PROGRESS",
    "SourceConfiguration": {
      "ImageRepository": {
```

```
        "ImageIdentifier": "public.ecr.aws/aws-containers/hello-app-runner:latest",
        "ImageConfiguration": {
          "Port": "8000"
        },
        "ImageRepositoryType": "ECR_PUBLIC"
      },
      "AutoDeploymentsEnabled": false
    },
    "InstanceConfiguration": {
      "Cpu": "1024",
      "Memory": "2048"
    },
    "HealthCheckConfiguration": {
      "Protocol": "TCP",
      "Path": "/",
      "Interval": 5,
      "Timeout": 2,
      "HealthyThreshold": 1,
      "UnhealthyThreshold": 5
    },
    "AutoScalingConfigurationSummary": {
      "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-2:123456789012:autoscalingconfiguration/
DefaultConfiguration/1/00000000000000000000000000000001",
      "AutoScalingConfigurationName": "DefaultConfiguration",
      "AutoScalingConfigurationRevision": 1
    },
    "NetworkConfiguration": {
      "IpAddressType": "DUAL_STACK",
      "EgressConfiguration": {
        "EgressType": "DEFAULT"
      },
      "IngressConfiguration": {
        "IsPubliclyAccessible": true
      }
    }
  },
  "OperationId": "24bd100b1e111ae1a1f0e1115c4f11de"
}
```

> **Note**
>
> Attualmente, App Runner supporta IPv6 solo gli endpoint pubblici. IPv6 gli endpoint non sono supportati per i servizi App Runner ospitati in un Amazon Virtual Private Cloud (Amazon VPC). Se aggiorni un servizio che utilizza un endpoint pubblico dual-stack a un endpoint privato, per impostazione predefinita il servizio App Runner supporta il traffico proveniente solo IPv4 dagli endpoint e non riceve traffico dagli endpoint. IPv6

Per ulteriori informazioni sul parametro API, consulta. NetworkConfiguration

Abilitazione dell'accesso VPC per il traffico in uscita

Per impostazione predefinita, l' AWS App Runner applicazione può inviare messaggi a endpoint pubblici. Ciò include le soluzioni personalizzate e qualsiasi altro sito Web o servizio Web pubblico. Servizi AWS La tua applicazione può persino inviare messaggi agli endpoint pubblici di applicazioni eseguite in un VPC da Amazon Virtual Private Cloud (Amazon VPC). Se non configuri un VPC all'avvio dell'ambiente, App Runner utilizza il VPC predefinito, che è pubblico.

Puoi scegliere di avviare il tuo ambiente in un VPC personalizzato per personalizzare le impostazioni di rete e sicurezza per il traffico in uscita. Puoi abilitare il tuo AWS App Runner servizio per accedere alle applicazioni eseguite in un VPC privato da Amazon Virtual Private Cloud (Amazon VPC). Dopo aver eseguito questa operazione, l'applicazione può connettersi e inviare messaggi ad altre applicazioni ospitate in un Amazon Virtual Private Cloud (Amazon VPC). Alcuni esempi sono un database Amazon RDS ElastiCache, Amazon e altri servizi privati ospitati in un VPC privato.

Connettore VPC

Puoi associare il tuo servizio a un VPC creando un endpoint VPC dalla console App Runner, chiamato VPC Connector. Per creare un connettore VPC, specifica il VPC, una o più sottoreti e, facoltativamente, uno o più gruppi di sicurezza. Dopo aver configurato un connettore VPC, puoi utilizzarlo con uno o più servizi App Runner.

Latenza una tantum

Se configuri il servizio App Runner con un connettore VPC personalizzato per il traffico in uscita, potrebbe verificarsi una latenza di avvio una tantum da due a cinque minuti. Il processo di avvio

attende che il connettore VPC sia pronto per la connessione ad altre risorse prima di impostare lo stato del servizio su In esecuzione. Puoi configurare un servizio con un connettore VPC personalizzato quando lo crei per la prima volta, oppure puoi farlo in seguito aggiornando il servizio.

Tieni presente che se riutilizzi la stessa configurazione del connettore VPC per un altro servizio non ci sarà alcuna latenza. La configurazione del connettore VPC si basa sulla combinazione di gruppo di sicurezza e sottorete. Per una determinata configurazione del connettore VPC, la latenza si verifica solo una volta, durante la creazione iniziale del connettore VPC Hyperplane ENIs (interfacce di rete elastiche).

Ulteriori informazioni sui connettori VPC personalizzati e Hyperplane AWS

I connettori VPC di App Runner si basano su AWS Hyperplane, il sistema di rete Amazon interno alla base di diverse AWS risorse, come Network Load Balancer, NAT Gateway e AWS. PrivateLink La tecnologia AWS Hyperplane offre capacità di throughput elevato e bassa latenza, oltre a un grado di condivisione più elevato. Un ENI Hyperplane viene creato nelle tue sottoreti quando crei un connettore VPC e lo associ al tuo servizio. La configurazione di un connettore VPC si basa su una combinazione di gruppo di sicurezza e sottorete ed è possibile fare riferimento allo stesso connettore VPC su più servizi App Runner. Di conseguenza, gli Hyperplane sottostanti ENIs vengono condivisi tra i servizi App Runner. Questa condivisione è fattibile anche se si aumenta il numero di attività necessarie per gestire il carico delle richieste e si traduce in un utilizzo più efficiente dello spazio IP nel VPC. Per ulteriori informazioni, consulta Deep Dive sulla rete VPC di AWS App Runner nel blog AWS Container.

Sottorete

Ogni sottorete si trova in una zona di disponibilità specifica. Per un'elevata disponibilità, si consiglia di selezionare sottoreti in almeno tre zone di disponibilità. Se la regione ha meno di tre zone di disponibilità, ti consigliamo di selezionare le sottoreti in tutte le zone di disponibilità supportate.

Quando selezioni una sottorete per il tuo VPC, assicurati di scegliere una sottorete privata, non una sottorete pubblica. Questo perché, quando si crea un connettore VPC, il servizio App Runner crea un ENI Hyperplane in ciascuna delle sottoreti. A ogni Hyperplane ENI viene assegnato solo un indirizzo IP privato e viene contrassegnato con un tag della chiave. AWSAppRunnerManaged Se scegli una sottorete pubblica, si verificheranno degli errori durante l'esecuzione del servizio App Runner. Tuttavia, se il servizio deve accedere ad alcuni servizi disponibili su Internet o su altri servizi pubblici Servizi AWS, consulta. the section called “Considerazioni sulla scelta di una sottorete ”

Considerazioni sulla scelta di una sottorete

- Quando connetti il tuo servizio a un VPC, il traffico in uscita non ha accesso alla rete Internet pubblica. Tutto il traffico in uscita dall'applicazione viene indirizzato attraverso il VPC a cui è connesso il servizio. Tutte le regole di rete per il VPC si applicano al traffico in uscita dell'applicazione. Ciò significa che i tuoi servizi non possono accedere alla rete Internet pubblica e. AWS APIs Per accedere, esegui una delle seguenti operazioni:
  - Connect le sottoreti a Internet tramite un gateway NAT.
  - Configura gli endpoint VPC per Servizi AWS i quali desideri accedere. Il tuo servizio rimane all'interno di Amazon VPC utilizzando. AWS PrivateLink
- Alcune zone di disponibilità in altre Regioni AWS non supportano le sottoreti che possono essere utilizzate con i servizi App Runner. Se scegli le sottoreti in queste zone di disponibilità, il servizio non verrà creato o aggiornato. In queste situazioni, App Runner fornisce un messaggio di errore dettagliato che indica le sottoreti e le zone di disponibilità non supportate. In questo caso, risolvi il problema rimuovendo le sottoreti non supportate dalla richiesta, quindi riprova.

Gruppo di sicurezza

Facoltativamente, puoi specificare i gruppi di sicurezza utilizzati da App Runner per accedere alle sottoreti specificate. AWS Se non specifichi i gruppi di sicurezza, App Runner utilizza il gruppo di sicurezza predefinito del VPC. Il gruppo di sicurezza predefinito consente tutto il traffico in uscita.

L'aggiunta di un gruppo di sicurezza fornisce un ulteriore livello di sicurezza ai connettori VCP, offrendo un maggiore controllo sul traffico di rete. Il connettore VPC viene utilizzato solo per le comunicazioni in uscita dall'applicazione. Utilizzate le regole in uscita per consentire la comunicazione con gli endpoint di destinazione desiderati. È inoltre necessario assicurarsi che tutti i gruppi di sicurezza associati alla risorsa di destinazione dispongano delle regole in entrata appropriate. Altrimenti, queste risorse non possono accettare il traffico proveniente dai gruppi di sicurezza di VPC Connector.

> **Note**
>
> Quando associ il tuo servizio a un VPC, il traffico seguente non viene influenzato:
>
> - Traffico in entrata: i messaggi in entrata ricevuti dall'applicazione non sono influenzati da un VPC associato. I messaggi vengono instradati attraverso il nome di dominio pubblico associato al tuo servizio e non interagiscono con il VPC.

- Traffico App Runner: App Runner gestisce diverse azioni per tuo conto, come l'estrazione del codice sorgente e delle immagini, l'invio di registri e il recupero di segreti. Il traffico generato da queste azioni non viene instradato attraverso il tuo VPC.

Per ulteriori informazioni su come AWS App Runner si integra con Amazon VPC, AWS consulta App Runner VPC Networking.

> **Note**
>
> Per il traffico in uscita, al momento App Runner supporta solo. IPv4

Gestisci l'accesso al VPC

> **Note**
>
> Se crei un connettore VPC per il traffico in uscita per un servizio, il processo di avvio del servizio che segue presenterà una latenza unica. È possibile impostare questa configurazione per un nuovo servizio al momento della creazione o in seguito, con un aggiornamento del servizio. Per ulteriori informazioni, consulta il Latenza una tantum capitolo Networking with App Runner di questa guida.

Gestisci l'accesso VPC per i tuoi servizi App Runner utilizzando uno dei seguenti metodi:

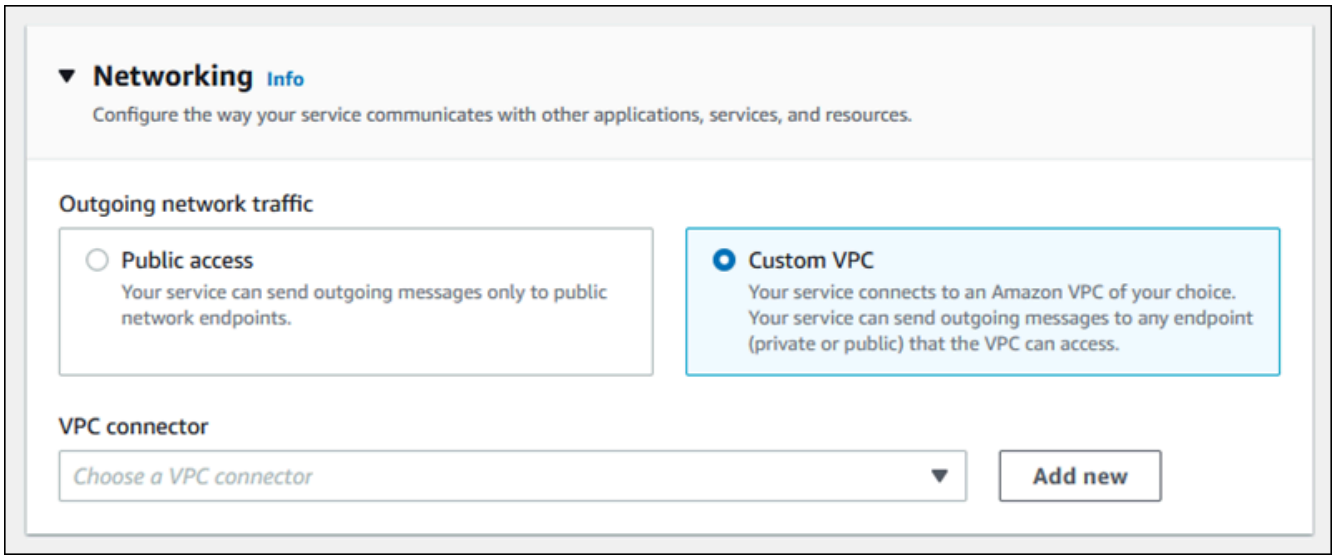
App Runner console

Quando crei un servizio utilizzando la console App Runner o quando ne aggiorni la configurazione in un secondo momento, puoi scegliere di configurare il traffico in uscita. Cerca la sezione Configurazione di rete nella pagina della console. Per il traffico di rete in uscita, scegli quanto segue:

- Accesso pubblico: per associare il tuo servizio agli endpoint pubblici di altri. Servizi AWS
- VPC personalizzato: per associare il servizio a un VPC di Amazon VPC. La tua applicazione può connettersi e inviare messaggi ad altre applicazioni ospitate in un Amazon VPC.

Per abilitare il VPC personalizzato

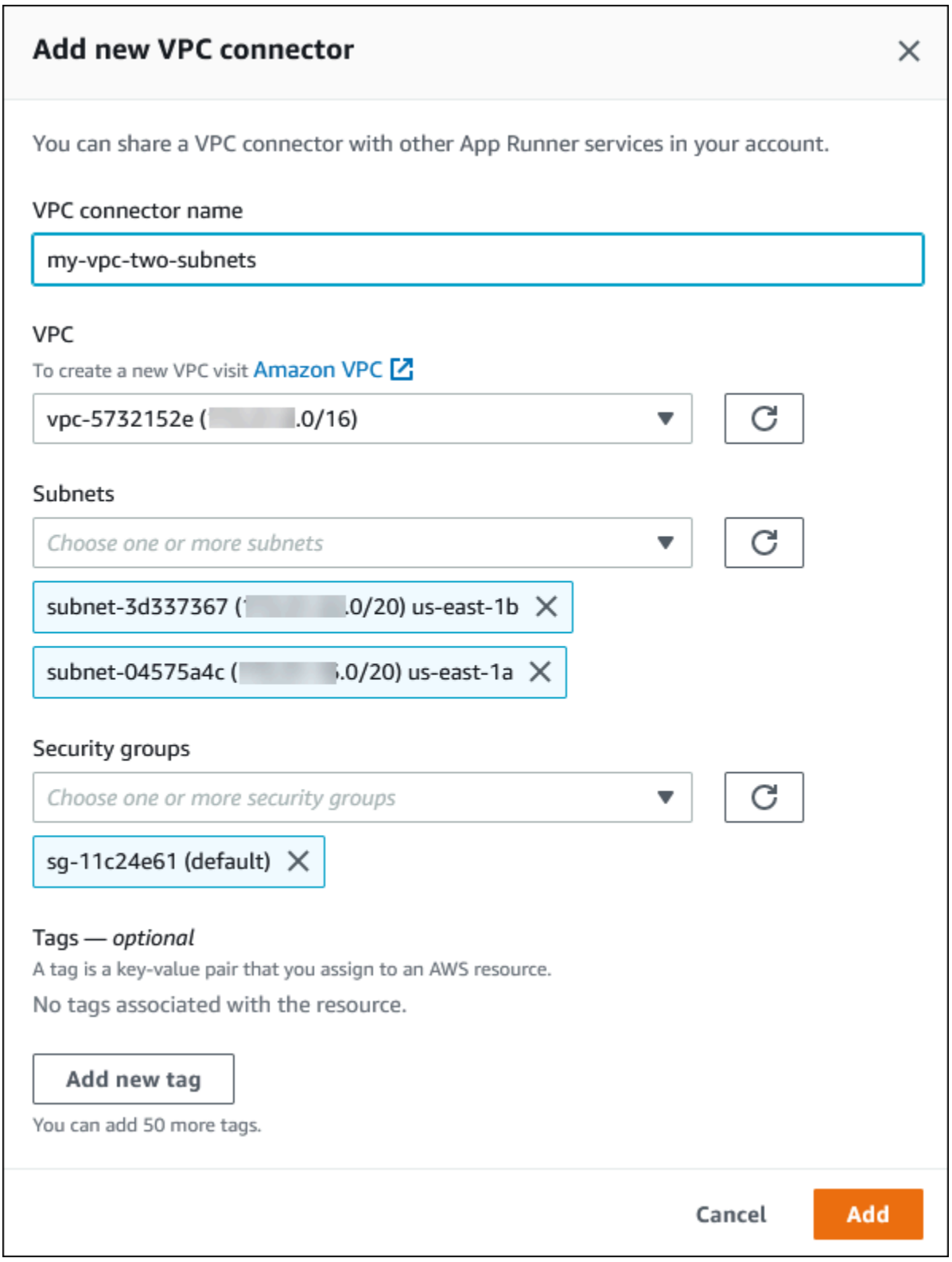
1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Vai alla sezione Rete in Configura servizio.



3. Scegli VPC personalizzato, per il traffico di rete in uscita.
4. Nel pannello di navigazione, scegli Connettore VPC.

   Se hai creato i connettori VPC, la console visualizza un elenco di connettori VPC nel tuo account. Puoi scegliere un connettore VPC esistente e scegliere Avanti per rivedere la configurazione. Quindi, vai all'ultimo passaggio. In alternativa, puoi aggiungere un nuovo connettore VPC utilizzando i seguenti passaggi.
5. Scegli Aggiungi nuovo per creare un nuovo connettore VPC per il tuo servizio.

   Viene quindi visualizzata la finestra di dialogo Aggiungi nuovo connettore VPC.



6. Inserisci un nome per il tuo connettore VPC e seleziona il VPC richiesto dall'elenco disponibile.

7. Per le sottoreti, seleziona una sottorete per ogni zona di disponibilità da cui intendi accedere al servizio App Runner. Per una migliore disponibilità, scegli tre sottoreti. Oppure, se sono presenti meno di tre sottoreti, scegli tutte le sottoreti disponibili.

   > **Note**
   >
   > Assicurati di assegnare sottoreti private al connettore VPC. Se assegni sottoreti pubbliche al connettore VPC, il servizio non riesce a creare o esegue il rollback automaticamente durante un aggiornamento.

8. (Facoltativo) Per il gruppo Security, selezionate i gruppi di sicurezza da associare alle interfacce di rete degli endpoint.
9. (Facoltativo) Per aggiungere un tag, scegliere Add new tag (Aggiungi nuovo tag) e immettere la chiave e il valore del tag.
10. Scegli Aggiungi.

    I dettagli del connettore VPC che hai creato vengono visualizzati in Connettore VPC.
11. Scegli Avanti per rivedere la configurazione, quindi scegli Crea e distribuisci.

    App Runner crea per te una risorsa di connettore VPC, quindi la associa al tuo servizio. Se il servizio viene creato correttamente, la console mostra la dashboard del servizio, con una panoramica del servizio del nuovo servizio.

App Runner API or AWS CLI

Quando richiami le azioni API CreateServiceo UpdateServiceApp Runner, utilizza il `EgressConfiguration` membro del `NetworkConfiguration` parametro per specificare una risorsa del connettore VPC per il tuo servizio.

Utilizza le seguenti azioni dell'API App Runner per gestire le risorse del connettore VPC.

- CreateVpcConnector— Crea un nuovo connettore VPC.
- ListVpcConnectors— Restituisce un elenco dei connettori VPC associati al tuo. Account AWS L'elenco include descrizioni complete.
- DescribeVpcConnector— Restituisce una descrizione completa di un connettore VPC.
- DeleteVpcConnector— Elimina un connettore VPC. Se raggiungi la quota di connettori VPC per il tuo Account AWS, potresti dover eliminare i connettori VPC non necessari.

Per distribuire un'applicazione su App Runner con accesso in uscita a un VPC, devi prima creare un connettore VPC. È possibile farlo specificando una o più sottoreti e gruppi di sicurezza da associare all'applicazione. È quindi possibile fare riferimento al connettore VPC in Create o UpdateServicetramite la CLI, come illustrato nell'esempio seguente:

```
            cat > vpc-connector.json <<EOF
{
"VpcConnectorName": "my-vpc-connector",
"Subnets": [
"subnet-a",
"subnet-b",
"subnet-c"
],
"SecurityGroups": [
"sg-1",
"sg-2"
]
}
EOF

aws apprunner create-vpc-connector \
--cli-input-json file:///vpc-connector.json

cat > service.json <<EOF

{
"ServiceName": "my-vpc-connected-service",
"SourceConfiguration": {
"ImageRepository": {
"ImageIdentifier": "<ecr-image-identifier> ",
"ImageConfiguration": {
"Port": "8000"
},
"ImageRepositoryType": "ECR"
}
},
"NetworkConfiguration": {
"EgressConfiguration": {
"EgressType": "VPC",
"VpcConnectorArn": "arn:aws:apprunner:..../my-vpc-connector"
}
}
```

```
}
EOF

aws apprunner create-service \
--cli-input-json file:///service.js
```

Osservabilità per il tuo servizio App Runner

AWS App Runner si integra con diversi AWS servizi per fornirti un'ampia suite di strumenti di osservabilità per il tuo servizio App Runner. Gli argomenti di questo capitolo descrivono queste funzionalità.

Argomenti

Monitoraggio dell'attività del servizio App Runner

AWS App Runner utilizza un elenco di operazioni per tenere traccia delle attività nel servizio App Runner. Un'operazione rappresenta una chiamata asincrona a un'azione API, come la creazione di un servizio, l'aggiornamento di una configurazione e la distribuzione di un servizio. Le sezioni seguenti mostrano come tenere traccia delle attività nella console App Runner e utilizzare l'API.

Tieni traccia dell'attività del servizio App Runner

Tieni traccia dell'attività del servizio App Runner utilizzando uno dei seguenti metodi:

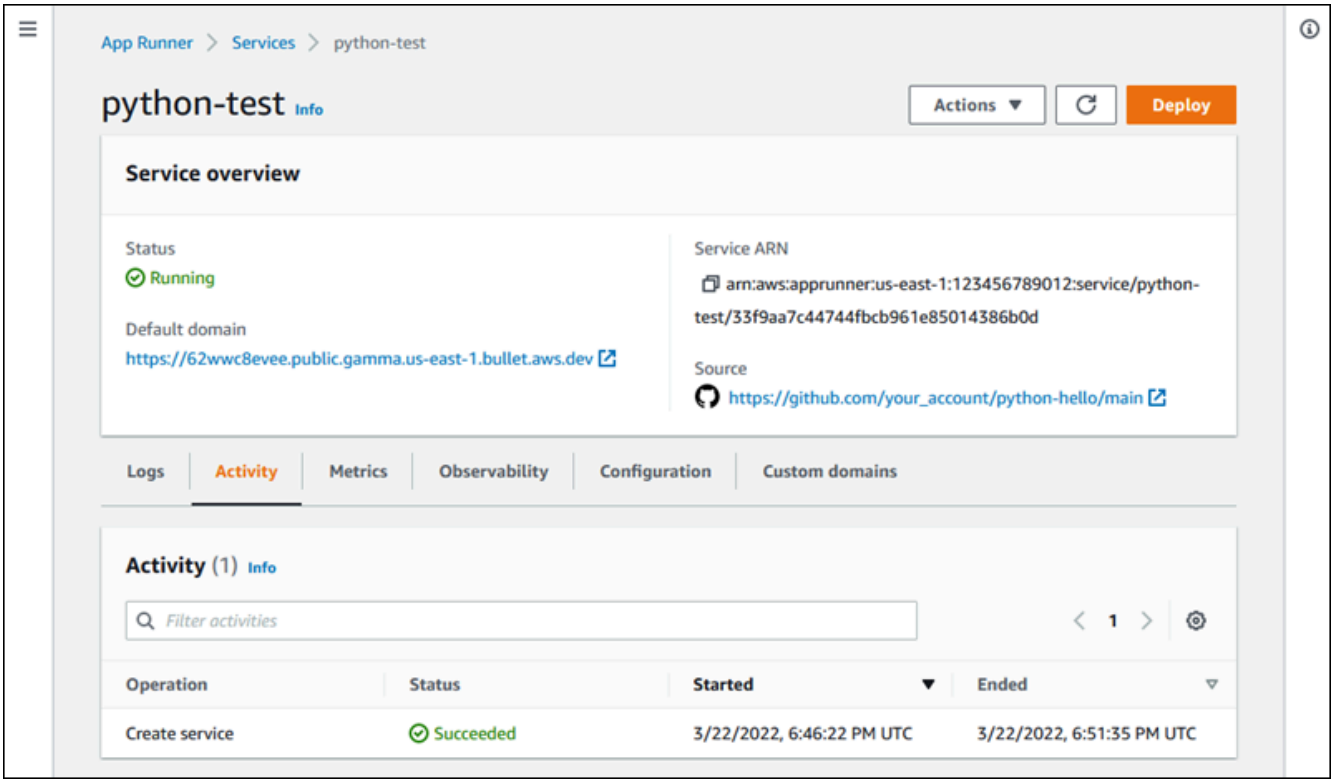
App Runner console

La console App Runner mostra l'attività del servizio App Runner e offre altri modi per esplorare le operazioni.

Per visualizzare l'attività del servizio

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua Regione AWS.
2. Nel riquadro di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

   La console mostra la dashboard del servizio con una panoramica del servizio.



3. Nella pagina della dashboard del servizio, scegli la scheda Attività, se non è già stata selezionata.

   La console mostra un elenco di operazioni.

4. Per trovare operazioni specifiche, scorri l'elenco inserendo un termine di ricerca. È possibile cercare qualsiasi valore visualizzato nella tabella.

5. Scegliete una delle operazioni elencate per visualizzare o scaricare il relativo registro.

App Runner API or AWS CLI

L'ListOperationsazione, dato l'Amazon Resource Name (ARN) di un servizio App Runner, restituisce un elenco di operazioni avvenute su questo servizio. Ogni elemento dell'elenco contiene un ID operativo e alcuni dettagli di tracciamento.

Visualizzazione dei log di App Runner trasmessi in streaming a Logs CloudWatch

Puoi utilizzare Amazon CloudWatch Logs per monitorare, archiviare e accedere ai file di registro generati dalle tue risorse in vari AWS servizi. Per ulteriori informazioni, consulta la Amazon CloudWatch Logs User Guide.

AWS App Runner raccoglie l'output delle distribuzioni delle applicazioni e del servizio attivo e lo trasmette a Logs. CloudWatch Le seguenti sezioni elencano i flussi di log di App Runner e mostrano come visualizzarli nella console App Runner.

Gruppi di log e stream di App Runner

CloudWatch Logs conserva i dati di log in flussi di log che poi organizza ulteriormente in gruppi di log. Un flusso di log è una sequenza di eventi di registro provenienti da una fonte specifica. Un gruppo di log è un gruppo di flussi di log che condividono le stesse impostazioni di conservazione, monitoraggio e controllo degli accessi.

App Runner definisce due gruppi di log CloudWatch Logs, ciascuno con più flussi di log, per ogni servizio App Runner dell'utente. Account AWS

Registri di servizio

Il gruppo di registri di servizio contiene l'output di registrazione generato da App Runner in quanto gestisce il servizio App Runner e agisce su di esso.

Nome del gruppo di log	Esempio
`/aws/apprunner/ service-name /service-id /service`	`/aws/apprunner/python-test/ ac7ec8b51ff34746bcb6654e0bc b23da/service`

All'interno del gruppo di log del servizio, App Runner crea un flusso di registro degli eventi per acquisire le attività nel ciclo di vita del servizio App Runner. Ad esempio, questo potrebbe essere l'avvio dell'applicazione o la sua sospensione.

Inoltre, App Runner crea un flusso di log per ogni operazione asincrona di lunga durata correlata al servizio. Il nome del flusso di registro riflette il tipo di operazione e l'ID operativo specifico.

Una distribuzione è un tipo di operazione. I registri di distribuzione contengono l'output di registrazione delle fasi di compilazione e distribuzione eseguite da App Runner quando si crea un servizio o si distribuisce una nuova versione dell'applicazione. I nomi dei flussi di log di distribuzione iniziano con `deployment/` e terminano con l'ID dell'operazione che esegue la distribuzione. Questa operazione è una CreateServicechiamata per la distribuzione iniziale dell'applicazione o una StartDeploymentchiamata per ogni ulteriore distribuzione.

All'interno di un registro di distribuzione, ogni messaggio di registro inizia con un prefisso:

- `[AppRunner]`— Output generato da App Runner durante la distribuzione.
- `[Build]`— Output dei propri script di compilazione.

Nome del flusso di log	Esempio
`events`	N/A (nome fisso)
`operation-type` `/`*`operation-id`*	`deployment/c2c8eeedea164f45 9cf78f12a8953390`

Log di applicazioni

Il gruppo di log dell'applicazione contiene l'output del codice dell'applicazione in esecuzione.

Nome del gruppo di log	Esempio
`/aws/apprunner/` *`service-name`* `/`*`service-id`* `/application`	`/aws/apprunner/python-test/ ac7ec8b51ff34746bcb6654e0bcb23da/ application`

All'interno del gruppo di log dell'applicazione, App Runner crea un flusso di log per ogni istanza (unità di scalabilità) che esegue l'applicazione.

Nome del flusso di log	Esempio
`instance/` *`instance-id`*	`instance/1a80bc9134a84699b7 b3432ebeebb591`

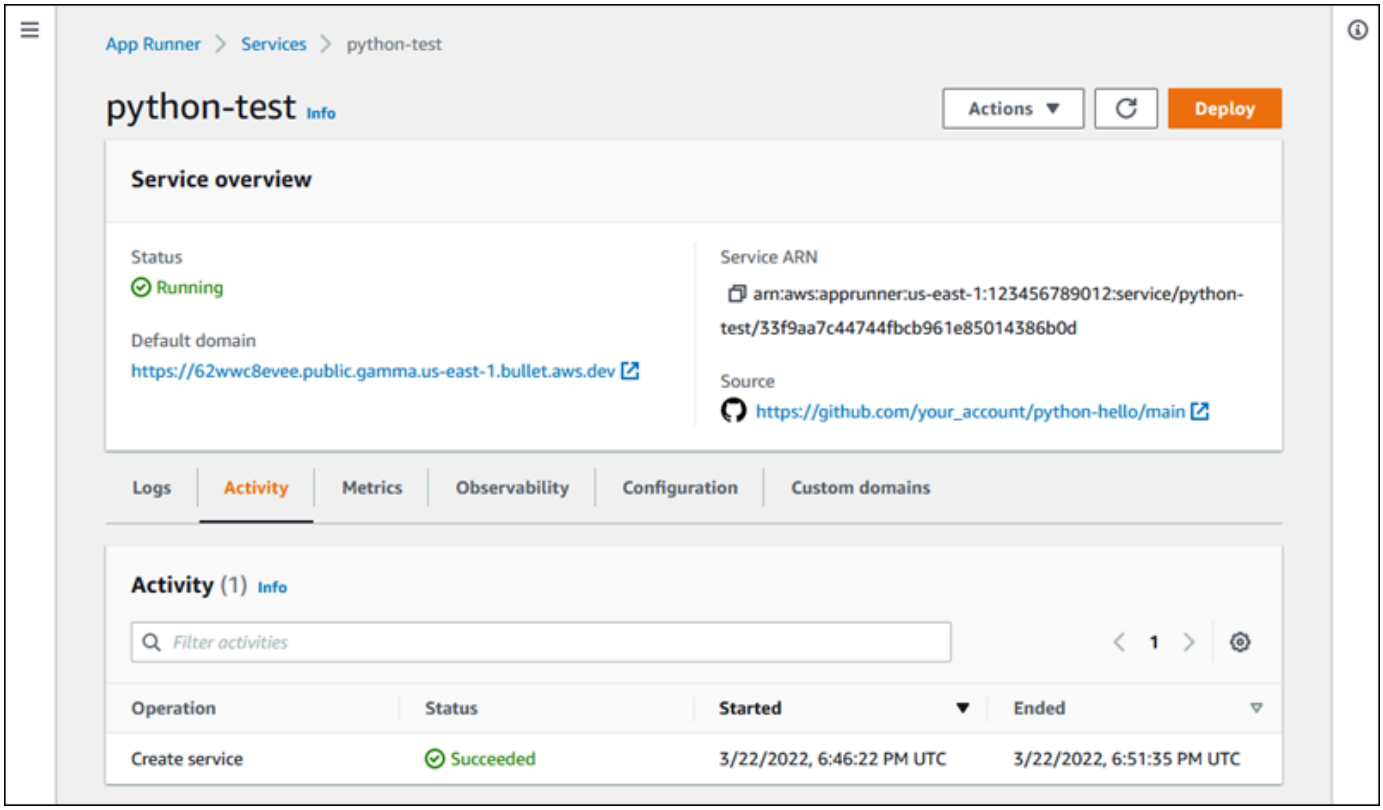
Visualizzazione dei log di App Runner nella console

La console App Runner mostra un riepilogo di tutti i log del servizio e consente di visualizzarli, esplorarli e scaricarli.

Per visualizzare i registri del servizio

1. Apri la console App Runner e, nell'elenco Regioni, seleziona il tuo. Regione AWS
2. Nel pannello di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

   La console mostra la dashboard del servizio con una panoramica del servizio.

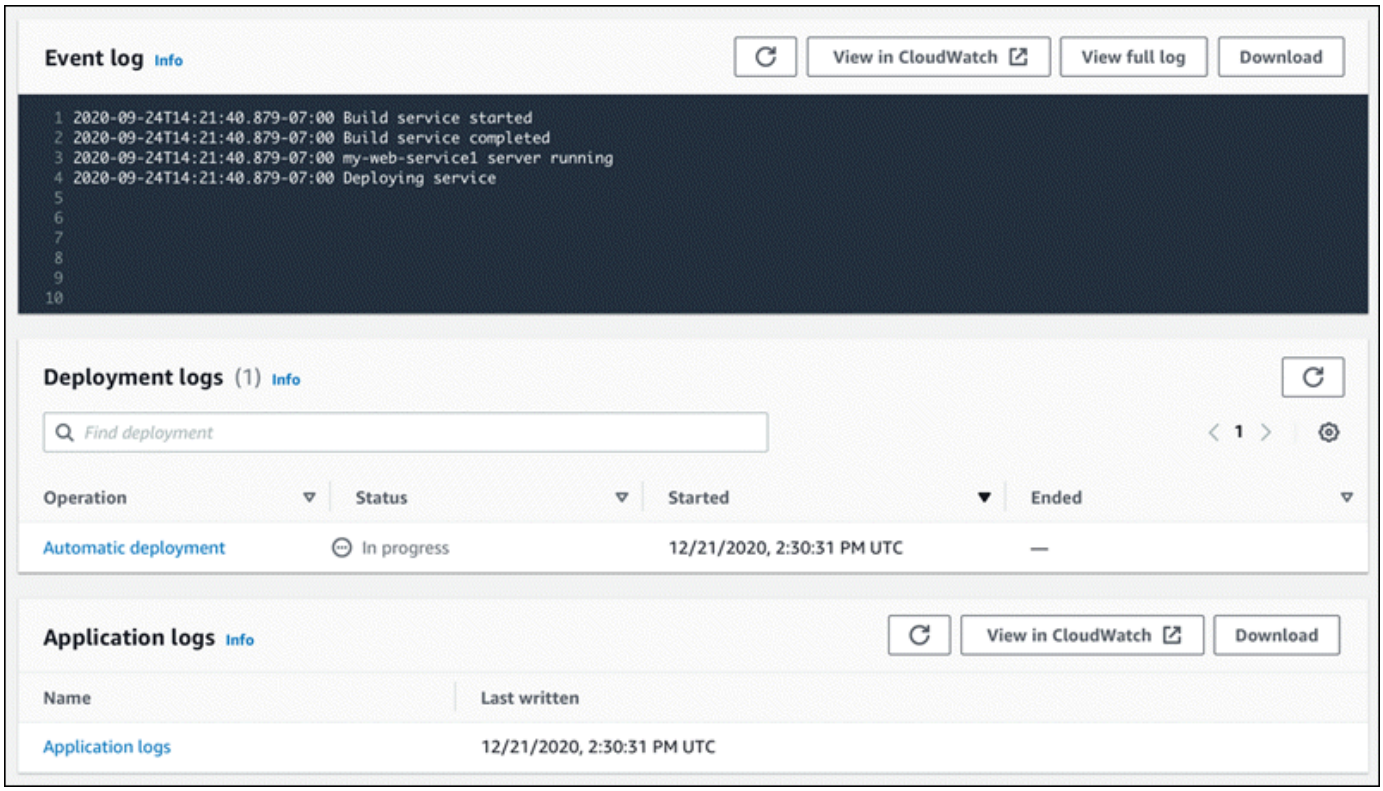


3. Nella pagina del dashboard del servizio, scegli la scheda Registri.

   La console mostra alcuni tipi di log in diverse sezioni:

   - Registro eventi: attività nel ciclo di vita del servizio App Runner. La console mostra gli ultimi eventi.
   - Registri di distribuzione: invia le distribuzioni del repository al servizio App Runner. La console visualizza un flusso di log separato per ogni distribuzione.

- Registri delle applicazioni: l'output dell'applicazione Web distribuita nel servizio App Runner. La console combina l'output di tutte le istanze in esecuzione in un unico flusso di log.



4. Per trovare implementazioni specifiche, scorri l'elenco dei log di distribuzione inserendo un termine di ricerca. È possibile cercare qualsiasi valore visualizzato nella tabella.
5. Per visualizzare il contenuto di un registro, scegli Visualizza registro completo (registro eventi) o il nome del flusso di registro (registri di distribuzione e applicazione).
6. Scegli Scarica per scaricare un registro. Per un flusso di log di distribuzione, seleziona prima un flusso di log.
7. Scegli Visualizza in CloudWatch per aprire la CloudWatch console e utilizzare tutte le sue funzionalità per esplorare i log di servizio di App Runner. Per un flusso di log di distribuzione, seleziona prima un flusso di log.

> **Note**
>
> La CloudWatch console è particolarmente utile se si desidera visualizzare i log delle applicazioni di istanze specifiche anziché il registro dell'applicazione combinato.

Visualizzazione delle metriche del servizio App Runner riportate a CloudWatch

Amazon CloudWatch monitora le tue risorse Amazon Web Services (AWS) e le applicazioni su cui esegui AWS in tempo reale. Puoi utilizzarlo CloudWatch per raccogliere e tenere traccia delle metriche, che sono variabili che puoi misurare per le tue risorse e applicazioni. Puoi anche usarlo per creare allarmi che controllano le metriche. Quando viene raggiunta una certa soglia, CloudWatch invia notifiche o apporta automaticamente modifiche alle risorse monitorate. Per ulteriori informazioni, consulta la Amazon CloudWatch User Guide.

AWS App Runner raccoglie una varietà di metriche che ti offrono una maggiore visibilità sull'utilizzo, le prestazioni e la disponibilità dei tuoi servizi App Runner. Alcune metriche tengono traccia delle singole istanze che eseguono il servizio Web, mentre altre riguardano il livello di servizio generale. Le seguenti sezioni elencano le metriche di App Runner e mostrano come visualizzarle nella console App Runner.

Metriche di App Runner

App Runner raccoglie le seguenti metriche relative al tuo servizio e le pubblica nel namespace. CloudWatch `AWS/AppRunner`

> **Note**
>
> Prima del 23 agosto 2023, le metriche relative all'utilizzo della CPU e della memoria si basavano sulle unità vCPU e sui megabyte di memoria utilizzati, anziché sulla percentuale di utilizzo, come calcolato oggi. Se la tua applicazione è stata eseguita su App Runner prima di questa data e scegli di tornare a visualizzare le metriche per questa data su App Runner o sulla CloudWatch console, vedrai una visualizzazione delle metriche in entrambe le unità e di conseguenza vedrai anche alcune irregolarità.

> **Important**
>
> Dovrai aggiornare tutti gli CloudWatch allarmi basati sui valori delle metriche relative all'utilizzo della CPU e della memoria prima del 23 agosto 2023. Aggiorna gli allarmi in modo che si attivino in base alla percentuale di utilizzo anziché alla vCPU o ai megabyte. Per ulteriori informazioni, consulta la Amazon CloudWatch User Guide.

Le metriche a livello di istanza vengono raccolte singolarmente per ogni istanza (unità di scala).

Cosa viene misurato?	Parametro	Descrizione
CPU utilization	`CPUUtilization`	La percentuale di utilizzo medio della CPU durante periodi di un minuto rispetto all'utilizzo totale della CPU riservato dalla configurazione del servizio.
Memory utilization	`MemoryUtilization`	La percentuale di utilizzo medio della memoria durante periodi di un minuto rispetto alla memoria totale riservata dalla configurazione del servizio.

Le metriche del livello di servizio vengono raccolte per l'intero servizio.

Cosa viene misurato?	Parametro	Descrizione
CPU utilization	`CPUUtilization`	La percentuale di utilizzo aggregato della CPU su tutte le istanze durante periodi di un minuto rispetto all'utilizzo totale della CPU riservato dalla configurazione del servizio.
Memory utilization	`MemoryUtilization`	La percentuale di utilizzo aggregato della memoria su tutte le istanze durante periodi di un minuto rispetto alla memoria totale riservata dalla configurazione del servizio.
Concurrency	`Concurrency`	Il numero approssimativo di richieste simultanee gestite dal servizio.
HTTP request count	`Requests`	Il numero di richieste HTTP ricevute dal servizio.
HTTP status counts	`2xxStatusResponses`	Il numero di richieste HTTP che hanno restituito ogni stato di risposta, raggruppate per categoria (2XX, 4XX, 5XX).

Cosa viene misurato?	Parametro	Descrizione

	`4xxStatus Responses`
	
	`5xxStatus Responses`	

HTTP request latency	`RequestLatency`	Il tempo, in millisecondi, impiegato dal servizio Web per elaborare le richieste HTTP.
Instance counts	`ActiveInstances`	Il numero di istanze che elaborano le richieste HTTP per il servizio. **Note** Se la `ActiveInstances` metrica mostra zero, significa che non ci sono richieste per il servizio. Non indica che il numero di istanze del servizio sia zero.

Visualizzazione delle metriche di App Runner nella console

La console App Runner mostra graficamente le metriche raccolte da App Runner per il tuo servizio e offre altri modi per esplorarle.

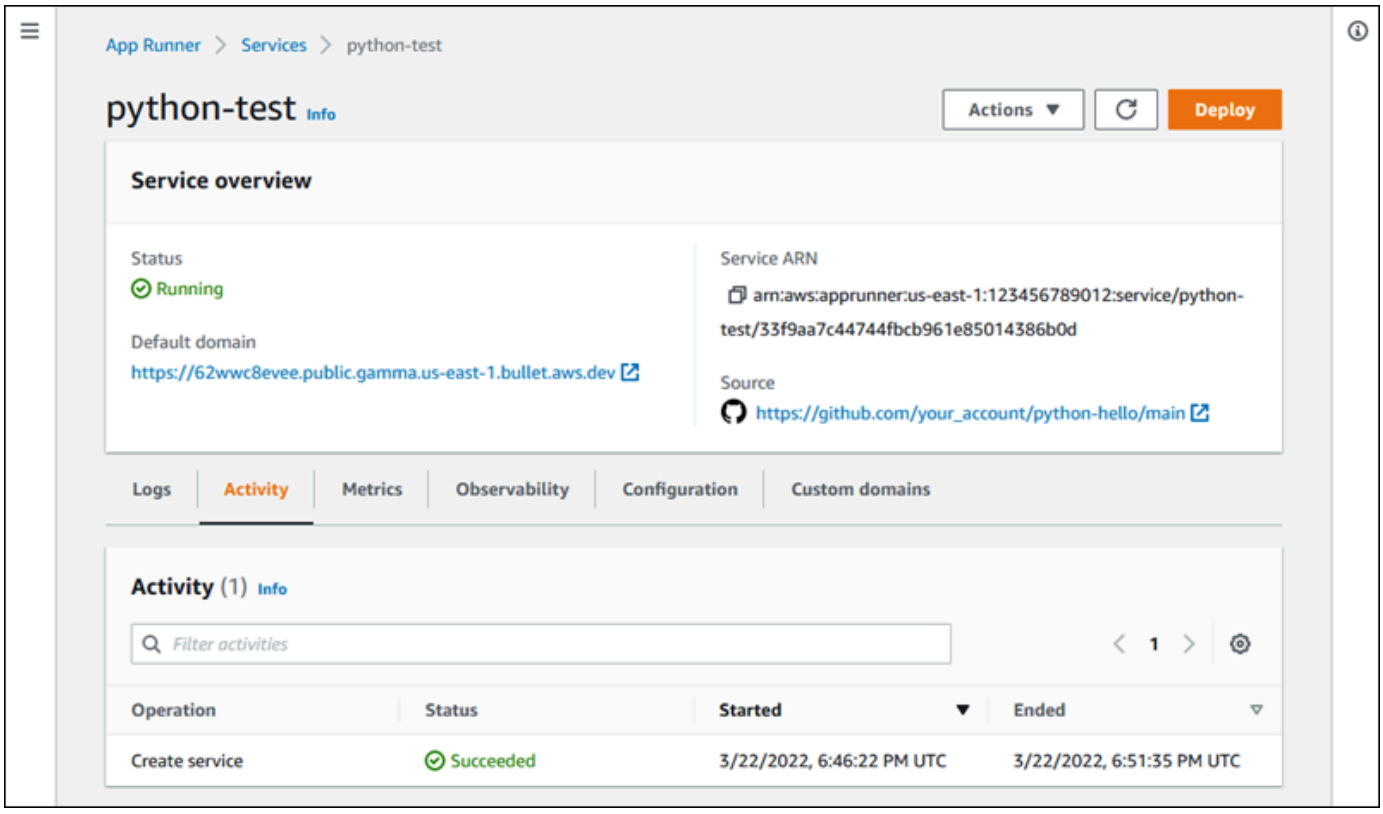
Note

Al momento, la console mostra solo le metriche di servizio. Per visualizzare le metriche delle istanze, usa la CloudWatch console.

Per visualizzare i log del tuo servizio

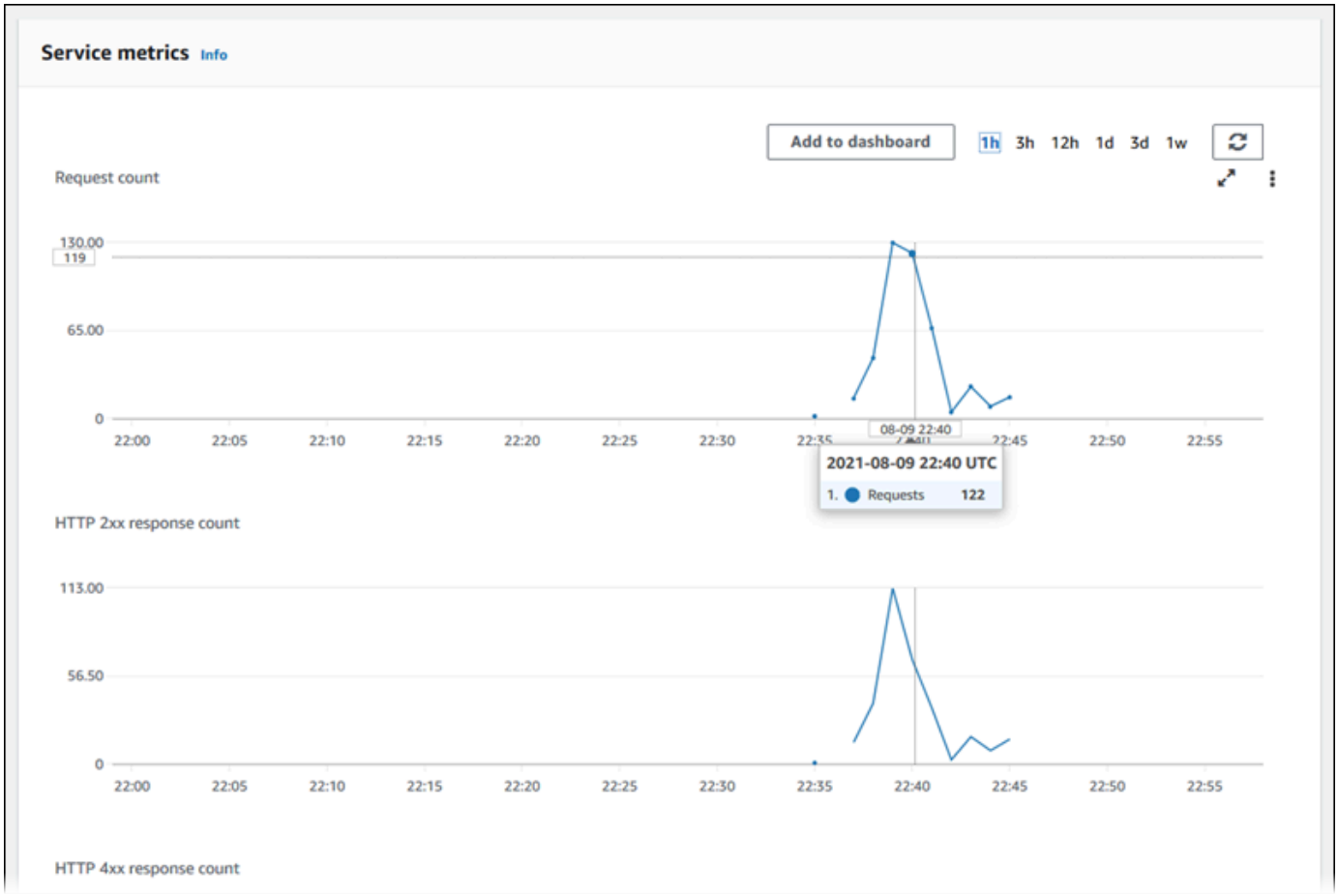
1. Apri la console App Runner e, nell'elenco Regioni, seleziona il tuo. Regione AWS
2. Nel riquadro di navigazione, scegli Servizi, quindi scegli il servizio App Runner.

La console mostra la dashboard del servizio con una panoramica del servizio.



3. Nella pagina del dashboard del servizio, scegli la scheda Metriche.

   La console mostra una serie di grafici delle metriche.



4. Scegli una durata (ad esempio, 12 ore) per adattare i grafici delle metriche al periodo recente di tale durata.
5. Scegli Aggiungi alla dashboard nella parte superiore di una delle sezioni del grafico o utilizza il menu su qualsiasi grafico per aggiungere le metriche pertinenti a una dashboard nella CloudWatch console per ulteriori indagini.

Gestione degli eventi App Runner in EventBridge

Con Amazon EventBridge, puoi configurare regole basate sugli eventi che monitorano un flusso di dati in tempo reale dal tuo AWS App Runner servizio per determinati modelli. Quando viene rispettato uno schema per una regola, EventBridge avvia un'azione in una destinazione come AWS Lambda Amazon ECS e Amazon AWS Batch SNS. Ad esempio, puoi impostare una regola per l'invio di notifiche e-mail segnalando un argomento di Amazon SNS ogni volta che una distribuzione del tuo servizio fallisce. In alternativa, puoi impostare una funzione Lambda per notificare a un canale Slack ogni volta che un aggiornamento del servizio fallisce. Per ulteriori informazioni su EventBridge, consulta Amazon EventBridge User Guide.

App Runner invia i seguenti tipi di eventi a EventBridge

- Modifica dello stato del servizio: modifica dello stato di un servizio App Runner. Ad esempio, lo stato di un servizio è cambiato in`DELETE_FAILED`.
- Modifica dello stato di funzionamento del servizio: modifica dello stato di un'operazione lunga e asincrona su un servizio App Runner. Ad esempio, la creazione di un servizio è iniziata, un aggiornamento del servizio è stato completato con successo o la distribuzione del servizio è stata completata con errori.

Creazione di una EventBridge regola per agire sugli eventi di App Runner

Un EventBridge evento è un oggetto che definisce alcuni EventBridge campi standard, come il AWS servizio di origine e il tipo di dettaglio (evento), e un insieme di campi specifici dell'evento con i dettagli dell'evento. Per creare una EventBridge regola, si utilizza la EventBridge console per definire uno schema di eventi (quali eventi devono essere registrati) e specificare un'azione mirata (cosa fare durante una partita). Uno schema di eventi è simile agli eventi a cui corrisponde. Si specifica un sottoinsieme di campi da abbinare e per ogni campo si specifica un elenco di valori possibili. Questo argomento fornisce esempi di eventi e modelli di eventi di App Runner.

Per ulteriori informazioni sulla creazione di EventBridge regole, consulta Creating a rule for an AWS service nella Amazon EventBridge User Guide.

> **Note**
>
> Alcuni servizi supportano modelli predefiniti in EventBridge. Ciò semplifica il modo in cui viene creato un modello di eventi. Seleziona i valori dei campi su un modulo e EventBridge genera lo schema automaticamente. Al momento, App Runner non supporta modelli predefiniti. Devi inserire il pattern come oggetto JSON. È possibile utilizzare gli esempi in questo argomento come punto di partenza.

Esempi di eventi App Runner

Questi sono alcuni esempi di eventi a cui App Runner invia. EventBridge

- Un evento di modifica dello stato del servizio. In particolare, un servizio che è cambiato dallo `RUNNING` stato `OPERATION_IN_PROGRESS` a.

```
{
  "version": "0",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "AppRunner Service Status Change",
  "source": "aws.apprunner",
  "account": "111122223333",
  "time": "2021-04-29T11:54:23Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:apprunner:us-east-2:123456789012:service/my-
app/8fe1e10304f84fd2b0df550fe98a71fa"
  ],
  "detail": {
    "previousServiceStatus": "OPERATION_IN_PROGRESS",
    "currentServiceStatus": "RUNNING",
    "serviceName": "my-app",
    "serviceId": "8fe1e10304f84fd2b0df550fe98a71fa",
    "message": "Service status is set to RUNNING.",
    "severity": "INFO"
  }
}
```

- Un evento di modifica dello stato dell'operazione. In particolare, un'`UpdateService`operazione completata con successo.

```
{
  "version": "0",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "AppRunner Service Operation Status Change",
  "source": "aws.apprunner",
  "account": "111122223333",
  "time": "2021-04-29T18:43:48Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:apprunner:us-east-2:123456789012:service/my-
app/8fe1e10304f84fd2b0df550fe98a71fa"
  ],
  "detail": {
    "operationStatus": "UpdateServiceCompletedSuccessfully",
    "serviceName": "my-app",
    "serviceId": "8fe1e10304f84fd2b0df550fe98a71fa",
```

```
    "message": "Service update completed successfully. New application and
 configuration is deployed.",
    "severity": "INFO"
  }
}
```

Esempi di pattern di eventi di App Runner

Gli esempi seguenti mostrano modelli di eventi che è possibile utilizzare nelle EventBridge regole per abbinare uno o più eventi di App Runner. Uno schema di eventi è simile a un evento. Includi solo i campi che desideri abbinare e fornisci un elenco anziché uno scalare per ciascuno di essi.

- Abbina tutti gli eventi di modifica dello stato del servizio per i servizi di un account specifico, in cui il servizio non è più in `RUNNING` stato.

```
{
  "detail-type": [ "AppRunner Service Status Change" ],
  "source": [ "aws.apprunner" ],
  "account": [ "111122223333" ],
  "detail": {
    "previousServiceStatus": [ "RUNNING" ]
  }
}
```

- Abbina tutti gli eventi di modifica dello stato dell'operazione per i servizi di un account specifico, in cui l'operazione non è riuscita.

```
{
  "detail-type": [ "AppRunner Service Operation Status Change" ],
  "source": [ "aws.apprunner" ],
  "account": [ "111122223333" ],
  "detail": {
    "operationStatus": [
      "CreateServiceFailed",
      "DeleteServiceFailed",
      "UpdateServiceFailed",
      "DeploymentFailed",
      "PauseServiceFailed",
      "ResumeServiceFailed"
    ]
  }
```

```
}
```

Riferimento all'evento App Runner

Modifica dello stato del servizio

Un evento di modifica dello stato del servizio è `detail-type` impostato su`AppRunner Service Status Change`. Ha i seguenti campi e valori di dettaglio:

```
"serviceId": "your service ID",
"serviceName": "your service name",
"message": "Service status is set to CurrentStatus.",
"previousServiceStatus": "any valid service status",
"currentServiceStatus": "any valid service status",
"severity": "varies"
```

Modifica dello stato dell'operazione

Un evento di modifica dello stato dell'operazione è `detail-type` impostato su`AppRunner Service Operation Status Change`. Ha i seguenti campi e valori di dettaglio:

```
"operationStatus": "see following table",
"serviceName": "your service name",
"serviceId": "your service ID",
"message": "see following table",
"severity": "varies"
```

La tabella seguente elenca tutti i possibili codici di stato e i messaggi correlati.

Stato	Messaggio
`CreateServiceStarted`	La creazione del servizio è iniziata.
`CreateServiceCompletedSuccessfully`	Creazione del servizio completata con successo.
`CreateServiceFailed`	Creazione del servizio non riuscita. Per i dettagli, consulta i registri di servizio.

Stato	Messaggio
`DeleteServiceStarted`	L'eliminazione del servizio è iniziata.
`DeleteServiceCompletedSuccessfully`	Eliminazione del servizio completata con successo.
`DeleteServiceFailed`	Eliminazione del servizio non riuscita.
`UpdateServiceStarted`	
`UpdateServiceCompletedSuccessfully`	Aggiornamento del servizio completato con successo. Viene implementata una nuova applicazione e configurazione.
	Aggiornamento del servizio completato con successo. È stata implementata una nuova configurazione.
`UpdateServiceFailed`	Aggiornamento del servizio non riuscito. Per i dettagli, consulta i registri di servizio.
`DeploymentStarted`	La distribuzione è iniziata.
`DeploymentCompletedSuccessfully`	Distribuzione completata con successo.
`DeploymentFailed`	Implementazione non riuscita. Per i dettagli, consulta i registri di servizio.
`PauseServiceStarted`	La pausa del servizio è iniziata.
`PauseServiceCompletedSuccessfully`	Sospensione del servizio completata con successo.
`PauseServiceFailed`	Sospensione del servizio non riuscita.
`ResumeServiceStarted`	Ripresa del servizio iniziata.
`ResumeServiceCompletedSuccessfully`	Ripresa del servizio completata con successo.
`ResumeServiceFailed`	Ripresa del servizio non riuscita.

Registrazione delle chiamate API App Runner con AWS CloudTrail

App Runner è integrato con AWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, ruolo o AWS servizio in App Runner. CloudTrailacquisisce tutte le chiamate API per App Runner come eventi. Le chiamate acquisite includono chiamate dalla console App Runner e chiamate di codice alle operazioni dell'API App Runner. Se crei un trail, puoi abilitare la distribuzione continua di CloudTrail eventi a un bucket Amazon S3, inclusi gli eventi per App Runner. Se non configuri un percorso, puoi comunque visualizzare gli eventi più recenti nella CloudTrail console nella cronologia degli eventi. Utilizzando le informazioni raccolte da CloudTrail, puoi determinare la richiesta che è stata fatta ad App Runner, l'indirizzo IP da cui è stata effettuata la richiesta, chi ha effettuato la richiesta, quando è stata effettuata e dettagli aggiuntivi.

Per ulteriori informazioni CloudTrail, consulta la Guida per l'AWS CloudTrail utente.

Informazioni su App Runner in CloudTrail

CloudTrail è abilitato sul tuo Account AWS quando crei l'account. Quando si verifica un'attività in App Runner, tale attività viene registrata in un CloudTrail evento insieme ad altri eventi di AWS servizio nella cronologia degli eventi. Puoi visualizzare, cercare e scaricare eventi recenti in. Account AWS Per ulteriori informazioni, consulta Visualizzazione degli eventi con la cronologia degli CloudTrail eventi.

Per una registrazione continua degli eventi del tuo sito Account AWS, compresi gli eventi per App Runner, crea un percorso. Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Per impostazione predefinita, quando si crea un percorso nella console, questo sarà valido in tutte le Regioni AWS. Il trail registra gli eventi di tutte le regioni della AWS partizione e consegna i file di log al bucket Amazon S3 specificato. Inoltre, puoi configurare altri AWS servizi per analizzare ulteriormente e agire in base ai dati sugli eventi raccolti nei log. CloudTrail Per ulteriori informazioni, consulta gli argomenti seguenti:

- Panoramica della creazione di un trail
- CloudTrail Servizi e integrazioni supportati
- Configurazione delle notifiche Amazon SNS per CloudTrail
- Ricezione di file di CloudTrail registro da più regioni e ricezione di file di CloudTrail registro da più account

Tutte le azioni di App Runner vengono registrate CloudTrail e documentate nell'API Reference. AWS App Runner Ad esempio, le chiamate a `CreateService`, `DeleteConnection`, e `StartDeployment` le azioni generano voci nei file di registro. CloudTrail

Ogni evento o voce di log contiene informazioni sull'utente che ha generato la richiesta. Le informazioni di identità consentono di determinare quanto segue:

- Se la richiesta è stata effettuata con le credenziali dell'utente IAM o root.
- Se la richiesta è stata effettuata con le credenziali di sicurezza temporanee per un ruolo o un utente federato.
- Se la richiesta è stata effettuata da un altro AWS servizio.

Per ulteriori informazioni, consulta Elemento CloudTrail userIdentity.

Comprensione delle voci dei file di registro di App Runner

Un trail è una configurazione che consente la distribuzione di eventi come file di log in un bucket Amazon S3 specificato dall'utente. CloudTrail i file di registro contengono una o più voci di registro. Un evento rappresenta una singola richiesta proveniente da qualsiasi fonte e include informazioni sull'azione richiesta, la data e l'ora dell'azione e i parametri della richiesta. CloudTrail i file di registro non sono una traccia ordinata delle chiamate API pubbliche, quindi non vengono visualizzati in un ordine specifico.

L'esempio seguente mostra una voce di CloudTrail registro che illustra l'`CreateService`azione.

> **Note**
>
> Per motivi di sicurezza, alcuni valori delle proprietà vengono oscurati nei registri e sostituiti con il testo. `HIDDEN_DUE_TO_SECURITY_REASONS` Ciò impedisce la divulgazione involontaria di informazioni segrete. Tuttavia, è ancora possibile vedere che queste proprietà sono state passate nella richiesta o restituite nella risposta.

Esempio di CloudTrail registrazione per l'azione **`CreateService`** App Runner

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
```

```
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/aws-user",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "aws-user"
  },
  "eventTime": "2020-10-02T23:25:33Z",
  "eventSource": "apprunner.amazonaws.com",
  "eventName": "CreateService",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_6) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/77.0.3865.75 Safari/537.36",
  "requestParameters": {
    "serviceName": "python-test",
    "sourceConfiguration": {
      "codeRepository": {
        "repositoryUrl": "https://github.com/github-user/python-hello",
        "sourceCodeVersion": {
          "type": "BRANCH",
          "value": "main"
        },
        "codeConfiguration": {
          "configurationSource": "API",
          "codeConfigurationValues": {
            "runtime": "python3",
            "buildCommand": "HIDDEN_DUE_TO_SECURITY_REASONS",
            "startCommand": "HIDDEN_DUE_TO_SECURITY_REASONS",
            "port": "8080",
            "runtimeEnvironmentVariables": "HIDDEN_DUE_TO_SECURITY_REASONS"
          }
        }
      },
      "autoDeploymentsEnabled": true,
      "authenticationConfiguration": {
        "connectionArn": "arn:aws:apprunner:us-east-2:123456789012:connection/your-
connection/e7656250f67242d7819feade6800f59e"
      }
    },
    "healthCheckConfiguration": {
      "protocol": "HTTP"
    },
    "instanceConfiguration": {
      "cpu": "256",
```

```
      "memory": "1024"
    }
  },
  "responseElements": {
    "service": {
        "serviceName": "python-test",
        "serviceId": "dfa2b7cc7bcb4b6fa6c1f0f4efff988a",
        "serviceArn": "arn:aws:apprunner:us-east-2:123456789012:service/python-test/
dfa2b7cc7bcb4b6fa6c1f0f4efff988a",
        "serviceUrl": "generated domain",
        "createdAt": "2020-10-02T23:25:32.650Z",
        "updatedAt": "2020-10-02T23:25:32.650Z",
        "status": "OPERATION_IN_PROGRESS",
        "sourceConfiguration": {
            "codeRepository": {
                "repositoryUrl": "https://github.com/github-user/python-hello",
                "sourceCodeVersion": {
                    "type": "Branch",
                    "value": "main"
                },
                "sourceDirectory": "/",
                "codeConfiguration": {
                    "codeConfigurationValues": {
                        "configurationSource": "API",
                        "runtime": "python3",
                        "buildCommand": "HIDDEN_DUE_TO_SECURITY_REASONS",
                        "startCommand": "HIDDEN_DUE_TO_SECURITY_REASONS",
                        "port": "8080",
                        "runtimeEnvironmentVariables": "HIDDEN_DUE_TO_SECURITY_REASONS"
                    }
                }
            },
            "autoDeploymentsEnabled": true,
            "authenticationConfiguration": {
                "connectionArn": "arn:aws:apprunner:us-east-2:123456789012:connection/
your-connection/e7656250f67242d7819feade6800f59e"
            }
        },
        "healthCheckConfiguration": {
            "protocol": "HTTP",
            "path": "/",
            "interval": 5,
            "timeout": 2,
            "healthyThreshold": 3,
```

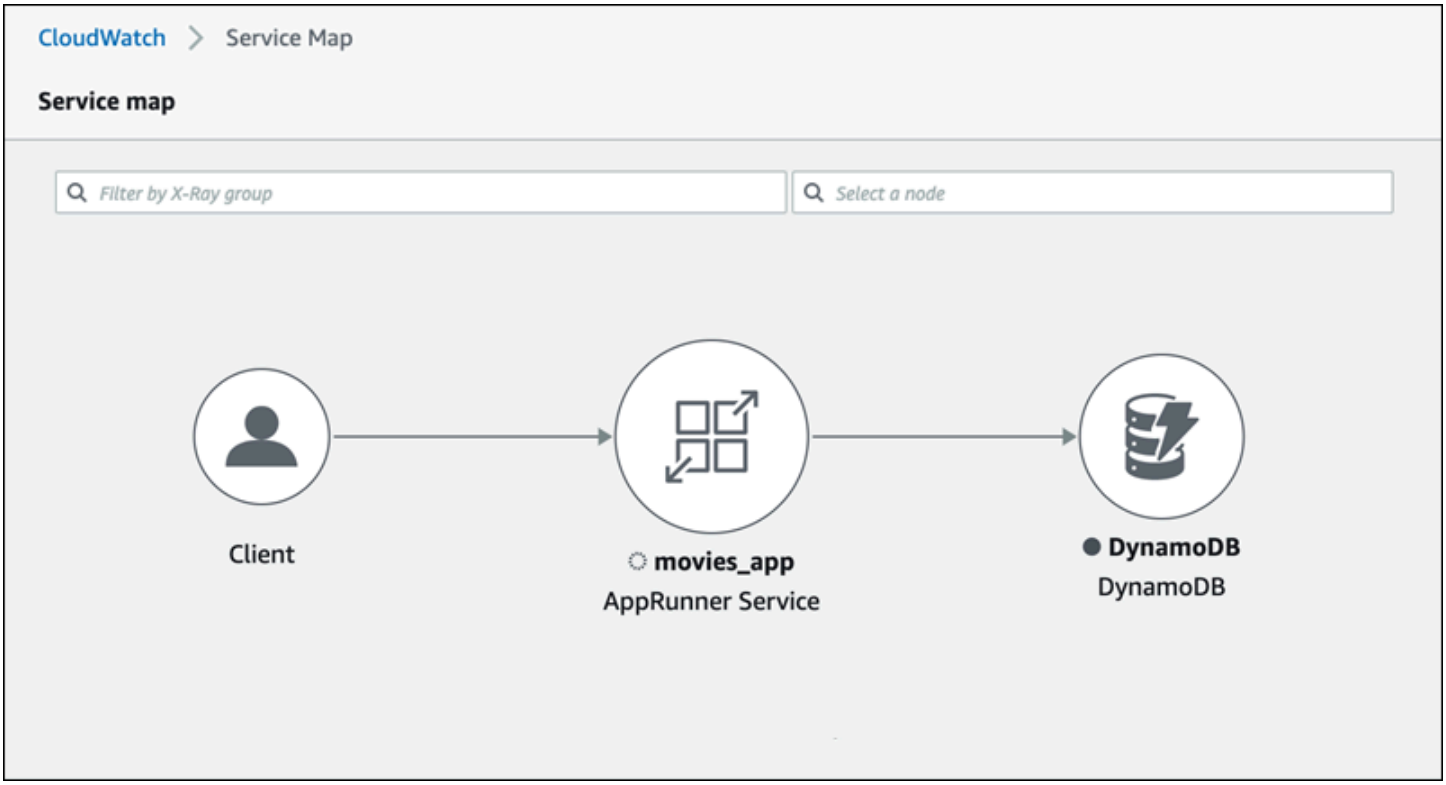
```
            "unhealthyThreshold": 5
        },
        "instanceConfiguration": {
            "cpu": "256",
            "memory": "1024"
        },
        "autoScalingConfigurationSummary": {
            "autoScalingConfigurationArn": "arn:aws:apprunner:us-
east-2:123456789012:autoscalingconfiguration/
DefaultConfiguration/1/00000000000000000000000000000001",
            "autoScalingConfigurationName": "DefaultConfiguration",
            "autoScalingConfigurationRevision": 1
        }
    }
},
  "requestID": "1a60af60-ecf5-4280-aa8f-64538319ba0a",
  "eventID": "e1a3f623-4d24-4390-a70b-bf08a0e24669",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```


Tracciamento per l'applicazione App Runner con X-Ray

AWS X-Ray è un servizio che raccoglie dati sulle richieste servite dall'applicazione e fornisce strumenti che è possibile utilizzare per visualizzare, filtrare e acquisire informazioni su tali dati per identificare problemi e opportunità di ottimizzazione. Per ogni richiesta tracciata all'applicazione, è possibile visualizzare informazioni dettagliate non solo sulla richiesta e sulla risposta, ma anche sulle chiamate effettuate dall'applicazione verso AWS risorse a valle, microservizi, database e siti Web HTTP. APIs

X-Ray utilizza i dati di traccia provenienti dalle AWS risorse che alimentano le applicazioni cloud per generare un grafico dettagliato dei servizi. Il grafo del servizio mostra il client, il tuo servizio di front-end e i servizi di back-end invocati dal servizio di front-end per elaborare le richieste e memorizzare i dati in modo persistente. Utilizza il grafo del servizio per identificare i colli di bottiglia, i picchi di latenza e altri problemi da risolvere per migliorare le prestazioni delle applicazioni.

Per ulteriori informazioni su X-Ray, consulta la Guida per gli sviluppatori di AWS X-Ray.



Strumenta la tua applicazione per il tracciamento

Strumenta la tua applicazione di servizio App Runner per il tracciamento utilizzando una specifica di OpenTelemetrytelemetria portatile. Al momento, App Runner supporta AWS Distro for OpenTelemetry (ADOT), un' OpenTelemetryimplementazione che raccoglie e presenta informazioni di telemetria utilizzando i servizi. AWS X-Ray implementa il componente di tracciamento.

A seconda dello specifico ADOT SDK utilizzato nell'applicazione, ADOT supporta fino a due approcci strumentali: automatico e manuale. Per ulteriori informazioni sulla strumentazione con il tuo SDK, consulta la documentazione ADOT e scegli il tuo SDK nel pannello di navigazione.

Configurazione del runtime

Di seguito sono riportate le istruzioni generali di configurazione del runtime per strumentare l'applicazione di servizio App Runner per il tracciamento.

Per configurare il tracciamento per il runtime

1. Segui le istruzioni fornite per il runtime in AWS Distro for OpenTelemetry (ADOT), per strumentare l'applicazione.

2. Installa le `OTEL` dipendenze richieste nella `build` sezione del `apprunner.yaml` file se stai usando il repository del codice sorgente o nel Dockerfile se stai usando un'immagine del contenitore.
3. Imposta le variabili di ambiente nel `apprunner.yaml` file se stai usando il repository del codice sorgente o nel Dockerfile se stai usando un'immagine del contenitore.

   Example Variabili di ambiente

   > **Note**
   >
   > L'esempio seguente elenca le importanti variabili di ambiente da aggiungere al file. `apprunner.yaml` Aggiungi queste variabili di ambiente al tuo Dockerfile se stai usando un'immagine del contenitore. Tuttavia, ogni runtime può avere le proprie idiosincrasie e potrebbe essere necessario aggiungere altre variabili di ambiente all'elenco seguente. Per ulteriori informazioni sulle istruzioni specifiche del runtime ed esempi su come configurare l'applicazione per il runtime, consultate AWS Distro for OpenTelemetry e vai al runtime, in Getting Started.

```
env:
    - name: OTEL_PROPAGATORS
      value: xray
    - name: OTEL_METRICS_EXPORTER
      value: none
    - name: OTEL_EXPORTER_OTLP_ENDPOINT
      value: http://localhost:4317
    - name: OTEL_RESOURCE_ATTRIBUTES
      value: 'service.name=example_app'
```

   > **Note**
   >
   > `OTEL_METRICS_EXPORTER=none`è una variabile di ambiente importante per App Runner poiché il raccoglitore App Runner Otel non accetta la registrazione delle metriche. Accetta solo il tracciamento delle metriche.

Esempio di configurazione del runtime

L'esempio seguente dimostra la strumentazione automatica dell'applicazione con ADOT Python SDK. L'SDK produce automaticamente intervalli con dati di telemetria che descrivono i valori utilizzati dai framework Python nell'applicazione senza aggiungere una sola riga di codice Python. È necessario aggiungere o modificare solo poche righe in due file sorgente.

Innanzitutto, aggiungete alcune dipendenze, come illustrato nell'esempio seguente.

Example requirements.txt

```
opentelemetry-distro[otlp]>=0.24b0
opentelemetry-sdk-extension-aws~=2.0
opentelemetry-propagator-aws-xray~=1.0
```

Quindi, strumentate la vostra applicazione. Il modo per farlo dipende dalla fonte del servizio, dall'immagine sorgente o dal codice sorgente.

Source image

Se la fonte del servizio è un'immagine, puoi utilizzare direttamente il Dockerfile che controlla la creazione dell'immagine del contenitore e l'esecuzione dell'applicazione nell'immagine. L'esempio seguente mostra un Dockerfile con strumenti per un'applicazione Python. Le aggiunte alla strumentazione sono enfatizzate in grassetto.

Example Dockerfile

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN yum install python3.7 -y && curl -O https://bootstrap.pypa.io/get-pip.py &&
 python3 get-pip.py && yum update -y
COPY . /app
WORKDIR /app
RUN pip3 install -r requirements.txt
RUN opentelemetry-bootstrap --action=install
ENV OTEL_PYTHON_DISABLED_INSTRUMENTATIONS=urllib3
ENV OTEL_METRICS_EXPORTER=none
ENV OTEL_RESOURCE_ATTRIBUTES='service.name=example_app'
CMD OTEL_PROPAGATORS=xray OTEL_PYTHON_ID_GENERATOR=xray opentelemetry-instrument
 python3 app.py
EXPOSE 8080
```

Source code repository

Quando la fonte del servizio è un repository contenente l'origine dell'applicazione, si strumenta indirettamente l'immagine utilizzando le impostazioni del file di configurazione di App Runner. Queste impostazioni controllano il Dockerfile che App Runner genera e utilizza per creare l'immagine per l'applicazione. L'esempio seguente mostra un file di configurazione App Runner strumentato per un'applicazione Python. Le aggiunte alla strumentazione sono evidenziate in grassetto.

Example apprunner.yaml

```
version: 1.0
runtime: python3
build:
  commands:
    build:
      - pip install -r requirements.txt
      - opentelemetry-bootstrap --action=install
run:
  command: opentelemetry-instrument python app.py
  network:
    port: 8080
  env:
    - name: OTEL_PROPAGATORS
      value: xray
    - name: OTEL_METRICS_EXPORTER
      value: none
    - name: OTEL_PYTHON_ID_GENERATOR
      value: xray
    - name: OTEL_PYTHON_DISABLED_INSTRUMENTATIONS
      value: urllib3
    - name: OTEL_RESOURCE_ATTRIBUTES
      value: 'service.name=example_app'
```

Aggiungi le autorizzazioni X-Ray al tuo ruolo di istanza del servizio App Runner

Per utilizzare il tracciamento a raggi X con il servizio App Runner, è necessario fornire alle istanze del servizio le autorizzazioni per interagire con il servizio X-Ray. Puoi farlo associando un ruolo di istanza al tuo servizio e aggiungendo una policy gestita con autorizzazioni X-Ray. Per ulteriori informazioni

su un ruolo di istanza di App Runner, consulta. the section called “Ruolo dell'istanza” Aggiungi la policy `AWSXRayDaemonWriteAccess` gestita al ruolo dell'istanza e assegnala al servizio durante la creazione.

Abilita il tracciamento X-Ray per il tuo servizio App Runner

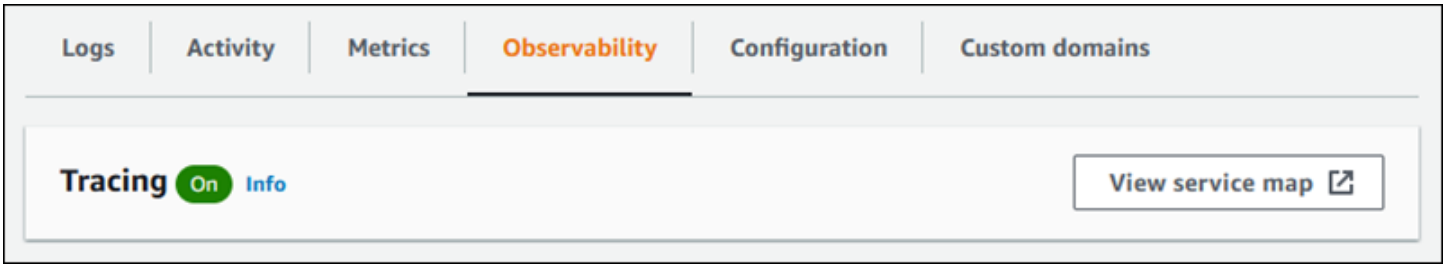
Quando crei un servizio, App Runner disabilita il tracciamento per impostazione predefinita. Puoi abilitare il tracciamento a raggi X per il tuo servizio come parte della configurazione dell'osservabilità. Per ulteriori informazioni, consulta the section called “Gestisci l'osservabilità”.

Se si utilizza l'API App Runner o AWS CLI, l'TraceConfigurationoggetto all'interno dell'oggetto ObservabilityConfigurationrisorsa contiene le impostazioni di tracciamento. Per mantenere disabilitato il tracciamento, non specificate un oggetto. `TraceConfiguration`

Sia nella console che nell'API, assicurati di associare il ruolo dell'istanza discusso nella sezione precedente al servizio App Runner.

Visualizza i dati di tracciamento X-Ray per il tuo servizio App Runner

Nella scheda Osservabilità della pagina del dashboard del servizio nella console App Runner, scegli Visualizza mappa del servizio per accedere alla console Amazon CloudWatch .



Usa la CloudWatch console Amazon per visualizzare le mappe e le tracce dei servizi per le richieste servite dalla tua applicazione. Le mappe dei servizi mostrano informazioni come la latenza delle richieste e le interazioni con altre applicazioni e AWS servizi. Le annotazioni personalizzate che aggiungi al codice ti consentono di cercare facilmente le tracce. Per ulteriori informazioni, consulta Using ServiceLens to monitoring the health of your application nella Amazon CloudWatch User Guide.

Associazione di un ACL AWS WAF Web al servizio

AWS WAF è un firewall per applicazioni Web che puoi utilizzare per proteggere il tuo servizio App Runner. Con AWS WAF gli elenchi di controllo degli accessi Web (web ACLs), puoi proteggere gli endpoint del servizio App Runner da exploit Web comuni e bot indesiderati.

Un ACL web ti offre un controllo dettagliato su tutte le richieste web in arrivo al tuo servizio App Runner. È possibile definire regole in un ACL Web per consentire, bloccare o monitorare il traffico Web, per garantire che solo le richieste autorizzate e legittime raggiungano le applicazioni Web e. APIs È possibile personalizzare le regole ACL Web in base alle esigenze aziendali e di sicurezza specifiche. Per ulteriori informazioni sulla sicurezza dell'infrastruttura e sulle best practice per l'applicazione della rete ACLs, consulta Control network traffic nella Amazon VPC User Guide.

> **Important**
>
> Le regole IP di origine per i servizi privati di App Runner associati al web WAF ACLs non rispettano le regole basate sull'IP. Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché le regole Web WAF. ACLs

Flusso di richieste web in entrata

Quando un ACL AWS WAF Web è associato a un servizio App Runner, le richieste Web in entrata passano attraverso il seguente processo:

1. App Runner inoltra il contenuto della richiesta di origine a. AWS WAF
2. AWS WAF ispeziona la richiesta e ne confronta il contenuto con le regole specificate nell'ACL web.
3. In base alla sua ispezione, AWS WAF restituisce una `block` risposta `allow` or ad App Runner.
   - Se viene restituita una `allow` risposta, App Runner inoltra la richiesta all'applicazione.
   - Se viene restituita una `block` risposta, App Runner impedisce alla richiesta di raggiungere l'applicazione web. Inoltra la `block` risposta AWS WAF all'applicazione.

> **Note**
>
> Per impostazione predefinita, App Runner blocca la richiesta se non viene restituita alcuna risposta. AWS WAF

Per ulteriori informazioni sul AWS WAF web ACLs, consulta Web access control lists (web ACLs) nella AWS WAF Developer Guide.

> **Note**
>
> Paghi un AWS WAF prezzo standard. Non è previsto alcun costo aggiuntivo per l'utilizzo del AWS WAF Web ACLs per i servizi App Runner. Per ulteriori informazioni sui prezzi, consulta Prezzi.AWS WAF

Associazione di WAF web ACLs al servizio App Runner

Di seguito è riportato il processo di alto livello per associare un ACL AWS WAF web al servizio App Runner:

1. Crea un ACL web nella console. AWS WAF Per ulteriori informazioni, consulta Creazione di un ACL Web nella Guida per gli AWS WAF sviluppatori.
2. Aggiorna le tue autorizzazioni AWS Identity and Access Management (IAM) per. AWS WAF Per ulteriori informazioni, consulta l'argomento relativo alle autorizzazioni .
3. Associate l'ACL Web al servizio App Runner utilizzando uno dei seguenti metodi:
   - Console App Runner: associa un ACL Web esistente utilizzando la console App Runner quando crei o aggiorni un servizio App Runner. Per istruzioni, consulta Gestione del web. AWS WAF ACLs
   - AWS WAF console: associa l'ACL Web utilizzando la AWS WAF console per un servizio App Runner esistente. Per ulteriori informazioni, consulta Associare o dissociare un ACL Web con una risorsa AWS nella Developer Guide.AWS WAF
   - AWS CLI: Associa l'ACL web utilizzando l'ACL pubblico. AWS WAF APIs Per ulteriori informazioni su AWS WAF public APIs, consulta AssociateWebACL nella AWS WAF API Reference Guide.

Considerazioni

- Le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP. Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché le regole Web WAF. ACLs
- Un servizio App Runner può essere associato a un solo ACL web. Tuttavia, è possibile associare un ACL Web a più servizi App Runner e a più risorse. AWS Gli esempi includono i pool di utenti di Amazon Cognito e le risorse Application Load Balancer.
- Quando crei un ACL Web, passa un breve lasso di tempo prima che l'ACL Web si propaghi completamente e sia disponibile per App Runner. Il tempo di propagazione può variare da pochi secondi a diversi minuti. AWS WAF restituisce a `WAFUnavailableEntityException` quando si tenta di associare un ACL Web prima che si sia completamente propagato.

  Se si aggiorna il browser o si esce dalla console App Runner prima che l'ACL Web sia completamente propagato, l'associazione non viene eseguita. Tuttavia, puoi navigare all'interno della console App Runner.
- AWS WAF restituisce un `WAFNonexistentItemException` errore quando si chiama una delle seguenti opzioni AWS WAF APIs per un servizio App Runner che si trova in uno stato non valido:
  - `AssociateWebACL`
  - `DisassociateWebACL`
  - `GetWebACLForResource`

  Gli stati non validi per il servizio App Runner includono:
  - `CREATE_FAILED`
  - `DELETE_FAILED`
  - `DELETED`
  - `OPERATION_IN_PROGRESS`

  > **Note**
  >
  > `OPERATION_IN_PROGRESS`Lo stato non è valido solo se il servizio App Runner viene eliminato.

- La tua richiesta potrebbe comportare un carico utile superiore ai limiti di ciò AWS WAF che può ispezionare. Per ulteriori informazioni su come AWS WAF gestisce le richieste di grandi dimensioni provenienti da App Runner, consulta Gestione dei componenti di richieste di grandi dimensioni nella Guida per gli AWS WAF sviluppatori per scoprire come AWS WAF gestire le richieste di grandi dimensioni provenienti da App Runner.
- Se non imposti regole appropriate o i modelli di traffico cambiano, un ACL web potrebbe non essere altrettanto efficace nel proteggere l'applicazione.

Autorizzazioni

Per utilizzare un ACL Web AWS App Runner, aggiungi le seguenti autorizzazioni IAM per: AWS WAF

- `apprunner:ListAssociatedServicesForWebAcl`
- `apprunner:DescribeWebAclForService`
- `apprunner:AssociateWebAcl`
- `apprunner:DisassociateWebAcl`

Per ulteriori informazioni sulle autorizzazioni IAM, consulta Policies and permissions in IAM nella IAM User Guide.

Di seguito è riportato un esempio della policy IAM aggiornata per. AWS WAF Questa policy IAM include le autorizzazioni necessarie per lavorare con un servizio App Runner.

Example

```
{
  {
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "wafv2:ListResourcesForWebACL",
            "wafv2:GetWebACLForResource",
            "wafv2:AssociateWebACL",
            "wafv2:DisassociateWebACL",
            "apprunner:ListAssociatedServicesForWebAcl",
            "apprunner:DescribeWebAclForService",
```

```
            "apprunner:AssociateWebAcl",
            "apprunner:DisassociateWebAcl"
         ],
         "Resource":"*"
      }
   ]
}
```

> **Note**
>
> Sebbene sia necessario concedere le autorizzazioni IAM, le azioni elencate fanno riferimento solo alle autorizzazioni e non corrispondono a un'operazione API.

Gestione del AWS WAF web ACLs

Gestisci il AWS WAF Web ACLs per il tuo servizio App Runner utilizzando uno dei seguenti metodi:

- the section called “Console App Runner”
- the section called “AWS CLI”

Console App Runner

Quando crei un servizio o ne aggiorni uno esistente sulla console App Runner, puoi associare o dissociare un AWS WAF ACL Web.

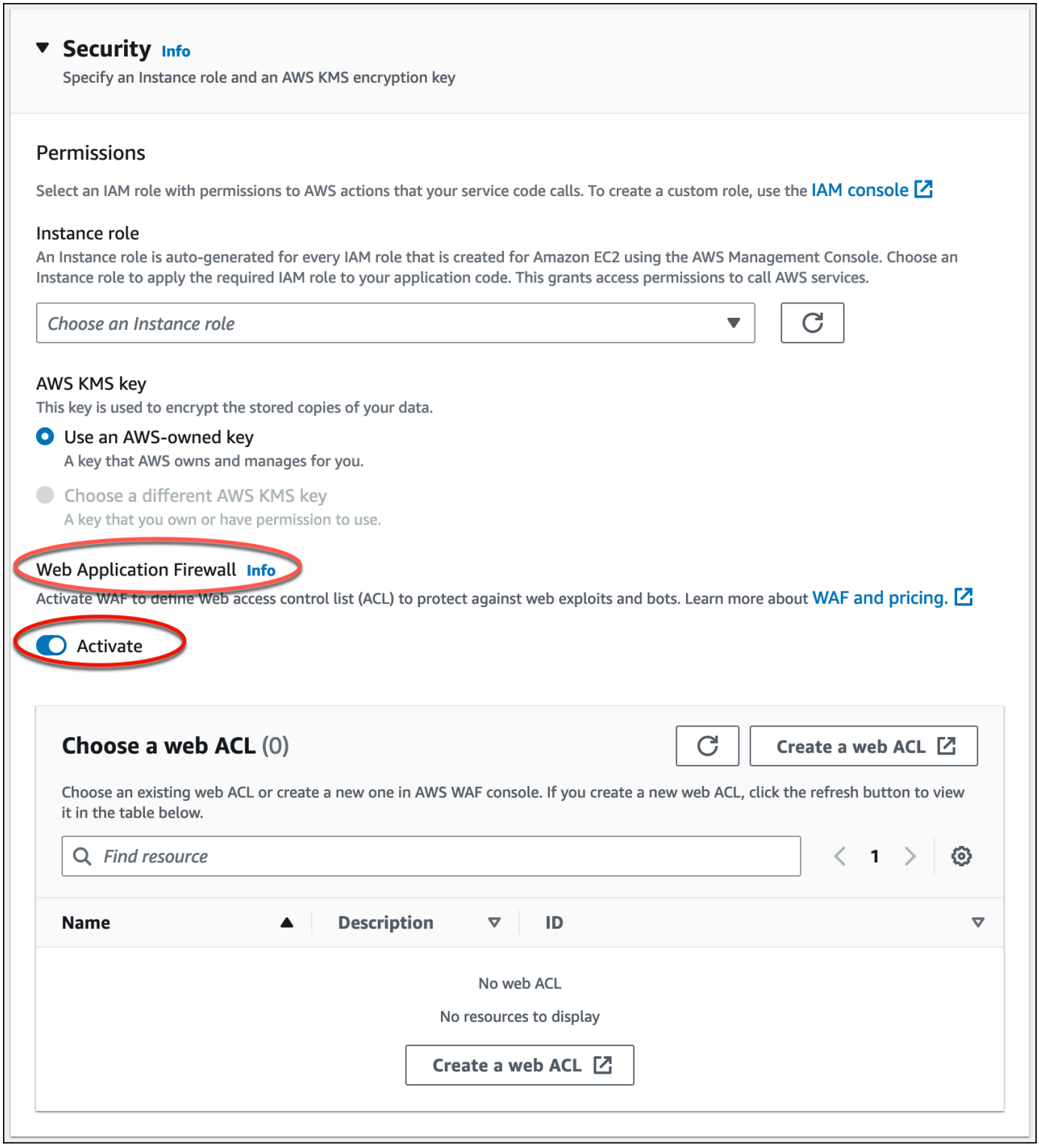
> **Note**
>
> - Un servizio App Runner può essere associato a un solo ACL web. Tuttavia, è possibile associare un ACL Web a più di un servizio App Runner oltre ad altre risorse. AWS
> - Prima di associare un ACL web, assicurati di aggiornare le autorizzazioni IAM per. AWS WAF Per ulteriori informazioni, consulta l'argomento relativo alle autorizzazioni .

Associazione AWS WAF di un ACL web

> **Important**
>
> Le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP. Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché le regole Web WAF. ACLs

AWS WAF Per associare un ACL web

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. A seconda che tu stia creando o aggiornando un servizio, esegui una delle seguenti operazioni:
   - Se stai creando un nuovo servizio, scegli Crea un servizio App Runner e vai a Configura servizio.
   - Se stai aggiornando un servizio esistente, scegli la scheda Configurazione, quindi scegli Modifica in Configura servizio.
3. Vai a Web Application Firewall in Sicurezza.
4. Scegli il pulsante Attiva per visualizzare le opzioni.



5. Eseguire una delle seguenti fasi:

   - Per associare un ACL Web esistente: scegli l'ACL Web richiesto dalla tabella Scegli un ACL Web da associare al servizio App Runner.

- Per creare un nuovo ACL Web: scegli Crea ACL Web per creare un nuovo ACL Web utilizzando la console. AWS WAF Per ulteriori informazioni, consulta Creazione di un ACL Web nella Guida per gli sviluppatori.AWS WAF
  1. Scegli il pulsante di aggiornamento per visualizzare l'ACL web appena creato nella tabella Scegli un ACL web.
  2. Seleziona l'ACL web richiesto.

6. Scegli Avanti se stai creando un nuovo servizio o Salva modifiche se stai aggiornando un servizio esistente. L'ACL web selezionato è associato al servizio App Runner.
7. Per verificare l'associazione ACL web, scegli la scheda Configurazione del tuo servizio e vai a Configura servizio. Scorri fino a Web Application Firewall in Sicurezza per visualizzare i dettagli dell'ACL web associato al tuo servizio.

> **Note**
>
> Quando crei un ACL Web, trascorre un breve periodo di tempo prima che l'ACL Web si propaghi completamente e sia disponibile per App Runner. Il tempo di propagazione può variare da pochi secondi a diversi minuti. AWS WAF restituisce a `WAFUnavailableEntityException` quando si tenta di associare un ACL Web prima che si sia completamente propagato.
> Se si aggiorna il browser o si esce dalla console App Runner prima che l'ACL Web sia completamente propagato, l'associazione non viene eseguita. Tuttavia, puoi navigare all'interno della console App Runner.

Dissociazione di un ACL Web AWS WAF

Puoi dissociare i AWS WAF siti Web ACl che non ti servono più aggiornando il servizio App Runner.

Per dissociare un Web AWS WAF ACl

1. Apri la console App Runner e, nell'elenco Regioni, seleziona la tua. Regione AWS
2. Vai alla scheda Configurazione del servizio che desideri aggiornare e scegli Modifica in Configura servizio.
3. Vai a Web Application Firewall in Sicurezza.
4. Disattiva il pulsante Attiva. Riceverai un messaggio per confermare l'eliminazione.
5. Scegli Conferma. L'ACL web è dissociato dal servizio App Runner.

> **Note**
>
> - Se desideri associare il tuo servizio a un altro ACL web, seleziona un ACL web dalla tabella Scegli un ACL web. App Runner dissocia l'ACL web corrente e avvia il processo di associazione all'ACL web selezionato.
> - Se nessun altro servizio o risorsa di App Runner utilizza un ACL web dissociato, valuta la possibilità di eliminare l'ACL web. In caso contrario, continuerai a sostenere dei costi. Per ulteriori informazioni sui prezzi, consulta Prezzi di AWS WAF. Per istruzioni su come eliminare un ACL Web, consulta DeleteWebACL nell'API Reference.AWS WAF
> - Non puoi eliminare un ACL web associato ad altri servizi App Runner attivi o ad altre risorse.

AWS CLI

È possibile associare o dissociare un ACL AWS WAF Web utilizzando il pubblico. AWS WAF APIs Il servizio App Runner, a cui si desidera associare o dissociare un ACL Web, deve trovarsi in uno stato valido.

AWS WAF restituisce un `WAFNonexistentItemException` errore quando si chiama una delle seguenti opzioni AWS WAF APIs per un servizio App Runner che si trova in uno stato non valido:

- `AssociateWebACL`
- `DisassociateWebACL`
- `GetWebACLForResource`

Gli stati non validi per il servizio App Runner includono:

- `CREATE_FAILED`
- `DELETE_FAILED`
- `DELETED`
- `OPERATION_IN_PROGRESS`

> **Note**
>
> `OPERATION_IN_PROGRESS`Lo stato non è valido solo se il servizio App Runner viene eliminato.

Per ulteriori informazioni su AWS WAF public APIs, consulta la AWS WAF API Reference Guide.

> **Note**
>
> Aggiorna le tue autorizzazioni IAM per AWS WAF. Per ulteriori informazioni, consulta l'argomento relativo alle autorizzazioni .

Associazione dell'ACL AWS WAF web tramite AWS CLI

> **Important**
>
> Le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP. Questo perché attualmente non supportiamo l'inoltro dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché le regole Web WAF. ACLs

AWS WAF Per associare un ACL web

1. Crea un ACL AWS WAF Web per il tuo servizio con il set preferito di azioni di regole `Allow` o richieste Web `Block` al tuo servizio. Per ulteriori informazioni in merito AWS WAF APIs, consulta CreateWebACL nella Guida di riferimento dell'AWS WAF API.

   Example Crea un ACL web - Richiesta

   ```
   aws wafv2
   create-web-acl
   --region <region>
   --name <web-acl-name>
   ```

```
--scope REGIONAL
--default-action Allow={}
--visibility-config <file-name.json>
# This is the file containing the WAF web ACL rules.
```

2. Associa l'ACL web che hai creato al servizio App Runner utilizzando l'`associate-web-acl` AWS WAF API pubblica. Per ulteriori informazioni in merito AWS WAF APIs, consulta AssociateWebACL nella AWS WAF API Reference Guide.

> **Note**
>
> Quando si crea un ACL Web, trascorre un breve lasso di tempo prima che l'ACL Web si propaghi completamente e sia disponibile per App Runner. Il tempo di propagazione può variare da pochi secondi a diversi minuti. AWS WAF restituisce a `WAFUnavailableEntityException` quando si tenta di associare un ACL Web prima che si sia completamente propagato.
> Se si aggiorna il browser o si esce dalla console App Runner prima che l'ACL Web sia completamente propagato, l'associazione non viene eseguita. Tuttavia, puoi navigare all'interno della console App Runner.

Example Associazione di un ACL web - Richiesta

```
aws wafv2 associate-web-acl
--resource-arn <apprunner_service_arn>
--web-acl-arn <web_acl_arn>
--region <region>
```

3. Verifica che l'ACL web sia associato al tuo servizio App Runner utilizzando l'API pubblica. `get-web-acl-for-resource` AWS WAF Per ulteriori informazioni in merito AWS WAF APIs, consulta GetWebACLForResource in the AWS WAF API Reference Guide.

Example Verifica l'ACL web per la risorsa - Richiesta

```
aws wafv2 get-web-acl-for-resource
--resource-arn <apprunner_service_arn>
--region <region>
```

Se non ci sono siti Web ACLs associati al tuo servizio, riceverai una risposta vuota.

Eliminazione di un AWS WAF ACL Web utilizzando AWS CLI

Non è possibile eliminare un ACL AWS WAF Web se è associato a un servizio App Runner.

Per eliminare un ACL web AWS WAF

1. Dissocia l'ACL web dal tuo servizio App Runner utilizzando l'API pubblica. `disassociate-web-acl` AWS WAF Per ulteriori informazioni in merito AWS WAF APIs, consulta DisassociateWebACL nella API Reference Guide.AWS WAF

   Example Dissociazione di un ACL Web - Richiesta

   ```
   aws wafv2 disassociate-web-acl
   --resource-arn <apprunner_service_arn>
   --region <region>
   ```

2. Verifica che l'ACL Web sia dissociato dal servizio App Runner utilizzando l'API pubblica. `get-web-acl-for-resource` AWS WAF

   Example Verifica che l'ACL web sia dissociato - Richiesta

   ```
   aws wafv2 get-web-acl-for-resource
   --resource-arn <apprunner_service_arn>
   --region <region>
   ```

   L'ACL web dissociato non è elencato per il tuo servizio App Runner. Se non ci sono siti Web ACLs associati al servizio, riceverai una risposta vuota.

3. Elimina l'ACL web dissociato utilizzando l'API `delete-web-acl` AWS WAF pubblica. Per ulteriori informazioni in merito AWS WAF APIs, consulta DeleteWebACL nella AWS WAF API Reference Guide.

   Example Eliminare un ACL web - Richiesta

   ```
   aws wafv2 delete-web-acl
   --name <web_acl_name>
   --scope REGIONAL
   --id <web_acl_id>
   --lock-token <web_acl_lock_token>
   --region <region>
   ```

4. Verifica che l'ACL Web venga eliminato utilizzando l'API `list-web-acl` AWS WAF pubblica. Per ulteriori informazioni ListWebACLsin merito AWS WAF APIs, consulta la Guida di riferimento dell'AWS WAF API.

   Example Verifica che l'ACL web sia eliminato - Richiesta

   ```
   aws wafv2 list-web-acls
   --scope REGIONAL
   --region <region>
   ```

   L'ACL web eliminato non è più presente nell'elenco.

   > **Note**
   >
   > Se un ACL Web è associato ad altri servizi App Runner attivi o ad altre risorse, come i pool di utenti di Amazon Cognito, l'ACL Web non può essere eliminato.

Elenco dei servizi App Runner associati a un ACL web

Un ACL web può essere associato a più servizi App Runner e altre risorse. Elenca i servizi App Runner associati a un ACL Web utilizzando l'API pubblica. `list-resources-for-web-acl` AWS WAF Per ulteriori informazioni in merito AWS WAF APIs, consulta ListResourcesForWebACL nella AWS WAF API Reference Guide.

Example Elenca i servizi App Runner associati a un ACL web - Request

```
aws wafv2 list-resources-for-web-acl
--web-acl-arn <WEB_ACL_ARN>
--resource-type APP_RUNNER_SERVICE
--region <REGION>
```

Example Elenca i servizi App Runner associati a un ACL web - Response

L'esempio seguente illustra la risposta quando non ci sono servizi App Runner associati a un ACL web.

```
{
  "ResourceArns": []
```

```
}
```

Example Elenca i servizi App Runner associati a un ACL web - Response

L'esempio seguente illustra la risposta quando esistono servizi App Runner associati a un ACL Web.

```
{
  "ResourceArns": [
    "arn:aws:apprunner:<region>:<aws_account_id>:service/<service_name>/<service_id>"
  ]
}
```

Test e registrazione web AWS WAF ACLs

Quando imposti un'azione della regola su Count nell'ACL web, AWS WAF aggiunge la richiesta a un conteggio di richieste che corrispondono alla regola. Per testare un ACL web con il servizio App Runner, imposta le azioni delle regole su Count e considera il volume di richieste che corrispondono a ciascuna regola. Ad esempio, imposti una regola per l'`Block`azione che corrisponde a un gran numero di richieste che ritieni essere il normale traffico utente. In tal caso, potrebbe essere necessario riconfigurare la regola. Per ulteriori informazioni, consulta Testare e ottimizzare AWS WAF le protezioni nella Guida per gli AWS WAF sviluppatori.

Puoi anche configurare AWS WAF per registrare le intestazioni delle richieste in un gruppo di log di Amazon CloudWatch Logs, un bucket Amazon Simple Storage Service (Amazon S3) o un Amazon Data Firehose. Per ulteriori informazioni, consulta Registrazione del traffico ACL Web nella Guida per gli sviluppatori di AWS WAF .

Per accedere ai log relativi all'ACL web associato al tuo servizio App Runner, consulta i seguenti campi di registro:

- `httpSourceName`: Contiene `APPRUNNER`
- `httpSourceId`: Contiene `customeraccountid-apprunnerserviceid`

Per ulteriori informazioni, consulta Log Examples nella AWS WAF Developer Guide.

> **Important**
>
> Le regole IP di origine per i servizi privati di App Runner associati a WAF web ACLs non rispettano le regole basate sull'IP. Questo perché attualmente non supportiamo l'inoltro

dei dati IP di origine della richiesta ai servizi privati di App Runner associati a WAF. Se l'applicazione App Runner richiede le regole di controllo del traffico in entrata IP/CIDR di origine, è necessario utilizzare le regole dei gruppi di sicurezza per gli endpoint privati anziché le regole Web WAF. ACLs

Impostazione delle opzioni del servizio App Runner utilizzando un file di configurazione

> **Note**
>
> I file di configurazione sono applicabili solo ai servizi basati sul codice sorgente. Non è possibile utilizzare i file di configurazione con servizi basati su immagini.

Quando si crea un AWS App Runner servizio utilizzando un repository di codice sorgente, sono AWS App Runner necessarie informazioni sulla creazione e l'avvio del servizio. È possibile fornire queste informazioni ogni volta che si crea un servizio utilizzando la console o l'API App Runner. In alternativa, è possibile impostare le opzioni di servizio utilizzando un file di configurazione. Le opzioni specificate in un file diventano parte dell'archivio di origine e tutte le modifiche a queste opzioni vengono registrate in modo analogo a come vengono tracciate le modifiche al codice sorgente. Puoi utilizzare il file di configurazione di App Runner per specificare più opzioni di quelle supportate dall'API. Non è necessario fornire un file di configurazione se sono necessarie solo le opzioni di base supportate dall'API.

Il file di configurazione di App Runner è un file YAML denominato `apprunner.yaml` nella directory dei sorgenti del repository dell'applicazione. Fornisce opzioni di compilazione e runtime per il tuo servizio. I valori in questo file indicano ad App Runner come creare e avviare il servizio e forniscono un contesto di runtime come le impostazioni di rete e le variabili di ambiente.

Il file di configurazione di App Runner non include impostazioni operative, come CPU e memoria.

Per esempi di file di configurazione di App Runner, vedi. the section called “Esempi” Per una guida di riferimento completa, vederethe section called “Documentazione di riferimento”.

Argomenti

- Esempi di file di configurazione di App Runner
- Riferimento al file di configurazione di App Runner

Esempi di file di configurazione di App Runner

> **Note**
>
> I file di configurazione sono applicabili solo ai servizi basati sul codice sorgente. Non è possibile utilizzare i file di configurazione con servizi basati su immagini.

Gli esempi seguenti mostrano i file di AWS App Runner configurazione. Alcuni sono minimi e contengono solo le impostazioni necessarie. Altre sono complete, incluse tutte le sezioni del file di configurazione. Per una panoramica dei file di configurazione di App Runner, vedereFile di configurazione di App Runner.

Esempi di file di configurazione

File di configurazione minimo

Con un file di configurazione minimo, App Runner fa le seguenti ipotesi:

- Non sono necessarie variabili di ambiente personalizzate durante la compilazione o l'esecuzione.
- Viene utilizzata la versione di runtime più recente.
- Vengono utilizzati il numero di porta e la variabile di ambiente di porta predefiniti.

Example apprunner.yaml

```
version: 1.0
runtime: python3
build:
  commands:
    build:
      - pip install pipenv
      - pipenv install
run:
  command: python app.py
```

File di configurazione completo

Questo esempio mostra l'uso di tutte le chiavi di configurazione nel formato `apprunner.yaml` originale con un runtime gestito.

Example apprunner.yaml

```
version: 1.0
runtime: python3
build:
  commands:
    pre-build:
      - wget -c https://s3.amazonaws.com/amzn-s3-demo-bucket/test-lib.tar.gz -O - | tar
 -xz
    build:
      - pip install pipenv
      - pipenv install
    post-build:
      - python manage.py test
  env:
    - name: DJANGO_SETTINGS_MODULE
      value: "django_apprunner.settings"
    - name: MY_VAR_EXAMPLE
      value: "example"
run:
  runtime-version: 3.7.7
  command: pipenv run gunicorn django_apprunner.wsgi --log-file -
  network:
    port: 8000
    env: MY_APP_PORT
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
  secrets:
    - name: my-secret
      value-from: "arn:aws:secretsmanager:us-
east-1:123456789012:secret:testingstackAppRunnerConstr-kJFXde2ULKbT-S7t8xR:username::"
    - name: my-parameter
      value-from: "arn:aws:ssm:us-east-1:123456789012:parameter/parameter-name"
    - name: my-parameter-only-name
      value-from: "parameter-name"
```

File di configurazione completo — (utilizza una build rivista)

Questo esempio mostra l'uso di tutte le chiavi di configurazione in `apprunner.yaml` un runtime gestito.

Il `pre-run` parametro è supportato solo dalla build aggiornata di App Runner. Non inserire questo parametro nel file di configurazione se l'applicazione utilizza versioni di runtime supportate dalla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

> **Note**
>
> Poiché questo esempio è per Python 3.11, utilizziamo i `pip3` comandi and. `python3` Per ulteriori informazioni, consultate l'Callout per versioni di runtime specificheargomento relativo alla piattaforma Python.

Example apprunner.yaml

```
version: 1.0
runtime: python311
build:
  commands:
    pre-build:
      - wget -c https://s3.amazonaws.com/amzn-s3-demo-bucket/test-lib.tar.gz -O - | tar
 -xz
    build:
      - pip3 install pipenv
      - pipenv install
    post-build:
      - python3 manage.py test
  env:
    - name: DJANGO_SETTINGS_MODULE
      value: "django_apprunner.settings"
    - name: MY_VAR_EXAMPLE
      value: "example"
run:
  runtime-version: 3.11
  pre-run:
    - pip3 install pipenv
    - pipenv install
    - python3 copy-global-files.py
  command: pipenv run gunicorn django_apprunner.wsgi --log-file -
  network:
    port: 8000
    env: MY_APP_PORT
  env:
```

```
    - name: MY_VAR_EXAMPLE
      value: "example"
  secrets:
    - name: my-secret
      value-from: "arn:aws:secretsmanager:us-
east-1:123456789012:secret:testingstackAppRunnerConstr-kJFXde2ULKbT-S7t8xR:username::"
    - name: my-parameter
      value-from: "arn:aws:ssm:us-east-1:123456789012:parameter/parameter-name"
    - name: my-parameter-only-name
      value-from: "parameter-name"
```

Per esempi di file di configurazione di runtime gestiti specifici, consultate il sottoargomento specifico del runtime sotto. Servizio basato su codice

Riferimento al file di configurazione di App Runner

> **Note**
>
> I file di configurazione sono applicabili solo ai servizi basati sul codice sorgente. Non è possibile utilizzare i file di configurazione con servizi basati su immagini.

Questo argomento è una guida di riferimento completa alla sintassi e alla semantica di un file di configurazione. AWS App Runner Per una panoramica dei file di configurazione di App Runner, vedere. File di configurazione di App Runner

Il file di configurazione di App Runner è un file YAML. Assegnagli un `apprunner.yaml` nome e inseriscilo nella directory dei sorgenti del repository dell'applicazione.

Panoramica della struttura

Il file di configurazione di App Runner è un file YAML. Assegnagli un `apprunner.yaml` nome e inseriscilo nella directory dei sorgenti del repository dell'applicazione.

Il file di configurazione di App Runner contiene le seguenti parti principali:

- Sezione superiore: contiene le chiavi di primo livello
- Sezione di compilazione: configura la fase di compilazione
- Sezione Esegui: configura la fase di runtime

Sezione superiore

Le chiavi nella parte superiore del file forniscono informazioni generali sul file e sul runtime del servizio. Sono disponibili le seguenti chiavi:

- `version`— Obbligatorio. La versione del file di configurazione di App Runner. Idealmente, usa la versione più recente.

  Sintassi

  ```
  version: version
  ```

  Example

  ```
  version: 1.0
  ```

- `runtime`— Richiesto. Il nome del runtime utilizzato dall'applicazione. Per informazioni sui runtime disponibili per le diverse piattaforme di programmazione offerte da App Runner, consulta. Servizio basato su codice

  > **Note**
  >
  > La convenzione di denominazione di un runtime gestito è. *<language-name><major-version>*

  Sintassi

  ```
  runtime: runtime-name
  ```

  Example

  ```
  runtime: python3
  ```

Compila la sezione

La sezione build configura la fase di compilazione dell'implementazione del servizio App Runner. È possibile specificare comandi di compilazione e variabili di ambiente. I comandi di compilazione sono obbligatori.

La sezione inizia con la `build:` chiave e contiene le seguenti sottochiavi:

- `commands`— Obbligatorio. Speciifica i comandi che App Runner esegue durante le varie fasi di compilazione. Include le seguenti sottochiavi:
  - `pre-build`— Facoltativo. I comandi che App Runner esegue prima della compilazione. Ad esempio, installa npm dipendenze o librerie di test.
  - `build`— Obbligatorio. I comandi che App Runner esegue per creare l'applicazione. Ad esempio, usapipenv.
  - `post-build`— Facoltativo. I comandi che App Runner esegue dopo la compilazione. Ad esempio, usa Maven per impacchettare gli artefatti della build in un file JAR o WAR o per eseguire un test.

  Sintassi

```
build:
  commands:
    pre-build:
      - command
      - …
    build:
      - command
      - …
    post-build:
      - command
      - …
```

  Example

```
build:
  commands:
    pre-build:
      - yum install openssl
    build:
      - pip install -r requirements.txt
```

```
    post-build:
      - python manage.py test
```

- `env`— Facoltativo. Specifica variabili di ambiente personalizzate per la fase di compilazione. Definito come mappature scalari nome-valore. Puoi fare riferimento a queste variabili per nome nei comandi di build.

> **Note**
>
> Ci sono due `env` voci distinte in due posizioni diverse in questo file di configurazione. Un set si trova nella sezione Build e l'altro nella sezione Run.
>
> - Il `env` set nella sezione Build può essere referenziato dai `pre-run` comandi`pre-build`, `build``post-build`, e durante il processo di compilazione.
>
> Importante: si noti che i `pre-run` comandi si trovano nella sezione Esegui di questo file, anche se possono accedere solo alle variabili di ambiente definite nella sezione Build.
> - Il `env` set nella sezione Run può essere referenziato dal `run` comando nell'ambiente di runtime.

Sintassi

```
build:
  env:
    - name: name1
      value: value1
    - name: name2
      value: value2
    - …
```

Example

```
build:
  env:
    - name: DJANGO_SETTINGS_MODULE
      value: "django_apprunner.settings"
    - name: MY_VAR_EXAMPLE
      value: "example"
```

Sezione Esegui

La sezione run configura la fase di esecuzione del contenitore della distribuzione dell'applicazione App Runner. È possibile specificare la versione di runtime, i comandi pre-esecuzione (solo formato rivisto), il comando di avvio, la porta di rete e le variabili di ambiente.

La sezione inizia con la `run:` chiave e contiene le seguenti sottochiavi:

- `runtime-version`— Facoltativo. Specificate una versione di runtime che desiderate bloccare per il servizio App Runner.

  Per impostazione predefinita, solo la versione principale è bloccata. App Runner utilizza le versioni secondarie e le patch più recenti disponibili per il runtime in ogni distribuzione o aggiornamento del servizio. Se si specificano versioni principali e secondarie, entrambe vengono bloccate e App Runner aggiorna solo le versioni patch. Se si specificano versioni principali, secondarie e patch, il servizio è bloccato su una versione di runtime specifica e App Runner non la aggiorna mai.

  Sintassi

  ```
  run:
    runtime-version: major[.minor[.patch]]
  ```

  > **Note**
  >
  > I runtime di alcune piattaforme hanno componenti di versione diversi. Per i dettagli, consulta gli argomenti specifici della piattaforma.

  Example

  ```
  runtime: python3
  run:
    runtime-version: 3.7
  ```

- `pre-run`— Facoltativo. Solo utilizzo rivisto della build. Speciifica i comandi che App Runner esegue dopo aver copiato l'applicazione dall'immagine di compilazione all'immagine di esecuzione. È possibile inserire i comandi qui per modificare l'immagine di esecuzione al di fuori della directory. `/app` Ad esempio, se è necessario installare dipendenze globali aggiuntive che risiedono all'esterno della `/app` directory, immettete i comandi richiesti in questa sottosezione a tale scopo.

Per ulteriori informazioni sul processo di compilazione di App Runner, consulta. Versioni di runtime gestite e build di App Runner

> **Note**
>
> - Importante: anche se i `pre-run` comandi sono elencati nella sezione Esegui, possono fare riferimento solo alle variabili di ambiente definite nella sezione Build di questo file di configurazione. Non possono fare riferimento alle variabili di ambiente definite in questa sezione Run.
> - Il `pre-run` parametro è supportato solo dalla build aggiornata di App Runner. Non inserire questo parametro nel file di configurazione se l'applicazione utilizza versioni di runtime supportate dalla build originale di App Runner. Per ulteriori informazioni, consulta Versioni di runtime gestite e build di App Runner.

Sintassi

```
run:
  pre-run:
      - command
      - …
```

- `command`— Obbligatorio. Il comando utilizzato da App Runner per eseguire l'applicazione dopo aver completato la compilazione dell'applicazione.

  Sintassi

  ```
  run:
    command: command
  ```

- `network`— Facoltativo. Specifica la porta che l'applicazione ascolta. Include le voci seguenti:
  - `port`— Facoltativo. Se specificato, questo è il numero di porta che l'applicazione ascolta. Il valore predefinito è `8080`.
  - `env`— Facoltativo. Se specificato, App Runner passa il numero di porta al contenitore in questa variabile di ambiente, oltre a (non invece di) passare lo stesso numero di porta nella variabile di ambiente predefinita,`PORT`. In altre parole, se si specifica`env`, App Runner passa il numero di porta in due variabili di ambiente.

Sintassi

```
run:
  network:
    port: port-number
    env: env-variable-name
```

Example

```
run:
  network:
    port: 8000
    env: MY_APP_PORT
```

- env— Facoltativo. Definizione di variabili di ambiente personalizzate per la fase di esecuzione. Definito come mappature scalari nome-valore. È possibile fare riferimento a queste variabili per nome nell'ambiente di runtime.

> **Note**
>
> In questo file di configurazione sono presenti due env voci distinte in due posizioni diverse. Un set si trova nella sezione Build e l'altro nella sezione Run.
>
> - Il env set nella sezione Build può essere referenziato dai pre-run comandipre-build, buildpost-build, e durante il processo di compilazione.
>
> Importante: si noti che i pre-run comandi si trovano nella sezione Esegui di questo file, anche se possono accedere solo alle variabili di ambiente definite nella sezione Build.
> - Il env set nella sezione Run può essere referenziato dal run comando nell'ambiente di runtime.

Sintassi

```
run:
  env:
    - name: name1
      value: value1
    - name: name2
      value: value2
```

```
  secrets:
    - name: name1
      value-from: arn:aws:secretsmanager:region:aws_account_id:secret:secret-id
    - name: name2
      value-from: arn:aws:ssm:region:aws_account_id:parameter/parameter-name
    - …
```

Example

```
run:
  env:
    - name: MY_VAR_EXAMPLE
      value: "example"
  secrets:
    - name: my-secret
      value-from: "arn:aws:secretsmanager:us-
east-1:123456789012:secret:testingstackAppRunnerConstr-kJFXde2ULKbT-
S7t8xR:username::"
    - name: my-parameter
      value-from: "arn:aws:ssm:us-east-1:123456789012:parameter/parameter-name"
    - name: my-parameter-only-name
      value-from: "parameter-name"
```

L'API App Runner

L'interfaccia di programmazione dell' AWS App Runner applicazione (API) è un' RESTful API per effettuare richieste al servizio App Runner. Puoi utilizzare l'API per creare, elencare, descrivere, aggiornare ed eliminare le risorse App Runner presenti nel tuo. Account AWS

Puoi chiamare l'API direttamente nel codice dell'applicazione oppure puoi utilizzare uno dei AWS SDKs.

Per informazioni complete di riferimento sull'API, consulta l'AWS App Runner API Reference.

Per ulteriori informazioni sugli strumenti per AWS sviluppatori, consulta Tools to Build on AWS.

Argomenti

Utilizzo di AWS CLI per lavorare con App Runner

Per gli script da riga di comando, utilizzare AWS CLIper effettuare chiamate al servizio App Runner. Per informazioni AWS CLI di riferimento complete, consulta l'apprunner nel Command Reference.AWS CLI

AWS CloudShell consente di saltare l'installazione di tale file AWS CLI nel proprio ambiente di sviluppo e di utilizzarlo invece in. AWS Management Console Oltre a evitare l'installazione, non è necessario configurare le credenziali e non è necessario specificare la regione. La tua AWS Management Console sessione fornisce questo contesto a. AWS CLI Per ulteriori informazioni CloudShell e per un esempio di utilizzo, vederethe section called “Usando AWS CloudShell”.

Utilizzo AWS CloudShell per lavorare con AWS App Runner

AWS CloudShell è una shell preautenticata basata su browser che puoi avviare direttamente da. AWS Management ConsoleÈ possibile eseguire AWS CLI comandi contro i AWS servizi (incluso AWS App Runner) utilizzando la shell preferita (Bash o Z shell). PowerShell E puoi farlo senza dover scaricare o installare strumenti da riga di comando.

Si avvia AWS CloudShell da e AWS Management Console le AWS credenziali utilizzate per accedere alla console sono automaticamente disponibili in una nuova sessione di shell. Questa

preautenticazione AWS CloudShell degli utenti consente di saltare la configurazione delle credenziali quando si interagisce con AWS servizi come App Runner utilizzando la AWS CLI versione 2 (preinstallata nell'ambiente di calcolo della shell).

Argomenti

Ottenere le autorizzazioni IAM per AWS CloudShell

Utilizzando le risorse di gestione degli accessi fornite da AWS Identity and Access Management, gli amministratori possono concedere le autorizzazioni agli utenti IAM in modo che possano accedere AWS CloudShell e utilizzare le funzionalità dell'ambiente.

Il modo più rapido per un amministratore di concedere l'accesso agli utenti è tramite una AWS policy gestita. Una policy gestita da AWS è una policy autonoma che viene creata e amministrata da AWS. La seguente policy AWS gestita per CloudShell può essere allegata alle identità IAM:

- `AWSCloudShellFullAccess`: concede l'autorizzazione all'uso AWS CloudShell con accesso completo a tutte le funzionalità.

Se desideri limitare l'ambito di azioni che un utente IAM può eseguire AWS CloudShell, puoi creare una policy personalizzata che utilizzi la policy `AWSCloudShellFullAccess` gestita come modello. Per ulteriori informazioni sulla limitazione delle azioni disponibili per gli utenti in CloudShell, consulta Gestire AWS CloudShell l'accesso e l'utilizzo con le politiche IAM nella Guida per l'AWS CloudShell utente.

> **Note**
>
> La tua identità IAM richiede anche una policy che conceda l'autorizzazione a effettuare chiamate ad App Runner. Per ulteriori informazioni, consulta the section called “App Runner e IAM”.

Interagire con App Runner utilizzando AWS CloudShell

Dopo l'avvio AWS CloudShell da AWS Management Console, puoi iniziare immediatamente a interagire con App Runner utilizzando l'interfaccia a riga di comando.

Nell'esempio seguente, recuperi informazioni su uno dei tuoi servizi App Runner utilizzando in. AWS CLI CloudShell

> **Note**
>
> Quando si utilizza AWS CLI in AWS CloudShell, non è necessario scaricare o installare risorse aggiuntive. Inoltre, poiché hai già eseguito l'autenticazione alla shell, non è necessario configurare le credenziali prima di effettuare chiamate.

Example Recupero delle informazioni sul servizio App Runner utilizzando AWS CloudShell

1. Da AWS Management Console, puoi avviarlo CloudShell scegliendo le seguenti opzioni disponibili nella barra di navigazione:

   - Scegli l' CloudShell icona.
   - Inizia a digitare **`cloudshell`** nella casella di ricerca, quindi scegli l'CloudShellopzione quando la vedi nei risultati della ricerca.

2. Per elencare tutti i servizi App Runner correnti nel tuo AWS account nella AWS regione della sessione della console, inserisci il seguente comando nella riga di CloudShell comando:

   ```
   $ aws apprunner list-services
   ```

   L'output elenca le informazioni di riepilogo per i tuoi servizi.

   ```
   {
     "ServiceSummaryList": [
       {
         "ServiceName": "my-app-1",
         "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
         "ServiceArn": "arn:aws:apprunner:us-east-2:123456789012:service/my-
   app-1/8fe1e10304f84fd2b0df550fe98a71fa",
         "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
         "CreatedAt": "2020-11-20T19:05:25Z",
   ```

```
      "UpdatedAt": "2020-11-23T12:41:37Z",
      "Status": "RUNNING"
    },
    {
      "ServiceName": "my-app-2",
      "ServiceId": "ab8f94cfe29a460fb8760afd2ee87555",
      "ServiceArn": "arn:aws:apprunner:us-east-2:123456789012:service/my-app-2/
ab8f94cfe29a460fb8760afd2ee87555",
      "ServiceUrl": "e2m8rrrx33.us-east-1.awsapprunner.com",
      "CreatedAt": "2020-11-06T23:15:30Z",
      "UpdatedAt": "2020-11-23T13:21:22Z",
      "Status": "RUNNING"
    }
  ]
}
```

3. Per ottenere una descrizione dettagliata di un particolare servizio App Runner, inserisci il seguente comando nella CloudShell riga di comando, utilizzando uno dei ARNs recuperati nel passaggio precedente:

```
$ aws apprunner describe-service --service-arn arn:aws:apprunner:us-
east-2:123456789012:service/my-app-1/8fe1e10304f84fd2b0df550fe98a71fa
```

L'output elenca una descrizione dettagliata del servizio specificato.

```
{
  "Service": {
    "ServiceName": "my-app-1",
    "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceArn": "arn:aws:apprunner:us-east-2:123456789012:service/my-
app-1/8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
    "CreatedAt": "2020-11-20T19:05:25Z",
    "UpdatedAt": "2020-11-23T12:41:37Z",
    "Status": "RUNNING",
    "SourceConfiguration": {
      "CodeRepository": {
        "RepositoryUrl": "https://github.com/my-account/python-hello",
        "SourceCodeVersion": {
          "Type": "BRANCH",
          "Value": "main"
        },
```

```
        "CodeConfiguration": {
          "CodeConfigurationValues": {
            "BuildCommand": "[pip install -r requirements.txt]",
            "Port": "8080",
            "Runtime": "PYTHON_3",
            "RuntimeEnvironmentVariables": [
              {
                "NAME": "Jane"
              }
            ],
            "StartCommand": "python server.py"
          },
          "ConfigurationSource": "API"
        }
      },
      "AutoDeploymentsEnabled": true,
      "AuthenticationConfiguration": {
        "ConnectionArn": "arn:aws:apprunner:us-east-2:123456789012:connection/my-
github-connection/e7656250f67242d7819feade6800f59e"
      }
    },
    "InstanceConfiguration": {
      "CPU": "1 vCPU",
      "Memory": "3 GB"
    },
    "HealthCheckConfiguration": {
      "Protocol": "TCP",
      "Path": "/",
      "Interval": 10,
      "Timeout": 5,
      "HealthyThreshold": 1,
      "UnhealthyThreshold": 5
    },
    "AutoScalingConfigurationSummary": {
      "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-2:123456789012:autoscalingconfiguration/
DefaultConfiguration/1/00000000000000000000000000000001",
      "AutoScalingConfigurationName": "DefaultConfiguration",
      "AutoScalingConfigurationRevision": 1
    }
  }
}
```

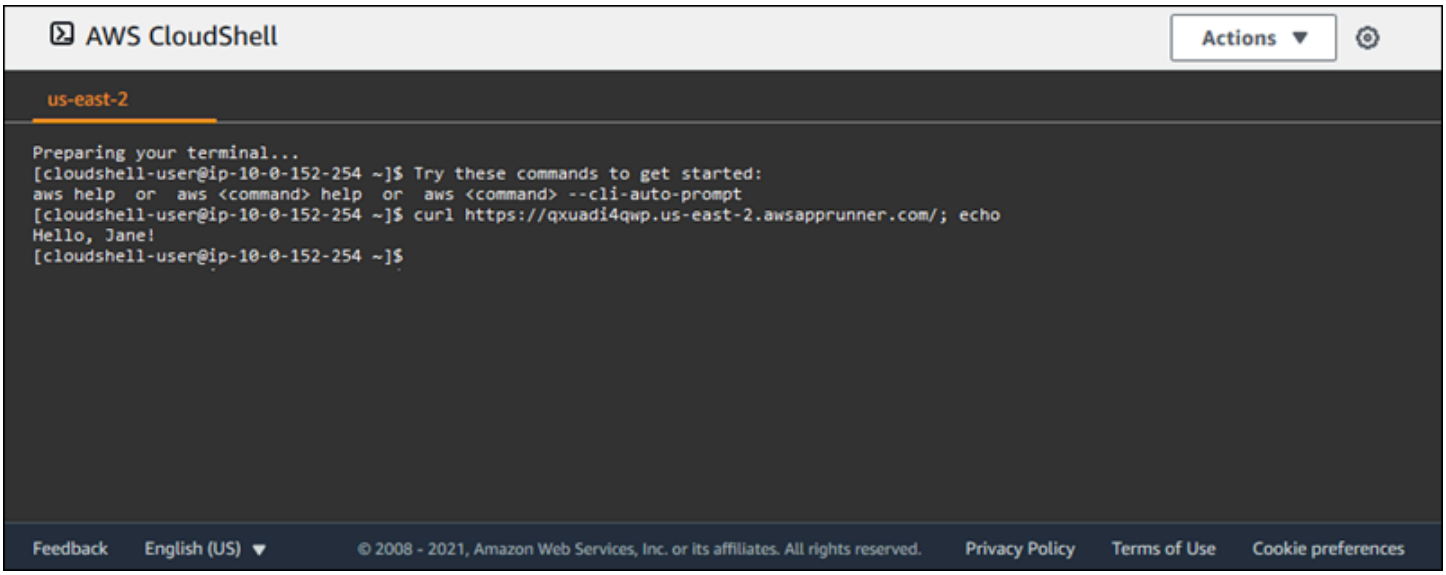
Verifica del servizio App Runner utilizzando AWS CloudShell

Quando crei un servizio App Runner, App Runner crea un dominio predefinito per il sito Web del servizio e lo mostra nella console (o lo restituisce nel risultato della chiamata API). Puoi utilizzarlo CloudShell per effettuare chiamate al tuo sito Web e verificare che funzioni correttamente.

Ad esempio, dopo aver creato un servizio App Runner come descritto inNozioni di base, esegui il seguente comando in CloudShell:

```
$ curl https://qxuadi4qwp.us-east-2.awsapprunner.com/; echo
```

L'output dovrebbe mostrare il contenuto della pagina previsto.



Risoluzione dei problemi

Questo capitolo fornisce le procedure per la risoluzione di errori e problemi comuni che potrebbero verificarsi durante l'utilizzo del AWS App Runner servizio. I messaggi di errore possono essere visualizzati nella console, nell'API o nella scheda Registri della pagina del servizio.

Per ulteriori suggerimenti sulla risoluzione dei problemi e per risposte a domande comuni relative al supporto, visitare il Knowledge Center di .

Argomenti

Quando il servizio non riesce a creare

Se il tentativo di creare un servizio App Runner fallisce, il servizio entra in uno `CREATE_FAILED` stato. Questo stato appare come Creazione non riuscita sulla console. La creazione di un servizio potrebbe non riuscire a causa di problemi correlati a uno o più dei seguenti fattori:

- Il codice dell'applicazione
- Il processo di compilazione
- Configurazione
- Quote delle risorse
- Problemi temporanei con Servizi AWS il sottostante utilizzato dal servizio

Per risolvere i problemi relativi alla mancata creazione di un servizio, si consiglia di effettuare le seguenti operazioni.

1. Leggi gli eventi e i registri del servizio per scoprire cosa ha causato la mancata creazione del servizio.
2. Apporta le modifiche necessarie al codice o alla configurazione.

3. Se hai raggiunto la quota di servizio, elimina uno o più servizi.
4. Se hai raggiunto un'altra quota di risorse, potresti essere in grado di aumentarla se è regolabile.
5. Prova a ricostruire nuovamente il servizio dopo aver completato tutti i passaggi precedenti. Per informazioni su come ricostruire il servizio, consulta. the section called “Ricostruisci un servizio fallito”

> **Note**
>
> Una delle quote di risorse regolabili che potrebbe causare un problema è la risorsa vCPU Fargate On-Demand.
> Il conteggio delle risorse vCPU determina il numero di istanze che App Runner può fornire al servizio. Si tratta di un valore di quota regolabile per il numero di risorse vCPU Fargate On-Demand che risiede nel servizio. AWS Fargate Per visualizzare le impostazioni della quota vCPU per il tuo account o per richiedere un aumento della quota, utilizza la console Service Quotas in. AWS Management Console Per ulteriori informazioni, consulta le quote AWS Fargate di servizio nella Amazon Elastic Container Service Developer Guide.

> **Important**
>
> Oltre al tentativo di creazione iniziale di un servizio non riuscito, non dovrai sostenere alcun costo aggiuntivo. Anche se il servizio guasto non è utilizzabile, viene comunque conteggiato ai fini della quota di servizio. App Runner non elimina automaticamente il servizio guasto, quindi assicurati di eliminarlo quando hai finito di analizzare l'errore.

Nomi di dominio personalizzati

Questa sezione illustra come risolvere i vari errori che potrebbero verificarsi durante il collegamento a un dominio personalizzato.

> **Note**
>
> Per aumentare la sicurezza delle applicazioni App Runner, il dominio*.awsapprunner.com è registrato nella Public Suffix List (PSL). Per una maggiore sicurezza, ti consigliamo di utilizzare i cookie con un `__Host-` prefisso se hai bisogno di impostare cookie sensibili nel nome di dominio predefinito per le tue applicazioni App Runner. Questa pratica ti aiuterà

a difendere il tuo dominio dai tentativi CSRF (cross-site request forgery). Per ulteriori informazioni, consulta la pagina Impostazione cookie nella pagina Mozilla Developer Network.

Errore Getting Create Fail per il dominio personalizzato

- Controlla se questo errore è dovuto a un problema con i record CAA. Se non ci sono record CAA in nessuna parte dell'albero DNS, ricevi un messaggio `fail open` ed AWS Certificate Manager emette un certificato per verificare il dominio personalizzato. Ciò consente a App Runner di accettare il dominio personalizzato. Se utilizzi le certificazioni CAA nei record DNS, assicurati che almeno i record CAA del dominio includano. `amazon.com` In caso contrario, ACM non riesce a emettere un certificato. Di conseguenza, il dominio personalizzato per App Runner non viene creato.

  L'esempio seguente utilizza lo strumento di ricerca DNS DiG per mostrare ai record CAA mancanti una voce obbligatoria. L'esempio utilizza `example.com` come dominio personalizzato. Esegui i seguenti comandi nell'esempio per controllare i record CAA.

```
...
;; QUESTION SECTION:
;example.com.            IN  CAA

;; ANSWER SECTION:
example.com.        7200    IN  CAA 0 iodef "mailto:hostmaster@example.com"
example.com.        7200    IN  CAA 0 issue "letsencrypt.org"
...note absence of "amazon.com" in any of the above CAA records...
```

- Correggete i record del dominio e assicuratevi che almeno un record CAA lo includa. `amazon.com`
- Riprova a collegare il dominio personalizzato con App Runner.

Per istruzioni su come risolvere gli errori CAA, consulta quanto segue:

- Problemi relativi all'autorizzazione dell'Autorità di certificazione (CAA)
- Come posso risolvere gli errori CAA per l'emissione o il rinnovo di un certificato ACM?

Errore in attesa di convalida del certificato DNS per un dominio personalizzato

- Verifica se hai saltato un passaggio importante nella configurazione del dominio personalizzato. Inoltre, controlla se hai configurato in modo errato un record DNS utilizzando uno strumento di ricerca DNS come DiG. In particolare, verifica la presenza dei seguenti errori:
  - Eventuali passaggi mancati.
  - Caratteri non supportati, ad esempio virgolette doppie nei record DNS.
- Correggi gli errori.
- Riprova a collegare il dominio personalizzato con App Runner.

Per istruzioni su come risolvere gli errori di convalida CAA, consulta quanto segue.

- Convalida DNS
- the section called “Nomi di dominio personalizzati”

Comandi base di risoluzione dei problemi

- Conferma che è possibile trovare un servizio.

```
aws apprunner list-services
```

- Descrivi un servizio e verificane lo stato.

```
aws apprunner describe-service --service-arn
```

- Controlla lo stato del dominio personalizzato.

```
aws apprunner describe-custom-domains --service-arn
```

- Elenca tutte le operazioni in corso.

```
aws apprunner list-operations --service-arn
```

Rinnovo del certificato di dominio personalizzato

Quando aggiungi un dominio personalizzato al tuo servizio, App Runner ti fornisce un set di record CNAME da aggiungere al tuo server DNS. Questi record CNAME includono i record dei certificati. App Runner utilizza AWS Certificate Manager (ACM) per verificare il dominio. App Runner convalida questi record DNS per garantire la proprietà continua di questo dominio. Se rimuovi i record CNAME dalla tua zona DNS, App Runner non può più convalidare i record DNS e il certificato di dominio personalizzato non si rinnova automaticamente.

Questa sezione illustra come risolvere i seguenti problemi di rinnovo dei certificati di dominio personalizzati:

- the section called “Il CNAME viene rimosso dal server DNS”.
- the section called “Il certificato è scaduto”.

Il CNAME viene rimosso dal server DNS

- Recupera i tuoi record CNAME utilizzando l'DescribeCustomDomainsAPI o dalle impostazioni del dominio personalizzato nella console App Runner. Per informazioni sugli archivi CNAMEs, consulta. CertificateValidationRecords
- Aggiungi i record CNAME di convalida del certificato al tuo server DNS. App Runner può quindi confermare che sei il proprietario del dominio. Dopo aver aggiunto i record CNAME, la propagazione dei record DNS può richiedere fino a 30 minuti. Inoltre, App Runner e ACM potrebbero impiegare diverse ore per riprovare il processo di rinnovo del certificato. Per istruzioni su come aggiungere record CNAME, consulta. the section called “Gestisci domini personalizzati”

Il certificato è scaduto

- Dissocia (scollega) e quindi associa (collega) il dominio personalizzato per il servizio App Runner utilizzando la console o l'API App Runner. App Runner crea un nuovo record CNAME di convalida del certificato.

- Aggiungi i nuovi record CNAME di convalida del certificato al tuo server DNS.

  Per istruzioni su come dissociare (scollegare) e associare (collegare) il dominio personalizzato, consulta. the section called “Gestisci domini personalizzati”

Come posso verificare che il certificato sia stato rinnovato correttamente

Puoi controllare lo stato dei record del certificato per verificare che il certificato sia stato rinnovato con successo. Puoi controllare lo stato dei certificati utilizzando strumenti come curl.

Per ulteriori informazioni sul rinnovo dei certificati, consulta i seguenti link:

- Perché il mio certificato ACM è contrassegnato come non idoneo al rinnovo?
- Rinnovo gestito per i certificati ACM
- Convalida DNS

Errore di routing della richiesta HTTP/HTTPS

Questa sezione illustra come risolvere i problemi e gli errori che potrebbero verificarsi durante il routing del traffico HTTP/HTTPS verso gli endpoint del servizio App Runner.

404 Errore non trovato durante l'invio del traffico HTTP/HTTPS agli endpoint del servizio App Runner

- Verifica che nella `Host Header` richiesta HTTP punti all'URL del servizio poiché App Runner utilizza le informazioni dell'intestazione dell'host per instradare le richieste. La maggior parte dei client`cURL`, like e browser Web indirizzano automaticamente l'intestazione dell'host all'URL del servizio. Se il tuo client non imposta l'URL del servizio come`Host Header`, ricevi un `404 Not Found` errore.

  Example Intestazione host errata

```
$ ~ curl -I -H "host: foobar.com" https://testservice.awsapprunner.com/
HTTP/1.1 404 Not Found
transfer-encoding: chunked
```

Example Intestazione host corretta

```
$ ~ curl -I -H "host: testservice.awsapprunner.com" https://
testservice.awsapprunner.com/
HTTP/1.1 200 OK
content-length: 11772
content-type: text/html; charset=utf-8
```

- Verifica che il client stia impostando correttamente l'indicatore del nome del server (SNI) per il routing delle richieste verso servizi pubblici o privati. Per la terminazione TLS e il routing delle richieste, App Runner utilizza lo SNI impostato nella connessione HTTPS.

Quando il servizio non riesce a connettersi ad Amazon RDS o al servizio downstream

Potrebbe esserci un problema di configurazione di rete con il tuo servizio se non riesce a connettersi a un database Amazon RDS o a un'altra applicazione o servizio downstream. Questo argomento illustra alcuni passaggi per determinare se ci sono problemi con la configurazione di rete e le opzioni per correggerli. Per ulteriori informazioni sulla configurazione del traffico in uscita per App Runner, consulta. Abilitazione dell'accesso VPC per il traffico in uscita

> **Note**
>
> Per visualizzare la configurazione del connettore VPC, dal riquadro di navigazione sinistro della console App Runner, seleziona Configurazione di rete. Quindi seleziona la scheda Traffico in uscita. Seleziona un connettore VPC. La pagina successiva mostra i dettagli sul connettore VPC. Da questa pagina è possibile visualizzare e approfondire quanto segue: sottoreti, gruppi di sicurezza e servizi App Runner che utilizzano il VPC.

Per restringere la causa dell'impossibilità dell'applicazione di connettersi a un altro servizio downstream

1. Assicurati che le sottoreti utilizzate nei connettori VPC siano sottoreti private. Se un connettore è configurato con una sottorete pubblica, il servizio riscontrerà degli errori, poiché l'Hyperplane ENIs (interfacce di rete elastiche) sottostante per ogni sottorete non dispone di uno spazio IP pubblico.

Se i tuoi connettori VPC utilizzano sottoreti pubbliche, hai le seguenti opzioni per correggere questa configurazione:

a. Crea una nuova sottorete privata e usala al posto della sottorete pubblica per il connettore VPC. Per ulteriori informazioni, consulta Subnet for your VPC nella Amazon VPC User Guide.

b. Indirizza la sottorete pubblica esistente tramite gateway NAT. Per ulteriori informazioni, consulta i gateway NAT nella Amazon VPC User Guide.

2. Verifica che le regole di ingresso e uscita del gruppo di sicurezza per il connettore VPC siano corrette. Dal riquadro di navigazione sinistro della console App Runner, seleziona Configurazione di rete > Traffico in uscita. Seleziona il connettore VPC dall'elenco. La pagina successiva elenca i gruppi di sicurezza che puoi selezionare per ispezionare.

3. Verifica che le regole in entrata e in uscita del gruppo di sicurezza siano corrette per l'istanza RDS o altro servizio downstream a cui stai tentando di connetterti. Per ulteriori informazioni, consulta la guida al servizio downstream a cui l'applicazione App Runner sta tentando di connettersi.

4. Per verificare che non vi siano altri tipi di problemi di configurazione della rete al di fuori delle configurazioni di App Runner, provate a connettervi all'RDS o al servizio downstream al di fuori di App Runner:

a. Da un' EC2 istanza Amazon nello stesso VPC, prova a connetterti all'istanza o al servizio RDS.

b. Se stai tentando di connetterti a un endpoint VPC di servizio, verifica la connettività accedendo allo stesso endpoint da un' EC2 istanza nello stesso VPC.

5. Se uno dei test di connessione nello Step 4 fallisce, molto probabilmente c'è un problema al di fuori delle configurazioni di App Runner con un'altra risorsa del tuo account. AWS Contatta AWS Support per ricevere assistenza per isolare e risolvere ulteriormente il problema con le altre configurazioni di rete.

6. Se ti connetti correttamente all'istanza RDS o al servizio downstream seguendo le istruzioni del passaggio 4, procedi con le istruzioni riportate in questo passaggio. Verificheremo se il traffico sta entrando nell'ENI abilitando e ispezionando i log di flusso ENI di Hyperplane.

> **Note**
>
> Per poter completare questi passaggi e ottenere le informazioni richieste sui log di flusso ENI, il tentativo di connessione al servizio RDS o al servizio downstream deve avvenire dopo che il servizio App Runner è stato avviato correttamente. L'applicazione deve eseguire l'operazione di connessione al servizio RDS o downstream quando è in esecuzione. Altrimenti, ENIs potrebbe essere ripulito come parte dei flussi di lavoro di rollback di App Runner. Questo approccio garantisce che ENIs rimangano disponibili per ulteriori indagini.

a. Dalla AWS console, avvia la EC2 console.

b. Nel riquadro di navigazione a sinistra, nel gruppo Rete e sicurezza, seleziona Interfacce di rete.

c. Scorri fino alle colonne Tipo di interfaccia e Descrizione per individuarle ENIs nelle sottoreti associate al connettore VPC. Avranno i seguenti schemi di denominazione.

   - Tipo di interfaccia: fargate
   - Descrizione: inizia con AWSAppRunner ENI(esempio: AWSAppRunner ENI - abcde123-abcd-1234-1234-abcde1233456)

d. Utilizza le caselle di controllo all'inizio delle righe per selezionare ENIs quelle pertinenti.

e. Dal menu Azioni, seleziona Crea log di flusso.

f. Inserisci le informazioni nei prompt e seleziona Create flow flog nella parte inferiore della pagina.

g. Ispeziona il log di flusso generato.

   - Se il traffico entrava nell'ENI durante il test della connessione, il problema non è correlato alla configurazione ENI. Potrebbero esserci problemi di configurazione della rete con un'altra risorsa del tuo AWS account oltre ai servizi App Runner. Contatta l' AWS assistenza per ulteriore assistenza.
   - Se il traffico non entrava nell'ENI durante il test della connessione, ti consigliamo di contattare l' AWS assistenza per verificare se ci sono problemi noti con il servizio Fargate.

h. Usa lo strumento Reachability Analyzer di rete. Questo strumento aiuta a determinare le configurazioni errate della rete identificando i componenti di blocco quando una fonte

nel percorso di rete virtuale non è raggiungibile. Per ulteriori informazioni, consulta Cos'è Reachability Analyzer? nella Amazon VPC Reachability Analyzer Guide.

Inserisci App Runner ENI come origine e RDS ENI come destinazione.

7. Se non riesci a restringere ulteriormente il problema o se non riesci ancora a connetterti all'RDS o al servizio downstream dopo aver completato i passaggi precedenti, ti consigliamo di contattare il AWS Supporto per ulteriore assistenza.

Quando gli indirizzi IP non sono sufficienti per avviare istanze o scalare

> **Note**
>
> Per i servizi pubblici, App Runner non crea un'interfaccia di rete elastica (ENI) nell'utente VPCs, quindi i servizi pubblici non sono interessati da questa modifica.

Questa guida ti aiuta a risolvere gli errori di esaurimento dell'IP che potresti riscontrare sui servizi App Runner con l'accesso VPC per il traffico in uscita abilitato.

App Runner avvierà le istanze nelle sottoreti associate al connettore VPC. App Runner crea 1 ENI per istanza nella sottorete in cui viene lanciata l'istanza. Ogni ENI utilizza un IP privato in quella sottorete. Le sottoreti hanno un numero fisso di sottoreti IPs disponibili, a seconda del blocco CIDR associato a quella sottorete. Se App Runner non riesce a trovare sottoreti con una quantità sufficiente IPs per creare un ENI, non riuscirà ad avviare nuove istanze per il servizio App Runner. Ciò potrebbe causare problemi con il potenziamento dei servizi. In questi casi, verranno visualizzati i registri degli eventi di App Runner che indicano che App Runner non è in grado di trovare le sottoreti disponibili. IPs Puoi aggiornare i tuoi servizi seguendo le istruzioni riportate di seguito per risolvere tali errori.

Come aggiornare i tuoi servizi per averne altri disponibili IPs

Il numero di indirizzi IP disponibili in una sottorete si basa sul blocco CIDR associato a quella sottorete. I blocchi CIDR associati a una sottorete non possono essere aggiornati dopo la creazione. Inoltre, i connettori VPC di App Runner non possono essere aggiornati una volta creati. Per fornire di più IPs ai tuoi servizi App Runner con l'accesso VPC per il traffico in uscita abilitato:

1. Crea nuove sottoreti con un blocco CIDR più grande.

2. Crea un nuovo connettore VPC con le nuove sottoreti.
3. Aggiorna il servizio App Runner per utilizzare il nuovo connettore VPC.

Calcolo IPs necessario per i tuoi servizi

Prima di provare a creare nuove sottoreti con blocchi CIDR più grandi, stabilisci il numero di sottoreti IPs che ti serviranno tra i tuoi servizi App Runner. Ti consigliamo di calcolare il numero di elementi IPs necessari nel connettore come segue:

1. Per ogni servizio con accesso VPC per il traffico in uscita abilitato, annota la dimensione massima (numero massimo di istanze) nella configurazione della scalabilità automatica.
2. Somma i valori di tutti i servizi.
3. Raddoppia questa somma per tenere conto delle nuove istanze lanciate durante le implementazioni blu-green.

Esempio

Consideriamo due servizi A e B che utilizzano lo stesso connettore VPC.

1. Il servizio A ha la dimensione massima configurata come 25.
2. La dimensione massima del servizio B è configurata come 15.

Richiesto IPs = 2 × (25 + 15) = 80

Assicurati che le tue sottoreti abbiano almeno 80 disponibili IPs combinate.

Crea nuove sottoreti

1. Determina la dimensione del blocco CIDR necessaria per IPv4 utilizzare questa formula (nota che 5 IPs sono riservati da AWS: Subnet Sizing)

```
Number of available IP addresses = 2^(32 - prefix length) - 5
```

```
Example :
For 192.168.1.0/24:
Prefix length is 24
```

```
Number of available IP addresses = 2^(32 - 24) - 5 = 2^8-5 = 251 IP addresses

For 10.0.0.0/16:
Prefix length is 16
Number of available IP addresses = 2^(32 - 16) - 5 = 2^16-5 = 65,531 IP addresses

Quick reference:
/24 = 251 IP addresses
/16 = 65,531 IP addresses
```

2. Crea una nuova sottorete utilizzando la CLI di AWS EC2 .

```
aws ec2 create-subnet --vpc-id <my-vpc-id> --cidr-block <cidr-block>
```

Esempio (crea una sottorete con 4.096): IPs

```
aws ec2 create-subnet --vpc-id my-vpc-id --cidr-block 10.0.0.0/20
```

3. Crea un nuovo connettore VPC. Vedi: Gestione dell'accesso al VPC
4. Aggiorna i tuoi servizi con il traffico in uscita verso VPC abilitato per utilizzare questo nuovo connettore VPC. App Runner inizierà a utilizzare le nuove sottoreti una volta aggiornato il servizio.

> **Note**
>
> VPCs sono inoltre limitati al numero di elementi disponibili IPs che possono essere assegnati alle sottoreti tramite blocchi CIDR. Se non riesci a creare sottoreti con blocchi CIDR più grandi, potresti dover aggiornare il tuo VPC con blocchi CIDR secondari prima di creare le nuove sottoreti.

Collegamento di blocchi CIDR secondari al tuo VPC

Associa il blocco CIDR secondario a questo VPC.

```
aws ec2 associate-vpc-cidr-block --vpc-id <my-vpc-id> --cidr-block <cidr-block>
```

Esempio:

```
aws ec2 associate-vpc-cidr-block --vpc-id my-vpc-id --cidr-block 10.1.0.0/16
```

Verifica

Dopo aver aggiornato il servizio. È possibile utilizzare quanto segue per eseguire la verifica della correzione

1. Monitora i registri degli eventi: monitora i registri degli eventi del servizio App Runner per verificare che non vengano visualizzati nuovi errori di indisponibilità IP o ENI
2. Controlla Service Scaling:
    1. Amplia completamente il servizio modificando il numero minimo di istanze nella configurazione di scalabilità automatica
    2. Verifica che tutte le nuove istanze vengano avviate senza errori relativi all'IP
    3. Monitora diversi eventi di scalabilità per garantire prestazioni costanti
3. Banner per la console: se utilizzi la Console di gestione AWS, verifica che App Runner non mostri più un banner di avviso insufficiente IPs.
4. Utilizzo di VPC e sottorete IP:
    1. Utilizza il dashboard VPC o i comandi CLI per controllare l'utilizzo degli indirizzi IP nelle nuove sottoreti.
    2. Verifica che ci sia ancora un buon margine di disponibilità dopo l'ampliamento del servizio IPs

Insidie comuni

Quando risolvi l'esaurimento dell'IP nei servizi App Runner, tieni presente questi potenziali problemi:

1. Pianificazione inadeguata degli indirizzi IP: sottovalutare le esigenze IP future può portare a problemi ricorrenti di esaurimento. Esegui una pianificazione approfondita della capacità, considerando la potenziale crescita del servizio e gli scenari di picco di utilizzo.
2. Informazioni sull'utilizzo dell'IP a livello di VPC: ricorda che anche altri servizi AWS all'interno dello stesso VPC utilizzano indirizzi IP. Considera i requisiti IP di tutti i servizi quando pianifichi le configurazioni del VPC e della sottorete.

3. Trascurare di aggiornare i servizi: dopo aver creato nuove sottoreti o connettori VPC, assicurati di aggiornare i servizi App Runner per utilizzare le nuove configurazioni. In caso contrario, si continuerà a utilizzare l'intervallo IP esaurito.
4. Incomprensione delle sovrapposizioni dei blocchi CIDR: quando aggiungi blocchi CIDR secondari a un VPC, assicurati che non si sovrappongano ai blocchi esistenti. La sovrapposizione di blocchi CIDR può causare conflitti di routing e ambiguità degli indirizzi IP.
5. Superamento dei limiti VPC: tieni presente che un VPC può avere un massimo di 5 blocchi CIDR (1 primario e 4 secondari). Pianifica l'espansione dello spazio degli indirizzi IP entro questi vincoli.
6. Ignorare la distribuzione della sottorete AZ: quando create nuove sottoreti, assicuratevi che siano distribuite su più zone di disponibilità per un'elevata disponibilità e tolleranza agli errori.
7. Trascurando i limiti ENI: ricorda che ci sono dei limiti al numero di istanze ENIs che possono essere allegate alle istanze. Verifica che i limiti del tuo account AWS siano in linea con l'utilizzo pianificato dell'interfaccia di rete.

Conoscendo queste insidie, puoi gestire in modo più efficace le tue risorse VPC ed evitare problemi di esaurimento dell'IP nei tuoi servizi App Runner.

Risorse aggiuntive

1. Documentazione AWS VPC
2. Comprendere i blocchi CIDR
3. Connettori VPC App Runner

Glossario

1. ENI: Elastic Network Interface, un'interfaccia di rete virtuale in AWS.
2. CIDR: Classless Inter-Domain Routing, un metodo per l'allocazione degli indirizzi IP.
3. Connettore VPC: una risorsa che consente ad App Runner di connettersi al tuo VPC.

Sicurezza in App Runner

La sicurezza del cloud AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di data center e architetture di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra te e te. AWS Il modello di responsabilità condivisa descrive questo aspetto come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi in Cloud AWS. AWS fornisce inoltre servizi che è possibile utilizzare in modo sicuro. I revisori esterni testano e verificano regolarmente l'efficacia della nostra sicurezza nell'ambito dei AWS Programmi di AWS conformità dei Programmi di conformità dei di . Per ulteriori informazioni sui programmi di conformità applicabili AWS App Runner, consulta AWS Servizi nell'ambito del programma di conformitàAWS .
- Sicurezza nel cloud: la tua responsabilità è determinata dal AWS servizio che utilizzi. Sei anche responsabile di altri fattori, tra cui la riservatezza dei dati, i requisiti della tua azienda e le leggi e normative vigenti.

Questa documentazione ti aiuta a capire come applicare il modello di responsabilità condivisa quando usi App Runner. I seguenti argomenti mostrano come configurare App Runner per soddisfare gli obiettivi di sicurezza e conformità. Scopri anche come utilizzare altri AWS servizi che ti aiutano a monitorare e proteggere le tue risorse di App Runner.

Argomenti

Protezione dei dati in App Runner

Il modello di responsabilità AWS condivisa modello di di si applica alla protezione dei dati in AWS App Runner. 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 or 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 lavori con App Runner 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.

Per altri argomenti sulla sicurezza di App Runner, consulta. Sicurezza

Argomenti

Protezione dei dati tramite crittografia

AWS App Runner legge il codice sorgente dell'applicazione (immagine sorgente o codice sorgente) da un repository specificato dall'utente e lo archivia per la distribuzione al servizio. Per ulteriori informazioni, consulta Architettura e concetti.

La protezione dei dati si riferisce alla protezione dei dati in transito (mentre viaggiano da e verso App Runner) e a riposo (mentre sono archiviati nei AWS data center).

Per ulteriori informazioni sulla protezione dei dati, vedere. the section called “Protezione dei dati”

Per altri argomenti sulla sicurezza di App Runner, consultaSicurezza.

Crittografia in transito

È possibile ottenere la protezione dei dati in transito in due modi: crittografare la connessione utilizzando Transport Layer Security (TLS) o utilizzare la crittografia lato client (in cui l'oggetto viene crittografato prima di essere inviato). Entrambi i metodi sono validi per proteggere i dati dell'applicazione. Per proteggere la connessione, crittografala utilizzando TLS ogni volta che l'applicazione, i suoi sviluppatori e amministratori e i suoi utenti finali inviano o ricevono oggetti. App Runner configura l'applicazione per ricevere traffico su TLS.

La crittografia lato client non è un metodo valido per proteggere l'immagine o il codice sorgente forniti ad App Runner per la distribuzione. App Runner deve accedere alla fonte dell'applicazione, quindi non può essere crittografata. Pertanto, assicurati di proteggere la connessione tra il tuo ambiente di sviluppo o distribuzione e App Runner.

Crittografia inattiva e gestione delle chiavi

Per proteggere i dati inattivi dell'applicazione, App Runner crittografa tutte le copie archiviate dell'immagine sorgente o del pacchetto sorgente dell'applicazione. Quando crei un servizio App Runner, puoi fornire un. AWS KMS key Se ne fornisci uno, App Runner utilizza la chiave fornita per crittografare la tua fonte. Se non ne fornisci una, App Runner utilizza invece una. Chiave gestita da AWS

Per informazioni dettagliate sui parametri di creazione del servizio App Runner, consulta. CreateService Per informazioni su AWS Key Management Service (AWS KMS), consulta la Guida per gli AWS Key Management Service sviluppatori.

Riservatezza del traffico Internet

App Runner utilizza Amazon Virtual Private Cloud (Amazon VPC) per creare confini tra le risorse dell'applicazione App Runner e controllare il traffico tra esse, la rete locale e Internet. Per ulteriori informazioni sulla sicurezza di Amazon VPC, consulta la privacy del traffico Internet in Amazon VPC nella Amazon VPC User Guide.

Per informazioni sull'associazione della tua applicazione App Runner a un Amazon VPC personalizzato, consulta. the section called “Traffico in uscita”

Per informazioni sulla protezione delle richieste ad App Runner utilizzando un endpoint VPC, consulta. the section called “Endpoint VPC”

Per ulteriori informazioni sulla protezione dei dati, consulta. the section called “Protezione dei dati”

Per altri argomenti sulla sicurezza di App Runner, consultaSicurezza.

Gestione delle identità e degli accessi per App Runner

AWS Identity and Access Management (IAM) è uno strumento Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle risorse. AWS Gli amministratori IAM controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (disporre delle autorizzazioni) a utilizzare le risorse di App Runner. IAM è uno strumento Servizio AWS che puoi utilizzare senza costi aggiuntivi.

Per altri argomenti sulla sicurezza di App Runner, consultaSicurezza.

Argomenti

- Destinatari
- Autenticazione con identità
- Gestione dell'accesso con policy
- Come funziona App Runner con IAM
- Esempi di policy basate sull'identità di App Runner

- Utilizzo di ruoli collegati ai servizi per App Runner
- AWS politiche gestite per AWS App Runner
- Risoluzione dei problemi relativi all'identità e all'accesso ad App Runner

Destinatari

Il modo in cui utilizzi AWS Identity and Access Management (IAM) varia a seconda del lavoro svolto in App Runner.

Utente del servizio: se utilizzi il servizio App Runner per svolgere il tuo lavoro, l'amministratore ti fornisce le credenziali e le autorizzazioni necessarie. Man mano che utilizzi più funzionalità di App Runner per svolgere il tuo lavoro, potresti aver bisogno di autorizzazioni aggiuntive. La comprensione della gestione dell'accesso ti consente di richiedere le autorizzazioni corrette all'amministratore. Se non riesci ad accedere a una funzionalità di App Runner, consulta. Risoluzione dei problemi relativi all'identità e all'accesso ad App Runner

Amministratore del servizio: se sei responsabile delle risorse di App Runner della tua azienda, probabilmente hai pieno accesso ad App Runner. È tuo compito determinare a quali funzionalità e risorse di App Runner devono accedere gli utenti del servizio. Devi inviare le richieste all'amministratore IAM per cambiare le autorizzazioni degli utenti del servizio. Esamina 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 App Runner, consulta. Come funziona App Runner con IAM

Amministratore IAM: se sei un amministratore IAM, potresti voler conoscere i dettagli su come scrivere policy per gestire l'accesso ad App Runner. Per visualizzare esempi di policy basate sull'identità di App Runner che puoi utilizzare in IAM, consulta. Esempi di policy basate sull'identità di App Runner

Autenticazione con identità

L'autenticazione è il modo in cui accedi AWS utilizzando le tue credenziali di identità. Devi essere autenticato (aver effettuato l' Utente root dell'account AWS accesso AWS) come utente IAM o assumendo un ruolo IAM.

Puoi accedere AWS come identità federata utilizzando le credenziali fornite tramite una fonte di identità. AWS IAM Identity Center Gli utenti (IAM Identity Center), l'autenticazione Single Sign-On della tua azienda e le tue credenziali di Google o Facebook sono esempi di identità federate. Se

accedi come identità federata, l'amministratore ha configurato in precedenza la federazione delle identità utilizzando i ruoli IAM. Quando accedi AWS utilizzando la federazione, assumi indirettamente un ruolo.

A seconda del tipo di utente, puoi accedere al AWS Management Console o al portale di AWS accesso. Per ulteriori informazioni sull'accesso a AWS, vedi Come accedere al tuo Account AWS nella Guida per l'Accedi ad AWS utente.

Se accedi a AWS livello di codice, AWS fornisce un kit di sviluppo software (SDK) e un'interfaccia a riga di comando (CLI) per firmare crittograficamente le tue richieste utilizzando le tue credenziali. Se non utilizzi AWS strumenti, devi firmare tu stesso le richieste. Per ulteriori informazioni sul metodo consigliato per la firma delle richieste, consulta Signature Version 4 AWS per le richieste API nella Guida per l'utente IAM.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. Ad esempio, ti AWS consiglia di utilizzare l'autenticazione a più fattori (MFA) per aumentare la sicurezza del tuo account. Per ulteriori informazioni, consulta Autenticazione a più fattori nella Guida per l'utente di AWS IAM Identity Center e Utilizzo dell'autenticazione a più fattori (MFA)AWS in IAM nella Guida per l'utente IAM.

Account AWS utente root

Quando si crea un account Account AWS, si inizia con un'identità di accesso che ha accesso completo a tutte Servizi AWS le risorse dell'account. Questa identità è denominata utente Account AWS root ed è accessibile effettuando l'accesso con l'indirizzo e-mail e la password utilizzati per creare l'account. Si consiglia vivamente di non utilizzare l'utente root per le attività quotidiane. Conserva le credenziali dell'utente root e utilizzale per eseguire le operazioni che solo l'utente root può eseguire. Per un elenco completo delle attività che richiedono l'accesso come utente root, consulta la sezione Attività che richiedono le credenziali dell'utente root nella Guida per l'utente IAM.

Utenti e gruppi IAM

Un utente IAM è un'identità interna Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Ove possibile, consigliamo di fare affidamento a credenziali temporanee invece di creare utenti IAM con credenziali a lungo termine come le password e le chiavi di accesso. Tuttavia, se si hanno casi d'uso specifici che richiedono credenziali a lungo termine con utenti IAM, si consiglia di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta la pagina Rotazione periodica delle chiavi di accesso per casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente IAM.

Un gruppo IAM è un'identità che specifica un insieme di utenti IAM. Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio, potresti avere un gruppo denominato IAMAdminse concedere a quel gruppo le autorizzazioni per amministrare le risorse IAM.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali temporanee. Per ulteriori informazioni, consulta Casi d'uso per utenti IAM nella Guida per l'utente IAM.

Ruoli IAM

Un ruolo IAM è un'identità interna all'utente Account AWS che dispone di autorizzazioni specifiche. È simile a un utente IAM, ma non è associato a una persona specifica. Per assumere temporaneamente un ruolo IAM in AWS Management Console, puoi passare da un ruolo utente a un ruolo IAM (console). Puoi assumere un ruolo chiamando un'operazione AWS CLI o AWS API o utilizzando un URL personalizzato. Per ulteriori informazioni sui metodi per l'utilizzo dei ruoli, consulta Utilizzo di ruoli IAM nella Guida per l'utente IAM.

I ruoli IAM con credenziali temporanee sono utili nelle seguenti situazioni:

- Accesso utente federato: per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per ulteriori informazioni sulla federazione dei ruoli, consulta Create a role for a third-party identity provider (federation) nella Guida per l'utente IAM. Se utilizzi IAM Identity Center, configura un set di autorizzazioni. IAM Identity Center mette in correlazione il set di autorizzazioni con un ruolo in IAM per controllare a cosa possono accedere le identità dopo l'autenticazione. Per informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .
- Autorizzazioni utente IAM temporanee: un utente IAM o un ruolo può assumere un ruolo IAM per ottenere temporaneamente autorizzazioni diverse per un'attività specifica.
- Accesso multi-account: è possibile utilizzare un ruolo IAM per permettere a un utente (un principale affidabile) con un account diverso di accedere alle risorse nell'account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, con alcuni Servizi AWS, è possibile allegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per

informazioni sulle differenze 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.

- Accesso a più servizi: alcuni Servizi AWS utilizzano le funzionalità di altri Servizi AWS. Ad esempio, quando effettui una chiamata in un servizio, è normale che quel servizio esegua applicazioni in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.
  - Sessioni di accesso inoltrato (FAS): quando utilizzi un utente o un ruolo IAM per eseguire azioni AWS, sei considerato un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra operazione in un servizio diverso. FAS utilizza le autorizzazioni del principale che chiama an Servizio AWS, combinate con la richiesta Servizio AWS per effettuare richieste ai servizi downstream. Le richieste FAS vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse per essere completata. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli delle policy relative alle richieste FAS, consulta Forward access sessions.
  - Ruolo di servizio: un ruolo di servizio è un ruolo IAM che un servizio assume per eseguire operazioni per tuo conto. Un amministratore IAM può creare, modificare ed eliminare un ruolo di servizio dall'interno di IAM. Per ulteriori informazioni, consulta la sezione Create a role to delegate permissions to an Servizio AWS nella Guida per l'utente IAM.
  - Ruolo collegato al servizio: un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'azione per tuo conto. I ruoli collegati al servizio vengono visualizzati nel tuo account Account AWS e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati ai servizi, ma non modificarle.
- Applicazioni in esecuzione su Amazon EC2: puoi utilizzare un ruolo IAM per gestire le credenziali temporanee per le applicazioni in esecuzione su un' EC2 istanza e che AWS CLI effettuano richieste AWS API. Questa soluzione è preferibile alla memorizzazione delle chiavi di accesso all'interno dell' EC2 istanza. Per assegnare un AWS ruolo a un' EC2 istanza e renderlo disponibile per tutte le sue applicazioni, create un profilo di istanza collegato all'istanza. Un profilo di istanza contiene il ruolo e consente ai programmi in esecuzione sull' EC2 istanza di ottenere credenziali temporanee. Per ulteriori informazioni, consulta Utilizzare un ruolo IAM per concedere le autorizzazioni alle applicazioni in esecuzione su EC2 istanze Amazon nella IAM User Guide.

Gestione dell'accesso con policy

Puoi controllare l'accesso AWS creando policy e collegandole a AWS identità o risorse. Una policy è un oggetto AWS che, se associato a un'identità o a una risorsa, ne definisce le autorizzazioni. AWS valuta queste politiche quando un principale (utente, utente root o sessione di ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle politiche viene archiviata AWS come documenti JSON. Per ulteriori informazioni sulla struttura e sui contenuti dei documenti delle policy JSON, consulta Panoramica delle policy JSON nella Guida per l'utente IAM.

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire operazioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti l'autorizzazione a eseguire operazioni sulle risorse di cui hanno bisogno, un amministratore IAM può creare policy IAM. L'amministratore può quindi aggiungere le policy IAM ai ruoli e gli utenti possono assumere i ruoli.

Le policy IAM definiscono le autorizzazioni relative a un'operazione, a prescindere dal metodo utilizzato per eseguirla. Ad esempio, supponiamo di disporre di una policy che consente l'operazione `iam:GetRole`. Un utente con tale policy può ottenere informazioni sul ruolo dall' AWS Management Console AWS CLI, dall'o dall' AWS API.

Policy basate sull'identità

Le policy basate su identità sono documenti di policy di autorizzazione JSON che è possibile allegare a un'identità (utente, gruppo di utenti o ruolo IAM). Tali policy definiscono le operazioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consulta Definizione di autorizzazioni personalizzate IAM con policy gestite dal cliente nella Guida per l'utente IAM.

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono integrate direttamente in un singolo utente, gruppo o ruolo. Le politiche gestite sono politiche autonome che puoi allegare a più utenti, gruppi e ruoli nel tuo Account AWS. Le politiche gestite includono politiche AWS gestite e politiche gestite dai clienti. Per informazioni su come scegliere tra una policy gestita o una policy inline, consulta Scelta fra policy gestite e policy inline nella Guida per l'utente IAM.

Policy basate sulle risorse

Le policy basate su risorse sono documenti di policy JSON che è possibile collegare a una risorsa. Esempi di policy basate sulle risorse sono le policy di attendibilità dei ruoli IAM e le policy dei bucket Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le operazioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Le policy basate sulle risorse sono policy inline che si trovano 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 e meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- Limiti delle autorizzazioni: un limite delle autorizzazioni è una funzionalità avanzata nella quale si imposta il numero massimo di autorizzazioni che una policy basata su identità può concedere a un'entità IAM (utente o ruolo IAM). È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal`sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità IAM nella Guida per l'utente IAM.
- Politiche di controllo del servizio (SCPs): SCPs sono politiche JSON che specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa (OU) in. AWS Organizations AWS Organizations è un servizio per il raggruppamento e la gestione centralizzata di più di

proprietà dell' Account AWS azienda. Se abiliti tutte le funzionalità di un'organizzazione, puoi applicare le politiche di controllo del servizio (SCPs) a uno o tutti i tuoi account. L'SCP limita le autorizzazioni per le entità presenti negli account dei membri, inclusa ciascuna di esse. Utente root dell'account AWS Per ulteriori informazioni su Organizations and SCPs, consulta le politiche di controllo dei servizi nella Guida AWS Organizations per l'utente.

- Politiche di controllo delle risorse (RCPs): RCPs sono politiche JSON che puoi utilizzare per impostare le autorizzazioni massime disponibili per le risorse nei tuoi account senza aggiornare le politiche IAM allegate a ciascuna risorsa di tua proprietà. L'RCP limita le autorizzazioni per le risorse negli account dei membri e può influire sulle autorizzazioni effettive per le identità, incluse le Utente root dell'account AWS, indipendentemente dal fatto che appartengano o meno all'organizzazione. Per ulteriori informazioni su Organizations e RCPs, incluso un elenco di Servizi AWS tale supporto RCPs, vedere Resource control policies (RCPs) nella Guida per l'AWS Organizations utente.
- Policy di sessione: le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta Policy di sessione nella Guida per l'utente IAM.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per scoprire come si AWS determina se consentire o meno una richiesta quando sono coinvolti più tipi di policy, consulta la logica di valutazione delle policy nella IAM User Guide.

Come funziona App Runner con IAM

Prima di utilizzare IAM per gestire l'accesso a AWS App Runner, è necessario comprendere quali funzionalità IAM sono disponibili per l'uso con App Runner. Per avere una visione di alto livello di come App Runner e altri AWS servizi funzionano con IAM, consulta AWS Services That Work with IAM nella IAM User Guide.

Per altri argomenti sulla sicurezza di App Runner, consulta. Sicurezza

Argomenti

- Politiche basate sull'identità di App Runner

- Politiche basate sulle risorse di App Runner
- Autorizzazione basata sui tag App Runner
- Autorizzazioni utente di App Runner
- Ruoli IAM di App Runner

Politiche basate sull'identità di App Runner

Con le policy basate su identità di IAM, è possibile specificare quali azioni e risorse sono consentite o rifiutate, nonché le condizioni in base alle quali le azioni sono consentite o rifiutate. App Runner supporta azioni, risorse e chiavi di condizione specifiche. Per informazioni su tutti gli elementi utilizzati in una policy JSON, consulta Documentazione di riferimento degli elementi delle policy JSON IAM nella Guida per l'utente IAM.

Azioni

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento `Action`di una policy JSON descrive le operazioni che è possibile utilizzare per consentire o negare l'accesso a un criterio. Le azioni politiche in genere hanno lo stesso nome dell'operazione AWS API associata. Ci sono alcune eccezioni, ad esempio le operazioni di sola autorizzazione che non hanno un'operazione API corrispondente. Esistono anche alcune operazioni che richiedono più operazioni in una policy. Queste operazioni aggiuntive sono denominate operazioni dipendenti.

Includi le operazioni in una policy per concedere le autorizzazioni a eseguire l'operazione associata.

Le azioni politiche in App Runner utilizzano il seguente prefisso prima dell'azione:. `apprunner:`
Ad esempio, per concedere a qualcuno l'autorizzazione a eseguire un' EC2 istanza Amazon con il funzionamento dell' EC2 `RunInstances`API Amazon, includi l'`ec2:RunInstances`azione nella sua politica. Le istruzioni della policy devono includere un elemento `Action` o `NotAction`. App Runner definisce il proprio set di azioni che descrivono le attività che puoi eseguire con questo servizio.

Per specificare più azioni in una sola istruzione, separa ciascuna di esse con una virgola come mostrato di seguito:

```
"Action": [
   "apprunner:CreateService",
   "apprunner:CreateConnection"
```

```
]
```

È possibile specificare più azioni tramite caratteri jolly (*). Ad esempio, per specificare tutte le azioni che iniziano con la parola `Describe`, includi la seguente azione:

```
"Action": "apprunner:Describe*"
```

Per visualizzare un elenco delle azioni di App Runner, consulta Azioni definite da AWS App Runner nel Service Authorization Reference.

Risorse

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento JSON `Resource`della policy specifica l'oggetto o gli oggetti ai quali si applica l'operazione. Le istruzioni devono includere un elemento `Resource`o un elemento `NotResource`. Come best practice, specifica una risorsa utilizzando il suo nome della risorsa Amazon (ARN). È possibile eseguire questa operazione per operazioni che supportano un tipo di risorsa specifico, note come autorizzazioni a livello di risorsa.

Per le azioni che non supportano le autorizzazioni a livello di risorsa, ad esempio le operazioni di elenco, utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"
```

Le risorse di App Runner hanno la seguente struttura ARN:

```
arn:aws:apprunner:region:account-id:resource-type/resource-name[/resource-id]
```

Per ulteriori informazioni sul formato di ARNs, consulta Amazon Resource Names (ARNs) e AWS Service Namespaces nel. Riferimenti generali di AWS

Ad esempio, per specificare il `my-service` servizio nella dichiarazione, utilizza il seguente ARN:

```
"Resource": "arn:aws:apprunner:us-east-1:123456789012:service/my-service"
```

Per specificare tutti i servizi che appartengono a un account specifico, usa il carattere jolly (*):

```
"Resource": "arn:aws:apprunner:us-east-1:123456789012:service/*"
```

Alcune azioni di App Runner, come quelle per la creazione di risorse, non possono essere eseguite su una risorsa specifica. In questi casi, è necessario utilizzare il carattere jolly (*).

```
"Resource": "*"
```

Per visualizzare un elenco dei tipi di risorse di App Runner e relativi ARNs, consulta Risorse definite da AWS App Runner nel Service Authorization Reference. Per informazioni sulle operazioni con cui è possibile specificare l'ARN di ogni risorsa, consulta la sezione Operazioni definite da AWS App Runner.

Chiavi di condizione

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento `Condition`(o blocco `Condition`) consente di specificare le condizioni in cui un'istruzione è in vigore. L'elemento `Condition`è facoltativo. È possibile compilare espressioni condizionali che utilizzano operatori di condizione, ad esempio uguale a o minore di, per soddisfare la condizione nella policy con i valori nella richiesta.

Se specifichi più elementi `Condition`in un'istruzione o più chiavi in un singolo elemento `Condition`, questi vengono valutati da AWS utilizzando un'operazione `AND`logica. Se si specificano più valori per una singola chiave di condizione, AWS valuta la condizione utilizzando un'operazione logica. `OR` Tutte le condizioni devono essere soddisfatte prima che le autorizzazioni dell'istruzione vengano concesse.

È possibile anche utilizzare variabili segnaposto quando specifichi le condizioni. Ad esempio, è possibile autorizzare un utente IAM ad accedere a una risorsa solo se è stata taggata con il relativo nome utente IAM. Per ulteriori informazioni, consulta Elementi delle policy IAM: variabili e tag nella Guida per l'utente di IAM.

AWS supporta chiavi di condizione globali e chiavi di condizione specifiche del servizio. Per visualizzare tutte le chiavi di condizione AWS globali, consulta le chiavi di contesto delle condizioni AWS globali nella Guida per l'utente IAM.

App Runner supporta l'utilizzo di alcune chiavi di condizione globali. Per visualizzare tutte le chiavi di condizione AWS globali, consulta AWS Global Condition Context Keys nella IAM User Guide.

App Runner definisce un set di chiavi di condizione specifiche del servizio. Inoltre, App Runner supporta il controllo degli accessi basato su tag, che viene implementato utilizzando chiavi condizionali. Per informazioni dettagliate, consultare the section called “Autorizzazione basata sui tag App Runner”.

Per visualizzare un elenco delle chiavi di condizione di App Runner, consulta Condition keys for AWS App Runner nel Service Authorization Reference. Per sapere con quali azioni e risorse puoi utilizzare una chiave di condizione, vedi Azioni definite da AWS App Runner.

Esempi

Per visualizzare esempi di politiche basate sull'identità di App Runner, consulta. Esempi di policy basate sull'identità di App Runner

Politiche basate sulle risorse di App Runner

App Runner non supporta politiche basate sulle risorse.

Autorizzazione basata sui tag App Runner

Puoi allegare tag alle risorse di App Runner o passare i tag in una richiesta ad App Runner. Per controllare l'accesso basato su tag, fornisci informazioni sui tag nell'elemento condizione di una policy utilizzando le chiavi di condizione `apprunner:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* o `aws:TagKeys`. Per ulteriori informazioni sull'etichettatura delle risorse di App Runner, consulta. the section called “Configurazione”

Per visualizzare una policy basata sulle identità di esempio per limitare l'accesso a una risorsa basata su tag su tale risorsa, consulta Controllo dell'accesso ai servizi App Runner in base ai tag.

Autorizzazioni utente di App Runner

Per utilizzare App Runner, gli utenti IAM necessitano delle autorizzazioni per le azioni di App Runner. Un modo comune per concedere le autorizzazioni agli utenti consiste nell'associare una policy agli utenti o ai gruppi IAM. Per ulteriori informazioni sulla gestione delle autorizzazioni degli utenti, consulta Modifica delle autorizzazioni per un utente IAM nella Guida per l'utente IAM.

App Runner fornisce due policy gestite che puoi allegare ai tuoi utenti.

- `AWSAppRunnerReadOnlyAccess`— Concede le autorizzazioni per elencare e visualizzare i dettagli sulle risorse di App Runner.
- `AWSAppRunnerFullAccess`— Concede le autorizzazioni a tutte le azioni di App Runner.

Per un controllo più granulare delle autorizzazioni degli utenti, puoi creare una policy personalizzata e allegarla ai tuoi utenti. Per i dettagli, consulta Creazione delle politiche IAM nella Guida per l'utente IAM.

Per esempi di policy per gli utenti, consultathe section called “Policy utente”.

AWSAppRunnerReadOnlyAccess

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        ":List*",
        ":Describe*"
      ],
      "Resource": "*"
    }
  ]
}
```

AWSAppRunnerFullAccess

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/apprunner.amazonaws.com/
AWSServiceRoleForAppRunner",
        "arn:aws:iam::*:role/aws-service-role/networking.apprunner.amazonaws.com/
AWSServiceRoleForAppRunnerNetworking"
      ],
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": [
```

```
            "apprunner.amazonaws.com",
            "networking.apprunner.amazonaws.com"
          ]
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:PassedToService": "apprunner.amazonaws.com"
        }
      }
    },
    {
      "Sid": "AppRunnerAdminAccess",
      "Effect": "Allow",
      "Action": "apprunner:*",
      "Resource": "*"
    }
  ]
}
```

Ruoli IAM di App Runner

Un ruolo IAM è un'entità interna all'utente Account AWS che dispone di autorizzazioni specifiche.

Ruoli collegati ai servizi

I ruoli collegati ai AWS servizi consentono ai servizi di accedere alle risorse di altri servizi per completare un'azione per tuo conto. I ruoli collegati ai servizi sono visualizzati nell'account IAM e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati ai servizi, ma non può modificarle.

App Runner supporta i ruoli collegati ai servizi. Per informazioni sulla creazione o la gestione di ruoli collegati ai servizi App Runner, consulta. the section called “Uso di ruoli collegati ai servizi”

Ruoli dei servizi

Questa caratteristica consente a un servizio di assumere un ruolo di servizio per conto dell'utente. Questo ruolo consente al servizio di accedere alle risorse in altri servizi per completare un'azione per

conto dell'utente. I ruoli dei servizi sono visualizzati nell'account IAM e sono di proprietà dell'account. Ciò significa che un utente IAM può modificare le autorizzazioni per questo ruolo. Tuttavia, il farlo potrebbe pregiudicare la funzionalità del servizio.

App Runner supporta alcuni ruoli di servizio.

Ruolo di accesso

Il ruolo di accesso è un ruolo utilizzato da App Runner per accedere alle immagini in Amazon Elastic Container Registry (Amazon ECR) nel tuo account. È necessario per accedere a un'immagine in Amazon ECR e non è richiesto con Amazon ECR Public. Prima di creare un servizio basato su un'immagine in Amazon ECR, utilizza IAM per creare un ruolo di servizio e utilizzare la policy `AWSAppRunnerServicePolicyForECRAccess` gestita al suo interno. Puoi quindi passare questo ruolo ad App Runner quando chiami l'CreateServiceAPI nel AuthenticationConfigurationmembro del SourceConfigurationparametro o quando usi la console App Runner per creare un servizio.

AWSAppRunnerServicePolicyForECRAccess

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchCheckLayerAvailability",
        "ecr:BatchGetImage",
        "ecr:DescribeImages",
        "ecr:GetAuthorizationToken"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Se crei una politica personalizzata per il tuo ruolo di accesso, assicurati di specificare `"Resource": "*"` l'azione`ecr:GetAuthorizationToken`. I token possono essere utilizzati per accedere a qualsiasi registro Amazon ECR a cui hai accesso.

Quando crei il tuo ruolo di accesso, assicurati di aggiungere una politica di fiducia che dichiari il responsabile del servizio App Runner `build.apprunner.amazonaws.com` come entità attendibile.

Politica di fiducia per un ruolo di accesso

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "build..amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Se utilizzi la console App Runner per creare un servizio, la console può creare automaticamente un ruolo di accesso per te e sceglierlo per il nuovo servizio. La console elenca anche altri ruoli nel tuo account e, se lo desideri, puoi selezionare un ruolo diverso.

Ruolo dell'istanza

Il ruolo di istanza è un ruolo opzionale utilizzato da App Runner per fornire le autorizzazioni alle azioni di AWS servizio necessarie alle istanze di calcolo del servizio. È necessario fornire un ruolo di istanza ad App Runner se il codice dell'applicazione chiama actions (). AWS APIs Incorpora le autorizzazioni richieste nel ruolo dell'istanza o crea una politica personalizzata e usala nel ruolo di istanza. Non abbiamo modo di anticipare le chiamate utilizzate dal tuo codice. Pertanto, non forniamo una politica gestita per questo scopo.

Prima di creare un servizio App Runner, utilizza IAM per creare un ruolo di servizio con le politiche personalizzate o integrate richieste. Puoi quindi passare questo ruolo ad App Runner come ruolo di istanza quando chiami l'CreateServiceAPI nel `InstanceRoleArn` membro del InstanceConfigurationparametro o quando usi la console App Runner per creare un servizio.

Quando crei il tuo ruolo di istanza, assicurati di aggiungere una politica di fiducia che dichiari il responsabile del servizio App Runner `tasks.apprunner.amazonaws.com` come entità attendibile.

Politica di fiducia per un ruolo di istanza

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "tasks..amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Se utilizzi la console App Runner per creare un servizio, la console elenca i ruoli del tuo account e puoi selezionare il ruolo che hai creato a tale scopo.

Per informazioni sulla creazione di un servizio, consultathe section called “Creazione”.

Esempi di policy basate sull'identità di App Runner

Per impostazione predefinita, gli utenti e i ruoli IAM non sono autorizzati a creare o modificare risorse. AWS App Runner Inoltre, non possono eseguire attività utilizzando l' AWS API AWS Management Console AWS CLI, o. Un amministratore IAM deve creare policy IAM che concedono a utenti e ruoli l'autorizzazione per eseguire operazioni API specifiche sulle risorse specificate di cui hanno bisogno. L'amministratore deve quindi allegare queste policy a utenti o IAM che richiedono tali autorizzazioni.

Per informazioni su come creare una policy basata su identità IAM utilizzando questi documenti di policy JSON di esempio, consulta Creazione di policy nella scheda JSON nella Guida per l'utente IAM.

Per altri argomenti sulla sicurezza di App Runner, consultaSicurezza.

Argomenti

- Best practice per le policy
- Policy utente
- Controllo dell'accesso ai servizi App Runner in base ai tag

Best practice per le policy

Le politiche basate sull'identità determinano se qualcuno può creare, accedere o eliminare le risorse di App Runner nel tuo account. Queste azioni possono comportare costi aggiuntivi per l' Account AWS. Quando crei o modifichi policy basate su identità, segui queste linee guida e raccomandazioni:

- Inizia con le policy AWS gestite e passa alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza le politiche gestite che concedono le autorizzazioni per molti casi d'uso comuni.AWS Sono disponibili nel tuo. Account AWS Ti consigliamo di ridurre ulteriormente le autorizzazioni definendo politiche gestite dai AWS clienti specifiche per i tuoi casi d'uso. Per ulteriori informazioni, consulta Policy gestite da AWSo Policy gestite da AWS per le funzioni dei processi nella Guida per l'utente IAM.
- Applica le autorizzazioni con privilegio minimo: quando imposti le autorizzazioni con le policy IAM, concedi solo le autorizzazioni richieste per eseguire un'attività. È possibile farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegi minimi. Per ulteriori informazioni sull'utilizzo di IAM per applicare le autorizzazioni, consulta Policy e autorizzazioni in IAM nella Guida per l'utente IAM.
- Condizioni d'uso nelle policy IAM per limitare ulteriormente l'accesso: per limitare l'accesso a operazioni e risorse è possibile aggiungere una condizione alle tue policy. Ad esempio, è possibile scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzando SSL. Puoi anche utilizzare le condizioni per concedere l'accesso alle azioni del servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio AWS CloudFormation. Per ulteriori informazioni, consulta la sezione Elementi delle policy JSON di IAM: condizione nella Guida per l'utente IAM.

- Utilizzo di IAM Access Analyzer per convalidare le policy IAM e garantire autorizzazioni sicure e funzionali: IAM Access Analyzer convalida le policy nuove ed esistenti in modo che aderiscano alla sintassi della policy IAM (JSON) e alle best practice di IAM. IAM Access Analyzer offre oltre 100 controlli delle policy e consigli utili per creare policy sicure e funzionali. Per ulteriori informazioni, consulta Convalida delle policy per il Sistema di analisi degli accessi IAM nella Guida per l'utente IAM.
- Richiedi l'autenticazione a più fattori (MFA): se hai uno scenario che richiede utenti IAM o un utente root nel Account AWS tuo, attiva l'MFA per una maggiore sicurezza. Per richiedere la MFA quando vengono chiamate le operazioni API, aggiungi le condizioni MFA alle policy. Per ulteriori informazioni, consulta Protezione dell'accesso API con MFA nella Guida per l'utente IAM.

Per maggiori informazioni sulle best practice in IAM, consulta Best practice di sicurezza in IAM nella Guida per l'utente di IAM.

Policy utente

Per accedere alla console App Runner, gli utenti IAM devono disporre di un set minimo di autorizzazioni. Queste autorizzazioni devono consentirti di elencare e visualizzare i dettagli sulle risorse di App Runner presenti nel tuo. Account AWS Se crei una politica basata sull'identità più restrittiva delle autorizzazioni minime richieste, la console non funzionerà come previsto per gli utenti con quella politica.

App Runner offre due policy gestite che puoi allegare ai tuoi utenti.

- `AWSAppRunnerReadOnlyAccess`— Concede le autorizzazioni per elencare e visualizzare i dettagli sulle risorse di App Runner.
- `AWSAppRunnerFullAccess`— Concede le autorizzazioni a tutte le azioni di App Runner.

Per garantire che gli utenti possano utilizzare la console App Runner, allega almeno la policy `AWSAppRunnerReadOnlyAccess` gestita agli utenti. Puoi invece allegare la politica `AWSAppRunnerFullAccess` gestita o aggiungere autorizzazioni aggiuntive specifiche per consentire agli utenti di creare, modificare ed eliminare la risorsa. Per ulteriori informazioni, consulta Aggiunta di autorizzazioni a un utente nella Guida per l'utente IAM.

Non è necessario consentire autorizzazioni minime per la console per gli utenti che effettuano chiamate solo verso AWS CLI o l' AWS API. Consenti invece l'accesso solo alle azioni che corrispondono all'operazione API che desideri consentire agli utenti di eseguire.

Gli esempi seguenti illustrano le politiche utente personalizzate. È possibile utilizzarli come punti di partenza per definire politiche utente personalizzate. Copia l'esempio e/o rimuovi le azioni, definisci l'ambito delle risorse e aggiungi condizioni.

Esempio: politica utente per la gestione della console e della connessione

Questa policy di esempio consente l'accesso alla console e consente la creazione e la gestione delle connessioni. Non consente la creazione e la gestione del servizio App Runner. Può essere collegato a un utente il cui ruolo è gestire l'accesso del servizio App Runner alle risorse del codice sorgente.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        ":List*",
        ":Describe*",
        ":CreateConnection",
        ":DeleteConnection"
      ],
      "Resource": "*"
    }
  ]
}
```

Esempio: politiche utente che utilizzano chiavi condizionali

Gli esempi in questa sezione mostrano le autorizzazioni condizionali che dipendono da alcune proprietà delle risorse o dai parametri di azione.

Questa politica di esempio consente la creazione di un servizio App Runner ma nega l'utilizzo di una connessione denominata. `prod`

JSON

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreateAppRunnerServiceWithNonProdConnections",
      "Effect": "Allow",
      "Action": ":CreateService",
      "Resource": "*",
      "Condition": {
        "ArnNotLike": {
          ":ConnectionArn": "arn:aws::*:*:connection/prod/*"
        }
      }
    }
  ]
}
```

Questo criterio di esempio consente l'aggiornamento di un servizio App Runner denominato `preprod` solo con una configurazione di scalabilità automatica denominata. `preprod`

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUpdatePreProdAppRunnerServiceWithPreProdASConfig",
      "Effect": "Allow",
      "Action": ":UpdateService",
      "Resource": "arn:aws::*:*:service/preprod/*",
      "Condition": {
        "ArnLike": {
          ":AutoScalingConfigurationArn": "arn:aws::*:*:autoscalingconfiguration/
preprod/*"
        }
      }
    }
  ]
}
```

Controllo dell'accesso ai servizi App Runner in base ai tag

Puoi utilizzare le condizioni della tua policy basata sull'identità per controllare l'accesso alle risorse di App Runner in base ai tag. Questo esempio mostra come è possibile creare una politica che consenta l'eliminazione di un servizio App Runner. Tuttavia, l'autorizzazione viene concessa solo se il valore del tag del servizio `0wner` è quello del nome utente dell'utente. Questa policy concede anche le autorizzazioni necessarie per completare questa azione nella console.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ListServicesInConsole",
      "Effect": "Allow",
      "Action": ":ListServices",
      "Resource": "*"
    },
    {
      "Sid": "DeleteServiceIfOwner",
      "Effect": "Allow",
      "Action": ":DeleteService",
      "Resource": "arn:aws::*:*:service/*",
      "Condition": {
        "StringEquals": {":ResourceTag/Owner": "${aws:username}"}
      }
    }
  ]
}
```

Puoi allegare questa policy agli utenti IAM nel tuo account. Se un utente denominato `richard-roe` tenta di eliminare un servizio App Runner, il servizio deve essere taggato o. `0wner=richard-roe` `owner=richard-roe` In caso contrario l'accesso è negato. La chiave di tag di condizione `0wner` corrisponde a `0wner` e `owner` perché i nomi delle chiavi di condizione non effettuano la distinzione tra maiuscole e minuscole. Per ulteriori informazioni, consulta la sezione Elementi delle policy JSON di IAM: condizione nella Guida per l'utente di IAM.

Utilizzo di ruoli collegati ai servizi per App Runner

AWS App Runner utilizza ruoli collegati ai AWS Identity and Access Management servizi (IAM). Un ruolo collegato ai servizi è un tipo unico di ruolo IAM collegato direttamente ad App Runner. I ruoli collegati ai servizi sono predefiniti da App Runner e includono tutte le autorizzazioni richieste dal servizio per chiamare altri servizi per tuo conto. AWS

Argomenti

Utilizzo dei ruoli per la gestione

AWS App Runner utilizza ruoli collegati ai AWS Identity and Access Management servizi (IAM). Un ruolo collegato ai servizi è un tipo unico di ruolo IAM collegato direttamente ad App Runner. I ruoli collegati ai servizi sono predefiniti da App Runner e includono tutte le autorizzazioni richieste dal servizio per chiamare altri servizi per tuo conto. AWS

Un ruolo collegato ai servizi semplifica la configurazione di App Runner perché non è necessario aggiungere manualmente le autorizzazioni necessarie. App Runner definisce le autorizzazioni dei suoi ruoli collegati ai servizi e, se non diversamente definito, solo App Runner può assumerne i ruoli. Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. Ciò protegge le tue risorse di App Runner perché non puoi rimuovere inavvertitamente l'autorizzazione ad accedere alle risorse.

Per informazioni sugli altri servizi che supportano i ruoli collegati ai servizi, consulta la sezione Servizi AWS che funzionano con IAM e cerca i servizi che riportano Sì nella colonna Ruolo associato ai servizi. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Autorizzazioni di ruolo collegate al servizio per App Runner

App Runner utilizza il ruolo collegato al servizio denominato. AWSServiceRoleForAppRunner

Il ruolo consente ad App Runner di eseguire le seguenti attività:

- Invia i log ai gruppi di log di Amazon CloudWatch Logs.

- Crea regole Amazon CloudWatch Events per iscriverti ai push di immagini di Amazon Elastic Container Registry (Amazon ECR).
- Invia le informazioni di tracciamento a. AWS X-Ray

Il ruolo AWSService RoleForAppRunner collegato al servizio prevede che i seguenti servizi assumano il ruolo:

- `apprunner.amazonaws.com`

Le politiche di autorizzazione del ruolo AWSService RoleForAppRunner collegato al servizio contengono tutte le autorizzazioni di cui App Runner ha bisogno per completare le azioni per conto dell'utente.

AppRunnerServiceRolePolicy politica gestita

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:PutRetentionPolicy"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:*:*:log-group:/aws/apprunner/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams"
      ],
      "Resource": [
        "arn:aws:logs:*:*:log-group:/aws/apprunner/*:log-stream:*"
      ]
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "events:DeleteRule",
        "events:RemoveTargets",
        "events:DescribeRule",
        "events:EnableRule",
        "events:DisableRule"
      ],
      "Resource": "arn:aws:events:*:*:rule/AWSAppRunnerManagedRule*"
    }
  ]
}
```

Politica per il tracciamento a raggi X

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "xray:PutTraceSegments",
        "xray:PutTelemetryRecords",
        "xray:GetSamplingRules",
        "xray:GetSamplingTargets",
        "xray:GetSamplingStatisticSummaries"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato al servizio per App Runner

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un servizio App Runner nell' AWS Management Console, nella o nell' AWS API AWS CLI, App Runner crea automaticamente il ruolo collegato al servizio.

Se elimini questo ruolo collegato ai servizi, è possibile ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando crei un servizio App Runner, App Runner crea nuovamente il ruolo collegato al servizio per te.

Modifica di un ruolo collegato al servizio per App Runner

App Runner non consente di modificare il ruolo collegato al servizio. AWSService RoleForAppRunner Dopo aver creato un ruolo collegato al servizio, non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. È possibile tuttavia modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta la sezione Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato al servizio per App Runner

Se non è più necessario utilizzare una funzionalità o un servizio che richiede un ruolo collegato al servizio, ti consigliamo di eliminare il ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario effettuare la pulizia delle risorse associate al ruolo collegato ai servizi prima di poterlo eliminare manualmente.

Pulizia di un ruolo collegato ai servizi

Prima di utilizzare IAM; per eliminare un ruolo collegato al servizio, è necessario prima rimuovere qualsiasi risorsa utilizzata dal ruolo.

In App Runner, ciò significa eliminare tutti i servizi App Runner nel tuo account. Per ulteriori informazioni sull'eliminazione dei servizi App Runner, consulta. the section called “Eliminazione”

> **Note**
>
> Se il servizio App Runner utilizza il ruolo quando si tenta di eliminare le risorse, l'eliminazione potrebbe non riuscire. In questo caso, attendi alcuni minuti e quindi ripeti l'operazione.

Eliminazione manuale del ruolo collegato ai servizi

Utilizza la console IAM AWS CLI, o l' AWS API per eliminare il ruolo collegato al AWSService RoleForAppRunner servizio. Per ulteriori informazioni, consulta Eliminazione del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Regioni supportate per i ruoli collegati al servizio App Runner

App Runner supporta l'utilizzo di ruoli collegati al servizio in tutte le regioni in cui il servizio è disponibile. Per ulteriori informazioni, consulta Endpoint e quote AWS App Runner nella Riferimenti generali di AWS.

Utilizzo dei ruoli per il networking

AWS App Runner utilizza ruoli collegati ai AWS Identity and Access Management servizi (IAM). Un ruolo collegato ai servizi è un tipo unico di ruolo IAM collegato direttamente ad App Runner. I ruoli collegati ai servizi sono predefiniti da App Runner e includono tutte le autorizzazioni richieste dal servizio per chiamare altri servizi per tuo conto. AWS

Un ruolo collegato ai servizi semplifica la configurazione di App Runner perché non è necessario aggiungere manualmente le autorizzazioni necessarie. App Runner definisce le autorizzazioni dei suoi ruoli collegati ai servizi e, se non diversamente definito, solo App Runner può assumerne i ruoli. Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. Ciò protegge le tue risorse di App Runner perché non puoi rimuovere inavvertitamente l'autorizzazione ad accedere alle risorse.

Per informazioni sugli altri servizi che supportano i ruoli collegati ai servizi, consulta la sezione Servizi AWS che funzionano con IAM e cerca i servizi che riportano Sì nella colonna Ruolo associato ai servizi. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Autorizzazioni di ruolo collegate al servizio per App Runner

App Runner utilizza il ruolo collegato al servizio denominato. AWSServiceRoleForAppRunnerNetworking

Il ruolo consente ad App Runner di eseguire le seguenti attività:

- Collega un VPC al tuo servizio App Runner e gestisci le interfacce di rete.

Il ruolo AWSService RoleForAppRunnerNetworking collegato al servizio prevede che i seguenti servizi assumano il ruolo:

- `networking.apprunner.amazonaws.com`

La politica di autorizzazione dei ruoli denominata AppRunnerNetworkingServiceRolePolicy contiene tutte le autorizzazioni di cui App Runner ha bisogno per completare le azioni per tuo conto.

AppRunnerNetworkingServiceRolePolicy

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeVpcs",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateNetworkInterface",
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringEquals": {
          "aws:TagKeys": [
            "AWSAppRunnerManaged"
          ]
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:*:*:network-interface/*",
```

```
      "Condition": {
        "StringEquals": {
          "ec2:CreateAction": "CreateNetworkInterface"
        },
        "StringLike": {
          "aws:RequestTag/AWSAppRunnerManaged": "*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:DeleteNetworkInterface",
      "Resource": "*",
      "Condition": {
        "Null": {
          "ec2:ResourceTag/AWSAppRunnerManaged": "false"
        }
      }
    }
  ]
}
```

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato al servizio per App Runner

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un connettore VPC nell' AWS API AWS Management Console, App Runner crea automaticamente il ruolo collegato al servizio. AWS CLI

Se elimini questo ruolo collegato ai servizi, è possibile ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando crei un connettore VPC, App Runner crea nuovamente il ruolo collegato al servizio per te.

Modifica di un ruolo collegato al servizio per App Runner

App Runner non consente di modificare il ruolo collegato al servizio. AWSService RoleForAppRunnerNetworking Dopo aver creato un ruolo collegato al servizio, non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. È possibile tuttavia modificarne la descrizione

utilizzando IAM. Per ulteriori informazioni, consulta la sezione Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato al servizio per App Runner

Se non è più necessario utilizzare una funzionalità o un servizio che richiede un ruolo collegato al servizio, ti consigliamo di eliminare il ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario effettuare la pulizia delle risorse associate al ruolo collegato ai servizi prima di poterlo eliminare manualmente.

Pulizia di un ruolo collegato ai servizi

Prima di utilizzare IAM; per eliminare un ruolo collegato al servizio, è necessario prima rimuovere qualsiasi risorsa utilizzata dal ruolo.

In App Runner, ciò significa dissociare i connettori VPC da tutti i servizi App Runner del tuo account ed eliminare i connettori VPC. Per ulteriori informazioni, consulta the section called “Traffico in uscita”.

> **Note**
>
> Se il servizio App Runner utilizza il ruolo quando si tenta di eliminare le risorse, l'eliminazione potrebbe non riuscire. In questo caso, attendi alcuni minuti e quindi ripeti l'operazione.

Eliminare manualmente il ruolo collegato al servizio

Utilizza la console IAM AWS CLI, o l' AWS API per eliminare il ruolo collegato al AWSService RoleForAppRunnerNetworking servizio. Per ulteriori informazioni, consulta Eliminazione del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Regioni supportate per i ruoli collegati al servizio App Runner

App Runner supporta l'utilizzo di ruoli collegati al servizio in tutte le regioni in cui il servizio è disponibile. Per ulteriori informazioni, consulta Endpoint e quote AWS App Runner nella Riferimenti generali di AWS.

AWS politiche gestite per AWS App Runner

Una politica AWS gestita è una politica autonoma creata e amministrata da. AWS AWS le politiche gestite sono progettate per fornire autorizzazioni per molti casi d'uso comuni, in modo da poter iniziare ad assegnare autorizzazioni a utenti, gruppi e ruoli.

Tieni presente che le policy AWS gestite potrebbero non concedere le autorizzazioni con il privilegio minimo per i tuoi casi d'uso specifici, poiché sono disponibili per tutti i clienti. AWS Ti consigliamo pertanto di ridurre ulteriormente le autorizzazioni definendo policy gestite dal cliente specifiche per i tuoi casi d'uso.

Non è possibile modificare le autorizzazioni definite nelle politiche gestite. AWS Se AWS aggiorna le autorizzazioni definite in una politica AWS gestita, l'aggiornamento ha effetto su tutte le identità principali (utenti, gruppi e ruoli) a cui è associata la politica. AWS è più probabile che aggiorni una policy AWS gestita quando ne Servizio AWS viene lanciata una nuova o quando diventano disponibili nuove operazioni API per i servizi esistenti.

Per ulteriori informazioni, consultare Policy gestite da AWSnella Guida per l'utente di IAM.

App Runner si aggiorna alle politiche AWS gestite

Visualizza i dettagli sugli aggiornamenti alle politiche AWS gestite per App Runner da quando questo servizio ha iniziato a tenere traccia di queste modifiche. Per ricevere avvisi automatici sulle modifiche a questa pagina, iscriviti al feed RSS nella pagina della cronologia dei documenti di App Runner.

Modifica	Descrizione	Data
AWSAppRunnerReadOnlyAccess: nuova policy	App Runner ha aggiunto una nuova politica per consentire agli utenti di elencare e visualizzare i dettagli sulle risorse di App Runner.	24 febbraio 2022
AWSAppRunnerFullAccess: aggiornamento a una policy esistente	App Runner ha aggiornato l'elenco delle risorse per l'`iam:CreateServiceLinkedRole` azione per consentire la creazione di ruoli collegati al `AWSServic`	8 febbraio 2022

Modifica	Descrizione	Data
	`eRoleForAppRunnerNetworking` servizio.	
AppRunnerNetworkingServiceRolePolicy: nuova policy	App Runner ha aggiunto una nuova policy per consentire ad App Runner di effettuare chiamate ad Amazon Virtual Private Cloud per collegare un VPC al servizio App Runner e gestire le interfacce di rete per conto dei servizi App Runner. La policy viene utilizzata nel ruolo collegato al servizio. `AWSServiceRoleForAppRunnerNetworking`	8 febbraio 2022
AWSAppRunnerFullAccess: nuova policy	App Runner ha aggiunto una nuova politica per consentire agli utenti di eseguire tutte le azioni di App Runner.	10 gennaio 2022
AppRunnerServiceRolePolicy: nuova policy	App Runner ha aggiunto una nuova policy per consentire ad App Runner di effettuare chiamate ad Amazon CloudWatch Logs e Amazon CloudWatch Events per conto dei servizi App Runner. La policy viene utilizzata nel ruolo collegato al servizio. `AWSServiceRoleForAppRunner`	1 marzo 2021
AWSAppRunnerServicePolicyForECRAccess: nuova policy	App Runner ha aggiunto una nuova policy per consentire ad App Runner di accedere alle immagini di Amazon Elastic Container Registry (Amazon ECR) nel tuo account.	1 marzo 2021
App Runner ha iniziato a tracciare le modifiche	App Runner ha iniziato a tenere traccia delle modifiche per le sue politiche AWS gestite.	1 marzo 2021

Risoluzione dei problemi relativi all'identità e all'accesso ad App Runner

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare quando lavori con AWS App Runner un IAM.

Per altri argomenti sulla sicurezza di App Runner, consulta. Sicurezza

Argomenti

- Non sono autorizzato a eseguire un'azione in App Runner
- Voglio consentire a persone esterne a me di accedere Account AWS alle mie risorse di App Runner

Non sono autorizzato a eseguire un'azione in App Runner

Se ti AWS Management Console dice che non sei autorizzato a eseguire un'azione, contatta l'amministratore per ricevere assistenza. L'amministratore è la persona che ti ha fornito le credenziali di AWS accesso.

Il seguente errore di esempio si verifica quando un utente IAM denominato `marymajor` tenta di utilizzare la console per visualizzare i dettagli su un servizio App Runner ma non dispone delle autorizzazioni. `apprunner:DescribeService`

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 apprunner:DescribeService on resource: my-example-service
```

In questo caso, Mary chiede al suo amministratore di aggiornare le sue politiche per consentirle di accedere alla *my-example-service* risorsa utilizzando l'azione`apprunner:DescribeService`.

Voglio consentire a persone esterne a me di accedere Account AWS alle mie risorse di App Runner

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue 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 consentire alle persone di accedere alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Per sapere se App Runner supporta queste funzionalità, consulta. Come funziona App Runner con IAM
- Per scoprire come fornire l'accesso alle tue risorse su tutto Account AWS ciò che possiedi, 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, consulta Fornire l'accesso a utenti autenticati esternamente (Federazione delle identità) nella Guida per l'utente 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.

Registrazione e monitoraggio in App Runner

Il monitoraggio è una parte importante del mantenimento dell'affidabilità, della disponibilità e delle prestazioni del AWS App Runner servizio. La raccolta dei dati di monitoraggio da tutte le parti della AWS soluzione consente di eseguire più facilmente il debug di un eventuale errore. App Runner si integra con diversi AWS strumenti per monitorare i servizi App Runner e rispondere a potenziali incidenti.

CloudWatch Allarmi Amazon

Con Amazon CloudWatch alarms, puoi monitorare una metrica di servizio per un periodo di tempo specificato. Se la metrica supera una determinata soglia per un determinato numero di periodi, ricevi una notifica.

App Runner raccoglie una serie di metriche sul servizio nel suo complesso e sulle istanze (unità di scalabilità) che eseguono il servizio web. Per ulteriori informazioni, consulta Metriche () CloudWatch.

Log di applicazioni

App Runner raccoglie l'output del codice dell'applicazione e lo trasmette ad Amazon Logs. CloudWatch Il contenuto di questo output dipende da te. Ad esempio, potresti includere registrazioni dettagliate delle richieste fatte al tuo servizio web. Questi record di registro potrebbero rivelarsi utili nei controlli di sicurezza e di accesso. Per ulteriori informazioni, consulta Registri (registri) CloudWatch .

AWS CloudTrail registri delle azioni

App Runner è integrato con AWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, ruolo o AWS servizio in App Runner. CloudTrail acquisisce tutte le chiamate API per App Runner come eventi. Puoi visualizzare gli eventi più recenti nella CloudTrail console e creare un percorso per consentire la distribuzione continua degli CloudTrail eventi a un bucket Amazon Simple Storage Service (Amazon S3). Per ulteriori informazioni, consulta Azioni API () CloudTrail.

Convalida della conformità per App Runner

I revisori esterni valutano la sicurezza e la conformità nell' AWS App Runner ambito di più programmi di AWS conformità. Questi includono SOC, PCI, FedRAMP, HIPAA e altri.

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. AWS fornisce le seguenti risorse per contribuire alla conformità:

- Governance e conformità per la sicurezza: queste guide all'implementazione di soluzioni illustrano considerazioni relative all'architettura e i passaggi per implementare le funzionalità di sicurezza e conformità.
- Riferimenti sui servizi conformi ai requisiti HIPAA: elenca i servizi HIPAA idonei. Non tutti Servizi AWS sono idonei alla normativa HIPAA.
- AWS Risorse per la per la conformità: questa raccolta di cartelle di lavoro e guide potrebbe essere valida per il tuo settore e la tua località.
- AWS Guide alla conformità dei clienti: comprendi il modello di responsabilità condivisa attraverso la lente della conformità. Le guide riassumono le migliori pratiche per la protezione Servizi AWS e mappano le linee guida per i controlli di sicurezza su più framework (tra cui il National Institute of Standards and Technology (NIST), il Payment Card Industry Security Standards Council (PCI) e l'International Organization for Standardization (ISO)).

- Valutazione delle risorse con regole nella Guida per gli AWS Config sviluppatori: il AWS Config servizio valuta la conformità delle configurazioni delle risorse alle pratiche interne, alle linee guida e alle normative del settore.
- AWS Security Hub— Ciò Servizio AWS fornisce una visione completa dello stato di sicurezza interno. AWS La Centrale di sicurezza utilizza i controlli di sicurezza per valutare le risorse AWS e verificare la conformità agli standard e alle best practice del settore della sicurezza. Per un elenco dei servizi e dei controlli supportati, consulta la pagina Documentazione di riferimento sui controlli della Centrale di sicurezza.
- Amazon GuardDuty: Servizio AWS rileva potenziali minacce ai tuoi carichi di lavoro Account AWS, ai contenitori e ai dati monitorando l'ambiente alla ricerca di attività sospette e dannose. GuardDuty può aiutarti a soddisfare vari requisiti di conformità, come lo standard PCI DSS, soddisfacendo i requisiti di rilevamento delle intrusioni imposti da determinati framework di conformità.
- AWS Audit Manager— Ciò Servizio AWS consente di verificare continuamente l' AWS utilizzo per semplificare la gestione del rischio e la conformità alle normative e agli standard di settore.

Per altri argomenti sulla sicurezza di App Runner, consultaSicurezza.

Resilienza in App Runner

L'infrastruttura AWS globale è costruita attorno a Regioni AWS zone di disponibilità. Regioni AWS forniscono più zone di disponibilità fisicamente separate e isolate, collegate con reti a bassa latenza, ad alto throughput e altamente ridondanti. Con le zone di disponibilità, è possibile progettare e gestire applicazioni e database che eseguono il failover automatico tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture tradizionali a data center singolo o multiplo.

Per ulteriori informazioni sulle zone di disponibilità, vedere Global Regioni AWS Infrastructure.AWS

AWS App Runner gestisce e automatizza l'uso dell'infrastruttura AWS globale per tuo conto. Quando si utilizza App Runner, si beneficia della disponibilità e dei meccanismi di tolleranza agli errori offerti. AWS

Per altri argomenti sulla sicurezza di App Runner, consulta. Sicurezza

Sicurezza dell'infrastruttura in AWS App Runner

In quanto servizio gestito, AWS App Runner è protetto dalle procedure di sicurezza della rete AWS globale descritte nel white paper Amazon Web Services: Overview of Security Processes.

Utilizzi chiamate API AWS pubblicate per accedere ad App Runner attraverso la rete. I client devono supportare Transport Layer Security (TLS) 1.2 o versioni successive. I client devono, inoltre, supportare le suite di cifratura con PFS (Perfect Forward Secrecy), ad esempio Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale IAM. O puoi utilizzare AWS Security Token Service (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Per altri argomenti sulla sicurezza di App Runner, consulta. Sicurezza

Utilizzo di App Runner con endpoint VPC

La tua AWS applicazione potrebbe integrare AWS App Runner servizi con altri Servizi AWS che vengono eseguiti in un VPC da Amazon Virtual Private Cloud (Amazon VPC). Alcune parti dell'applicazione potrebbero inviare richieste ad App Runner dall'interno del VPC. Ad esempio, potresti AWS CodePipeline utilizzarlo per eseguire l'implementazione continua sul tuo servizio App Runner. Un modo per migliorare la sicurezza dell'applicazione consiste nell'inviare queste richieste App Runner (e richieste ad altri Servizi AWS) tramite un endpoint VPC.

Utilizzando un endpoint VPC, puoi connettere privatamente il tuo VPC a servizi endpoint VPC supportati Servizi AWS e alimentati da. AWS PrivateLink Non è necessario un gateway Internet, un dispositivo NAT, una connessione VPN o una connessione. AWS Direct Connect

Le risorse nel tuo VPC non utilizzano indirizzi IP pubblici per interagire con le risorse di App Runner. Il traffico tra il tuo VPC e App Runner non esce dalla rete Amazon. Per ulteriori informazioni sugli endpoint VPC, consulta Endpoint VPC nella Guida.AWS PrivateLink

> **Note**
>
> Per impostazione predefinita, l'applicazione Web del servizio App Runner viene eseguita in un VPC fornito e configurato da App Runner. Questo VPC è pubblico. Significa che

è connesso a Internet. Facoltativamente, puoi associare la tua applicazione a un VPC personalizzato. Per ulteriori informazioni, consulta the section called “Traffico in uscita”.
Puoi configurare i tuoi servizi per accedere a Internet AWS APIs, anche quando il servizio è connesso a un VPC. Per istruzioni su come abilitare l'accesso pubblico a Internet per il traffico VPC in uscita, consulta. the section called “Considerazioni sulla scelta di una sottorete ”
App Runner non supporta la creazione di un endpoint VPC per la tua applicazione.

Configurazione di un endpoint VPC per App Runner

Per creare l'endpoint VPC dell'interfaccia per il servizio App Runner nel tuo VPC, segui la procedura Crea un endpoint di interfaccia nella Guida.AWS PrivateLink Per Service Name (Nome del servizio), selezionare `com.amazonaws.region.apprunner`.

Considerazioni sulla privacy della rete VPC

Important

L'utilizzo di un endpoint VPC per App Runner non garantisce che tutto il traffico proveniente dal tuo VPC rimanga lontano da Internet. Il VPC potrebbe essere pubblico. Inoltre, alcune parti della soluzione potrebbero non utilizzare gli endpoint VPC per effettuare AWS chiamate API. Ad esempio, Servizi AWS potrebbe chiamare altri servizi utilizzando i relativi endpoint pubblici. Se la privacy del traffico è necessaria per la soluzione nel tuo VPC, leggi questa sezione.

Per garantire la privacy del traffico di rete nel tuo VPC, considera quanto segue:

- Abilita il nome DNS: alcune parti dell'applicazione potrebbero comunque inviare richieste ad App Runner tramite Internet utilizzando l'`apprunner.region.amazonaws.com`endpoint pubblico. Se il tuo VPC è configurato con l'accesso a Internet, queste richieste vanno a buon fine senza che tu te ne dia alcuna indicazione. Puoi evitare che ciò accada assicurandoti che Enable DNS name sia abilitato quando crei l'endpoint. Per impostazione predefinita, è impostato su true. In questo modo viene aggiunta una voce DNS nel VPC che associa l'endpoint del servizio pubblico all'endpoint VPC dell'interfaccia.

- Configura gli endpoint VPC per servizi aggiuntivi: la tua soluzione potrebbe inviare richieste ad altri. Servizi AWS Ad esempio, AWS CodePipeline potrebbe inviare richieste a. AWS CodeBuild Configura gli endpoint VPC per questi servizi e abilita i nomi DNS su questi endpoint.
- Configura un VPC privato: se possibile (se la tua soluzione non richiede affatto l'accesso a Internet), configura il tuo VPC come privato, il che significa che non ha una connessione Internet. Ciò garantisce che un endpoint VPC mancante provochi un errore visibile, in modo da poter aggiungere l'endpoint mancante.

Utilizzo dei criteri endpoint per controllare l'accesso con gli endpoint VPC

Le policy degli endpoint VPC sono supportate per App Runner. Per impostazione predefinita, l'accesso completo ad App Runner è consentito tramite l'endpoint dell'interfaccia. Le policy degli endpoint VPC possono essere utilizzate per controllare quali responsabili AWS possono accedere all'endpoint App Runner. In alternativa, puoi associare un gruppo di sicurezza alle interfacce di rete degli endpoint per controllare il traffico verso App Runner attraverso l'endpoint dell'interfaccia.

Integrazione con l'endpoint dell'interfaccia

App Runner supporta AWS PrivateLink, che fornisce connettività privata ad App Runner ed elimina l'esposizione del traffico a Internet. Per consentire all'applicazione di inviare richieste a App Runner utilizzando AWS PrivateLink, configura un tipo di endpoint VPC noto come endpoint di interfaccia. Per ulteriori informazioni, consulta Interface VPC endpoints (AWS PrivateLink) nella Guida.AWS PrivateLink

Configurazione e analisi delle vulnerabilità in App Runner

AWS e i nostri clienti condividono la responsabilità di raggiungere un elevato livello di sicurezza e conformità dei componenti software. Per ulteriori informazioni, consulta il modello di responsabilità AWS condivisa.

Immagini del contenitore di patch

L'applicazione di patch all'immagine del contenitore fa parte della responsabilità del cliente nel modello di sicurezza condiviso. Il proprietario dell'immagine è responsabile dell'aggiornamento e della correzione periodica dell'immagine del contenitore. Ti consigliamo di stabilire una pianificazione di routine per il controllo e l'applicazione degli aggiornamenti alle immagini del contenitore. Per

ulteriori informazioni su come scansionare le immagini alla ricerca di vulnerabilità, consulta la documentazione di AWS App Runner

Per altri argomenti sulla sicurezza di App Runner, consulta. Sicurezza

Best practice di sicurezza per App Runner

AWS App Runner offre diverse funzionalità di sicurezza da considerare durante lo sviluppo e l'implementazione delle proprie politiche di sicurezza. Le seguenti best practice sono linee guida generali e non rappresentano una soluzione di sicurezza completa. Poiché queste best practice potrebbero non essere appropriate o sufficienti per il tuo ambiente, gestiscile come considerazioni utili anziché prescrizioni.

Per altri argomenti sulla sicurezza di App Runner, consultaSicurezza.

Best practice relative alla sicurezza preventiva

I controlli di sicurezza preventivi tentano di prevenire gli incidenti prima che si verifichino.

Implementazione dell'accesso con privilegi minimi

App Runner fornisce policy gestite AWS Identity and Access Management (IAM) per gli utenti IAM e il ruolo di accesso. Queste politiche gestite specificano tutte le autorizzazioni che potrebbero essere necessarie per il corretto funzionamento del servizio App Runner.

La tua applicazione potrebbe non richiedere tutte le autorizzazioni nelle nostre policy gestite. Puoi personalizzarle e concedere solo le autorizzazioni necessarie agli utenti e al servizio App Runner per eseguire le loro attività. Ciò è particolarmente rilevante per le policy utente, dove ruoli utente diversi potrebbero avere esigenze di autorizzazione diverse. L'applicazione dell'accesso con privilegio minimo è fondamentale per ridurre i rischi di sicurezza e l'impatto risultante da errori o intenzioni dannose.

Scansione delle immagini per individuare eventuali vulnerabilità

Puoi utilizzare Amazon ECR APIs per identificare le vulnerabilità del software nelle immagini dei container. Per ulteriori informazioni, consulta la documentazione di Amazon ECR.

Best practice relative alla sicurezza di rilevamento

I controlli di sicurezza di rilevamento identificano le violazioni della sicurezza dopo che si sono verificate. Possono aiutarti a rilevare potenziali minacce o incidenti alla sicurezza.

Implementazione del monitoraggio

Il monitoraggio è una parte importante per mantenere l'affidabilità, la sicurezza, la disponibilità e le prestazioni delle soluzioni App Runner. AWS fornisce diversi strumenti e servizi per aiutarti a monitorare i tuoi AWS servizi.

Di seguito sono elencati alcuni esempi di elementi da monitorare:

- CloudWatch Metriche Amazon per App Runner: imposta allarmi per le metriche chiave di App Runner e per le metriche personalizzate della tua applicazione. Per informazioni dettagliate, consultare Metriche () CloudWatch.
- AWS CloudTrail voci: tieni traccia delle azioni che potrebbero influire sulla disponibilità, come o. `PauseService` `DeleteConnection` Per informazioni dettagliate, consultare Azioni API () CloudTrail.

AWS Glossario

Per la AWS terminologia più recente, consultate il AWS glossario nella sezione Reference.Glossario AWS

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.