aws

Leitfaden für bewährte Verfahren

Amazon Elastic Container Service

Amazon Elastic Container Service: Leitfaden für bewährte Verfahren

Table of Contents

Einführung

Amazon Elastic Container Service (Amazon ECS) ist ein hoch skalierbarer und schneller Container-Management-Service, mit dem Sie Container in einem Cluster verwalten können. Dieser Leitfaden behandelt viele der wichtigsten betrieblichen Best Practices für Amazon ECS. Außerdem werden mehrere Kernkonzepte beschrieben, die für die Funktionsweise von Amazon ECS-basierten Anwendungen eine Rolle spielen. Ziel ist es, einen konkreten und umsetzbaren Ansatz für den Betrieb und die Fehlerbehebung von Amazon ECS-basierten Anwendungen bereitzustellen.

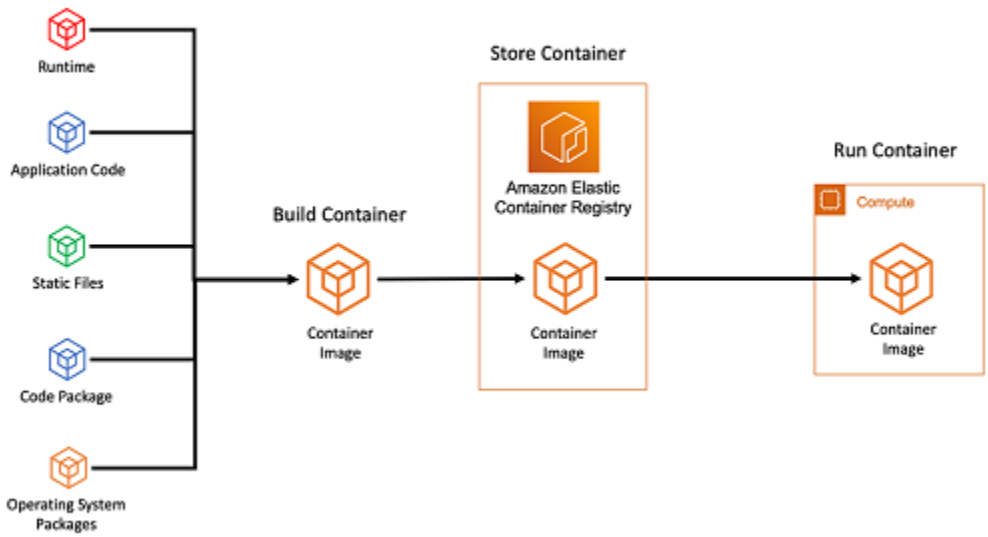
- Bewährte Methoden — Ausführung Ihrer Anwendung mit Amazon ECS
- Bewährte Methoden — Netzwerke
- Bewährte Methoden — Automatische Skalierung und Kapazitätsmanagement
- Bewährte Methoden — Persistenter Speicher
- Bewährte Methoden — Schnellerer Start von Aufgaben
- Bewährte Methoden — Schnellere Bereitstellungen
- Bewährte Methoden — Betrieb von Amazon ECS im großen Maßstab
- Bewährte Methoden — Sicherheit

Bewährte Methoden — Ausführung Ihrer Anwendung mit Amazon ECS

Bevor Sie eine Anwendung mit Amazon Elastic Container Service ausführen, sollten Sie sicherstellen, dass Sie wissen, wie die verschiedenen Aspekte Ihrer Anwendung mit Funktionen in Amazon ECS funktionieren. Dieses Handbuch behandelt die wichtigsten Amazon ECS-Ressourcentypen, wofür sie verwendet werden, und bewährte Methoden für die Verwendung der einzelnen Ressourcentypen.

Container-Image

Ein Container-Image enthält Ihren Anwendungscode und alle Abhängigkeiten, die Ihr Anwendungscode zur Ausführung benötigt. Zu den Anwendungsabhängigkeiten gehören die Quellcodepakete, auf denen Ihr Anwendungscode basiert sowie eine Sprachlaufzeit für interpretierte Sprachen und Binärpakete, auf denen Ihr dynamisch verlinkter Code basiert.



Container-Images durchlaufen einen dreistufigen Prozess.

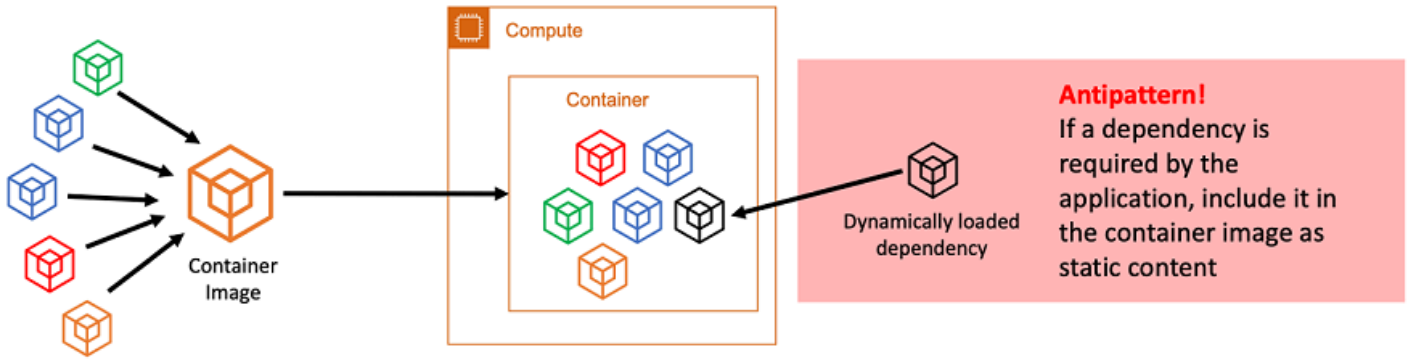
1. Erstellen — Sammeln Sie Ihre Anwendung und all ihre Abhängigkeiten in einem Container-Image.
2. Speichern — Laden Sie das Container-Image in eine Container-Registry hoch.
3. Ausführen — Laden Sie das Container-Image auf einen Computer herunter, entpacken Sie es und starten Sie die Anwendung.

Beachten Sie beim Erstellen Ihres eigenen Container-Images die in den folgenden Abschnitten beschriebenen bewährten Methoden.

Machen Sie Container-Images vollständig und statisch

Im Idealfall soll ein Container-Image eine vollständige Momentaufnahme von allem sein, was die Anwendung benötigt, um zu funktionieren. Mit einem vollständigen Container-Image können Sie eine Anwendung ausführen, indem Sie ein Container-Image von einem Ort herunterladen. Sie müssen nicht mehrere separate Teile von verschiedenen Speicherorten herunterladen. Es hat sich daher bewährt, alle Anwendungsabhängigkeiten als statische Dateien im Container-Image zu speichern.

Gleichzeitig sollten Sie Bibliotheken, Abhängigkeiten oder wichtige Daten beim Start der Anwendung nicht dynamisch herunterladen. Fügen Sie diese Dinge stattdessen als statische Dateien in das Container-Image ein. Wenn Sie später etwas am Container-Image ändern möchten, erstellen Sie ein neues Container-Image, auf das die Änderungen angewendet werden.



Es gibt einige Gründe, warum wir diese bewährte Methode empfehlen.

- Durch die Aufnahme aller Abhängigkeiten als statische Dateien in das Container-Image wird die Anzahl potenziell gefährlicher Ereignisse reduziert, die während der Bereitstellung auftreten können. Wenn Sie auf Dutzende, Hunderte oder sogar Tausende von Kopien Ihres Containers skalieren, vereinfacht das Herunterladen eines einzelnen Container-Images, anstatt es von zwei oder drei verschiedenen Orten herunterzuladen, Ihre Arbeitslast, da potenzielle Schwachstellen begrenzt werden. Gehen Sie beispielsweise davon aus, dass Sie 100 Kopien Ihrer Anwendung bereitstellen und jede Kopie der Anwendung Teile aus drei verschiedenen Quellen herunterladen muss. Es gibt 300 Downloads, die fehlschlagen können. Wenn Sie ein Container-Image herunterladen, gibt es nur 100 Abhängigkeiten, die unterbrochen werden können.
- Container-Image-Downloads sind für das parallel Herunterladen der Anwendungsabhängigkeiten optimiert. Standardmäßig besteht ein Container-Image aus Ebenen, die parallel heruntergeladen

und entpackt werden können. Das bedeutet, dass mit einem Container-Image-Download all Ihre Abhängigkeiten schneller auf Ihren Computer übertragen werden können als mit einem handcodierten Skript, das jede Abhängigkeit nacheinander herunterlädt.

- Indem Sie all Ihre Abhängigkeiten innerhalb des Images behalten, sind Ihre Bereitstellungen zuverlässiger und reproduzierbarer. Wenn Sie eine dynamisch geladene Abhängigkeit ändern, kann dies dazu führen, dass die Anwendung im Container-Image beschädigt wird. Wenn der Container jedoch wirklich eigenständig ist, können Sie ihn jederzeit erneut bereitstellen, auch in future. Das liegt daran, dass er bereits die richtigen Versionen und Abhängigkeiten enthält.

Sorgen Sie für schnelle Startzeiten von Containern, indem Sie Container-Images so klein wie möglich halten

Vollständige Container enthalten alles, was für die Ausführung Ihrer Anwendung benötigt wird, aber sie müssen nicht Ihre Build-Tools enthalten. Betrachten Sie dieses Beispiel. Gehen Sie davon aus, dass Sie einen Container für eine Node.js Anwendung erstellen. Sie benötigen den NPM Paketmanager, um Pakete für Ihre Anwendung herunterzuladen. Sie benötigen ihn jedoch nicht mehr, NPM wenn die Anwendung ausgeführt wird. Sie können einen mehrstufigen Docker Build verwenden, um dieses Problem zu lösen.

Im Folgenden finden Sie ein Beispiel dafür, wie ein solcher mehrstufiger `Dockerfile` Vorgang für eine Node.js Anwendung aussehen könnte, in der Abhängigkeiten bestehen. NPM

```
FROM node:14 AS build
WORKDIR /srv
ADD package.json .
RUN npm install

FROM node:14-slim
COPY --from=build /srv .
ADD . .
EXPOSE 3000
CMD ["node", "index.js"]
```

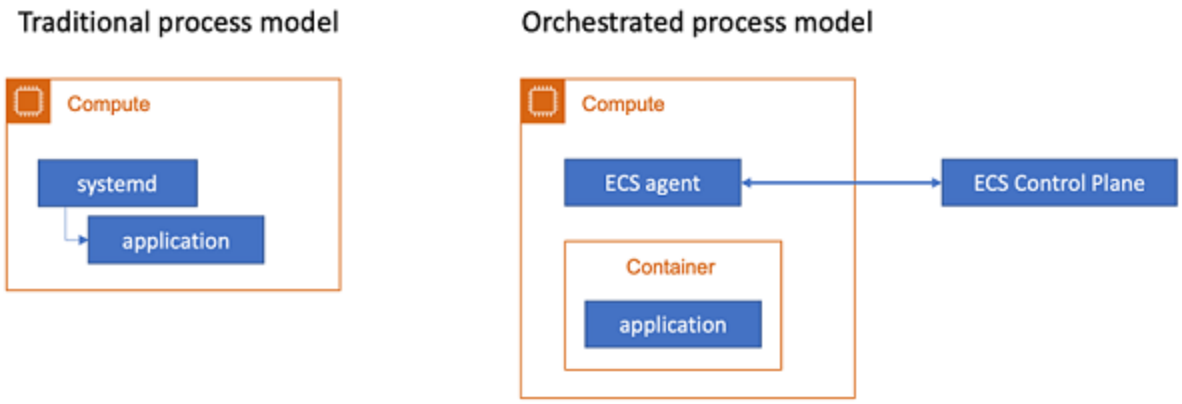
Die erste Phase verwendet eine vollständige Node.js Umgebung mit und einen Compiler zum Erstellen systemeigener Codebindungen für Pakete. NPM Die zweite Phase beinhaltet nichts als die Node.js Laufzeit. Es kann die heruntergeladenen NPM Pakete aus der ersten Phase kopieren. Das Endprodukt ist ein minimales Image, das die Node.js Laufzeit, die NPM Pakete und den Anwendungscode enthält. Es enthält nicht die vollständige NPM Build-Toolchain.

Halten Sie Ihre Container-Images so klein wie möglich und verwenden Sie gemeinsam genutzte Ebenen. Wenn Sie beispielsweise mehrere Anwendungen haben, die denselben Datensatz verwenden, können Sie ein gemeinsames Basisimage erstellen, das diesen Datensatz enthält. Erstellen Sie dann zwei verschiedene Image-Varianten auf der Grundlage desselben gemeinsamen Basis-Images. Dadurch kann die Container-Image-Ebene mit dem Datensatz einmal und nicht zweimal heruntergeladen werden.

Der Hauptvorteil der Verwendung kleinerer Container-Images besteht darin, dass diese Bilder schneller auf Computerhardware heruntergeladen werden können. Dadurch kann Ihre Anwendung schneller skaliert und nach unerwarteten Abstürzen oder Neustarts schnell wiederhergestellt werden.

Führen Sie nur einen einzigen Anwendungsprozess mit einem Container-Image aus

In einer herkömmlichen Umgebung mit virtuellen Maschinen ist es üblich, einen Daemon auf hoher Ebene `systemd` wie den Root-Prozess auszuführen. Dieser Daemon ist dann dafür verantwortlich, Ihren Anwendungsprozess zu starten und den Anwendungsprozess neu zu starten, falls er abstürzt. Wir empfehlen dies nicht bei der Verwendung von Containern. Führen Sie stattdessen nur einen einzigen Anwendungsprozess mit einem Container aus.



Wenn der Anwendungsprozess abstürzt oder beendet wird, wird auch der Container beendet. Wenn die Anwendung bei einem Absturz neu gestartet werden muss, lassen Sie Amazon ECS den Anwendungsneustart extern verwalten. Der Amazon ECS-Agent meldet der Amazon ECS-Steuerebene, dass der Anwendungscontainer abgestürzt ist. Anschließend bestimmt die Kontrollebene, ob ein Ersatzcontainer gestartet werden soll, und wenn ja, wo er gestartet werden soll. Der Ersatzcontainer kann auf demselben Host oder auf einem anderen Host im Cluster platziert werden.

Behandeln Sie Container als kurzlebige Ressourcen. Sie gelten nur für die Dauer des Hauptantragsverfahrens. Starten Sie Anwendungsprozesse nicht ständig innerhalb eines Containers neu, um zu versuchen, den Container am Laufen zu halten. Lassen Sie Amazon ECS Container nach Bedarf austauschen.

Diese bewährte Methode hat zwei wichtige Vorteile.

- Es mildert Szenarien, in denen eine Anwendung aufgrund einer Mutation im lokalen Container-Dateisystem abgestürzt ist. Anstatt dieselbe mutierte Container-Umgebung wiederzuverwenden, startet der Orchestrator einen neuen Container, der auf dem ursprünglichen Container-Image basiert. Das bedeutet, dass Sie sicher sein können, dass der Ersatzcontainer über eine saubere Basisumgebung verfügt.
- Abgestürzte Prozesse werden durch einen zentralen Entscheidungsprozess in der Amazon-ECS-Steuerebene ersetzt. Die Steuerebene trifft intelligentere Entscheidungen darüber, wo der Austauschprozess gestartet werden soll. Die Steuerungsebene kann beispielsweise versuchen, den Ersatz auf einer anderen Hardware in einer anderen Availability Zone zu starten. Dadurch ist die gesamte Bereitstellung robuster, als wenn jede einzelne Rechen-Instance versucht, ihre eigenen Prozesse lokal neu zu starten.

Handhaben von **`SIGTERM`** innerhalb der Anwendung

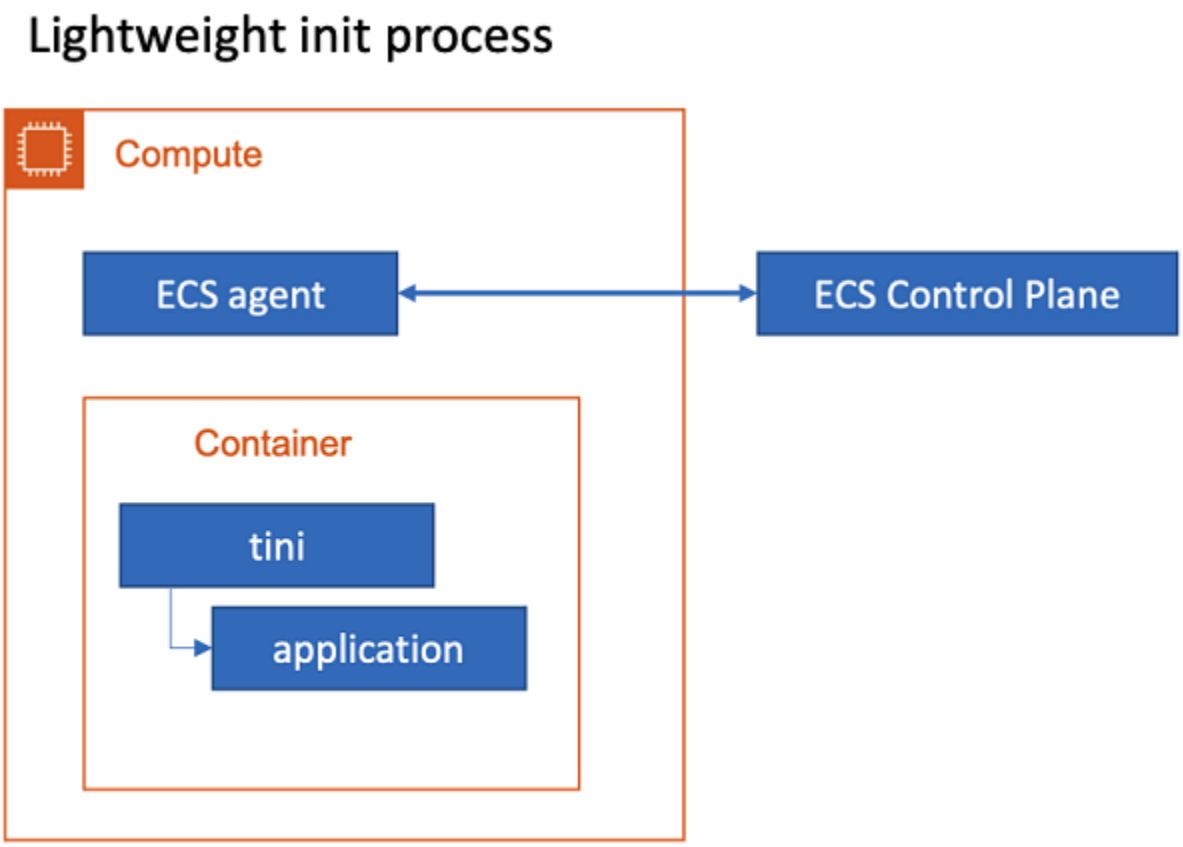
Wenn Sie die Anweisungen des vorherigen Abschnitts befolgen, erlauben Sie Amazon ECS, Aufgaben an anderer Stelle im Cluster zu ersetzen, anstatt die abstürzende Anwendung neu zu starten. In anderen Fällen kann es vorkommen, dass eine Aufgabe gestoppt wird, auf die die Anwendung keinen Einfluss hat. Aufgaben können aufgrund von Anwendungsfehlern, fehlgeschlagenen Integritätsprüfungen, Abschluss von Geschäftsabläufen oder sogar manueller Beendigung durch einen Benutzer gestoppt werden.

Wenn eine Aufgabe von ECS gestoppt wird, folgt ECS den unter beschriebenen Schritten und der KonfigurationReaktionsfähigkeit von SIGTERM.

Um Ihre Anwendung vorzubereiten, müssen Sie ermitteln, wie lange es dauert, bis Ihre Anwendung ihre Arbeit abgeschlossen hat, und sicherstellen, dass Ihre Anwendungen das `SIGTERM`-Signal verarbeiten. Im Rahmen der Signalverarbeitung der Anwendung müssen Sie verhindern, dass die Anwendung neue Aufgaben annimmt, und sicherstellen, dass die laufende Arbeit abgeschlossen oder unerledigte Arbeiten außerhalb der Aufgabe gespeichert werden, falls die Bearbeitung zu lange dauern würde.

Nach dem Senden des `SIGTERM` Signals wartet Amazon ECS auf die `StopTimeout` in der Aufgabendefinition angegebene Zeit. Dann wird das `SIGKILL` Signal gesendet. Stellen Sie den `StopTimeout` Wert so lang ein, dass Ihre Anwendung den `SIGTERM` Handler in allen Situationen abschließt, bevor der gesendet `SIGKILL` wird.

Bei Webanwendungen müssen Sie auch offene Verbindungen berücksichtigen, die sich im Leerlauf befinden. Weitere Informationen finden Sie auf der folgenden Seite dieses HandbuchsNetwork Load Balancer.



Wenn Sie in Ihrem Container einen Init-Prozess verwenden, verwenden Sie einen einfachen Init-Prozess wie. `tini` Dieser Init-Prozess übernimmt die Verantwortung für das Abrufen von Zombie-Prozessen, falls Ihre Anwendung Worker-Prozesse erzeugt. Wenn Ihre Anwendung das `SIGTERM` Signal nicht richtig verarbeitet, `tini` kann sie dieses Signal für Sie catch und Ihren Bewerbungsprozess beenden. Wenn Ihr Bewerbungsprozess jedoch abstürzt, `tini` wird er nicht neu gestartet. Wird stattdessen `tini` beendet, sodass der Container beendet und durch Container-Orchestrierung ersetzt werden kann. Weitere Informationen finden Sie unter `tini`. GitHub

Konfigurieren Sie containerisierte Anwendungen für das Schreiben von Protokollen in und `stdoutstderr`

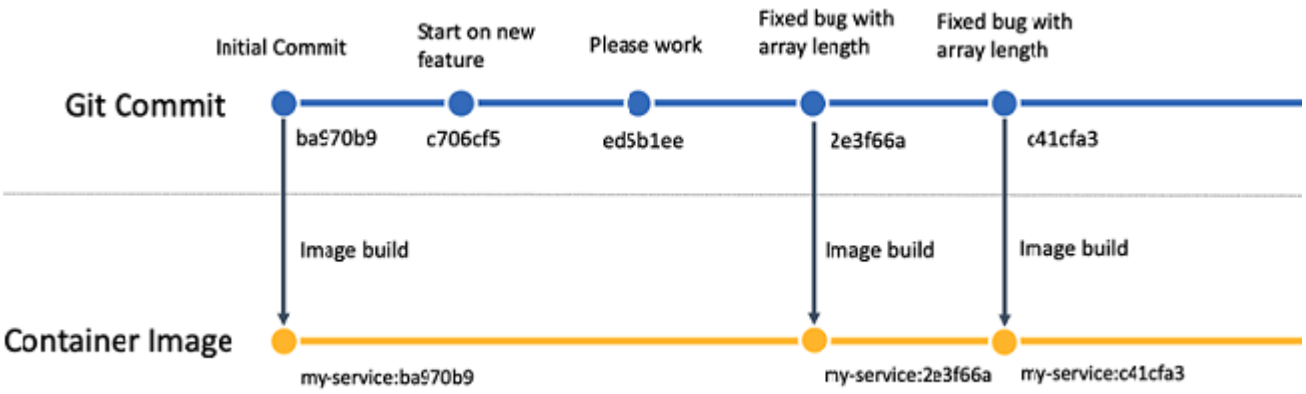
Es gibt viele verschiedene Möglichkeiten, die Protokollierung durchzuführen. Für einige Anwendungsframeworks ist es üblich, eine Anwendungsprotokollierungsbibliothek zu verwenden, die direkt auf Festplattendateien schreibt. Es ist auch üblich, eine zu verwenden, die Protokolle direkt in einen ELK (OpenSearch,Logstash,Kibana) -Stack oder eine ähnliche Protokollierungseinrichtung streamt. Wir empfehlen jedoch, eine containerisierte Anwendung so zu konfigurieren, dass Anwendungsprotokolle direkt in die `stdout` und `stderr` -Streams geschrieben werden.

Dockerumfasst eine Vielzahl von Protokollierungstreibern, die Datenströme aufnehmen `stdout` und `stderr` protokollieren und verarbeiten. Sie können wählen, ob Sie die Streams auf die lokale Instance`syslog`, auf der der Container ausgeführt wird, auf die Festplatte schreiben oder einen Protokollierungstreiber verwenden, um die Protokolle anFluentd, Splunk CloudWatch, und andere Ziele zu senden. Mit Amazon ECS können Sie wählen, ob Sie den FireLens Protokollierungstreiber konfigurieren möchten. Dieser Treiber kann Amazon ECS-Metadaten an Protokolle anhängen, Protokolle filtern und Protokolle anhand von Kriterien wie dem HTTP-Statuscode an verschiedene Ziele weiterleiten. Weitere Informationen zu Docker Protokollierungstreibern finden Sie unter Protokollierungstreiber konfigurieren. Weitere Informationen zu finden Sie FireLens unter Verwenden von FireLens.

Wenn Sie die Protokollverarbeitung von Ihrem Anwendungscode entkoppeln, erhalten Sie mehr Flexibilität bei der Anpassung der Protokollverarbeitung auf Infrastrukturebene. Gehen Sie davon aus, dass Sie von einem Protokollierungssystem zu einem anderen wechseln möchten. Sie können dies tun, indem Sie einige Einstellungen auf Container-Orchestrator-Ebene anpassen, anstatt den Code in all Ihren Services ändern, ein neues Container-Image erstellen und es bereitstellen zu müssen.

Container-Images mithilfe von Tags versionieren

Container-Images werden in einer Container-Registrierung gespeichert. Jedes Image in einer Registrierung wird durch ein Tag identifiziert. Es gibt ein Tag namens `latest`. Dieses Tag dient als Hinweis auf die neueste Version des Anwendungs-Container-Images, ähnlich wie `HEAD` in einem Git-Repository. Wir empfehlen, dass Sie das `latest`-Tag nur für Tests verwenden. Es hat sich bewährt, Container-Images mit einem eindeutigen Tag für jeden Build zu kennzeichnen. Wir empfehlen, dass Sie Ihre Images mit dem git-SHA für den Git-Commit taggen, der zum Erstellen des Images verwendet wurde.



Sie müssen nicht für jeden Commit ein Container-Image erstellen. Wir empfehlen jedoch, dass Sie jedes Mal, wenn Sie einen bestimmten Code-Commit für die Produktionsumgebung veröffentlichen, ein neues Container-Image erstellen. Wir empfehlen außerdem, das Image mit einem Tag zu versehen, das dem git-Commit des Codes entspricht, der sich im Image befindet. Wenn Sie das Image mit dem Tag des git-Commits versehen, können Sie schneller herausfinden, welche Version des Codes auf dem Image ausgeführt wird.

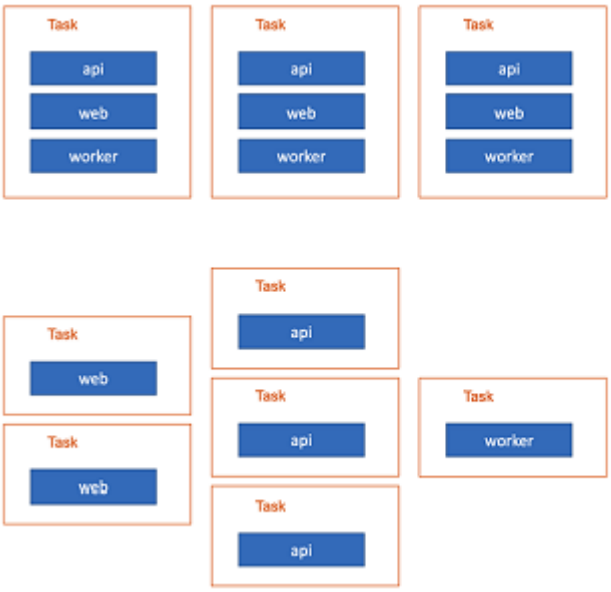
Wir empfehlen außerdem, dass Sie unveränderliche Image-Tags in der Amazon Elastic Container Registry aktivieren. Mit dieser Einstellung können Sie das Container-Image, auf das ein Tag verweist, nicht ändern. Stattdessen erzwingt Amazon ECR, dass ein neues Images zu einem neuen Tag hochgeladen werden muss, anstatt ein bereits vorhandenes Tag zu überschreiben. Weitere Informationen finden Sie unter Veränderlichkeit von Image-Tags im Amazon-ECR-Benutzerhandbuch.

Aufgabendefinition

Die Aufgabendefinition ist ein Dokument, das beschreibt, welche Container-Images zusammen ausgeführt werden sollen und welche Einstellungen bei der Ausführung der Container-Images zu verwenden sind. Zu diesen Einstellungen gehören die Menge an CPU und Arbeitsspeicher, die der Container benötigt. Sie enthalten auch alle Umgebungsvariablen, die dem Container zur Verfügung gestellt werden, und alle Datenvolumes, die in den Container eingebunden sind. Aufgabendefinitionen werden nach den Dimensionen Familie und Revision gruppiert.

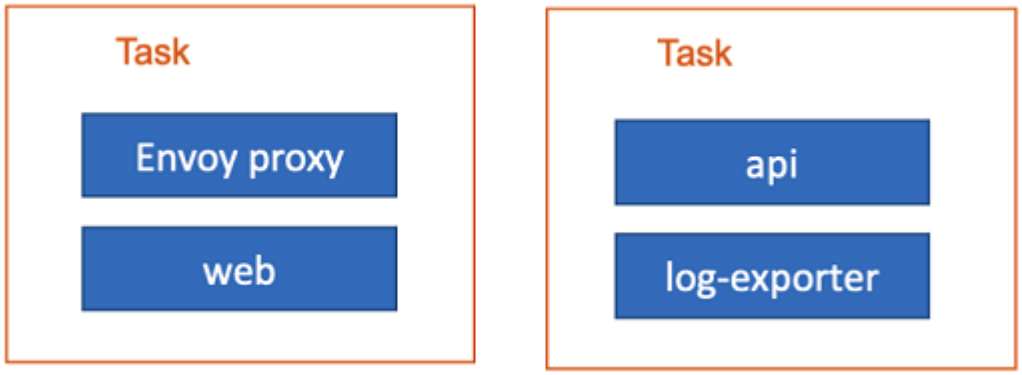
Verwenden Sie jede Aufgabendefinitionsfamilie nur für einen Geschäftszweck

Sie können eine Amazon ECS-Aufgabendefinition verwenden, um mehrere Container anzugeben. Alle von Ihnen angegebenen Container werden mit derselben Rechenkapazität bereitgestellt. Verwenden Sie diese Funktion nicht, um derselben Aufgabendefinition mehrere Anwendungscontainer hinzuzufügen, da dadurch verhindert wird, dass Kopien jeder Anwendung separat skaliert werden. Stellen Sie sich zum Beispiel diese Situation vor. Angenommen, Sie haben einen Webserver-Container, einen API-Container und einen Worker-Service-Container. Es hat sich bewährt, für jeden dieser containerisierten Codeteile eine eigene Aufgabendefinitionsfamilie zu verwenden.



Wenn Sie mehrere Typen von Anwendungscontainern in derselben Aufgabendefinition zusammenfassen, können Sie diese Container nicht unabhängig voneinander skalieren. Es ist beispielsweise unwahrscheinlich, dass sowohl eine Website als auch eine API mit derselben Geschwindigkeit aufskaliert werden müssen. Mit steigendem Datenverkehr wird eine andere Anzahl von Web-Containern benötigt als API-Container. Wenn diese beiden Container in derselben Aufgabendefinition bereitgestellt werden, führt jede Aufgabe dieselbe Anzahl von Webcontainern und API-Containern aus.

Wir empfehlen, jeden Containertyp je nach Bedarf unabhängig voneinander zu skalieren.



Es wird nicht empfohlen, mehrere Container in einer einzigen Aufgabendefinition zu verwenden, um verschiedene Typen von Anwendungscontainern zu gruppieren. Der Zweck, mehrere Container in einer einzigen Aufgabendefinition zu haben, besteht darin, dass Sie Sidecars bereitstellen können, kleine Zusatzcontainer, die einen einzelnen Containertyp erweitern. Ein Sidecar kann bei der Protokollierung und Beobachtbarkeit, beim Routing des Datenverkehrs oder bei anderen Zusatzfunktionen hilfreich sein.

Es wird empfohlen, Sidecars zu verwenden, um zusätzliche Funktionen hinzuzufügen, aber die Aufgabe sollte nur eine einzige Geschäftsfunktion haben.

Ordnen Sie jeder Anwendungsversion eine Revision der Aufgabendefinition innerhalb einer Aufgabendefinitionsfamilie zu

Eine Aufgabendefinition kann so konfiguriert werden, dass sie auf ein beliebiges Container-Image-Tag verweist, einschließlich des Tags „latest“. Wir empfehlen jedoch nicht, das Tag „latest“ in Ihrer Aufgabendefinition zu verwenden. Das liegt daran, dass das „latest“ -Tag als veränderbarer Zeiger fungiert, sodass sich der Inhalt des Bildes, auf das es zeigt, ändern kann, während Amazon ECS die Änderung nicht identifiziert.

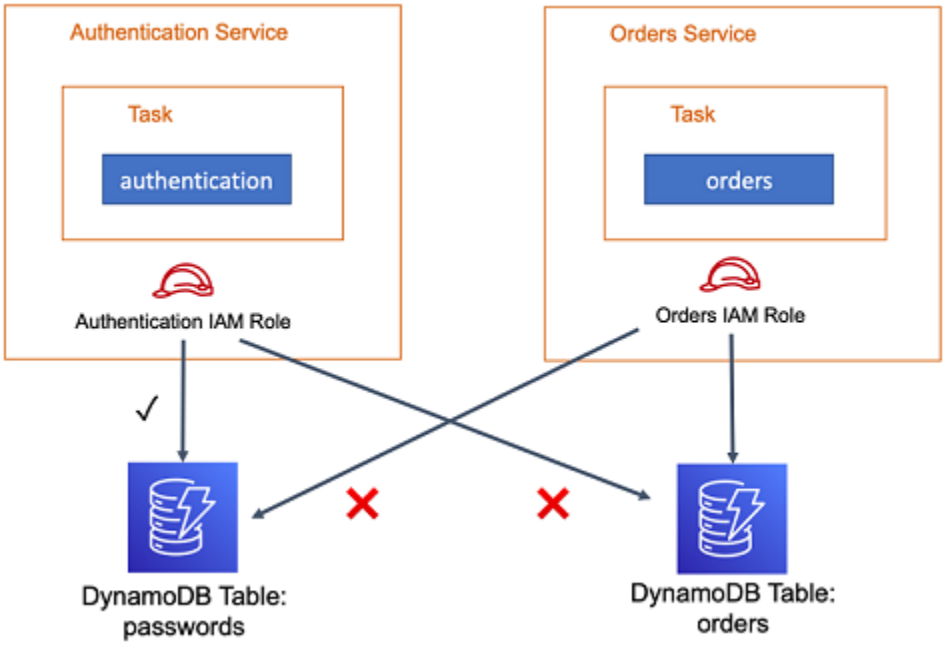
Betrachten Sie innerhalb einer Aufgabendefinitionsfamilie jede Version der Aufgabendefinition als eine Momentaufnahme der Einstellungen für ein bestimmtes Container-Image. Dies ist vergleichbar mit der Art und Weise, wie der Container eine Momentaufnahme aller Dinge ist, die zur Ausführung einer bestimmten Version Ihres Anwendungscodes erforderlich sind.

Stellen Sie sicher, dass eine one-to-one Zuordnung zwischen einer Version des Anwendungscodes, einem Container-Image-Tag und einer Revision der Aufgabendefinition besteht. Ein typischer Release-Prozess beinhaltet einen Git-Commit, der in ein Container-Image umgewandelt wird, das mit dem Git-Commit-SHA gekennzeichnet ist. Dann erhält dieses Container-Image-Tag seine eigene Version der Amazon-ECS-Aufgabendefinition. Zuletzt wird der Amazon-ECS-Service aktualisiert, um ihm mitzuteilen, dass er die neue Version der Aufgabendefinition bereitstellen soll.

Mit diesem Ansatz können Sie bei der Einführung neuer Versionen Ihrer Anwendung die Konsistenz zwischen Einstellungen und Anwendungscode wahren. Gehen Sie beispielsweise davon aus, dass Sie eine neue Version Ihrer Anwendung erstellen, die eine neue Umgebungsvariable verwendet. Die neue Aufgabendefinition, die dieser Änderung entspricht, definiert auch den Wert für die Umgebungsvariable.

Verwenden Sie für jede Aufgabendefinitionsfamilie unterschiedliche IAM-Rollen

Sie können verschiedene IAM-Rollen für verschiedene Aufgaben in Amazon ECS definieren. Verwenden Sie die Aufgabendefinition, um eine IAM-Rolle für diese Anwendung anzugeben. Wenn die Container in dieser Aufgabendefinition ausgeführt werden, können sie AWS APIs aufrufen, die auf den in der IAM-Rolle definierten Richtlinien basieren. Weitere Informationen finden Sie unter IAM-Rollen für Aufgaben.



Definieren Sie jede Aufgabendefinition mit einer eigenen IAM-Rolle. Diese Empfehlung sollte mit unserer Empfehlung einhergehen, jeder Geschäftskomponente eine eigene Aufgabendefinitionsfamilie zur Verfügung zu stellen. Durch die Implementierung dieser beiden bewährten Methoden können Sie einschränken, wie viel Zugriff jeder Dienst auf Ressourcen in Ihrem Konto hat. AWS Sie können Ihrem Authentifizierungsservice beispielsweise Zugriff gewähren, um eine Verbindung zu Ihrer Kennwortdatenbank herzustellen. Gleichzeitig können Sie auch sicherstellen, dass nur Ihr Bestellservice Zugriff auf die Kreditkartenzahlungsinformationen hat.

Amazon-ECS-Service

ECS verwendet die Serviceressource, um identische Aufgaben zu gruppieren, zu überwachen, zu ersetzen und zu skalieren. Die Serviceressource bestimmt, welche Aufgabendefinition und Revision Amazon ECS veröffentlicht. Sie bestimmt auch, wie viele Kopien der Aufgabendefinition

gestartet werden und welche Ressourcen mit den gestarteten Aufgaben verbunden sind. Zu diesen verbundenen Ressourcen gehören Load Balancer und Service Discovery. Die Dienstressource definiert auch Regeln für das Netzwerk und die Platzierung der Aufgaben auf der Hardware.

Verwenden Sie den **`awsvpc`** Netzwerkmodus und weisen Sie jedem Dienst eine eigene Sicherheitsgruppe zu

Wir empfehlen, den `awsvpc` Netzwerkmodus für Aufgaben in Amazon EC2 zu verwenden. Dadurch kann jede Aufgabe über eine eindeutige IP-Adresse mit einer Sicherheitsgruppe auf Service-Ebene verfügen. Dadurch werden Sicherheitsgruppenregeln pro Dienst anstelle von Sicherheitsgruppen auf Instanzebene erstellt, die in anderen Netzwerkmodi verwendet werden. Mithilfe von Sicherheitsgruppenregeln pro Service können Sie beispielsweise einen Service autorisieren, mit einer Amazon RDS-Datenbank zu kommunizieren. Einem anderen Dienst mit einer anderen Sicherheitsgruppe wird das Öffnen einer Verbindung zu dieser Amazon RDS-Datenbank verweigert.

Von Amazon ECS verwaltete Tags und Tag-Propagierung aktivieren

Nachdem Sie die von Amazon ECS verwalteten Tags und die Tag-Propagierung aktiviert haben, kann Amazon ECS Tags an die Aufgaben anhängen und weitergeben, die der Service startet. Sie können diese Tags anpassen und sie verwenden, um Tag-Dimensionen wie `environment=production` oder `team=web` oder `application=storefront` zu erstellen. Diese Tags werden in Nutzungs- und Abrechnungsberichten verwendet. Wenn Sie die Tags richtig eingerichtet haben, können Sie sie verwenden, um zu sehen, wie viele vCPU-Stunden oder GB-Stunden eine bestimmte Umgebung, ein bestimmtes Team oder eine bestimmte Anwendung verwendet hat. Dies kann Ihnen helfen, die Gesamtkosten Ihrer Infrastruktur anhand verschiedener Dimensionen abzuschätzen.

Bewährte Methoden — Netzwerke

Moderne Anwendungen bestehen in der Regel aus mehreren verteilten Komponenten, die miteinander kommunizieren. Beispielsweise kann eine Mobil- oder Webanwendung mit einem API-Endpunkt kommunizieren, und die API kann von mehreren Microservices betrieben werden, die über das Internet kommunizieren.

In diesem Leitfaden werden die bewährten Methoden für den Aufbau eines Netzwerks vorgestellt, in dem die Komponenten Ihrer Anwendung sicher und skalierbar miteinander kommunizieren können.

Themen

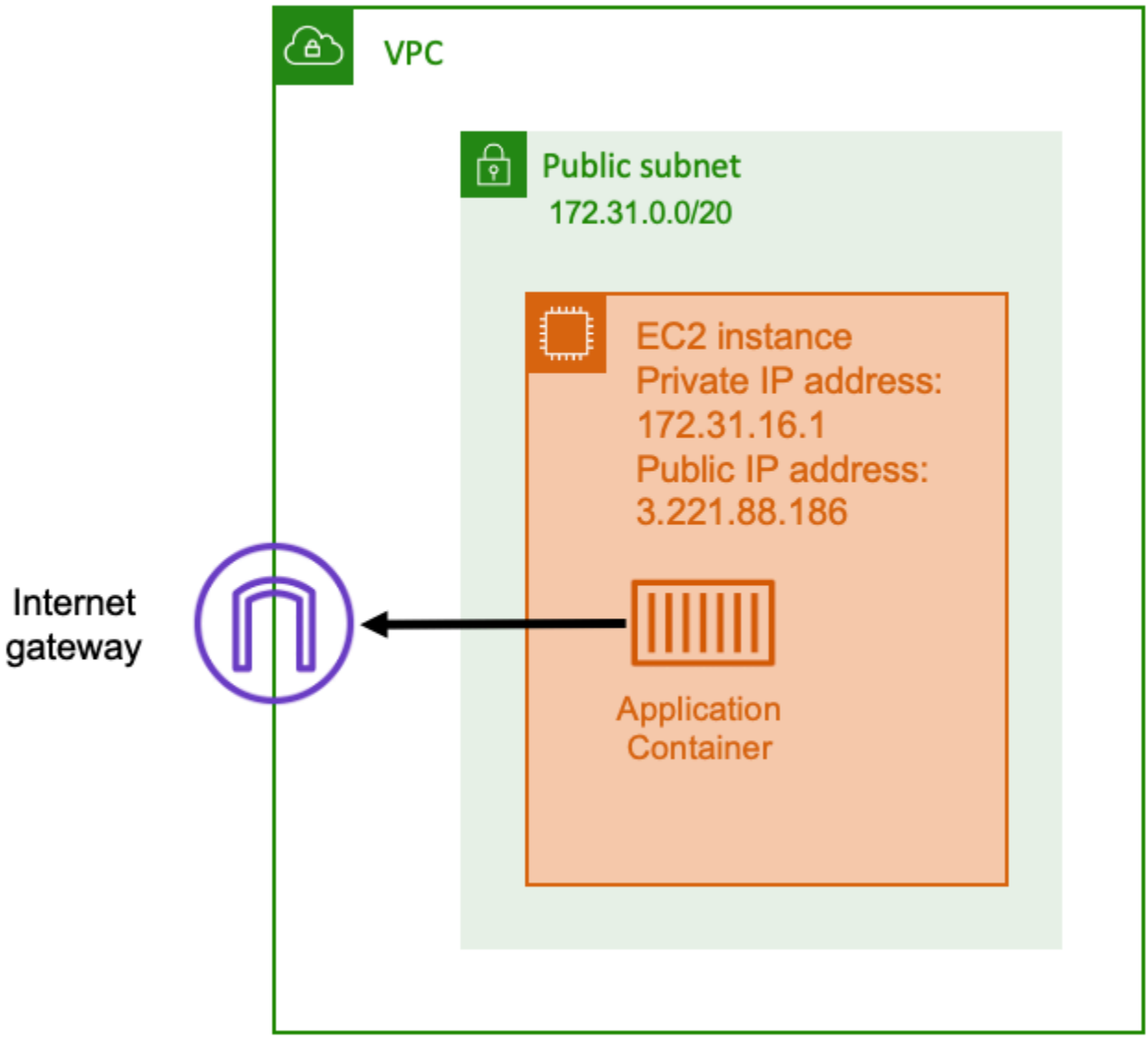
Verbindung zum Internet herstellen

Die meisten containerisierten Anwendungen haben zumindest einige Komponenten, die ausgehenden Zugriff auf das Internet benötigen. Beispielsweise erfordert das Backend für eine mobile App ausgehenden Zugriff auf Push-Benachrichtigungen.

Amazon Virtual Private Cloud bietet zwei Hauptmethoden zur Erleichterung der Kommunikation zwischen Ihrer VPC und dem Internet.

Themen

Verwenden eines öffentlichen Subnetzes und eines Internet-Gateways



Durch die Verwendung eines öffentlichen Subnetzes mit einer Route zu einem Internet-Gateway kann Ihre containerisierte Anwendung auf einem Host innerhalb einer VPC in einem öffentlichen Subnetz ausgeführt werden. Dem Host, der Ihren Container ausführt, wird eine öffentliche IP-Adresse zugewiesen. Diese öffentliche IP-Adresse kann vom Internet aus geroutet werden. Weitere Informationen finden Sie unter Internet-Gateways im Amazon-VPC-Benutzerhandbuch.

Diese Netzwerkarchitektur ermöglicht die direkte Kommunikation zwischen dem Host, auf dem Ihre Anwendung ausgeführt wird, und anderen Hosts im Internet. Die Kommunikation ist bidirektional. Das bedeutet, dass Sie nicht nur eine ausgehende Verbindung zu einem anderen Host im Internet

herstellen können, sondern dass auch andere Hosts im Internet versuchen können, eine Verbindung zu Ihrem Host herzustellen. Daher sollten Sie Ihre Sicherheitsgruppen- und Firewallregeln genau beachten. Dadurch wird sichergestellt, dass andere Hosts im Internet keine Verbindungen öffnen können, von denen Sie nicht möchten, dass sie geöffnet werden.

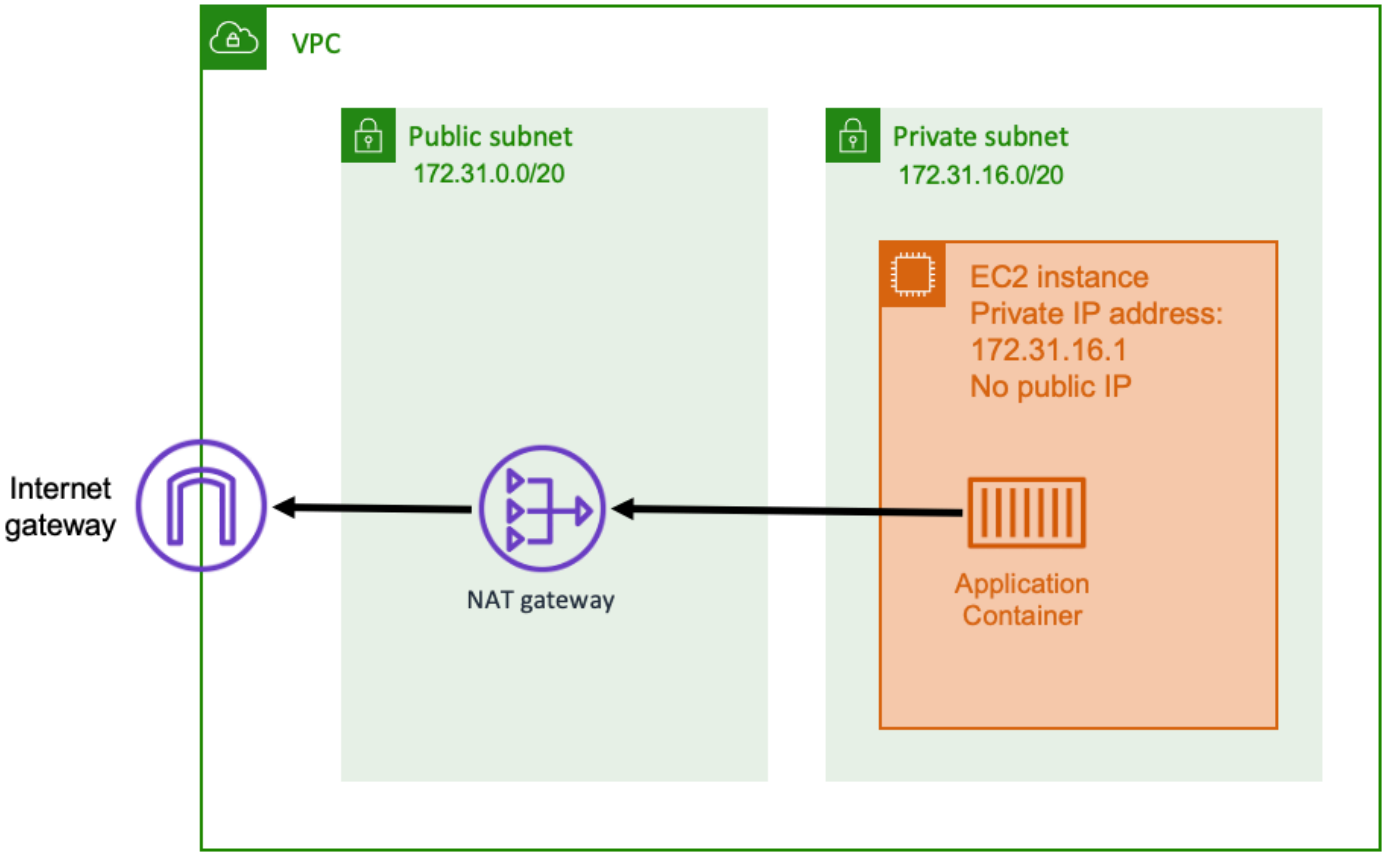
Wenn Ihre Anwendung beispielsweise auf Amazon EC2 läuft, stellen Sie sicher, dass Port 22 für den SSH-Zugriff nicht geöffnet ist. Andernfalls könnte Ihre Instance ständig SSH-Verbindungsversuche von bösartigen Bots im Internet erhalten. Diese Bots durchsuchen öffentliche IP-Adressen. Nachdem sie einen offenen SSH-Port gefunden haben, versuchen sie, mit Brute-Force-Passwörtern auf Ihre Instance zuzugreifen. Aus diesem Grund schränken viele Organisationen die Nutzung öffentlicher Subnetze ein und ziehen es vor, die meisten, wenn nicht sogar alle Ressourcen in privaten Subnetzen zu haben.

Die Verwendung öffentlicher Subnetze für Netzwerke eignet sich für öffentliche Anwendungen, die viel Bandbreite oder minimale Latenz erfordern. Zu den anwendbaren Anwendungsfällen gehören Videostreaming- und Spieledienste.

Dieser Netzwerkansatz wird sowohl bei der Verwendung von Amazon ECS auf Amazon EC2 als auch bei der Verwendung auf AWS Fargate unterstützt.

- Verwenden von Amazon EC2 — Sie können EC2-Instances in einem öffentlichen Subnetz starten. Amazon ECS verwendet diese EC2-Instances als Cluster-Kapazität, und alle Container, die auf den Instances ausgeführt werden, können die zugrunde liegende öffentliche IP-Adresse des Hosts für ausgehende Netzwerke verwenden. Dies gilt sowohl für den als auch für den `host bridge` Netzwerkmodus. Der `awsvpc` Netzwerkmodus stellt jedoch keine öffentlichen IP-Adressen für Task-ENIs bereit. Daher können sie ein Internet-Gateway nicht direkt nutzen.
- Fargate verwenden — Wenn Sie Ihren Amazon ECS-Service erstellen, geben Sie öffentliche Subnetze für die Netzwerkkonfiguration Ihres Service an und stellen Sie sicher, dass die Option Öffentliche IP-Adresse zuweisen aktiviert ist. Jede Fargate-Aufgabe ist im öffentlichen Subnetz vernetzt und verfügt über eine eigene öffentliche IP-Adresse für die direkte Kommunikation mit dem Internet.

Verwendung eines privaten Subnetzes und eines NAT-Gateways



Durch die Verwendung eines privaten Subnetzes und eines NAT-Gateways können Sie Ihre containerisierte Anwendung auf einem Host ausführen, der sich in einem privaten Subnetz befindet. Daher hat dieser Host eine private IP-Adresse, die innerhalb Ihrer VPC routbar ist, aber nicht vom Internet aus routbar ist. Das bedeutet, dass andere Hosts innerhalb der VPC über ihre private IP-Adresse Verbindungen zum Host herstellen können, andere Hosts im Internet jedoch keine eingehende Kommunikation mit dem Host herstellen können.

Bei einem privaten Subnetz können Sie ein NAT-Gateway (Network Address Translation) verwenden, um einem Host in einem privaten Subnetz die Verbindung zum Internet zu ermöglichen. Hosts im Internet erhalten eine eingehende Verbindung, die anscheinend von der öffentlichen IP-Adresse des NAT-Gateways stammt, das sich in einem öffentlichen Subnetz befindet. Das NAT-Gateway dient als Brücke zwischen dem Internet und der privaten VPC. Diese Konfiguration wird häufig aus Sicherheitsgründen bevorzugt, da sie bedeutet, dass Ihre VPC vor direktem Zugriff durch Angreifer im Internet geschützt ist. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch.

Dieser private Netzwerkansatz eignet sich für Szenarien, in denen Sie Ihre Container vor direktem externen Zugriff schützen möchten. Zu den anwendbaren Szenarien gehören Zahlungsabwicklungssysteme oder Container, in denen Benutzerdaten und Passwörter gespeichert werden. Für das Erstellen und Betreiben eines NAT-Gateways in Ihrem Konto fallen entsprechende Gebühren an. Es gelten auch die stündlichen Nutzungs- und Datenverarbeitungsgebühren für das NAT-Gateway. Aus Redundanzgründen sollten Sie in jeder Availability Zone über ein NAT-Gateway verfügen. Auf diese Weise beeinträchtigt der Verlust der Verfügbarkeit einer einzelnen Availability Zone Ihre ausgehende Konnektivität nicht. Aus diesem Grund kann es bei einer geringen Arbeitslast kostengünstiger sein, private Subnetze und NAT-Gateways zu verwenden.

Dieser Netzwerkansatz wird sowohl bei der Verwendung von Amazon ECS auf Amazon EC2 als auch bei der Verwendung auf AWS Fargate unterstützt.

- Verwenden von Amazon EC2 — Sie können EC2-Instances in einem privaten Subnetz starten. Die Container, die auf diesen EC2-Hosts ausgeführt werden, verwenden das zugrunde liegende Host-Netzwerk, und ausgehende Anfragen werden über das NAT-Gateway abgewickelt.
- Fargate verwenden — Wenn Sie Ihren Amazon ECS-Service erstellen, geben Sie private Subnetze für die Netzwerkkonfiguration Ihres Service an und aktivieren Sie nicht die Option Öffentliche IP-Adresse zuweisen. Jede Fargate-Aufgabe wird in einem privaten Subnetz gehostet. Der ausgehende Datenverkehr wird über jedes NAT-Gateway geleitet, das Sie mit diesem privaten Subnetz verknüpft haben.

Empfangen eingehender Verbindungen aus dem Internet

Wenn Sie einen öffentlichen Dienst betreiben, müssen Sie eingehenden Datenverkehr aus dem Internet akzeptieren. Ihre öffentliche Website muss beispielsweise eingehende HTTP-Anfragen von Browsern akzeptieren. In einem solchen Fall müssen auch andere Hosts im Internet eine eingehende Verbindung zum Host Ihrer Anwendung herstellen.

Ein Ansatz zur Lösung dieses Problems besteht darin, Ihre Container auf Hosts zu starten, die sich in einem öffentlichen Subnetz mit einer öffentlichen IP-Adresse befinden. Wir empfehlen dies jedoch nicht für umfangreiche Anwendungen. Für diese ist eine skalierbare Eingabeschicht, die sich zwischen dem Internet und Ihrer Anwendung befindet, ein besserer Ansatz. Für diesen Ansatz können Sie jeden der in diesem Abschnitt aufgeführten AWS Dienste als Eingabe verwenden.

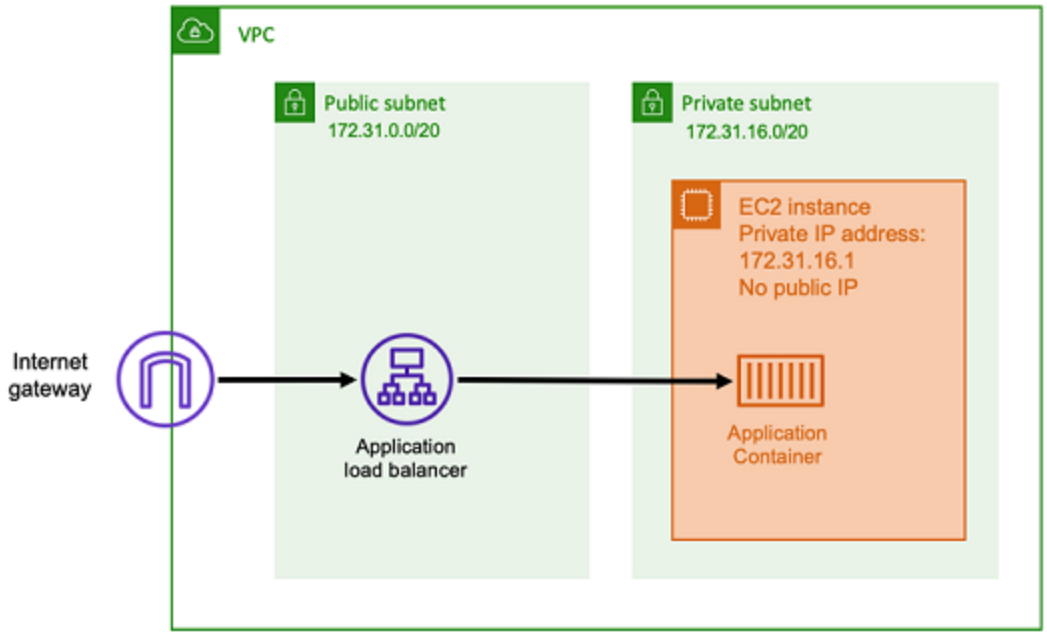
Themen

- Application Load Balancer

- Network Load Balancer
- Amazon API Gateway API-Gateway-HTTP-API

Application Load Balancer

Ein Application Load Balancer funktioniert auf der Anwendungsebene. Es ist die siebte Schicht des Open Systems Interconnection (OSI) -Modells. Dadurch eignet sich ein Application Load Balancer für öffentliche HTTP-Dienste. Wenn Sie über eine Website oder eine HTTP-REST-API verfügen, ist ein Application Load Balancer ein geeigneter Load Balancer für diesen Workload. Weitere Informationen finden Sie unter Was ist ein Application Load Balancer? im Benutzerhandbuch für Application Load Balancers.



Mit dieser Architektur erstellen Sie einen Application Load Balancer in einem öffentlichen Subnetz, sodass er über eine öffentliche IP-Adresse verfügt und eingehende Verbindungen aus dem Internet empfangen kann. Wenn der Application Load Balancer eine eingehende Verbindung, genauer gesagt eine HTTP-Anfrage, empfängt, öffnet er über seine private IP-Adresse eine Verbindung zur Anwendung. Anschließend leitet er die Anfrage über die interne Verbindung weiter.

Ein Application Load Balancer bietet die folgenden Vorteile.

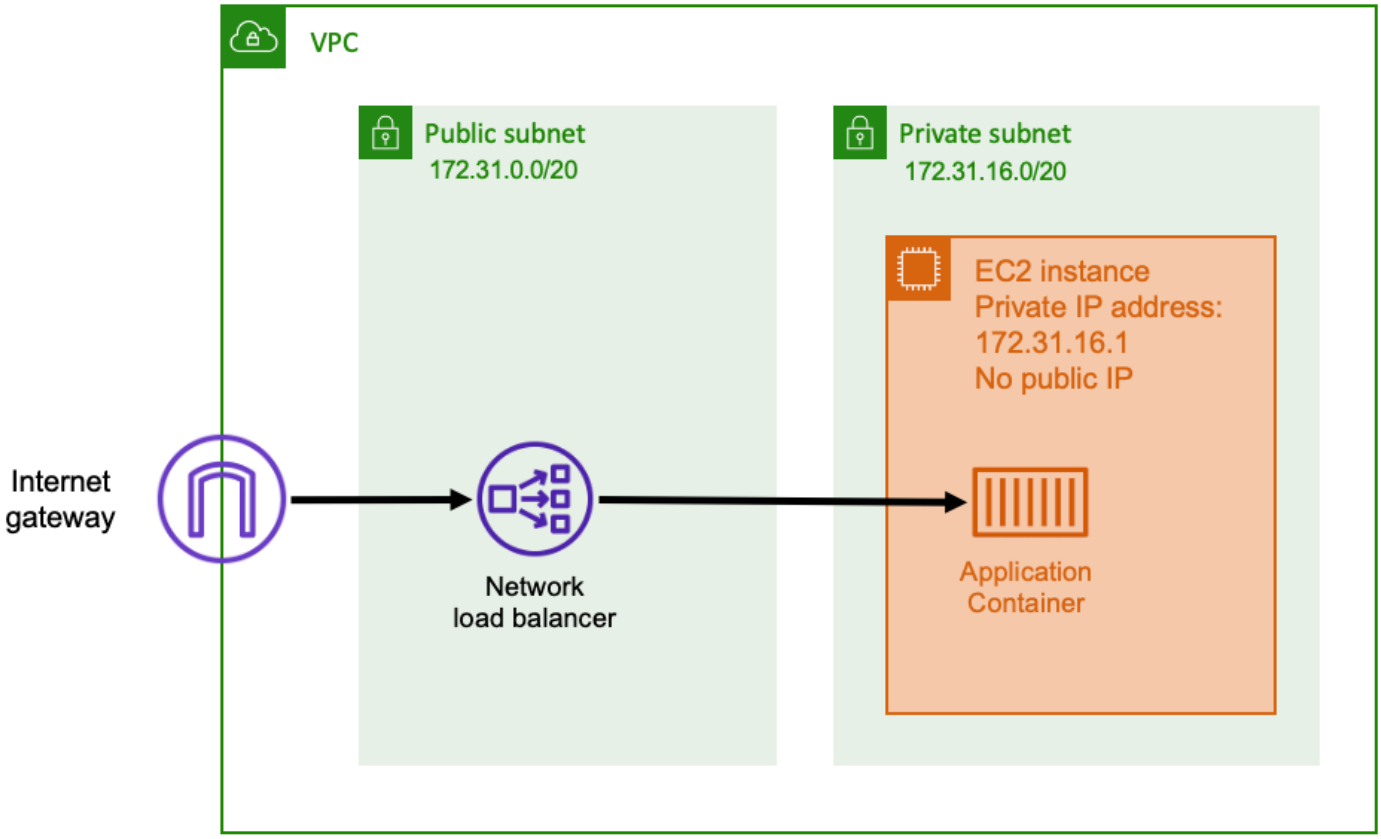
- SSL/TLS-Terminierung — Ein Application Load Balancer kann sichere HTTPS-Kommunikation und Zertifikate für die Kommunikation mit Clients aufrechterhalten. Er kann optional die SSL-

Verbindung auf Load Balancer-Ebene beenden, sodass Sie Zertifikate nicht in Ihrer eigenen Anwendung verwalten müssen.

- Erweitertes Routing — Ein Application Load Balancer kann mehrere DNS-Hostnamen haben. Er verfügt außerdem über erweiterte Routing-Funktionen, mit denen eingehende HTTP-Anfragen anhand von Metriken wie dem Hostnamen oder dem Pfad der Anfrage an verschiedene Ziele gesendet werden können. Das bedeutet, dass Sie einen einzelnen Application Load Balancer als Eingabe für viele verschiedene interne Dienste oder sogar Microservices auf verschiedenen Pfaden einer REST-API verwenden können.
- gRPC-Unterstützung und Websockets — Ein Application Load Balancer kann mehr als nur HTTP verarbeiten. Es kann auch den Lastausgleich von gRPC- und Websocket-basierten Diensten mit HTTP/2-Unterstützung durchführen.
- Sicherheit — Ein Application Load Balancer schützt Ihre Anwendung vor bösartigem Datenverkehr. Er umfasst Funktionen wie Abwehr von HTTP-Synchronisierungen und ist in die AWS Web Application Firewall ()AWS WAF integriert. AWS WAF kann außerdem bösartigen Datenverkehr herausfiltern, der Angriffsmuster wie SQL-Injection oder Cross-Site-Scripting enthalten könnte.

Network Load Balancer

Ein Network Load Balancer arbeitet auf der vierten Ebene der OSI-Modells (Open Systems Interconnection). Es eignet sich für Nicht-HTTP-Protokolle oder Szenarien, in denen end-to-end Verschlüsselung erforderlich ist, verfügt aber nicht über dieselben HTTP-spezifischen Funktionen wie ein Application Load Balancer. Daher eignet sich ein Network Load Balancer am besten für Anwendungen, die kein HTTP verwenden. Weitere Informationen finden Sie unter Was ist ein Network Load Balancer? im Benutzerhandbuch für Network Load Balancers.



Wenn ein Network Load Balancer als Eingabe verwendet wird, funktioniert er ähnlich wie ein Application Load Balancer. Das liegt daran, dass er in einem öffentlichen Subnetz erstellt wurde und über eine öffentliche IP-Adresse verfügt, auf die über das Internet zugegriffen werden kann. Der Network Load Balancer öffnet dann eine Verbindung zur privaten IP-Adresse des Hosts, auf dem Ihr Container läuft, und sendet die Pakete von der öffentlichen Seite an die private Seite.

Funktionen des Network Load Balancer

Da der Network Load Balancer auf einer niedrigeren Ebene des Netzwerkstapels arbeitet, verfügt er nicht über dieselben Funktionen wie Application Load Balancer. Er verfügt jedoch über die folgenden wichtigen Funktionen.

- nd-to-end E-Verschlüsselung — Da ein Network Load Balancer auf der vierten Ebene des OSI-Modells arbeitet, liest er den Inhalt von Paketen nicht. Dadurch eignet er sich für den Lastenausgleich von Kommunikationen, die end-to-end verschlüsselt werden müssen.
- TLS-Verschlüsselung — Zusätzlich zur end-to-end Verschlüsselung kann Network Load Balancer auch TLS-Verbindungen beenden. Auf diese Weise müssen Ihre Backend-Anwendungen kein eigenes TLS implementieren.

- UDP-Unterstützung — Da ein Network Load Balancer auf der vierten Ebene des OSI-Modells arbeitet, eignet er sich für Nicht-HTTP-Workloads und andere Protokolle als TCP.

Verbindungen schließen

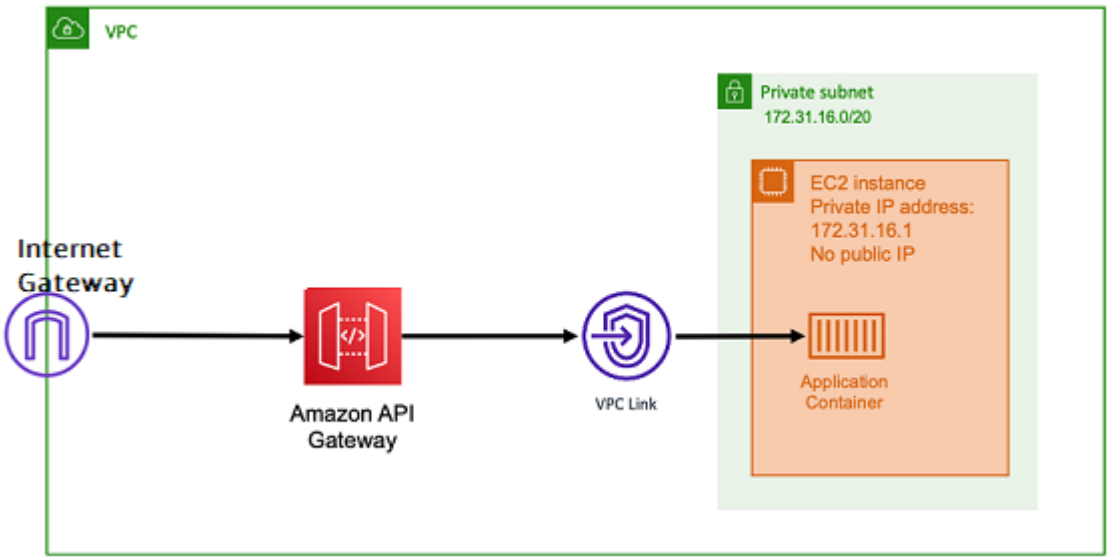
Da der Network Load Balancer das Anwendungsprotokoll auf den höheren Schichten des OSI-Modells nicht beobachtet, kann er in diesen Protokollen keine Abschlussnachrichten an die Clients senden. Im Gegensatz zum Application Load Balancer müssen diese Verbindungen von der Anwendung geschlossen werden. Sie können den Network Load Balancer auch so konfigurieren, dass er die Verbindungen der vierten Ebene schließt, wenn eine Aufgabe gestoppt oder ersetzt wird. Informationen zur Einstellung des Verbindungsabbruchs für Network Load Balancer Balancer-Zielgruppen finden Sie in der Network Load Balancer Balancer-Dokumentation.

Wenn der Network Load Balancer Verbindungen auf der vierten Ebene schließt, kann dies dazu führen, dass Clients unerwünschte Fehlermeldungen anzeigen, wenn der Client sie nicht verarbeitet. Weitere Informationen zur empfohlenen Clientkonfiguration finden Sie hier in der Entwicklerbibliothek.

Die Methoden zum Schließen von Verbindungen variieren je nach Anwendung. Eine Möglichkeit besteht jedoch darin, sicherzustellen, dass die Verzögerung bei der Deregistrierung des Network Load Balancer Balancer-Ziels länger ist als das Verbindungstimeout des Clients. Der Client würde zuerst das Timeout überschreiten und über den Network Load Balancer die Verbindung zur nächsten Aufgabe wieder herstellen, während die alte Aufgabe langsam alle Clients leert. Weitere Informationen zur Verzögerung der Network Load Balancer Balancer-Zielabmeldung finden Sie in der Network Load Balancer Balancer-Dokumentation.

Amazon API Gateway API-Gateway-HTTP-API

Die Amazon API Gateway HTTP API ist ein serverloser Eingang, der sich für HTTP-Anwendungen mit plötzlichen Anforderungsmengen oder niedrigen Anforderungsvolumina eignet. Weitere Informationen finden Sie unter Was ist Amazon API Gateway? im API Gateway Developer Guide.



Das Preismodell für Application Load Balancer und Network Load Balancer beinhaltet einen Stundenpreis, damit die Load Balancer jederzeit für die Annahme eingehender Verbindungen verfügbar sind. Im Gegensatz dazu berechnet API Gateway für jede Anfrage separat. Dies hat zur Folge, dass keine Gebühren anfallen, wenn keine Anfragen eingehen. Bei hoher Traffic-Auslastung kann ein Application Load Balancer oder Network Load Balancer ein größeres Anforderungsvolumen zu einem günstigeren Preis pro Anfrage verarbeiten als API Gateway. Wenn Sie jedoch insgesamt nur eine geringe Anzahl von Anfragen oder Zeiten mit geringem Traffic haben, sollte der Gesamtpreis für die Nutzung des API Gateway kostengünstiger sein als die Zahlung einer Stundengebühr für die Wartung eines Load Balancers, der nicht ausgelastet ist. Das API Gateway kann auch API-Antworten zwischenspeichern, was zu niedrigeren Backend-Anforderungsraten führen kann.

API-Gateway-Funktionen, die einen VPC-Link verwenden, der AWS es dem verwalteten Dienst ermöglicht, mithilfe seiner privaten IP-Adresse eine Verbindung zu Hosts innerhalb des privaten Subnetzes Ihrer VPC herzustellen. Es kann diese privaten IP-Adressen anhand von AWS Cloud Map Service Discovery-Datensätzen erkennen, die von Amazon ECS Service Discovery verwaltet werden.

API Gateway unterstützt die folgenden Funktionen.

- Der API-Gateway-Vorgang ähnelt einem Load Balancer, verfügt jedoch über zusätzliche Funktionen, die nur für das API-Management verfügbar sind
- Das API Gateway bietet zusätzliche Funktionen in Bezug auf Client-Autorisierung, Nutzungsstufen und Änderung von Anforderungen/Antworten. Weitere Informationen finden Sie unter Amazon API Gateway Gateway-Funktionen.

- Das API Gateway kann Edge-API-Gateway-Endpunkte, regionale und private API-Gateway-Endpunkte unterstützen. Edge-Endpunkte sind über eine verwaltete CloudFront Distribution verfügbar. Sowohl regionale als auch private Endpunkte befinden sich lokal in einer Region.
- SSL/TLS-Terminierung
- Routing verschiedener HTTP-Pfade zu verschiedenen Backend-Microservices

Neben den oben genannten Funktionen unterstützt API Gateway auch die Verwendung benutzerdefinierter Lambda-Autorisierer, mit denen Sie Ihre API vor unbefugter Nutzung schützen können. Weitere Informationen finden Sie unter Feldnotizen: Serverlose Container-basierte APIs mit Amazon ECS und Amazon API Gateway.

Einen Netzwerkmodus wählen

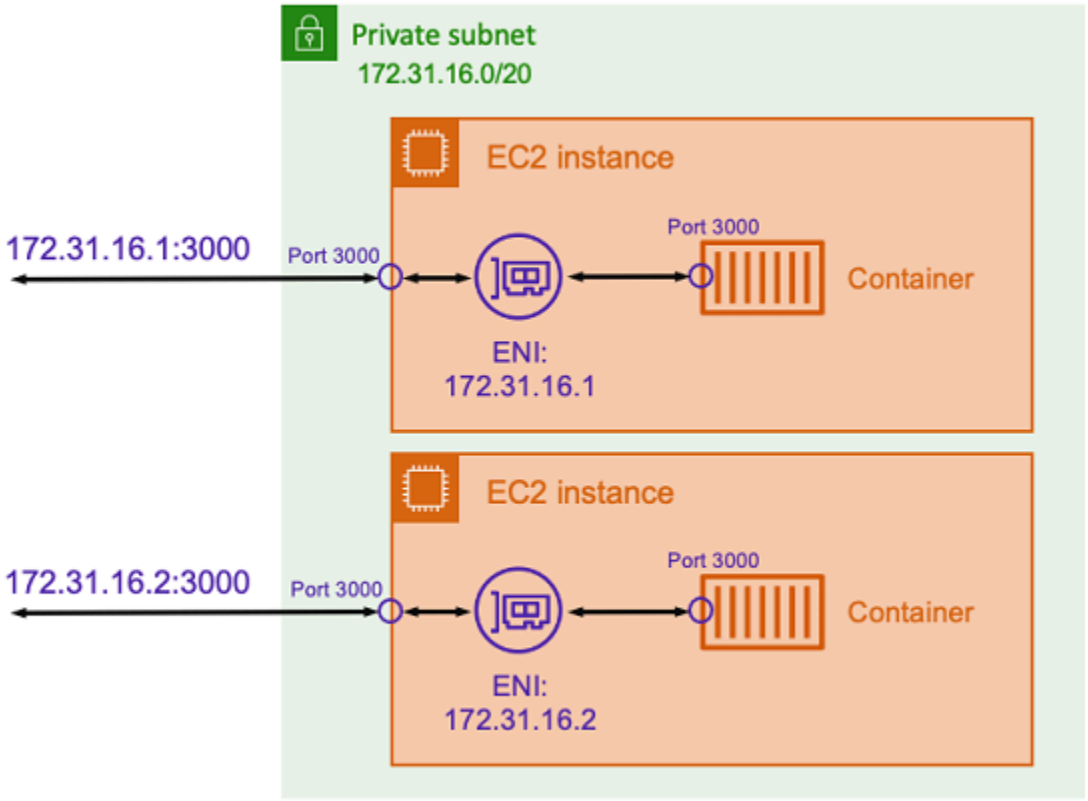
Die zuvor genannten Ansätze für die Architektur eingehender und ausgehender Netzwerkverbindungen können auf alle Ihre Workloads angewendet werden AWS, auch wenn sie sich nicht in einem Container befinden. Wenn Sie Container darauf ausführen AWS, müssen Sie eine andere Netzwerkebene in Betracht ziehen. Einer der Hauptvorteile der Verwendung von Containern besteht darin, dass Sie mehrere Container auf einem einzigen Host packen können. Dabei müssen Sie auswählen, wie Sie die Container, die auf demselben Host laufen, vernetzen möchten. Im Folgenden sind die Optionen aufgeführt, aus denen Sie wählen können.

- the section called “Hostmodus”- Der `host` Netzwerkmodus ist der grundlegendste Netzwerkmodus, der in Amazon ECS unterstützt wird.
- the section called “Bridge-Modus”- Im `bridge` Netzwerkmodus können Sie eine virtuelle Netzwerkbrücke verwenden, um eine Ebene zwischen dem Host und dem Netzwerk des Containers zu erstellen.

- the section called “AWSVPC Modus”- Im `awsvpc` Netzwerkmodus erstellt und verwaltet Amazon ECS für jede Aufgabe ein Elastic Network Interface (ENI), und jede Aufgabe erhält ihre eigene private IP-Adresse innerhalb der VPC.

Hostmodus

Der `host`-Netzwerkmodus ist der einfachste Netzwerkmodus, der in Amazon ECS unterstützt wird. Im Host-Modus ist das Netzwerk des Containers direkt an den zugrunde liegenden Host gebunden, auf dem der Container ausgeführt wird.



Gehen Sie davon aus, dass Sie einen Node.js-Container mit einer Express-Anwendung ausführen, die Listener auf einem `3000`-Port ist, der dem im vorherigen Diagramm dargestellten ähnelt. Wenn der `host`-Netzwerkmodus verwendet wird, empfängt der Container Datenverkehr auf Port 3000 unter Verwendung der IP-Adresse der zugrunde liegenden Host-Amazon-EC2-Instance. Wir raten davon ab, diesen Modus zu verwenden.

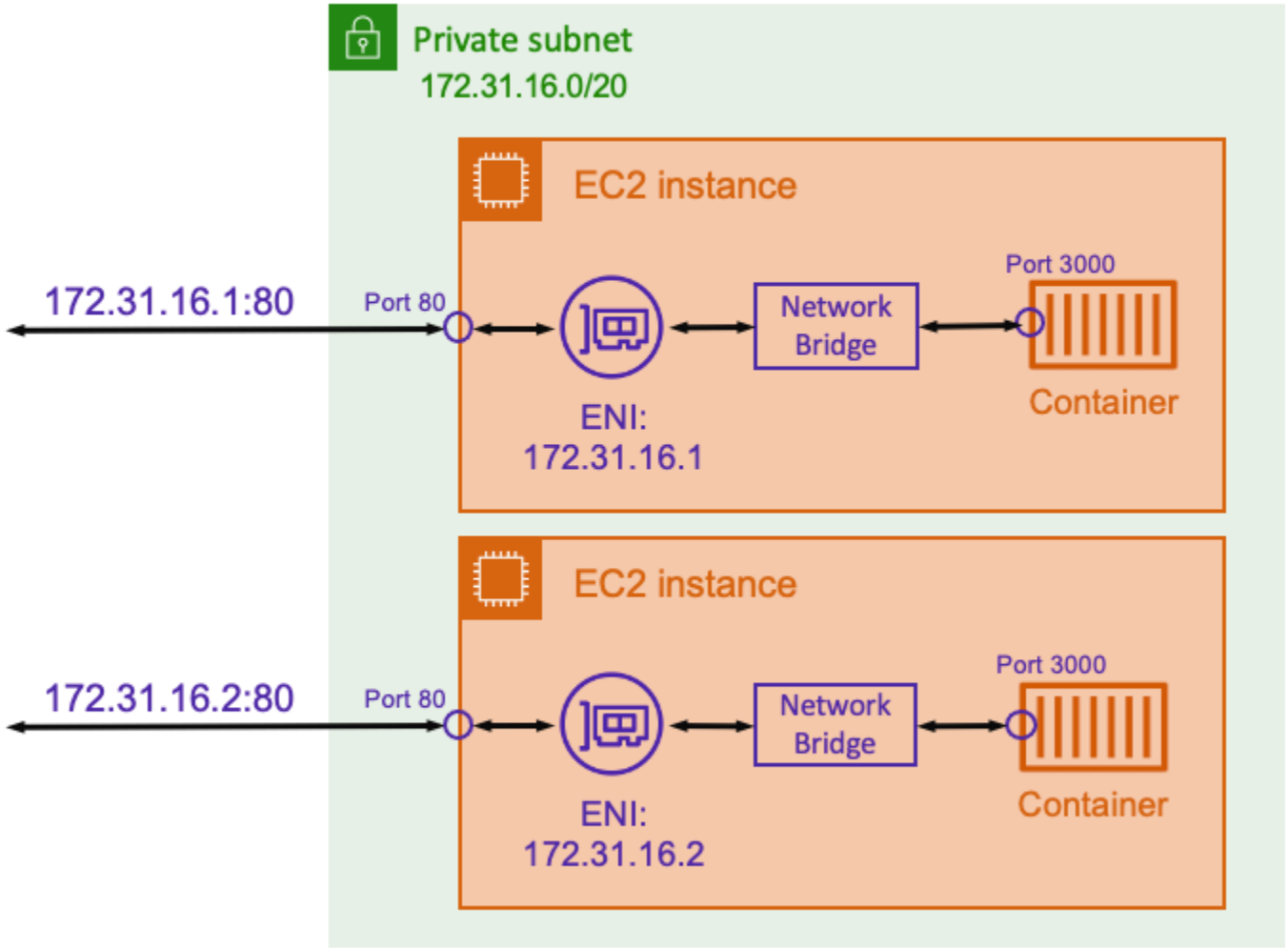
Die Verwendung dieses Netzwerkmodus hat erhebliche Nachteile. Sie können auf jedem Host nur eine einzige Instanziierung eines Tasks ausführen. Dies liegt daran, dass nur die erste Aufgabe an den erforderlichen Port auf der Amazon-EC2-Instance gebunden werden kann. Es gibt auch keine Möglichkeit, einen Container-Port neu zuzuordnen, wenn er den `host`-Netzwerkmodus verwendet. Wenn eine Anwendung beispielsweise Listener auf einer bestimmten Portnummer sein muss, können Sie die Portnummer nicht direkt neu zuordnen. Stattdessen müssen Sie alle Portkonflikte lösen, indem Sie die Anwendungskonfiguration ändern.

Die Verwendung des `host`-Netzwerkmodus hat auch Auswirkungen auf die Sicherheit. In diesem Modus können Container die Identität des Hosts annehmen und Container können sich mit privaten Loopback-Netzwerkservices auf dem Host verbinden.

Der `host`-Netzwerkmodus wird nur für Amazon-ECS-Aufgaben unterstützt, die auf Amazon-EC2-Instances gehostet werden. Bei der Verwendung von Amazon ECS auf Fargate wird dies nicht unterstützt.

Bridge-Modus

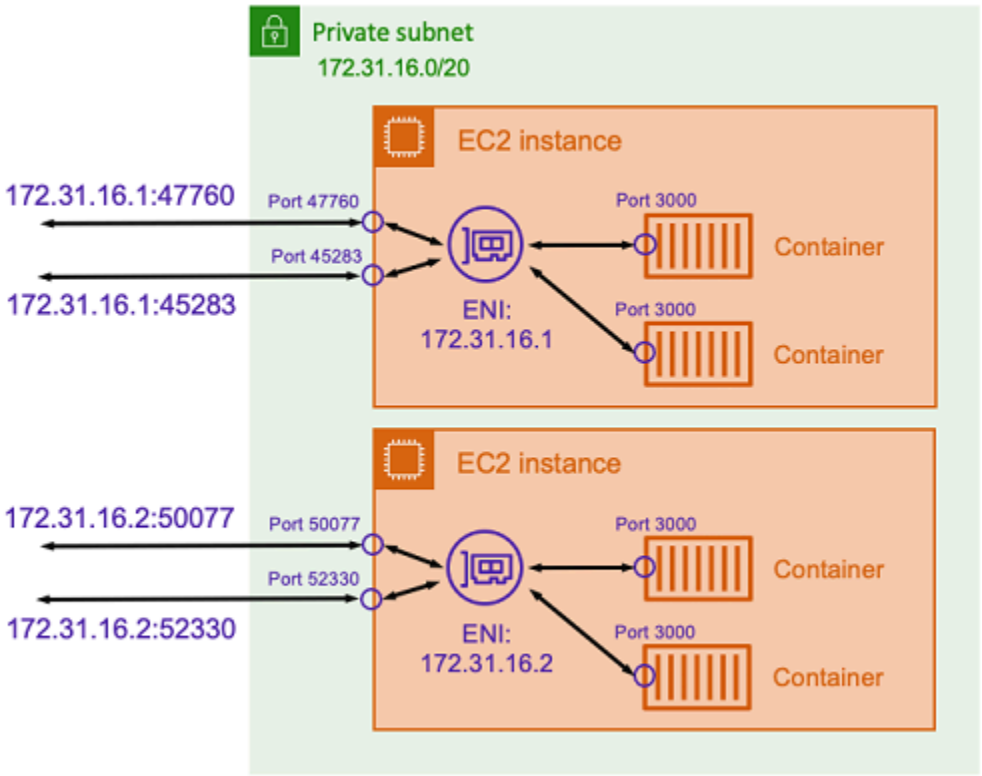
Im `bridge`-Modus verwenden Sie eine virtuelle Netzwerkbrücke, um eine Ebene zwischen dem Host und dem Netzwerk des Containers zu erstellen. Auf diese Weise können Sie Portzuordnungen erstellen, die einen Host-Port einem Container-Port neu zuordnen. Die Zuordnungen können statisch oder dynamisch sein.



Mit einer statischen Port-Zuordnung können Sie explizit definieren, welchen Host-Port Sie einem Container-Port zuordnen möchten. Im obigen Beispiel wird der Port `80` auf dem Host dem Port `3000` auf dem Container zugeordnet. Um mit der containerisierten Anwendung zu kommunizieren, senden Sie den Datenverkehr an Port `80` an die IP-Adresse der Amazon-EC2-Instance. Aus der Sicht der containerisierten Anwendung sieht sie den eingehenden Datenverkehr an Port `3000`.

Wenn Sie nur den Datenverkehr-Port ändern möchten, eignen sich statische Port-Zuordnungen. Dies hat jedoch immer noch den gleichen Nachteil wie die Verwendung des `host`-Netzwerkmodus. Sie können auf jedem Host nur eine einzige Instanziierung einer Aufgabe ausführen. Dies liegt daran, dass bei einer statischen Port-Zuordnung nur ein einziger Container Port 80 zugeordnet werden kann.

Um dieses Problem zu lösen, sollten Sie erwägen, den `bridge`-Netzwerkmodus mit einer dynamischen Portzuweisung zu verwenden, wie in der folgenden Abbildung dargestellt.



Wenn Sie in der Port-Zuordnung keinen Host-Port angeben, können Sie Docker veranlassen, einen zufälligen, ungenutzten Port aus dem flüchtigen Portbereich auszuwählen und ihn als öffentlichen Host-Port für den Container zuzuweisen. Beispielsweise könnte der Anwendung Node.js, die den Port `3000` auf dem Container überwacht, ein zufälliger Port mit hoher Nummer zugewiesen werden, z. B. `47760` auf dem Amazon-EC2-Host. Dies bedeutet, dass Sie mehrere Kopien dieses Containers

auf dem Host ausführen können. Darüber hinaus kann jedem Container ein eigener Port auf dem Host zugewiesen werden. Jede Kopie des Containers empfängt Datenverkehr über Port `3000`. Clients, die Datenverkehr an diese Container senden, verwenden jedoch die nach dem Zufallsprinzip zugewiesenen Host-Ports.

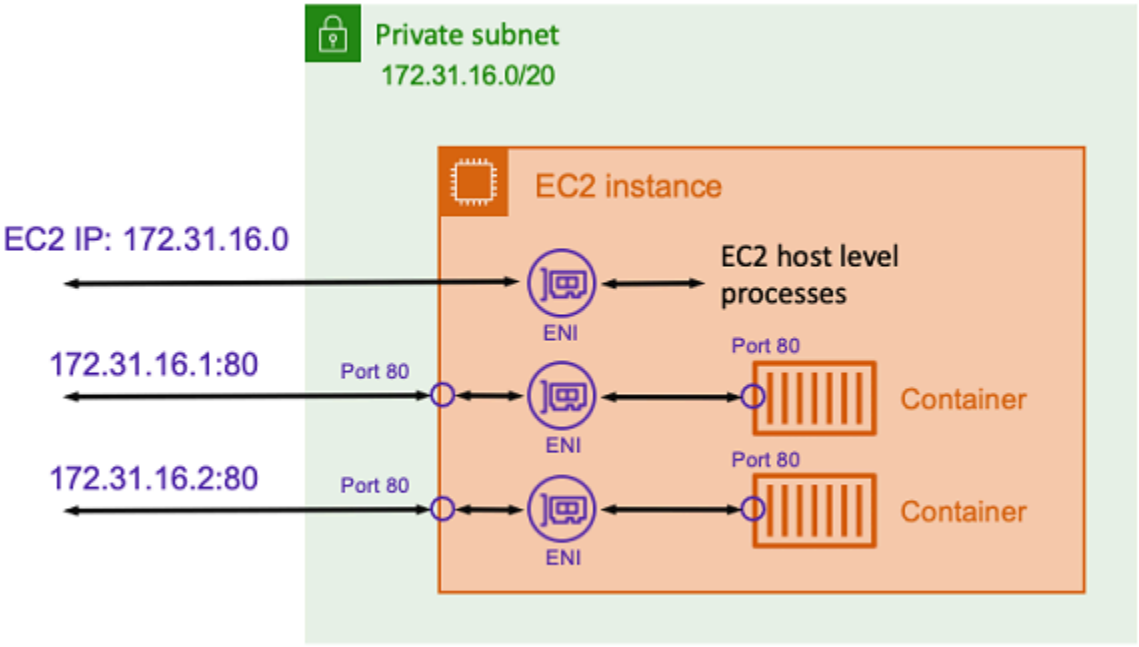
Amazon ECS hilft Ihnen dabei, den Überblick über die zufällig zugewiesenen Ports für jede Aufgabe zu behalten. Zu diesem Zweck werden die Zielgruppen und die AWS Cloud Map Serviceerkennung des Load Balancers automatisch aktualisiert, sodass sie über die Liste der Task-IP-Adressen und -Ports verfügen. Dies erleichtert die Nutzung von Services, die im `bridge`-Modus mit dynamischen Ports betrieben werden.

Ein Nachteil der Verwendung des `bridge`-Netzwerkmodus besteht jedoch darin, dass es schwierig ist, die Kommunikation zwischen Services zu sperren. Da Services jedem beliebigen, ungenutzten Port zugewiesen werden können, ist es notwendig, breite Portbereiche zwischen Hosts zu öffnen. Es ist jedoch nicht einfach, spezifische Regeln zu erstellen, sodass ein bestimmter Service nur mit einem bestimmten anderen Service kommunizieren kann. Die Services haben keine spezifischen Ports, die für Netzwerkregeln für Sicherheitsgruppen verwendet werden können.

Der `bridge`-Netzwerkmodus wird nur für Amazon-ECS-Aufgaben unterstützt, die auf Amazon-EC2-Instances gehostet werden. Wird bei der Verwendung von Amazon ECS auf Fargate nicht unterstützt.

AWSVPC Modus

Im `awsvpc` Netzwerkmodus erstellt und verwaltet Amazon ECS für jede Aufgabe ein Elastic Network Interface (ENI), und jede Aufgabe erhält ihre eigene private IP-Adresse innerhalb der VPC. Diese ENI ist von der ENI der zugrunde liegenden Hosts getrennt. Wenn eine Amazon EC2 EC2-Instance mehrere Aufgaben ausführt, ist die ENI jeder Aufgabe ebenfalls separat.



Im vorherigen Beispiel ist die Amazon EC2 EC2-Instance einer ENI zugewiesen. Die ENI steht für die IP-Adresse der EC2-Instance, die für die Netzwerkkommunikation auf Host-Ebene verwendet wird. Jede Aufgabe hat auch eine entsprechende ENI und eine private IP-Adresse. Da jede ENI separat ist, kann jeder Container an den Port 80 der Task-ENI gebunden werden. Daher müssen Sie die Portnummern nicht im Auge behalten. Stattdessen können Sie Datenverkehr an den Port 80 an der IP-Adresse der Aufgabe ENI senden.

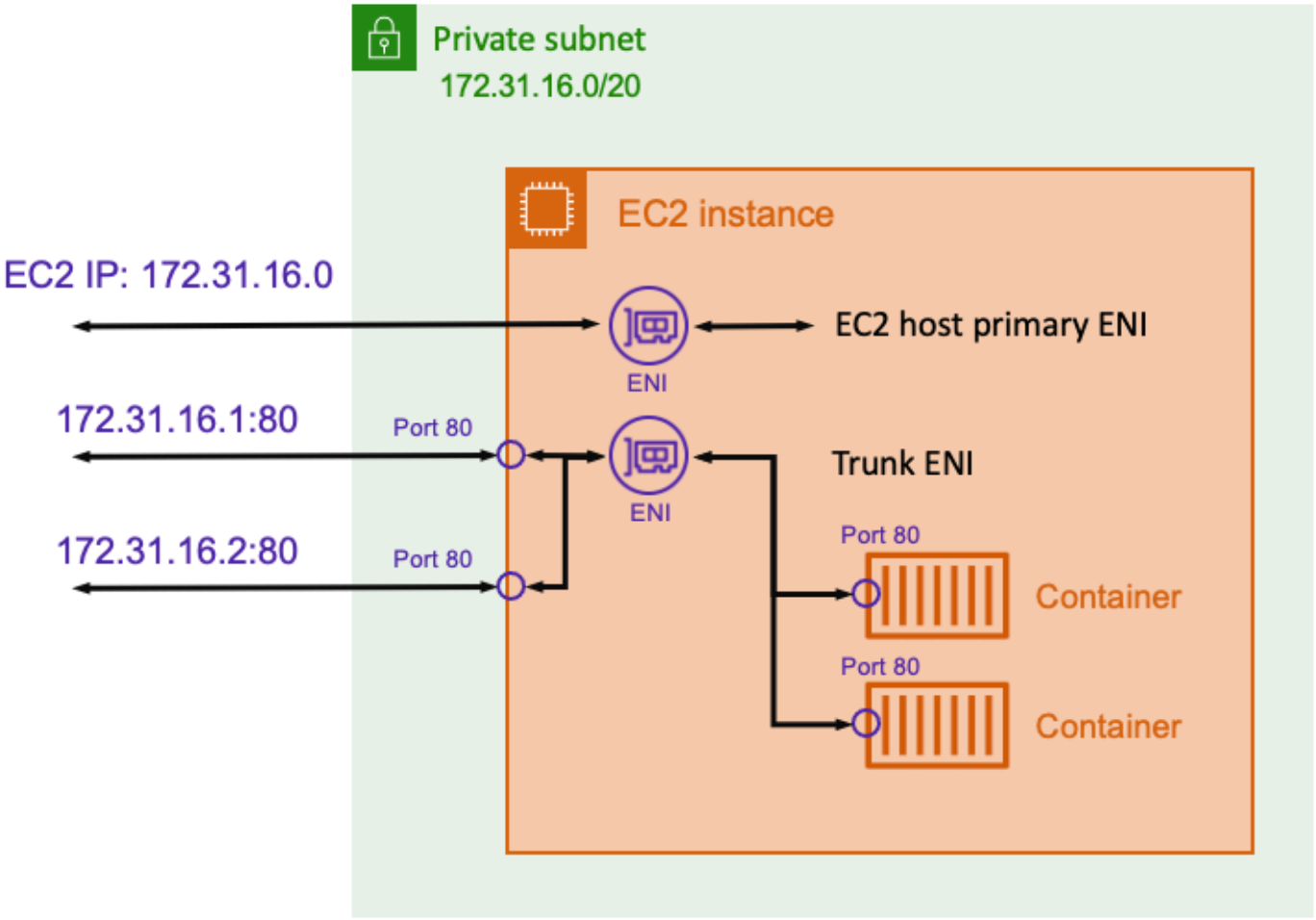
Der Vorteil der Verwendung des `awsvpc` Netzwerkmodus besteht darin, dass jede Aufgabe über eine separate Sicherheitsgruppe verfügt, mit der der Datenverkehr zugelassen oder verweigert werden kann. Dies bedeutet, dass Sie flexibler sind, um die Kommunikation zwischen Aufgaben und Diensten detaillierter zu steuern. Sie können eine Aufgabe auch so konfigurieren, dass eingehender Datenverkehr von einer anderen Aufgabe, die sich auf demselben Host befindet, verweigert wird.

Der `awsvpc` Netzwerkmodus wird für Amazon ECS-Aufgaben unterstützt, die sowohl auf Amazon EC2 als auch auf Fargate gehostet werden. Beachten Sie, dass bei der Verwendung von Fargate der `awsvpc` Netzwerkmodus erforderlich ist.

Bei der Verwendung des `awsvpc` Netzwerkmodus gibt es einige Herausforderungen, auf die Sie achten sollten.

Erhöhung der Aufgabendichte mit ENI Trunking

Der größte Nachteil der Verwendung des `awsvpc` Netzwerkmodus für Aufgaben, die auf Amazon EC2 EC2-Instances gehostet werden, besteht darin, dass EC2-Instances eine begrenzte Anzahl von ENIs haben, die an sie angehängt werden können. Dadurch wird begrenzt, wie viele Aufgaben Sie jeder Instance zuweisen können. Amazon ECS bietet die ENI-Trunking-Funktion, mit der die Anzahl der verfügbaren ENIs erhöht wird, um eine höhere Aufgabendichte zu erreichen.



Bei Verwendung von ENI-Trunking werden standardmäßig zwei ENI-Anhänge verwendet. Die erste ist die primäre ENI der Instance, die für alle Prozesse auf Hostebene verwendet wird. Der zweite ist der Trunk ENI, den Amazon ECS erstellt. Diese Funktion wird nur auf bestimmten Amazon EC2 EC2-Instance-Typen unterstützt.

Betrachten Sie dieses Beispiel. Ohne ENI-Trunking kann eine `c5.large` Instance mit zwei vCPUs nur zwei Aufgaben hosten. Mit ENI-Trunking kann eine `c5.large` Instanz mit zwei vCPUs jedoch bis zu zehn Aufgaben hosten. Jede Aufgabe hat eine andere IP-Adresse und Sicherheitsgruppe. Weitere

Informationen zu verfügbaren Instance-Typen und ihrer Dichte finden Sie unter Unterstützte Amazon EC2 EC2-Instance-Typen im Amazon Elastic Container Service Developer Guide.

ENI-Trunking hat keine Auswirkungen auf die Laufzeitleistung in Bezug auf Latenz oder Bandbreite.

Weitere Informationen finden Sie unter Elastic Network Interface Trunking im Amazon Elastic Container Service Developer Guide.

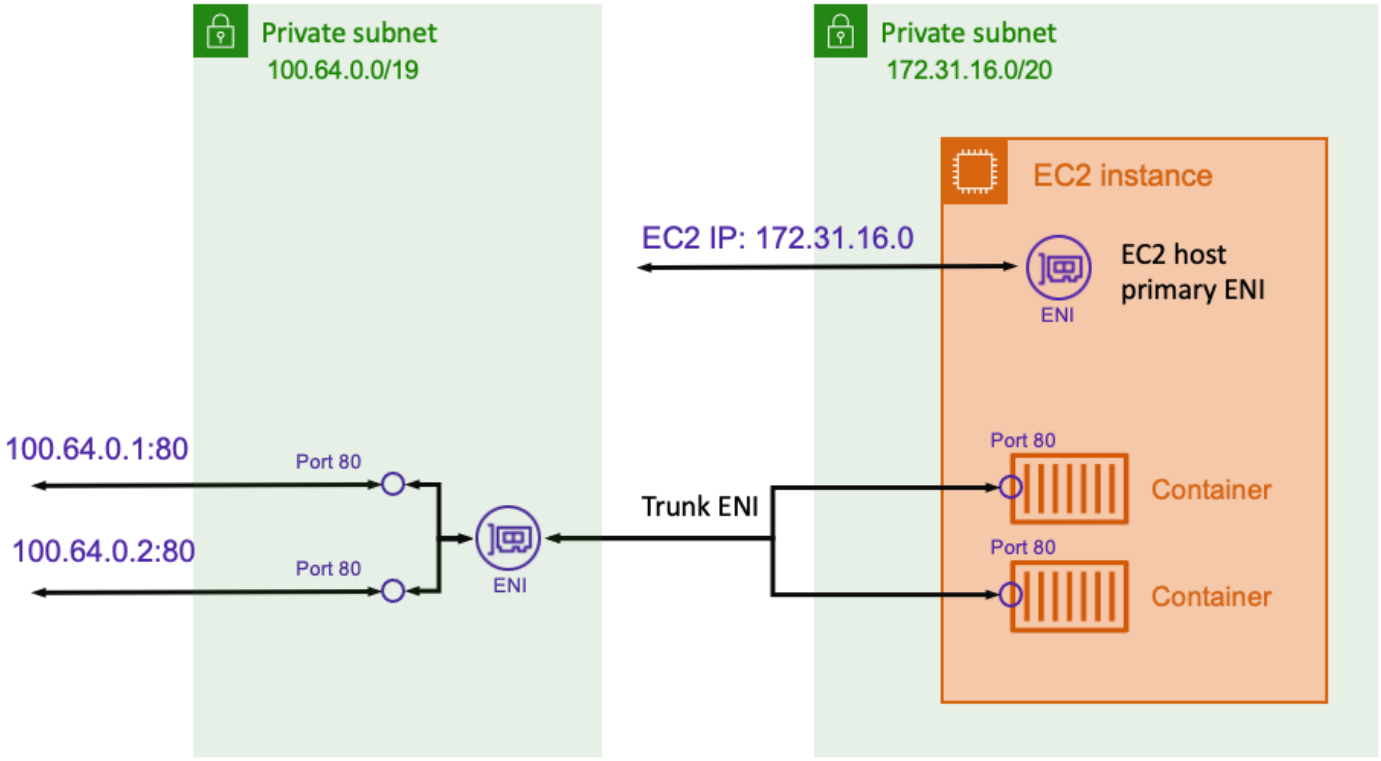
Verhinderung der Erschöpfung von IP-Adressen

Indem Sie jeder Aufgabe eine separate IP-Adresse zuweisen, können Sie Ihre gesamte Infrastruktur vereinfachen und Sicherheitsgruppen einrichten, die ein hohes Maß an Sicherheit bieten. Diese Konfiguration kann jedoch zu einer IP-Erschöpfung führen.

Die Standard-VPC in Ihrem AWS Konto verfügt über vorab bereitgestellte Subnetze mit einem CIDR-Bereich. `/20` Das bedeutet, dass jedes Subnetz 4.091 verfügbare IP-Adressen hat. Beachten Sie, dass mehrere IP-Adressen innerhalb des `/20` Bereichs für AWS eine bestimmte Verwendung reserviert sind. Betrachten Sie dieses Beispiel. Sie verteilen Ihre Anwendungen auf drei Subnetze in drei Availability Zones, um eine hohe Verfügbarkeit zu gewährleisten. In diesem Fall können Sie ungefähr 12.000 IP-Adressen in den drei Subnetzen verwenden.

Bei Verwendung von ENI-Trunking benötigt jede Amazon EC2 EC2-Instance, die Sie starten, zwei IP-Adressen. Eine IP-Adresse wird für das primäre ENI verwendet, und die andere IP-Adresse wird für das Trunk-ENI verwendet. Jede Amazon ECS-Aufgabe auf der Instance benötigt eine IP-Adresse. Wenn Sie einen extrem großen Workload starten, könnten Ihnen die verfügbaren IP-Adressen ausgehen. Dies kann zu Fehlern beim Starten von Amazon EC2 oder beim Starten von Aufgaben führen. Diese Fehler treten auf, weil die ENIs keine IP-Adressen innerhalb der VPC hinzufügen können, wenn keine IP-Adressen verfügbar sind.

Wenn Sie den `awsvpc` Netzwerkmodus verwenden, sollten Sie Ihre IP-Adressanforderungen überprüfen und sicherstellen, dass die CIDR-Bereiche Ihres Subnetzes Ihren Anforderungen entsprechen. Wenn Sie bereits mit der Verwendung einer VPC mit kleinen Subnetzen begonnen haben und der Adressraum langsam knapp wird, können Sie ein sekundäres Subnetz hinzufügen.



Durch die Verwendung von ENI-Trunking kann das Amazon VPC CNI so konfiguriert werden, dass ENIs in einem anderen IP-Adressraum als der Host verwendet werden. Auf diese Weise können Sie Ihrem Amazon EC2 EC2-Host und Ihren Aufgaben unterschiedliche IP-Adressbereiche zuweisen, die sich nicht überschneiden. Im Beispieldiagramm befindet sich die EC2-Host-IP-Adresse in einem Subnetz mit dem `172.31.16.0/20` IP-Bereich. Aufgaben, die auf dem Host ausgeführt werden, werden jedoch IP-Adressen im `100.64.0.0/19` Bereich zugewiesen. Durch die Verwendung von zwei unabhängigen IP-Bereichen müssen Sie sich keine Gedanken darüber machen, dass Aufgaben zu viele IP-Adressen verbrauchen und nicht genügend IP-Adressen für Instanzen übrig lassen.

Verwenden Sie den IPv6-Dual-Stack-Modus

Der `awsvpc` Netzwerkmodus ist mit VPCs kompatibel, die für den IPv6-Dual-Stack-Modus konfiguriert sind. Eine VPC, die den Dual-Stack-Modus verwendet, kann über IPv4, IPv6 oder beides kommunizieren. Jedes Subnetz in der VPC kann sowohl einen IPv4-CIDR-Bereich als auch einen IPv6-CIDR-Bereich haben. Weitere Informationen finden Sie unter IP-Adressierung in Ihrer VPC im Amazon VPC-Benutzerhandbuch.

Sie können die IPv4-Unterstützung für Ihre VPC und Subnetze nicht deaktivieren, um Probleme mit der IPv4-Erschöpfung zu beheben. Mit der IPv6-Unterstützung können Sie jedoch einige neue

Funktionen nutzen, insbesondere das Internet-Gateway nur für ausgehenden Datenverkehr. Ein Internet-Gateway nur für ausgehenden Datenverkehr ermöglicht es Aufgaben, ihre öffentlich routbare IPv6-Adresse zu verwenden, um ausgehende Verbindungen zum Internet zu initiieren. Das Internet-Gateway nur für ausgehenden Datenverkehr erlaubt jedoch keine Verbindungen aus dem Internet. Weitere Informationen finden Sie unter Internet-Gateways nur für ausgehenden Ausgang im Amazon VPC-Benutzerhandbuch.

Von Ihrer AWS VPC aus eine Verbindung zu Diensten herstellen

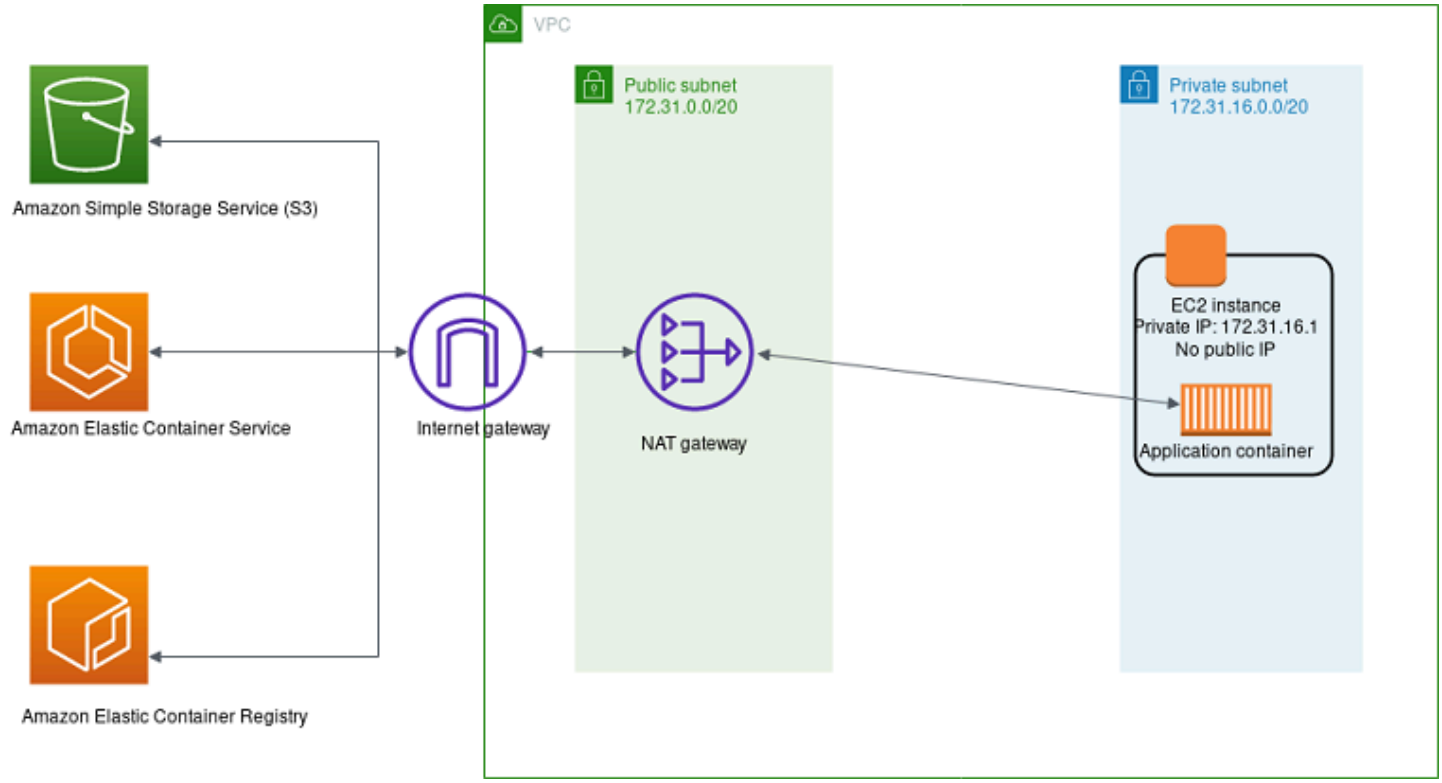
Damit Amazon ECS ordnungsgemäß funktioniert, muss der ECS-Container-Agent, der auf jedem Host ausgeführt wird, mit der Amazon ECS-Steuerebene kommunizieren. Wenn Sie Ihre Container-Images in Amazon ECR speichern, müssen die Amazon EC2 EC2-Hosts mit dem Amazon ECR-Serviceendpunkt und mit Amazon S3 kommunizieren, wo die Bildebenen gespeichert sind. Wenn Sie andere AWS Dienste für Ihre containerisierte Anwendung verwenden, z. B. persistente Daten, die in DynamoDB gespeichert sind, überprüfen Sie, ob diese Dienste auch über die erforderliche Netzwerkunterstützung verfügen.

Themen

- NAT-Gateway
- AWS PrivateLink

NAT-Gateway

Die Verwendung eines NAT-Gateways ist der einfachste Weg, um sicherzustellen, dass Ihre Amazon ECS-Aufgaben auf andere AWS Services zugreifen können. Weitere Informationen zu diesem Ansatz finden Sie unterVerwendung eines privaten Subnetzes und eines NAT-Gateways.



Im Folgenden sind die Nachteile dieses Ansatzes aufgeführt:

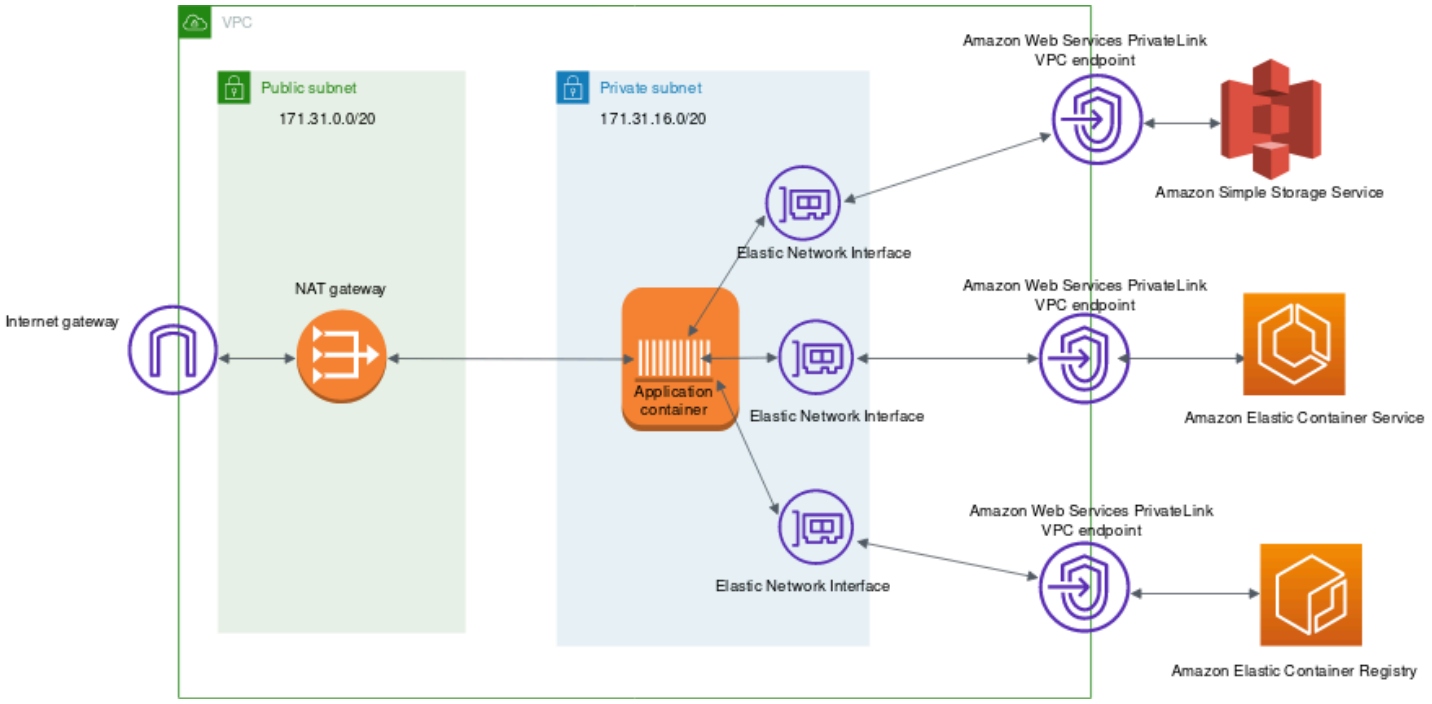
- Sie können nicht einschränken, mit welchen Zielen das NAT-Gateway kommunizieren kann. Sie können auch nicht einschränken, mit welchen Zielen Ihre Back-End-Stufe kommunizieren kann, ohne die gesamte ausgehende Kommunikation von Ihrer VPC zu unterbrechen.
- NAT-Gateways berechnen für jedes GB an Daten, das übertragen wird. Wenn Sie das NAT-Gateway verwenden, um große Dateien von Amazon S3 herunterzuladen oder eine große Anzahl von Datenbankabfragen an DynamoDB durchzuführen, wird Ihnen jedes GB Bandbreite in Rechnung gestellt. Darüber hinaus unterstützen NAT-Gateways eine Bandbreite von 5 Gbit/s und skalieren automatisch auf bis zu 45 Gbit/s. Wenn Sie über ein einzelnes NAT-Gateway routen, können Anwendungen, die Verbindungen mit sehr hoher Bandbreite benötigen, auf Netzwerkeinschränkungen stoßen. Um das Problem zu umgehen, können Sie Ihre Arbeitslast auf mehrere Subnetze aufteilen und jedem Subnetz ein eigenes NAT-Gateway zuweisen.

AWS PrivateLink

AWS PrivateLink bietet private Konnektivität zwischen VPCs, AWS Diensten und Ihren lokalen Netzwerken, ohne dass Ihr Datenverkehr dem öffentlichen Internet ausgesetzt wird.

Eine der dafür verwendeten Technologien ist der VPC-Endpunkt. Ein VPC-Endpunkt ermöglicht private Verbindungen zwischen Ihrer VPC und unterstützten AWS Diensten und VPC-Endpunktdiensten. Der Datenverkehr zwischen Ihrer VPC und dem anderen Service verlässt das Amazon-Netzwerk nicht. Ein VPC-Endpunkt benötigt kein Internet-Gateway, kein Virtual Private Gateway, kein NAT-Gerät, keine VPN-Verbindung oder AWS Direct Connect Verbindung. Amazon EC2 EC2-Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen, um mit Ressourcen im Service zu kommunizieren.

Das folgende Diagramm zeigt, wie die Kommunikation mit AWS Diensten funktioniert, wenn Sie VPC-Endpunkte anstelle eines Internet-Gateways verwenden. AWS PrivateLink stellt Elastic Network Interfaces (ENIs) innerhalb des Subnetzes bereit, und VPC-Routing-Regeln werden verwendet, um jegliche Kommunikation an den Service-Hostnamen über die ENI direkt an den Zieldienst zu senden. AWS Dieser Datenverkehr muss nicht mehr das NAT-Gateway oder das Internet-Gateway verwenden.



Im Folgenden sind einige der gängigen VPC-Endpunkte aufgeführt, die mit dem Amazon ECS-Service verwendet werden.

- VPC-Endpunkt des S3-Gateways
- DynamoDB-VPC-Endpunkt
- Amazon ECS VPC-Endpunkt
- Amazon ECR VPC-Endpunkt

Viele andere AWS Dienste unterstützen VPC-Endpunkte. Wenn Sie einen AWS Dienst intensiv nutzen, sollten Sie die spezifische Dokumentation für diesen Dienst und die Erstellung eines VPC-Endpunkts für diesen Datenverkehr nachschlagen.

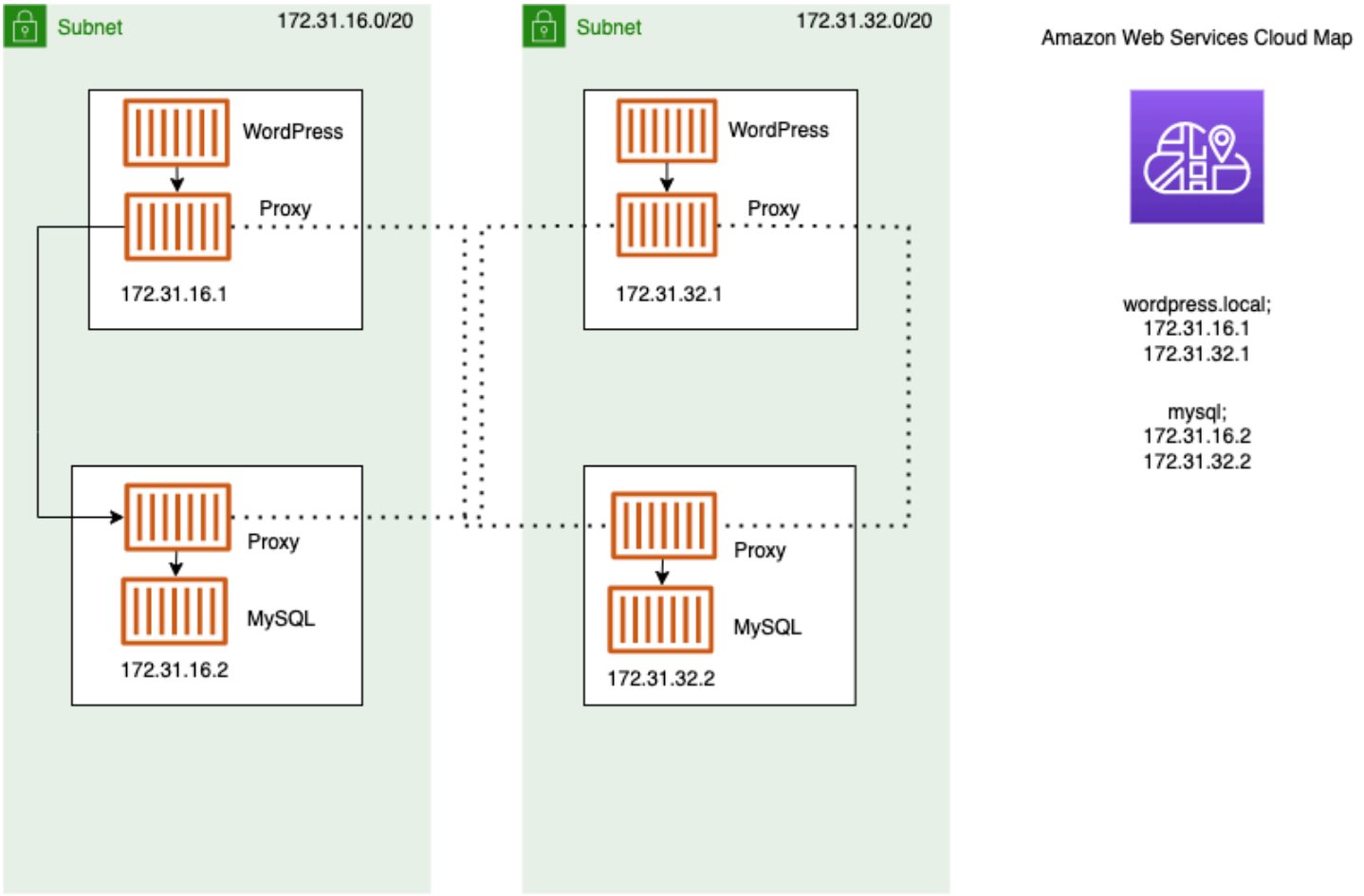
Vernetzung zwischen Amazon ECS-Services in einer VPC

Mithilfe von Amazon ECS-Aufgaben in einer VPC können Sie monolithische Anwendungen in separate Teile aufteilen, die unabhängig voneinander in einer sicheren Umgebung bereitgestellt und skaliert werden können. Diese Architektur wird als serviceorientierte Architektur (SOA) oder Microservices bezeichnet. Es kann jedoch schwierig sein, sicherzustellen, dass all diese Teile, sowohl innerhalb als auch außerhalb einer VPC, miteinander kommunizieren können. Es gibt verschiedene Ansätze zur Erleichterung der Kommunikation, die alle unterschiedliche Vor- und Nachteile haben.

Service Connect verwenden

Wir empfehlen Amazon ECS Service Connect, das eine Amazon ECS-Konfiguration für Serviceerkennung, Konnektivität und Verkehrsüberwachung bietet. Mit Service Connect können Ihre Anwendungen Kurznamen und Standardports verwenden, um eine Verbindung zu ECS-Diensten im selben Cluster oder in anderen Clustern herzustellen, auch zwischen VPCs im selben AWS-Region. Weitere Informationen finden Sie unter Amazon ECS Service Connect im Amazon Elastic Container Service Developer Guide.

Wenn Sie Service Connect verwenden, verwaltet ECS alle Teile der Serviceerkennung: die Erstellung der Namen, die erkannt werden können, die dynamische Verwaltung von Einträgen für jede Aufgabe, wenn die Aufgaben gestartet und beendet werden, die Ausführung eines Agenten in jeder Aufgabe, die so konfiguriert ist, dass sie die Namen erkennt. Ihre Anwendung kann die Namen mithilfe der Standardfunktionen für DNS-Namen und das Herstellen von Verbindungen nachschlagen. Wenn Ihre Anwendung dies bereits tut, müssen Sie Ihre Anwendung nicht ändern, um Service Connect verwenden zu können.



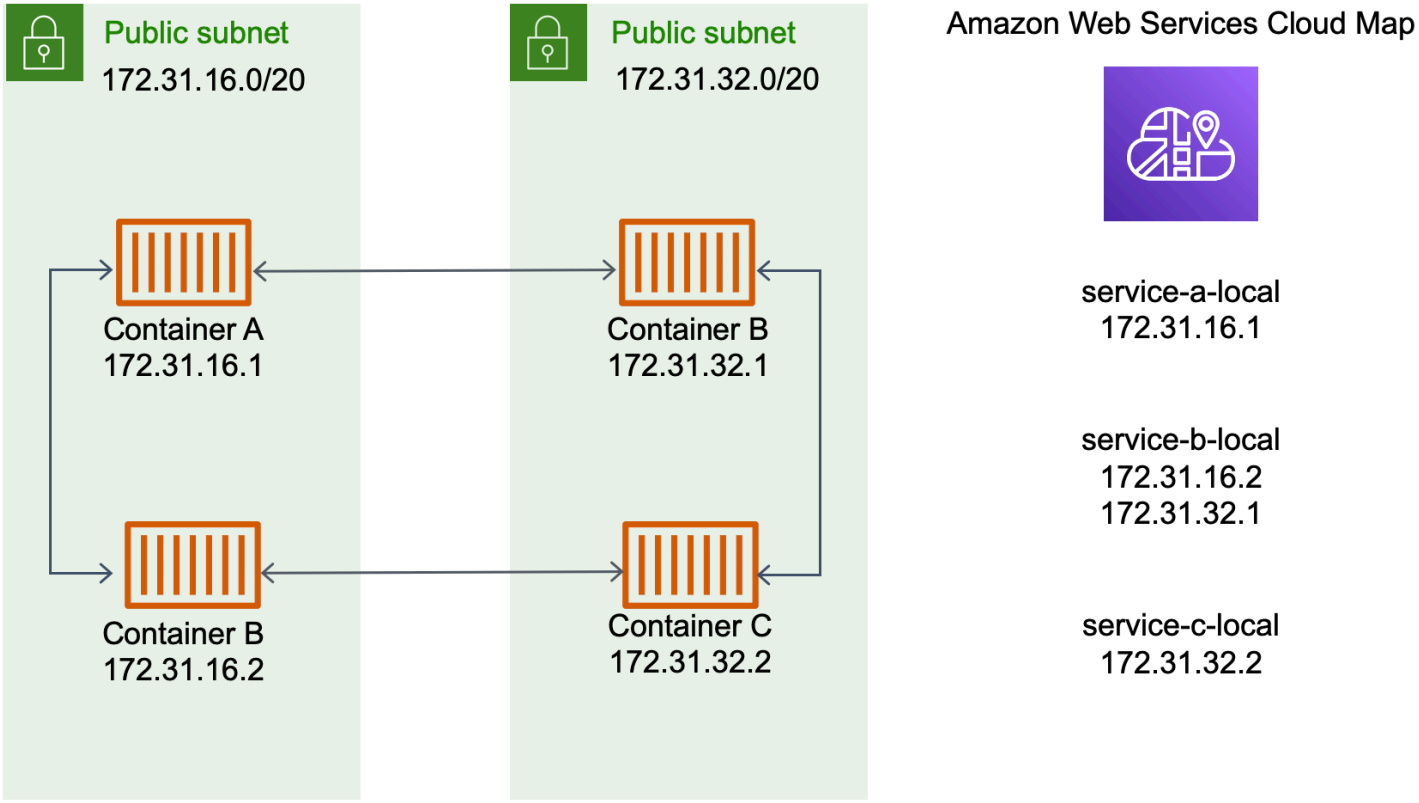
Änderungen treten nur während der Bereitstellung auf

Sie stellen die vollständige Konfiguration in jeder ECS-Service- und Aufgabendefinition bereit. ECS verwaltet Änderungen an dieser Konfiguration in jeder Servicebereitstellung, um sicherzustellen, dass sich alle Aufgaben in einer Bereitstellung auf die gleiche Weise verhalten. Ein häufiges Problem bei DNS as Service Discovery ist beispielsweise die Steuerung einer Migration. Wenn Sie einen DNS-Namen so ändern, dass er auf die neuen Ersatz-IP-Adressen verweist, kann es die maximale TTL-Zeit dauern, bis alle Clients den neuen Dienst verwenden. Mit Service Connect aktualisiert die Client-Bereitstellung die Konfiguration, indem sie die Client-Tasks ersetzt. Sie können den Deployment Circuit Breaker und andere Bereitstellungskonfigurationen so konfigurieren, dass sie Service Connect-Änderungen genauso beeinflussen wie jede andere Bereitstellung.

Verwenden von Service Discovery

Ein weiterer service-to-service Kommunikationsansatz ist die direkte Kommunikation mithilfe von Service Discovery. Bei diesem Ansatz können Sie die AWS Cloud Map Service Discovery-Integration mit Amazon ECS verwenden. Mithilfe von Service Discovery synchronisiert Amazon ECS die Liste

der gestarteten Aufgaben mit AWS Cloud Map, das einen DNS-Hostnamen verwaltet, der in die internen IP-Adressen einer oder mehrerer Aufgaben von diesem bestimmten Service aufgelöst wird. Andere Dienste in der Amazon VPC können diesen DNS-Hostnamen verwenden, um Traffic mithilfe seiner internen IP-Adresse direkt an einen anderen Container zu senden. Weitere Informationen finden Sie unter Service Discovery im Amazon Elastic Container Service Developer Guide.



Im obigen Diagramm gibt es drei Dienste. `serviceA`hat einen Container und kommuniziert mit`serviceB`, der zwei Container hat. `serviceB`muss auch mit kommunizieren`serviceC`, der einen Container hat. Jeder Container in allen drei Diensten kann die internen DNS-Namen von verwenden, AWS Cloud Map um die internen IP-Adressen eines Containers aus dem Downstream-Dienst zu ermitteln, mit dem er kommunizieren muss.

Dieser service-to-service Kommunikationsansatz bietet eine geringe Latenz. Auf den ersten Blick ist es auch einfach, da sich zwischen den Containern keine zusätzlichen Komponenten befinden. Der Verkehr wird direkt von einem Container zum anderen Container geleitet.

Dieser Ansatz eignet sich für den `awsvpc` Netzwerkmodus, in dem jede Aufgabe ihre eigene eindeutige IP-Adresse hat. Die meiste Software unterstützt nur die Verwendung von `A` DNS-Einträgen, die direkt in IP-Adressen aufgelöst werden. Bei Verwendung des `awsvpc` Netzwerkmodus handelt es sich bei den IP-Adressen für jede Aufgabe um einen `A` Datensatz. Wenn Sie jedoch

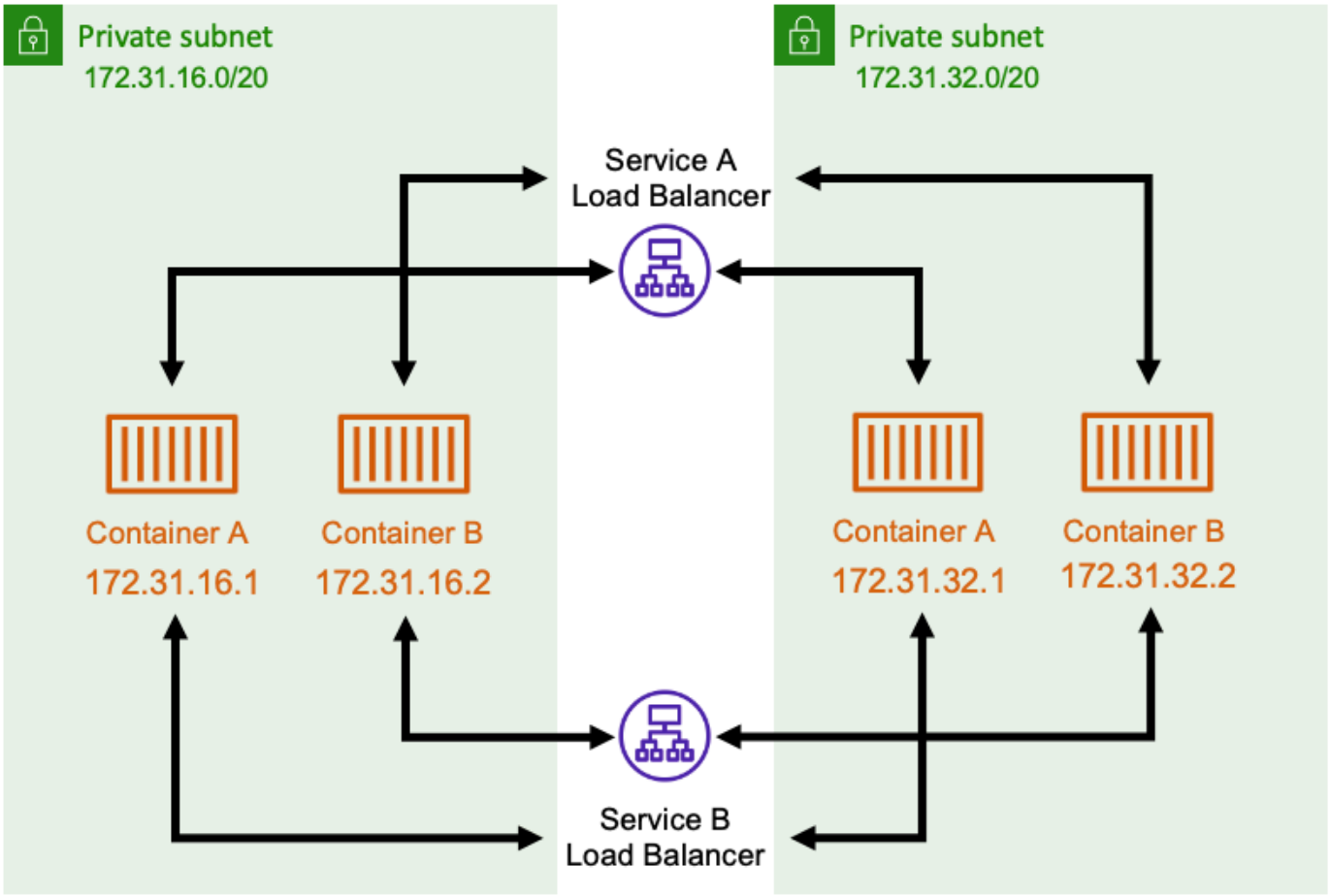
den `bridge` Netzwerkmodus verwenden, können sich mehrere Container dieselbe IP-Adresse teilen. Darüber hinaus führen dynamische Portzuordnungen dazu, dass den Containern nach dem Zufallsprinzip Portnummern für diese einzelne IP-Adresse zugewiesen werden. Zu diesem Zeitpunkt reicht ein `A` Datensatz für die Diensterkennung nicht mehr aus. Sie müssen auch einen `SRV` Datensatz verwenden. Dieser Datensatztyp kann sowohl IP-Adressen als auch Portnummern nachverfolgen, erfordert jedoch, dass Sie die Anwendungen entsprechend konfigurieren. Einige vorgefertigte Anwendungen, die Sie verwenden, unterstützen möglicherweise keine `SRV` Datensätze.

Ein weiterer Vorteil des `awsvpc` Netzwerkmodus besteht darin, dass Sie für jeden Dienst eine eigene Sicherheitsgruppe haben. Sie können diese Sicherheitsgruppe so konfigurieren, dass eingehende Verbindungen nur von den spezifischen Upstream-Diensten zugelassen werden, die mit diesem Dienst kommunizieren müssen.

Der Hauptnachteil der direkten service-to-service Kommunikation mithilfe von Service Discovery besteht darin, dass Sie zusätzliche Logik implementieren müssen, um Wiederholungsversuche durchzuführen und Verbindungsfehler zu beheben. DNS-Einträge haben einen Zeitraum time-to-live (TTL), der bestimmt, wie lange sie zwischengespeichert werden. Es dauert einige Zeit, bis der DNS-Eintrag aktualisiert ist und der Cache abläuft, sodass Ihre Anwendungen die neueste Version des DNS-Eintrags abrufen können. Daher löst Ihre Anwendung den DNS-Eintrag möglicherweise so auf, dass er auf einen anderen Container verweist, der nicht mehr vorhanden ist. Ihre Anwendung muss Wiederholungsversuche verarbeiten und über eine Logik verfügen, um fehlerhafte Backends zu ignorieren.

Verwenden Sie einen internen Load Balancer

Ein anderer service-to-service Kommunikationsansatz ist die Verwendung eines internen Load Balancers. Ein interner Load Balancer befindet sich vollständig in Ihrer VPC und ist nur für Dienste innerhalb Ihrer VPC zugänglich.



Der Load Balancer sorgt für eine hohe Verfügbarkeit, indem er redundante Ressourcen in jedem Subnetz bereitstellt. Wenn ein Container von mit einem Container von kommunizieren `serviceA` muss`serviceB`, öffnet er eine Verbindung zum Load Balancer. Der Load Balancer öffnet dann eine Verbindung zu einem Container von. `service B` Der Load Balancer dient als zentraler Ort für die Verwaltung aller Verbindungen zwischen den einzelnen Diensten.

Wenn ein Container von `serviceB` stoppt, kann der Load Balancer diesen Container aus dem Pool entfernen. Der Load Balancer führt außerdem Integritätsprüfungen für jedes Downstream-Ziel in seinem Pool durch und kann fehlerhafte Ziele automatisch aus dem Pool entfernen, bis sie wieder fehlerfrei sind. Die Anwendungen müssen nicht mehr wissen, wie viele Downstream-Container es gibt. Sie öffnen einfach ihre Verbindungen zum Load Balancer.

Dieser Ansatz ist für alle Netzwerkmodi von Vorteil. Der Load Balancer kann die IP-Adressen der Aufgaben im `awsvpc` Netzwerkmodus sowie erweiterte Kombinationen von IP-Adressen und Ports im `bridge` Netzwerkmodus verfolgen. Es verteilt den Verkehr gleichmäßig auf alle Kombinationen von

IP-Adressen und Ports, auch wenn mehrere Container tatsächlich auf derselben Amazon EC2 EC2-Instance gehostet werden, nur auf verschiedenen Ports.

Der einzige Nachteil dieses Ansatzes sind die Kosten. Um hochverfügbar zu sein, muss der Load Balancer über Ressourcen in jeder Availability Zone verfügen. Dies führt zu zusätzlichen Kosten, da der Aufwand für die Bezahlung des Load Balancers und die Menge des Datenverkehrs, der über den Load Balancer fließt, verursacht werden.

Sie können jedoch die Gemeinkosten reduzieren, indem sich mehrere Dienste einen Load Balancer teilen. Dies ist besonders für REST-Dienste geeignet, die einen Application Load Balancer verwenden. Sie können pfadbasierte Routing-Regeln erstellen, die den Datenverkehr an verschiedene Dienste weiterleiten. Kann beispielsweise an einen `/api/user/*` Container weiterleiten, der Teil des `user` Dienstes ist, wohingegen `/api/order/*` die Weiterleitung an den zugehörigen `order` Dienst möglich ist. Bei diesem Ansatz zahlen Sie nur für einen Application Load Balancer und haben eine einheitliche URL für Ihre API. Sie können den Datenverkehr jedoch auf verschiedene Microservices im Backend aufteilen.

Netzwerkdienste für AWS Konten und VPCs

Wenn Sie Teil einer Organisation mit mehreren Teams und Abteilungen sind, stellen Sie Dienste wahrscheinlich unabhängig voneinander in separaten VPCs innerhalb eines gemeinsamen AWS Kontos oder in VPCs bereit, die mehreren einzelnen Konten zugeordnet sind. AWS Unabhängig davon, wie Sie Ihre Dienste bereitstellen, empfehlen wir Ihnen, Ihre Netzwerkkomponenten zu ergänzen, um den Datenverkehr zwischen VPCs weiterzuleiten. Zu diesem Zweck können mehrere AWS Dienste verwendet werden, um Ihre vorhandenen Netzwerkkomponenten zu ergänzen.

- AWS Transit Gateway — Sie sollten zuerst diesen Netzwerkdienst in Betracht ziehen. Dieser Service dient als zentraler Knotenpunkt für das Routing Ihrer Verbindungen zwischen Amazon-VPCs, AWS Konten und lokalen Netzwerken. Weitere Informationen finden Sie unter Was ist ein Transit-Gateway? im Amazon VPC Transit Gateways Guide.
- Amazon VPC- und VPN-Unterstützung — Sie können diesen Service verwenden, um site-to-site VPN-Verbindungen für die Verbindung von lokalen Netzwerken mit Ihrer VPC herzustellen. Weitere Informationen finden Sie unter Was ist? AWS Site-to-Site VPN im AWS Site-to-Site VPN Benutzerhandbuch.
- Amazon VPC — Sie können Amazon VPC Peering verwenden, um mehrere VPCs zu verbinden, entweder im selben Konto oder kontenübergreifend. Weitere Informationen finden Sie unter Was ist VPC Peering? im Amazon VPC Peering Guide.

- Gemeinsam genutzte VPCs — Sie können eine VPC und VPC-Subnetze für mehrere Konten verwenden. AWS Weitere Informationen finden Sie unter Arbeiten mit gemeinsam genutzten VPCs im Amazon VPC-Benutzerhandbuch.

Optimierung und Problembehebung

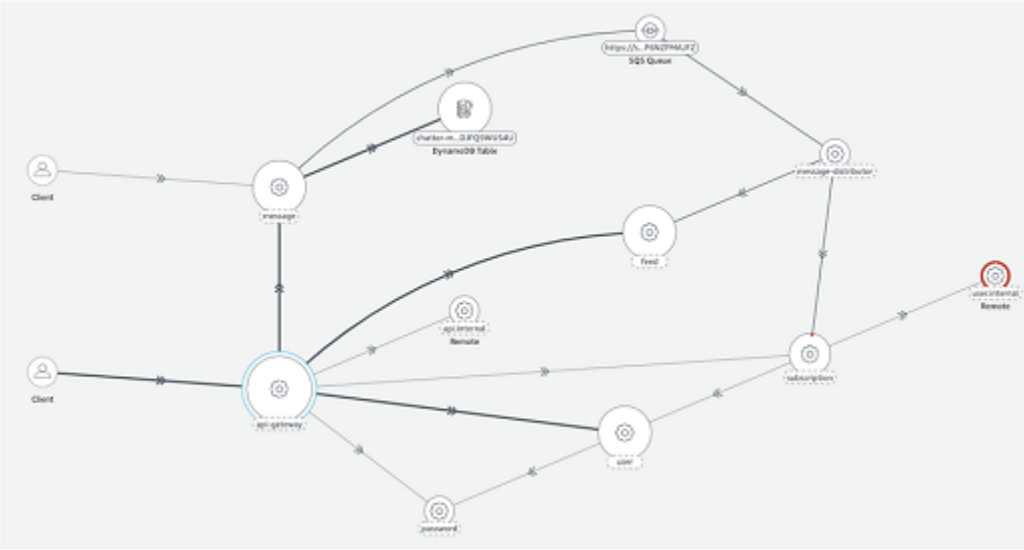
Die folgenden Dienste und Funktionen können Ihnen helfen, Einblicke in Ihre Netzwerk- und Dienstkonfigurationen zu gewinnen. Sie können diese Informationen verwenden, um Netzwerkprobleme zu beheben und Ihre Dienste besser zu optimieren.

CloudWatch Einblicke in Container

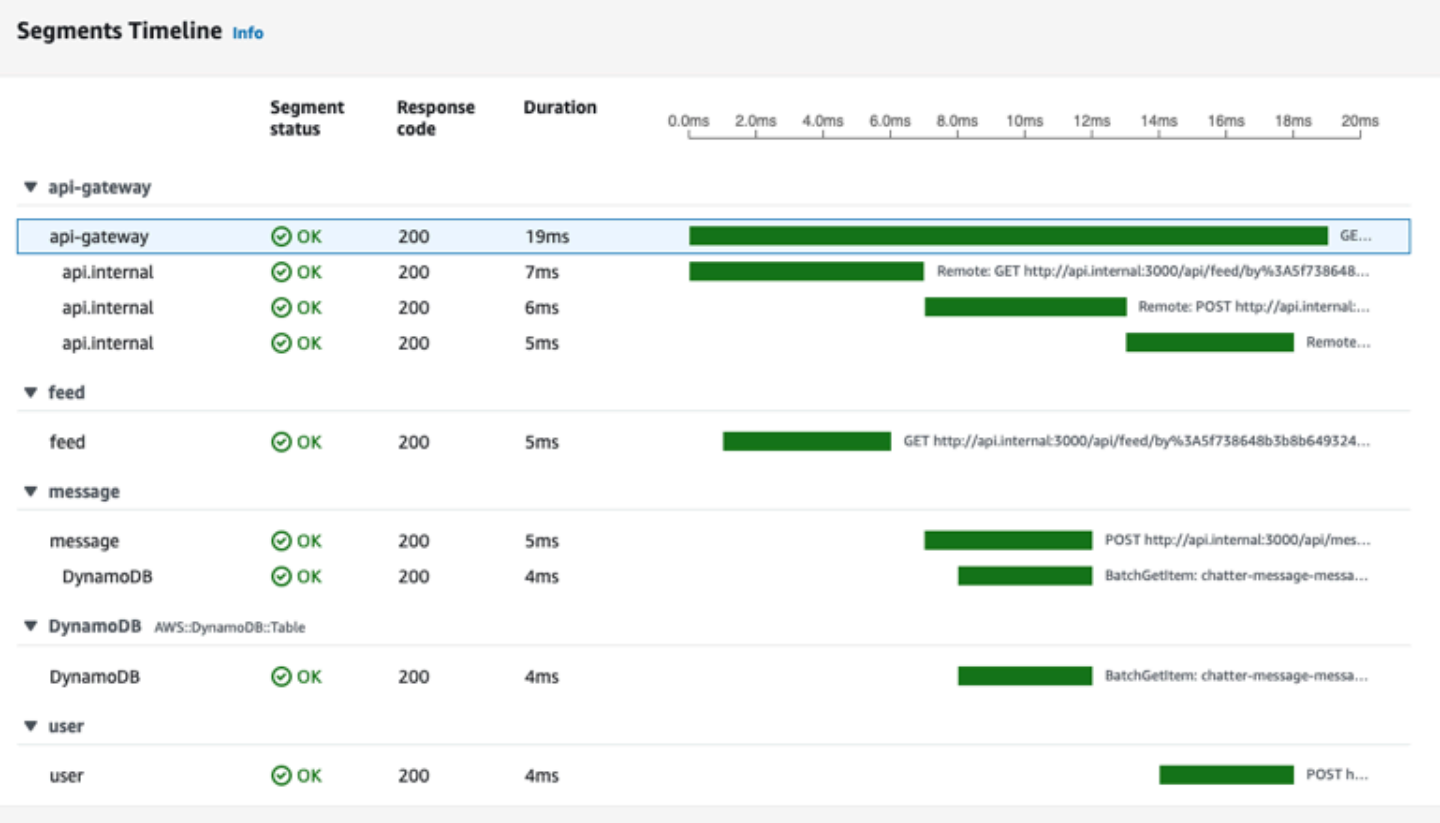
CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Protokolle aus Ihren containerisierten Anwendungen und Microservices zusammen. Zu den Metriken gehört die Nutzung von Ressourcen wie CPU, Arbeitsspeicher, Festplatte und Netzwerk. Sie sind in CloudWatch automatischen Dashboards verfügbar. Weitere Informationen finden Sie unter Setting up Container Insights on Amazon ECS im CloudWatch Amazon-Benutzerhandbuch.

AWS X-Ray

AWS X-Ray ist ein Tracing-Service, mit dem Sie Informationen über die Netzwerkanfragen sammeln können, die Ihre Anwendung stellt. Sie können das SDK verwenden, um Ihre Anwendung zu instrumentieren und die Zeitabläufe und Antwortcodes des Datenverkehrs zwischen Ihren Diensten sowie zwischen Ihren Diensten und AWS Dienstendpunkten zu erfassen. Weitere Informationen finden Sie unter Was ist AWS X-Ray im AWS X-Ray -Entwicklerhandbuch.



Sie können sich auch AWS X-Ray Grafiken ansehen, die zeigen, wie Ihre Dienste miteinander vernetzt sind. Oder verwenden Sie sie, um aggregierte Statistiken über die Leistung der einzelnen service-to-service Links zu untersuchen. Schließlich können Sie sich eingehender mit jeder einzelnen Transaktion befassen, um zu sehen, wie Segmente, die Netzwerkanrufe darstellen, mit dieser bestimmten Transaktion verknüpft sind.



Mithilfe dieser Funktionen können Sie feststellen, ob ein Netzwerkengpass vorliegt oder ob ein bestimmter Dienst in Ihrem Netzwerk nicht wie erwartet funktioniert.

VPC Flow Logs

Sie können Amazon VPC-Flow-Logs verwenden, um die Netzwerkleistung zu analysieren und Verbindungsprobleme zu debuggen. Wenn VPC-Flow-Logs aktiviert sind, können Sie ein Protokoll aller Verbindungen in Ihrer VPC erfassen. Dazu gehören Verbindungen zu Netzwerkschnittstellen, die mit Elastic Load Balancing, Amazon RDS, NAT-Gateways und anderen wichtigen AWS Diensten verknüpft sind, die Sie möglicherweise verwenden. Weitere Informationen finden Sie unter VPC-Flow-Protokolle im Amazon-VPC-Benutzerhandbuch.

Tipps zur Netzwerkoptimierung

Es gibt einige Einstellungen, die Sie verfeinern können, um Ihr Netzwerk zu verbessern.

kein Datei-Ulimit

Wenn Sie davon ausgehen, dass Ihre Anwendung viel Verkehr hat und viele gleichzeitige Verbindungen verarbeiten wird, sollten Sie das Systemkontingent für die Anzahl der erlaubten Dateien berücksichtigen. Wenn viele Netzwerk-Sockets geöffnet sind, muss jeder einzelne durch einen Dateideskriptor repräsentiert werden. Wenn Ihr Dateideskriptorkontingent zu niedrig ist, werden Ihre Netzwerk-Sockets begrenzt. Dies führt zu fehlgeschlagenen Verbindungen oder Fehlern. Sie können das containerspezifische Kontingent für die Anzahl der Dateien in der Amazon ECS-Aufgabendefinition aktualisieren. Wenn Sie Amazon EC2 (statt AWS Fargate) verwenden, müssen Sie möglicherweise auch diese Kontingente für Ihre zugrunde liegende Amazon EC2 EC2-Instance anpassen.

sysctl.net

Eine weitere Kategorie einstellbarer Einstellungen sind die `sysctl` Netzwerkeinstellungen. Sie sollten sich an die spezifischen Einstellungen für die Linux-Distribution Ihrer Wahl halten. Viele dieser Einstellungen passen die Größe der Lese- und Schreibpuffer an. Dies kann in einigen Situationen hilfreich sein, wenn große Amazon EC2 EC2-Instances ausgeführt werden, auf denen sich viele Container befinden.

Bewährte Methoden — Automatische Skalierung und Kapazitätsmanagement

Amazon ECS wird verwendet, um containerisierte Anwendungs-Workloads aller Größen auszuführen. Dazu gehören sowohl die Extreme minimaler Testumgebungen als auch großer Produktionsumgebungen, die auf globaler Ebene betrieben werden.

Bei Amazon ECS zahlen Sie, wie bei allen AWS Services, nur für das, was Sie tatsächlich nutzen. Bei entsprechender Architektur können Sie Kosten sparen, indem Ihre Anwendung nur die Ressourcen verbraucht, die sie zu dem Zeitpunkt benötigt, zu dem sie benötigt wird. Dieser Leitfaden mit bewährten Methoden zeigt, wie Sie Ihre Amazon ECS-Workloads so ausführen, dass sie Ihre Service-Level-Ziele erfüllen und gleichzeitig kostengünstig arbeiten.

Themen

Bestimmung der Aufgabengröße

Eine der wichtigsten Entscheidungen, die Sie bei der Bereitstellung von Containern auf Amazon ECS treffen müssen, ist Ihre Container- und Aufgabengröße. Sowohl Ihre Container- als auch Ihre Aufgabengröße sind für die Skalierung und Kapazitätsplanung von entscheidender Bedeutung. In Amazon ECS gibt es zwei Ressourcenmetriken, die für die Kapazität verwendet werden: CPU und Arbeitsspeicher. Die CPU wird in Einheiten von 1/1024 einer vollen vCPU gemessen (wobei 1024 Einheiten einer ganzen vCPU entsprechen). Der Arbeitsspeicher wird in Megabyte gemessen. In Ihrer Aufgabendefinition können Sie Ressourcenreservierungen und -limits deklarieren.

Wenn Sie eine Reservierung deklarieren, geben Sie die Mindestmenge an Ressourcen an, die eine Aufgabe benötigt. Ihre Aufgabe erhält mindestens die angeforderte Menge an Ressourcen. Ihre Anwendung kann möglicherweise mehr CPU oder Arbeitsspeicher verwenden als die Reservierung, die Sie deklariert haben. Dies unterliegt jedoch allen Beschränkungen, die Sie ebenfalls deklariert haben. Wenn Sie mehr als den Reservierungsbetrag verwenden, wird dies als Bursting bezeichnet. In Amazon ECS sind Reservierungen garantiert. Wenn Sie beispielsweise Amazon EC2 EC2-Instances verwenden, um Kapazität bereitzustellen, platziert Amazon ECS keine Aufgabe auf einer Instance, bei der die Reservierung nicht erfüllt werden kann.

Ein Limit ist die maximale Menge an CPU-Einheiten oder Arbeitsspeicher, die Ihr Container oder Ihre Aufgabe verwenden kann. Jeder Versuch, mehr CPUs als diesen Grenzwert zu verwenden, führt zu einer Drosselung. Jeder Versuch, mehr Speicher zu verwenden, führt dazu, dass Ihr Container gestoppt wird.

Die Auswahl dieser Werte kann eine Herausforderung sein. Das liegt daran, dass die Werte, die für Ihre Anwendung am besten geeignet sind, stark von den Ressourcenanforderungen Ihrer Anwendung abhängen. Auslastungstests Ihrer Anwendung sind der Schlüssel zu einer erfolgreichen Planung des Ressourcenbedarfs und zu einem besseren Verständnis der Anforderungen Ihrer Anwendung.

Zustandslose Anwendungen

Für statusfreie Anwendungen, die horizontal skaliert werden, wie z. B. eine Anwendung hinter einem Load Balancer, empfehlen wir, dass Sie zunächst ermitteln, wie viel Speicher Ihre Anwendung bei der Bearbeitung von Anfragen verbraucht. Zu diesem Zweck können Sie herkömmliche Tools wie `ps` oder oder oder `top` Überwachungslösungen wie CloudWatch Container Insights verwenden.

Denken Sie bei der Festlegung einer CPU-Reservierung darüber nach, wie Sie Ihre Anwendung skalieren möchten, um Ihren Geschäftsanforderungen gerecht zu werden. Sie können kleinere CPU-Reservierungen, wie z. B. 256 CPU-Einheiten (oder 1/4 vCPU), verwenden, um feinkörnig zu skalieren und so die Kosten zu minimieren. Sie können jedoch möglicherweise nicht schnell genug skaliert werden, um erhebliche Nachfragespitzen zu bewältigen. Sie können größere CPU-Reservierungen verwenden, um schneller ein- und auszuskalieren und so Nachfragespitzen schneller zu begegnen. Größere CPU-Reservierungen sind jedoch teurer.

Andere Anwendungen

Bei Anwendungen, die nicht horizontal skaliert werden können, wie z. B. Singleton Worker oder Datenbankserver, stellen die verfügbaren Kapazitäten und Kosten die wichtigsten Überlegungen dar.

Sie sollten die Größe des Speichers und der CPU auf der Grundlage der Belastungstests auswählen, dass Sie Datenverkehr bereitstellen müssen, um Ihr Service-Level-Ziel zu erreichen. Amazon ECS stellt sicher, dass die Anwendung auf einem Host mit ausreichender Kapazität platziert wird.

Konfiguration von Service Auto Scaling

Ein Amazon ECS-Service ist eine verwaltete Sammlung von Aufgaben. Jedem Service ist eine Aufgabendefinition, eine gewünschte Anzahl von Aufgaben und eine optionale Platzierungsstrategie zugeordnet. Amazon ECS Service Auto Scaling wird über den Application Auto Scaling-Service implementiert. Application Auto Scaling verwendet CloudWatch Metriken als Quelle für Skalierungsmetriken. Es verwendet auch CloudWatch Alarme, um Schwellenwerte dafür festzulegen, wann Ihr Service erweitert oder verkleinert werden soll. Sie geben die Schwellenwerte für die Skalierung an, indem Sie entweder ein metrisches Ziel festlegen, das als Zielverfolgungsskalierung bezeichnet wird, oder indem Sie Schwellenwerte angeben, die als schrittweise Skalierung bezeichnet werden. Nachdem Application Auto Scaling konfiguriert wurde, berechnet es kontinuierlich die entsprechende gewünschte Task-Anzahl für den Service. Es benachrichtigt Amazon ECS auch, wenn sich die gewünschte Anzahl an Aufgaben ändern sollte, entweder durch Verkleinern oder Vergrößern.

Um Service Auto Scaling effektiv nutzen zu können, müssen Sie eine geeignete Skalierungsmetrik auswählen. In den folgenden Abschnitten wird erläutert, wie Sie eine Metrik auswählen.

Charakterisierung Ihrer Anwendung

Um eine Anwendung richtig zu skalieren, müssen Sie die Bedingungen kennen, unter denen die Anwendung skaliert werden sollte und wann sie skaliert werden sollte. Im Wesentlichen sollte eine Anwendung skaliert werden, wenn prognostiziert wird, dass die Nachfrage die Kapazität übersteigt. Umgekehrt kann eine Anwendung skaliert werden, um Kosten zu sparen, wenn die Ressourcen den Bedarf übersteigen.

Identifizieren einer Nutzungsmetrik

Für eine effektive Skalierung ist es wichtig, eine Kennzahl zu identifizieren, die auf Auslastung oder Sättigung hinweist. Diese Metrik muss die folgenden Eigenschaften aufweisen, um für die Skalierung nützlich zu sein.

- Die Metrik muss mit der Nachfrage korreliert sein. Wenn die Ressourcen konstant gehalten werden, sich die Nachfrage jedoch ändert, muss sich auch der Metrikwert ändern. Die Kennzahl sollte steigen oder sinken, wenn die Nachfrage steigt oder sinkt.

- Der metrische Wert muss proportional zur Kapazität skaliert werden. Wenn die Nachfrage konstant bleibt, muss das Hinzufügen weiterer Ressourcen zu einer proportionalen Änderung des metrischen Werts führen. Eine Verdoppelung der Anzahl der Aufgaben sollte also zu einer Verringerung der Metrik um 50% führen.

Der beste Weg, eine Nutzungsmetrik zu ermitteln, sind Belastungstests in einer Vorproduktionsumgebung, wie z. B. einer Staging-Umgebung. Kommerzielle und Open-Source-Lösungen für Lasttests sind weit verbreitet. Diese Lösungen können in der Regel entweder synthetische Last erzeugen oder echten Benutzerverkehr simulieren.

Um mit dem Auslastungstest zu beginnen, sollten Sie zunächst Dashboards für die Nutzungsmetriken Ihrer Anwendung erstellen. Zu diesen Metriken gehören CPU-Auslastung, Speicherauslastung, I/O-Operationen, I/O-Warteschlangentiefe und Netzwerkdurchsatz. Sie können diese Metriken mit einem Service wie CloudWatch Container Insights sammeln. Oder nutzen Sie dazu Amazon Managed Service for Prometheus zusammen mit Amazon Managed Grafana. Stellen Sie während dieses Vorgangs sicher, dass Sie Kennzahlen zu den Antwortzeiten oder den Abschlussquoten Ihrer Anwendung erfassen und grafisch darstellen.

Beginnen Sie beim Auslastungstest mit einer geringen Anfrage- oder Stelleneinführungsrate. Halten Sie diese Rate mehrere Minuten lang konstant, damit sich Ihre Bewerbung aufwärmen kann. Erhöhen Sie dann langsam die Geschwindigkeit und halten Sie sie einige Minuten lang konstant. Wiederholen Sie diesen Zyklus und erhöhen Sie die Rate jedes Mal, bis die Antwort- oder Abschlusszeiten Ihrer Anwendung zu langsam sind, um Ihre Service Level Objectives (SLOs) zu erreichen.

Untersuchen Sie beim Auslastungstest die einzelnen Nutzungskennzahlen. Die Metriken, die mit der Auslastung steigen, eignen sich am besten als Ihre besten Nutzungskennzahlen.

Identifizieren Sie als Nächstes die Ressource, deren Sättigung erreicht ist. Untersuchen Sie gleichzeitig auch die Nutzungskennzahlen, um festzustellen, welche sich auf hoher Ebene zuerst abflachen. Oder untersuchen Sie, welches System seinen Höhepunkt erreicht und dann zuerst Ihre Anwendung zum Absturz bringt. Wenn die CPU-Auslastung beispielsweise von 0 auf 70 bis 80% steigt, wenn Sie mehr Last hinzufügen, dann kann man mit Sicherheit sagen, dass die CPU ausgelastet ist. Je nach CPU-Architektur erreicht sie möglicherweise nie 100%. Gehen Sie beispielsweise davon aus, dass die Speicherauslastung zunimmt, wenn Sie mehr Last hinzufügen, und Ihre Anwendung dann plötzlich abstürzt, wenn sie das Speicherlimit für Aufgaben oder Amazon EC2 EC2-Instances erreicht. In dieser Situation ist es wahrscheinlich der Fall, dass der Speicher vollständig verbraucht wurde. Möglicherweise werden von Ihrer Anwendung mehrere Ressourcen verbraucht. Wählen Sie daher die Metrik aus, die die Ressource darstellt, die zuerst erschöpft ist.

Versuchen Sie abschließend erneut, die Auslastung zu testen, nachdem Sie die Anzahl der Aufgaben oder Amazon EC2 EC2-Instances verdoppelt haben. Gehen Sie davon aus, dass die Schlüsselkennzahl halb so schnell wie zuvor zunimmt oder abnimmt. Wenn dies der Fall ist, ist die Metrik proportional zur Kapazität. Dies ist eine gute Nutzungsmetrik für Auto Scaling.

Betrachten Sie nun dieses hypothetische Szenario. Nehmen wir an, Sie führen einen Lasttest für eine Anwendung durch und stellen fest, dass die CPU-Auslastung bei 100 Anfragen pro Sekunde irgendwann 80% erreicht. Wenn mehr Last hinzugefügt wird, erhöht sich die CPU-Auslastung nicht mehr. Dadurch reagiert Ihre Anwendung jedoch langsamer. Dann führen Sie den Auslastungstest erneut durch und verdoppeln dabei die Anzahl der Aufgaben, halten aber die Geschwindigkeit auf dem vorherigen Spitzenwert. Wenn Sie feststellen, dass die durchschnittliche CPU-Auslastung auf etwa 40% sinkt, ist die durchschnittliche CPU-Auslastung ein guter Kandidat für eine Skalierungsmetrik. Bleibt die CPU-Auslastung dagegen bei 80%, nachdem die Anzahl der Aufgaben erhöht wurde, ist die durchschnittliche CPU-Auslastung keine gute Skalierungsmetrik. In diesem Fall sind weitere Untersuchungen erforderlich, um eine geeignete Metrik zu finden.

Allgemeine Anwendungsmodelle und Skalierungseigenschaften

Es wird Software aller Art verwendet AWS. Viele Workloads sind selbst entwickelt, während andere auf beliebter Open-Source-Software basieren. Unabhängig davon, woher sie stammen, haben wir einige gängige Entwurfsmuster für Dienste beobachtet. Wie effektiv skaliert werden kann, hängt zu einem großen Teil vom Muster ab.

Der effiziente CPU-gebundene Server

Der effiziente CPU-gebundene Server nutzt fast keine anderen Ressourcen als den CPU- und Netzwerkdurchsatz. Jede Anfrage kann von der Anwendung alleine bearbeitet werden. Anfragen hängen nicht von anderen Diensten wie Datenbanken ab. Die Anwendung kann Hunderttausende von gleichzeitigen Anfragen verarbeiten und dafür mehrere CPUs effizient nutzen. Jede Anfrage wird entweder von einem dedizierten Thread mit geringem Speicheraufwand bedient, oder es gibt eine asynchrone Ereignisschleife, die auf jeder CPU läuft, die Anfragen bearbeitet. Jedes Replikat der Anwendung ist gleichermaßen in der Lage, eine Anfrage zu verarbeiten. Die einzige Ressource, die vor der CPU aufgebraucht sein könnte, ist die Netzwerkbandbreite. Bei CPU-gebundenen Diensten macht die Speicherauslastung selbst bei Spitzendurchsatz nur einen Bruchteil der verfügbaren Ressourcen aus.

Diese Art von Anwendung eignet sich für CPU-basiertes Auto-Scaling. Die Anwendung bietet maximale Flexibilität in Bezug auf die Skalierung. Es kann vertikal skaliert werden, indem größere Amazon EC2 EC2-Instances oder Fargate-vCPUs bereitgestellt werden. Und es kann auch horizontal

skaliert werden, indem weitere Replikate hinzugefügt werden. Durch das Hinzufügen weiterer Replikate oder die Verdoppelung der Instance-Größe wird die durchschnittliche CPU-Auslastung im Verhältnis zur Kapazität um die Hälfte reduziert.

Wenn Sie Amazon EC2 EC2-Kapazität für diese Anwendung verwenden, sollten Sie erwägen, sie auf rechenoptimierten Instances wie der `c5` OR-Familie zu platzieren. `c6g`

Der effiziente speichergebundene Server

Der effiziente speichergebundene Server weist pro Anforderung eine beträchtliche Menge an Speicher zu. Bei maximaler Parallelität, aber nicht unbedingt bei Durchsatz, wird der Arbeitsspeicher erschöpft, bevor die CPU-Ressourcen aufgebraucht sind. Der einer Anforderung zugeordnete Speicher wird freigegeben, wenn die Anforderung endet. Zusätzliche Anfragen können akzeptiert werden, solange Speicherplatz verfügbar ist.

Diese Art von Anwendung eignet sich für speicherbasiertes Auto-Scaling. Die Anwendung bietet maximale Flexibilität in Bezug auf die Skalierung. Es kann sowohl vertikal skaliert werden, indem ihm größere Amazon EC2- oder Fargate-Speicherressourcen zur Verfügung gestellt werden. Und es kann auch horizontal skaliert werden, indem weitere Replikate hinzugefügt werden. Durch das Hinzufügen weiterer Replikate oder die Verdoppelung der Instance-Größe kann die durchschnittliche Speicherauslastung im Verhältnis zur Kapazität um die Hälfte reduziert werden.

Wenn Sie Amazon EC2 EC2-Kapazität für diese Anwendung verwenden, sollten Sie erwägen, sie auf speicheroptimierten Instances wie der `r5` OR-Familie zu platzieren. `r6g`

Einige speichergebundene Anwendungen geben den Speicher, der einer Anforderung zugeordnet ist, nicht frei, wenn diese beendet ist, sodass eine Verringerung der Parallelität nicht zu einer Verringerung des verwendeten Speichers führt. Aus diesem Grund empfehlen wir nicht, die speicherbasierte Skalierung zu verwenden.

Der Worker-basierte Server

Der Worker-basierte Server verarbeitet nacheinander eine Anfrage für jeden einzelnen Worker-Thread. Bei den Worker-Threads kann es sich um Lightweight-Threads wie POSIX-Threads handeln. Sie können auch Threads mit höherem Gewicht sein, wie z. B. UNIX-Prozesse. Unabhängig davon, um welchen Thread es sich handelt, gibt es immer eine maximale Parallelität, die die Anwendung unterstützen kann. Normalerweise wird das Parallelitätslimit proportional zu den verfügbaren Speicherressourcen festgelegt. Wenn das Parallelitätslimit erreicht ist, werden zusätzliche Anfragen in eine Backlog-Warteschlange gestellt. Wenn die Backlog-Warteschlange überläuft, werden

zusätzliche eingehende Anfragen sofort abgelehnt. Zu den gängigen Anwendungen, die diesem Muster entsprechen, gehören Apache Webserver und Gunicorn.

Die Parallelität von Anfragen ist normalerweise die beste Metrik für die Skalierung dieser Anwendung. Da es für jedes Replikat ein Limit für die Parallelität gibt, ist es wichtig, eine Skalierung vorzunehmen, bevor das durchschnittliche Limit erreicht wird.

Der beste Weg, Metriken zur Parallelität von Anfragen zu erhalten, besteht darin, sie von Ihrer Anwendung melden zu lassen. CloudWatch Jedes Replikat Ihrer Anwendung kann die Anzahl der gleichzeitigen Anfragen als benutzerdefinierte Metrik mit hoher Frequenz veröffentlichen. Wir empfehlen, die Frequenz auf mindestens einmal pro Minute festzulegen. Nachdem mehrere Berichte gesammelt wurden, können Sie die durchschnittliche Parallelität als Skalierungsmetrik verwenden. Diese Metrik wird berechnet, indem die gesamte Parallelität durch die Anzahl der Replikate dividiert wird. Wenn beispielsweise die Gesamtzahl der Parallelität 1000 beträgt und die Anzahl der Replikate 10 beträgt, beträgt die durchschnittliche Parallelität 100.

Wenn Ihre Anwendung hinter einem Application Load Balancer steht, können Sie die `ActiveConnectionCount` Metrik für den Load Balancer auch als Faktor in der Skalierungsmetrik verwenden. Die `ActiveConnectionCount` Metrik muss durch die Anzahl der Replikate dividiert werden, um einen Durchschnittswert zu erhalten. Der Durchschnittswert muss für die Skalierung verwendet werden, nicht der Rohzählwert.

Damit dieses Design optimal funktioniert, sollte die Standardabweichung der Antwortlatenz bei niedrigen Anforderungsraten gering sein. Wir empfehlen, dass in Zeiten geringer Nachfrage die meisten Anfragen innerhalb kurzer Zeit beantwortet werden, und es gibt nicht viele Anfragen, deren Beantwortung deutlich länger als der Durchschnitt dauert. Die durchschnittliche Antwortzeit sollte in der Nähe des 95. Perzentils der Antwortzeit liegen. Andernfalls kann es zu Warteschlangenüberläufen kommen. Dies führt zu Fehlern. Wir empfehlen, dass Sie bei Bedarf zusätzliche Replikate bereitstellen, um das Risiko eines Überlaufs zu minimieren.

Der wartende Server

Der wartende Server verarbeitet jede Anfrage in gewissem Maße, ist jedoch in hohem Maße von einem oder mehreren nachgelagerten Diensten abhängig, damit er funktioniert. Containeranwendungen nutzen häufig stark nachgelagerte Dienste wie Datenbanken und andere API-Dienste. Es kann einige Zeit dauern, bis diese Dienste reagieren, insbesondere in Szenarien mit hoher Kapazität oder hoher Parallelität. Dies liegt daran, dass diese Anwendungen in der Regel nur wenige CPU-Ressourcen verwenden und gleichzeitig den verfügbaren Arbeitsspeicher maximal ausnutzen.

Der Wartedienst eignet sich entweder für das speichergebundene Servermuster oder das Worker-basierte Servermuster, je nachdem, wie die Anwendung konzipiert ist. Wenn die Parallelität der Anwendung nur durch den Arbeitsspeicher begrenzt ist, sollte die durchschnittliche Speicherauslastung als Skalierungsmetrik verwendet werden. Wenn die Parallelität der Anwendung auf einem Worker-Limit basiert, sollte die durchschnittliche Parallelität als Skalierungsmetrik verwendet werden.

Der Java-basierte Server

Wenn Ihr Java-basierter Server CPU-gebunden ist und proportional zu den CPU-Ressourcen skaliert, ist er möglicherweise für das effiziente CPU-gebundene Servermuster geeignet. Wenn das der Fall ist, könnte die durchschnittliche CPU-Auslastung als Skalierungsmetrik geeignet sein. Viele Java-Anwendungen sind jedoch nicht CPU-gebunden, was ihre Skalierung schwierig macht.

Für eine optimale Leistung empfehlen wir, dem Java Virtual Machine (JVM) -Heap so viel Speicher wie möglich zuzuweisen. Neuere Versionen der JVM, einschließlich Java 8 Update 191 oder höher, legen die Heap-Größe automatisch so groß wie möglich fest, damit sie in den Container passt. Das bedeutet, dass in Java die Speicherauslastung selten proportional zur Anwendungsauslastung ist. Mit steigender Anforderungsrate und Parallelität bleibt die Speicherauslastung konstant. Aus diesem Grund empfehlen wir nicht, Java-basierte Server auf der Grundlage der Speicherauslastung zu skalieren. Stattdessen empfehlen wir in der Regel, je nach CPU-Auslastung zu skalieren.

In einigen Fällen kommt es bei Java-basierten Servern zu einer Heap-Erschöpfung, bevor die CPU-Auslastung erreicht wird. Wenn Ihre Anwendung bei hoher Parallelität zur Heap-Erschöpfung neigt, sind durchschnittliche Verbindungen die beste Skalierungsmetrik. Wenn Ihre Anwendung bei hohem Durchsatz zur Heap-Erschöpfung neigt, ist die durchschnittliche Anforderungsrate die beste Skalierungsmetrik.

Server, die andere Laufzeiten verwenden, bei denen Müll gesammelt wurde

Viele Serveranwendungen wie .NET und Ruby basieren auf Laufzeiten, die Garbage-Collection durchführen. Diese Serveranwendungen passen möglicherweise in eines der zuvor beschriebenen Muster. Wie bei Java empfehlen wir jedoch nicht, diese Anwendungen auf der Grundlage des Speichers zu skalieren, da ihre beobachtete durchschnittliche Speicherauslastung oft nicht mit dem Durchsatz oder der Parallelität korreliert.

Für diese Anwendungen empfehlen wir, die CPU-Auslastung entsprechend zu skalieren, wenn die Anwendung CPU-gebunden ist. Andernfalls empfehlen wir, die Skalierung auf der Grundlage Ihrer Lasttestergebnisse auf der Grundlage des durchschnittlichen Durchsatzes oder der durchschnittlichen Parallelität durchzuführen.

Jobprozessoren

Viele Workloads beinhalten asynchrone Auftragsverarbeitung. Dazu gehören Anwendungen, die Anfragen nicht in Echtzeit empfangen, sondern stattdessen eine Arbeitswarteschlange abonnieren, um Jobs zu erhalten. Für diese Art von Anwendungen ist die richtige Skalierungsmetrik fast immer die Warteschlangentiefe. Ein Anstieg der Warteschlange ist ein Hinweis darauf, dass die ausstehende Arbeit die Verarbeitungskapazität übersteigt, wohingegen eine leere Warteschlange darauf hindeutet, dass mehr Kapazität als zu erledigende Arbeit vorhanden ist.

AWS Messaging-Dienste wie Amazon SQS und Amazon Kinesis Data Streams bieten CloudWatch Metriken, die für die Skalierung verwendet werden können. Für Amazon SQS `ApproximateNumberOfMessagesVisible` ist dies die beste Metrik. Für Kinesis Data Streams sollten Sie die `MillisBehindLatest` Metrik verwenden, die von der Kinesis Client Library (KCL) veröffentlicht wurde. Diese Metrik sollte für alle Verbraucher gemittelt werden, bevor sie für die Skalierung verwendet wird.

Kapazität und Verfügbarkeit

Die Verfügbarkeit von Anwendungen ist entscheidend für ein fehlerfreies Erlebnis und für die Minimierung der Anwendungslatenz. Die Verfügbarkeit hängt davon ab, ob Ressourcen verfügbar sind und über ausreichend Kapazität verfügen, um den Bedarf zu decken. AWS bietet mehrere Mechanismen zur Verwaltung der Verfügbarkeit. Für Anwendungen, die auf Amazon ECS gehostet werden, gehören dazu Autoscaling und Availability Zones (AZs). Autoscaling verwaltet die Anzahl der Aufgaben oder Instances auf der Grundlage von Metriken, die Sie definieren, während Availability Zones es Ihnen ermöglichen, Ihre Anwendung an isolierten, aber geografisch nahe gelegenen Standorten zu hosten.

Wie bei der Aufgabengröße stellen Kapazität und Verfügbarkeit gewisse Kompromisse dar, die Sie berücksichtigen müssen. Im Idealfall wäre die Kapazität perfekt auf die Nachfrage abgestimmt. Es würde immer gerade genug Kapazität zur Verfügung stehen, um Anfragen zu bearbeiten und Jobs so zu bearbeiten, dass Service Level Objectives (SLOs), einschließlich einer niedrigen Latenz und Fehlerrate, erfüllt werden. Die Kapazität wäre niemals zu hoch, was zu übermäßigen Kosten führen würde, und sie würde auch niemals zu niedrig sein, was zu hohen Latenzen und Fehlerraten führen würde.

Autoscaling ist ein latenter Prozess. Zunächst müssen Echtzeit-Metriken bereitgestellt werden. CloudWatch Anschließend müssen sie für die Analyse aggregiert werden, was je nach Granularität der Metrik bis zu mehreren Minuten dauern kann. CloudWatch vergleicht die Metriken mit den

Alarmschwellenwerten, um einen Mangel oder Überschuss an Ressourcen zu ermitteln. Um Instabilität zu vermeiden, sollten Sie Alarme so konfigurieren, dass der eingestellte Schwellenwert einige Minuten lang überschritten werden muss, bevor der Alarm ausgelöst wird. Außerdem nimmt es Zeit in Anspruch, neue Aufgaben bereitzustellen und Aufgaben zu beenden, die nicht mehr benötigt werden.

Aufgrund dieser potenziellen Verzögerungen im beschriebenen System ist es wichtig, dass Sie durch Überprovisionierung einen gewissen Spielraum wahren. Dies kann dazu beitragen, kurzfristigen Nachfrageschüben Rechnung zu tragen. Auf diese Weise kann Ihre Anwendung auch zusätzliche Anfragen bearbeiten, ohne dass eine Überlastung eintritt. Als bewährte Methode können Sie Ihr Skalierungsziel zwischen 60 und 80% der Auslastung festlegen. Auf diese Weise kann Ihre Anwendung zusätzliche Bedarfsspitzen besser bewältigen, während zusätzliche Kapazität noch bereitgestellt wird.

Ein weiterer Grund, warum wir eine Überversorgung empfehlen, besteht darin, dass Sie schnell auf Ausfälle in der Availability Zone reagieren können. AWS empfiehlt, dass Produktionsworkloads von mehreren Availability Zones aus bedient werden. Dies liegt daran, dass Ihre Aufgaben, die in den verbleibenden Availability Zones ausgeführt werden, auch bei einem Ausfall der Availability Zone den Bedarf decken können. Wenn Ihre Anwendung in zwei Availability Zones ausgeführt wird, müssen Sie die Anzahl Ihrer normalen Aufgaben verdoppeln. Auf diese Weise können Sie bei einem möglichen Ausfall sofort Kapazität bereitstellen. Wenn Ihre Anwendung in drei Availability Zones ausgeführt wird, empfehlen wir, dass Sie das 1,5-fache Ihrer normalen Task-Anzahl ausführen. Führen Sie also drei Aufgaben für jeweils zwei Aufgaben aus, die für die normale Ausführung erforderlich sind.

Maximierung der Skalierungsgeschwindigkeit

Autoscaling ist ein reaktiver Prozess, dessen Wirksamkeit einige Zeit in Anspruch nimmt. Es gibt jedoch einige Möglichkeiten, den Zeitaufwand für die Skalierung zu minimieren.

Minimiert die Bildgröße. Das Herunterladen und Entpacken größerer Bilder aus einem Bild-Repository dauert länger. Wenn Sie also die Bildgrößen kleiner halten, verringert sich die Zeit, die ein Container benötigt, um zu starten. Um die Bildgröße zu reduzieren, können Sie die folgenden spezifischen Empfehlungen befolgen:

- Wenn Sie eine statische Binärdatei erstellen oder Golang verwenden können, erstellen Sie Ihren `FROM` Image-Scratch und fügen Sie nur Ihre Binäranwendung in das resultierende Image ein.
- Verwenden Sie minimierte Basis-Images von Upstream-Distributionsanbietern wie Amazon Linux oder Ubuntu.

- Nehmen Sie keine Build-Artefakte in Ihr endgültiges Image auf. Die Verwendung mehrstufiger Builds kann dabei helfen.
- Kompakte `RUN` Stufen, wo immer möglich. `RUN`In jeder Phase wird eine neue Bildebene erstellt, was zu einem zusätzlichen Roundtrip zum Herunterladen der Ebene führt. Eine einzelne `RUN` Phase, in der mehrere Befehle miteinander verknüpft sind, && hat weniger Ebenen als eine Phase mit mehreren `RUN` Stufen.
- Wenn Sie Daten, wie z. B. ML-Inferenzdaten, in Ihr endgültiges Bild aufnehmen möchten, schließen Sie nur die Daten ein, die für den Start und die Bereitstellung des Datenverkehrs erforderlich sind. Wenn Sie Daten bei Bedarf von Amazon S3 oder einem anderen Speicher abrufen, ohne den Service zu beeinträchtigen, speichern Sie Ihre Daten stattdessen an diesen Orten.

Halten Sie Ihre Bilder in der Nähe. Je höher die Netzwerklatenz, desto länger dauert es, das Bild herunterzuladen. Hosten Sie Ihre Bilder in einem Repository in derselben AWS Region, in der sich Ihr Workload befindet. Amazon ECR ist ein leistungsstarkes Image-Repository, das in jeder Region verfügbar ist, in der Amazon ECS verfügbar ist. Vermeiden Sie es, das Internet oder einen VPN-Link zu nutzen, um Container-Images herunterzuladen. Das Hosten Ihrer Bilder in derselben Region verbessert die allgemeine Zuverlässigkeit. Dadurch wird das Risiko von Netzwerkverbindungs- und Verfügbarkeitsproblemen in einer anderen Region gemindert. Alternativ können Sie auch die regionsübergreifende Amazon ECR-Replikation implementieren, um dies zu unterstützen.

Reduzieren Sie die Schwellenwerte für die Integritätsprüfung des Load Balancers. Load Balancer führen Integritätsprüfungen durch, bevor sie Datenverkehr an Ihre Anwendung senden. Die Standardkonfiguration für Integritätsprüfungen für eine Zielgruppe kann 90 Sekunden oder länger dauern. Währenddessen überprüft der Load Balancer den Integritätsstatus und empfängt Anfragen. Wenn Sie das Intervall für Integritätsprüfungen und die Anzahl der Schwellenwerte verringern, kann Ihre Anwendung den Datenverkehr schneller annehmen und die Belastung anderer Aufgaben verringern.

Ziehen Sie die Kaltstartleistung in Betracht. Einige Anwendungen verwenden Laufzeiten, wie z. B. Java, führen Just-In-Time-Kompilierung (JIT) durch. Der Kompilierungsprozess kann zumindest beim Start die Leistung der Anwendung belegen. Eine Behelfslösung besteht darin, die latenzkritischen Teile Ihres Workloads in Sprachen umzuschreiben, die keine Leistungseinbußen beim Kaltstart zur Folge haben.

Verwenden Sie schrittweise Skalierungsrichtlinien, nicht zielgerichtete Skalierungsrichtlinien. Sie haben mehrere Application-Auto-Scaling-Optionen für Amazon-ECS-Aufgaben. Die Zielverfolgung

ist der am einfachsten zu verwendende Modus. Damit müssen Sie lediglich einen Zielwert für eine Metrik festlegen, z. B. die durchschnittliche CPU-Auslastung. Anschließend verwaltet der Autoscaler automatisch die Anzahl der Aufgaben, die erforderlich sind, um diesen Wert zu erreichen. Mit der schrittweisen Skalierung können Sie schneller auf Bedarfsänderungen reagieren, da Sie die spezifischen Schwellenwerte für Ihre Skalierungsmetriken definieren und festlegen, wie viele Aufgaben hinzugefügt oder entfernt werden müssen, wenn die Schwellenwerte überschritten werden. Und was noch wichtiger ist: Sie können sehr schnell auf Änderungen der Nachfrage reagieren, indem Sie die Zeitspanne, in der ein Schwellenwertalarm überschritten wird, auf ein Minimum reduzieren. Weitere Informationen finden Sie unter Service Auto Scaling im Amazon Elastic Container Service Developer Guide.

Wenn Sie Amazon EC2 EC2-Instances zur Bereitstellung von Cluster-Kapazität verwenden, sollten Sie die folgenden Empfehlungen berücksichtigen:

Verwenden Sie größere Amazon EC2 EC2-Instances und schnellere Amazon EBS-Volumes. Sie können die Geschwindigkeit beim Herunterladen und Vorbereiten von Bildern verbessern, indem Sie eine größere Amazon EC2 EC2-Instance und ein schnelleres Amazon EBS-Volume verwenden. Innerhalb einer bestimmten Amazon EC2 EC2-Instance-Familie steigt der maximale Durchsatz von Netzwerk und Amazon EBS mit zunehmender Instance-Größe (z. B. von `m5.xlarge` bis`m5.2xlarge`). Darüber hinaus können Sie Amazon EBS-Volumes anpassen, um deren Durchsatz und IOPS zu erhöhen. Wenn Sie beispielsweise Volumes verwenden, sollten Sie größere `gp2` Volumes verwenden, die einen höheren Basisdurchsatz bieten. Wenn Sie `gp3` Volumes verwenden, geben Sie bei der Erstellung des Volumes den Durchsatz und die IOPS an.

Verwenden Sie den Bridge-Netzwerkmodus für Aufgaben, die auf Amazon EC2 EC2-Instances ausgeführt werden. Aufgaben, die den `bridge` Netzwerkmodus auf Amazon EC2 verwenden, starten schneller als Aufgaben, die den `awsvpc` Netzwerkmodus verwenden. Wenn der `awsvpc` Netzwerkmodus verwendet wird, fügt Amazon ECS der Instance ein elastic network interface (ENI) hinzu, bevor die Aufgabe gestartet wird. Dies führt zu zusätzlicher Latenz. Bei der Verwendung von Bridge-Netzwerken gibt es jedoch mehrere Kompromisse. Für diese Aufgaben gibt es keine eigene Sicherheitsgruppe, und es gibt einige Auswirkungen auf den Lastenausgleich. Weitere Informationen finden Sie unter Load Balancer-Zielgruppen im Elastic Load Balancing User Guide.

Umgang mit Nachfrageschocks

Bei einigen Anwendungen kommt es zu plötzlichen starken Nachfrageschocks. Dies geschieht aus einer Vielzahl von Gründen: ein Nachrichtenereignis, ein großer Verkauf, ein Medienereignis oder ein anderes Ereignis, das sich viral verbreitet und dazu führt, dass der Traffic in sehr kurzer Zeit schnell

und deutlich zunimmt. Wenn dies nicht geplant ist, kann dies dazu führen, dass die Nachfrage die verfügbaren Ressourcen schnell übersteigt.

Der beste Weg, mit Nachfrageschocks umzugehen, besteht darin, sie zu antizipieren und entsprechend zu planen. Da die automatische Skalierung einige Zeit in Anspruch nehmen kann, empfehlen wir, dass Sie Ihre Anwendung skalieren, bevor der Nachfrageschock einsetzt. Um optimale Ergebnisse zu erzielen, empfehlen wir, einen Geschäftsplan zu erstellen, der eine enge Zusammenarbeit zwischen Teams vorsieht, die einen gemeinsamen Kalender verwenden. Das Team, das die Veranstaltung plant, sollte im Voraus eng mit dem für die Bewerbung zuständigen Team zusammenarbeiten. Dies gibt dem Team genügend Zeit, um einen klaren Terminplan zu haben. Sie können die Kapazität so planen, dass sie vor der Veranstaltung skaliert und nach der Veranstaltung wieder erhöht wird. Weitere Informationen finden Sie unter Geplante Skalierung im Benutzerhandbuch für Application Auto Scaling.

Wenn Sie einen Enterprise Support-Plan haben, sollten Sie auch mit Ihrem Technical Account Manager (TAM) zusammenarbeiten. Ihr TAM kann Ihre Servicekontingenten überprüfen und sicherstellen, dass alle erforderlichen Kontingente erhöht werden, bevor die Veranstaltung beginnt. Auf diese Weise erreichen Sie nicht versehentlich irgendwelche Servicekontingenten. Sie können Ihnen auch helfen, indem sie Dienste wie Load Balancer vorwärmen, um sicherzustellen, dass Ihre Veranstaltung reibungslos abläuft.

Der Umgang mit ungeplanten Nachfrageschocks ist ein schwierigeres Problem. Außerplanmäßige Schocks können, sofern sie eine ausreichend große Amplitude haben, schnell dazu führen, dass die Nachfrage die Kapazität übersteigt. Sie kann auch die Reaktionsfähigkeit der automatischen Skalierung übersteigen. Der beste Weg, sich auf ungeplante Schocks vorzubereiten, besteht darin, zu viele Ressourcen bereitzustellen. Sie müssen über genügend Ressourcen verfügen, um den zu erwartenden maximalen Verkehrsbedarf jederzeit bewältigen zu können.

Die Aufrechterhaltung der maximalen Kapazität im Vorgriff auf ungeplante Nachfrageschocks kann kostspielig sein. Um die Auswirkungen auf die Kosten zu minimieren, suchen Sie nach einer Frühindikatorkennzahl oder einem Ereignis, das darauf hindeutet, dass ein großer Nachfrageschock unmittelbar bevorsteht. Wenn die Kennzahl oder das Ereignis zuverlässig eine deutliche Vorwarnung bietet, beginnen Sie sofort mit dem Skalierungsprozess, wenn das Ereignis eintritt oder wenn die Kennzahl den von Ihnen festgelegten Schwellenwert überschreitet.

Wenn Ihre Anwendung zu plötzlichen, ungeplanten Nachfrageschocks neigt, sollten Sie erwägen, Ihrer Anwendung einen Hochleistungsmodus hinzuzufügen, der unkritische Funktionen einbüßt, aber wichtige Funktionen für einen Kunden beibehält. Gehen Sie beispielsweise davon aus, dass

Ihre Anwendung von der Generierung teurer benutzerdefinierter Antworten zur Bereitstellung einer statischen Antwortseite übergehen kann. In diesem Szenario können Sie den Durchsatz erheblich erhöhen, ohne die Anwendung überhaupt skalieren zu müssen.

Schließlich können Sie erwägen, monolithische Dienste auseinanderzunehmen, um Nachfrageschocks besser bewältigen zu können. Wenn es sich bei Ihrer Anwendung um einen monolithischen Dienst handelt, der teuer in der Ausführung und langsam zu skalieren ist, können Sie möglicherweise leistungskritische Teile extrahieren oder neu schreiben und sie als separate Dienste ausführen. Diese neuen Dienste können dann unabhängig von weniger kritischen Komponenten skaliert werden. Wenn Sie die Flexibilität haben, leistungskritische Funktionen getrennt von anderen Teilen Ihrer Anwendung zu skalieren, können Sie sowohl den Zeitaufwand für die Kapazitätserweiterung reduzieren als auch Kosten sparen.

Cluster-Kapazität

Sie können einem Amazon ECS-Cluster auf verschiedene Weise Kapazität zur Verfügung stellen. Sie können beispielsweise Amazon EC2 EC2-Instances starten und sie beim Start mit dem Amazon ECS-Container-Agenten im Cluster registrieren. Diese Methode kann jedoch schwierig sein, da Sie die Skalierung selbst verwalten müssen. Daher empfehlen wir Ihnen, Amazon ECS-Kapazitätsanbieter zu verwenden. Sie verwalten die Ressourcenskalierung für Sie. Es gibt drei Arten von Kapazitätsanbietern: Amazon EC2, Fargate und Fargate Spot. Weitere Informationen zu Fargate-Kapazitätsanbietern finden Sie unter Amazon ECS-Cluster für Fargate-Workloads vom Starttyp und für den EC2-Starttyp unter Amazon ECS-Cluster für EC2-Workloads vom Starttyp EC2 im Amazon Elastic Container Service Developer Guide.

Die Kapazitätsanbieter Fargate und Fargate Spot übernehmen den Lebenszyklus der Fargate-Aufgaben für Sie. Fargate bietet On-Demand-Kapazität und Fargate Spot Spot-Kapazität. Wenn eine Aufgabe gestartet wird, stellt ECS eine Fargate-Ressource für Sie bereit. Diese Fargate-Ressource enthält die Speicher- und CPU-Einheiten, die direkt den Grenzwerten auf Aufgabenebene entsprechen, die Sie in Ihrer Aufgabendefinition deklariert haben. Jede Aufgabe erhält ihre eigene Fargate-Ressource, wodurch eine 1:1 -Beziehung zwischen der Aufgabe und den Rechenressourcen hergestellt wird.

Aufgaben, die auf Fargate Spot ausgeführt werden, können unterbrochen werden. Unterbrechungen treten nach einer zweiminütigen Warnung auf. Diese treten in Zeiten starker Nachfrage auf. Fargate Spot eignet sich am besten für unterbrechungstolerante Workloads wie Batch-Jobs, Entwicklungs- oder Staging-Umgebungen. Sie eignen sich auch für jedes andere Szenario, in dem hohe Verfügbarkeit und geringe Latenz keine Voraussetzung sind.

Sie können Fargate Spot-Aufgaben zusammen mit Fargate-On-Demand-Aufgaben ausführen. Wenn Sie sie zusammen verwenden, erhalten Sie Bereitstellungskapazität zu geringeren Kosten.

ECS kann auch die Amazon EC2 EC2-Instance-Kapazität für Ihre Aufgaben verwalten. Jeder Amazon EC2-Kapazitätsanbieter ist einer von Ihnen angegebenen Amazon EC2 Auto Scaling Scaling-Gruppe zugeordnet. Wenn Sie den Amazon EC2 Capacity Provider verwenden, behält ECS Cluster Auto Scaling die Größe der Amazon EC2 Auto Scaling-Gruppe bei, um sicherzustellen, dass alle geplanten Aufgaben platziert werden können.

Auswahl der Fargate-Aufgabengrößen

Bei AWS Fargate Aufgabendefinitionen müssen Sie CPU und Arbeitsspeicher auf Aufgabenebene angeben und müssen keinen Overhead berücksichtigen. Sie können auch die CPU- und Speicherauslastung für Fargate-Aufgaben auf Container-Ebene festlegen. Das ist jedoch optional. Die Ressourcenlimits müssen größer oder gleich den von Ihnen deklarierten Reservierungen sein. In den meisten Fällen können Sie sie auf die Summe der Reservierungen der einzelnen Container festlegen, die in Ihrer Aufgabendefinition deklariert sind. Runden Sie dann die Zahl auf die nächste Fargate-Größe auf. Weitere Informationen zu den verfügbaren Größen finden Sie unter Task CPU and memory im Amazon Elastic Container Service Developer Guide.

Beschleunigung der Cluster-Kapazitätsbereitstellung mit Kapazitätsanbietern auf Amazon EC2

Kunden, die Amazon ECS auf Amazon EC2 ausführen, können Amazon ECS Cluster Auto Scaling (CAS) nutzen, um die Skalierung von Amazon EC2 Auto Scaling Scaling-Gruppen (ASG) zu verwalten. Mit CAS können Sie Amazon ECS so konfigurieren, dass Ihre ASG automatisch skaliert wird, sodass Sie sich ganz auf die Ausführung Ihrer Aufgaben konzentrieren können. Amazon ECS stellt sicher, dass das ASG nach Bedarf ein- und ausgeblendet wird, ohne dass weitere Eingriffe erforderlich sind. Amazon ECS-Kapazitätsanbieter werden verwendet, um die Infrastruktur in Ihrem Cluster zu verwalten, indem sie sicherstellen, dass genügend Container-Instances vorhanden sind, um die Anforderungen Ihrer Anwendung zu erfüllen. Weitere Informationen zur Funktionsweise von Amazon ECS CAS unter der Haube finden Sie unter Deep Dive on Amazon ECS Cluster Auto Scaling.

Da CAS bei der Anpassung der Cluster-Kapazität auf einer CloudWatch basierten Integration mit ASG basiert, hat es eine inhärente Latenz, die mit der Veröffentlichung der CloudWatch Metriken, der Zeit, die die Metrik benötigt, `CapacityProviderReservation` um CloudWatch Alarme zu

verletzen (sowohl hoch als auch niedrig), und der Zeit, die eine neu gestartete Amazon EC2 EC2-Instance zum Aufwärmen benötigt, verbunden. Sie können die folgenden Maßnahmen ergreifen, um Amazon ECS CAS für schnellere Bereitstellungen reaktionsfähiger zu machen:

Skalierung der Größen durch Kapazitätsanbieter

Amazon ECS-Kapazitätsanbieter werden die Container-Instances irgendwann vergrößern oder verkleinern, um den Anforderungen Ihrer Anwendung gerecht zu werden. Die Mindestanzahl von Instances, die Amazon ECS startet, ist standardmäßig auf 1 festgelegt. Dies kann Ihre Bereitstellungen verlängern, wenn mehrere Instances für die Ausführung Ihrer ausstehenden Aufgaben erforderlich sind. Sie können die `minimumScalingStepSize`über die Amazon ECS-API erhöhen, um die Mindestanzahl von Instances zu erhöhen, die Amazon ECS gleichzeitig ein- oder ausskaliert. Ein zu niedriger Wert kann einschränken `maximumScalingStepSize`, wie viele Container-Instances gleichzeitig ein- oder ausgeschaltet werden, was Ihre Bereitstellungen verlangsamen kann.

> **Note**
>
> Diese Konfiguration ist derzeit nur über die APIs `CreateCapacityProvider`oder `UpdateCapacityProvider`verfügbar.

Aufwärmphase der Instanz

Die Instance-Aufwärmphase ist der Zeitraum, nach dem eine neu gestartete Amazon EC2 EC2-Instance zu den CloudWatch Metriken für die Auto Scaling Scaling-Gruppe beitragen kann. Sobald die angegebene Aufwärmphase abgelaufen ist, wird die Instance auf die aggregierten Kennzahlen der ASG angerechnet, und CAS fährt mit der nächsten Berechnungsrunde fort, um die Anzahl der benötigten Instances zu schätzen.

Der Standardwert für `instanceWarmupPeriod`ist 300 Sekunden, den Sie über die `UpdateCapacityProvider`APIs `CreateCapacityProvider`oder auf einen niedrigeren Wert konfigurieren können, um eine reaktionsschnellere Skalierung zu erzielen.

Reservekapazität

Wenn Ihr Kapazitätsanbieter keine Container-Instances für die Platzierung von Aufgaben zur Verfügung hat, muss er die Clusterkapazität erhöhen (skalieren), indem er Amazon EC2 EC2-Instances im laufenden Betrieb startet und wartet, bis sie hochgefahren sind, bevor er Container auf

ihnen starten kann. Dies kann die Startrate von Aufgaben erheblich senken. Sie haben hier zwei Möglichkeiten.

In diesem Fall erhöht sich die effektive Startrate von Aufgaben, wenn Amazon EC2 EC2-Kapazitäten bereits gestartet und bereit sind, Aufgaben auszuführen, vorhanden sind. Sie können die `Target Capacity` Konfiguration verwenden, um anzugeben, dass Sie freie Kapazitäten in Ihren Clustern beibehalten möchten. Wenn Sie beispielsweise einen Wert von 80% festlegen`Target Capacity`, geben Sie an, dass Ihr Cluster jederzeit 20% freie Kapazität benötigt. Dank dieser freien Kapazität können alle eigenständigen Aufgaben sofort gestartet werden, wodurch sichergestellt wird, dass das Starten von Aufgaben nicht gedrosselt wird. Der Nachteil dieses Ansatzes sind potenziell höhere Kosten für die Beibehaltung von Clusterkapazitäten.

Ein alternativer Ansatz, den Sie in Betracht ziehen können, besteht darin, Ihrem Service mehr Spielraum zu verleihen, nicht dem Kapazitätsanbieter. Das bedeutet, dass Sie, anstatt die `Target Capacity` Konfiguration zu reduzieren, um freie Kapazitäten bereitzustellen, die Anzahl der Replikate in Ihrem Service erhöhen können, indem Sie die Ziel-Tracking-Skalierungsmetrik oder die Schwellenwerte für die schrittweise Skalierung des Service Auto Scaling ändern. Beachten Sie, dass dieser Ansatz nur bei stark beanspruchten Workloads hilfreich ist, aber keine Auswirkungen hat, wenn Sie neue Services bereitstellen und zum ersten Mal von 0 auf N Aufgaben wechseln. Weitere Informationen zu den zugehörigen Skalierungsrichtlinien finden Sie unter Target Tracking Scaling Policies oder Step Scaling Policies im Amazon Elastic Container Service Developer Guide.

Auswählen des Amazon-EC2-Instance-Typs

Wenn Sie Amazon EC2 verwenden, um Kapazität für Ihren ECS-Cluster bereitzustellen, können Sie aus einer großen Auswahl an Instance-Typen wählen. Alle Amazon EC2 EC2-Instance-Typen und -Familien sind mit ECS kompatibel.

Um zu ermitteln, welche Instance-Typen Sie verwenden können, entfernen Sie zunächst die Instance-Typen oder Instance-Familien, die den spezifischen Anforderungen Ihrer Anwendung nicht entsprechen. Wenn Ihre Anwendung beispielsweise eine GPU benötigt, können Sie alle Instance-Typen ausschließen, die keine GPU haben. Sie sollten jedoch auch andere Anforderungen berücksichtigen. Denken Sie beispielsweise an die CPU-Architektur, den Netzwerkdurchsatz und ob Instance-Speicher erforderlich ist. Untersuchen Sie als Nächstes die Menge an CPU und Arbeitsspeicher, die von jedem Instance-Typ bereitgestellt werden. Als allgemeine Regel gilt, dass die CPU und der Arbeitsspeicher groß genug sein müssen, um mindestens ein Replikat der Aufgabe aufzunehmen, die Sie ausführen möchten.

Sie können aus den Instance-Typen wählen, die mit Ihrer Anwendung kompatibel sind. Bei größeren Instances können Sie mehr Aufgaben gleichzeitig starten. Und bei kleineren Instances können Sie differenzierter aufskalieren, um Kosten zu sparen. Sie müssen sich nicht für einen einzigen Amazon EC2 EC2-Instance-Typ entscheiden, der für alle Anwendungen in Ihrem Cluster geeignet ist. Stattdessen können Sie mehrere Auto Scaling Scaling-Gruppen erstellen. Jede Gruppe kann einen anderen Instance-Typ haben. Anschließend können Sie für jede Gruppe einen Amazon EC2 Capacity Provider erstellen. Schließlich können Sie in der Kapazitätsanbieter-Strategie für Ihren Service und Ihre Aufgabe den Kapazitätsanbieter auswählen, der Ihren Anforderungen am besten entspricht. Weitere Informationen finden Sie unter Instance-Typen im Amazon-EC2-Benutzerhandbuch zu Linux-Instances.

Verwenden von Amazon EC2 Spot und FARGATE_SPOT

Spot-Kapazität kann im Vergleich zu On-Demand-Instances zu erheblichen Kosteneinsparungen führen. Spot-Kapazität ist überschüssige Kapazität, deren Preis deutlich unter dem Preis für On-Demand-Kapazität oder reservierter Kapazität liegt. Spot-Kapazität eignet sich für Batch-Verarbeitungs- und Machine-Learning-Workloads sowie für Entwicklungs- und Staging-Umgebungen. Generell ist sie für alle Workloads geeignet, die vorübergehende Ausfallzeiten tolerieren.

Seien Sie sich über die folgenden Konsequenzen im Klaren, denn die Spot-Kapazität ist möglicherweise nicht immer verfügbar.

- Erstens ist Spot-Kapazität in Zeiten extrem hoher Nachfrage möglicherweise nicht verfügbar. Dies kann dazu führen, dass der Start von Fargate Spot-Aufgaben und Amazon EC2-Spot-Instances verzögert wird. In diesen Fällen versuchen ECS-Services erneut, Aufgaben zu starten, und Amazon EC2 Auto Scaling Scaling-Gruppen versuchen ebenfalls erneut, Instances zu starten, bis die erforderliche Kapazität verfügbar ist. Fargate und Amazon EC2 ersetzen Spot-Kapazität nicht durch On-Demand-Kapazität.
- Zweitens: Wenn der Gesamtbedarf an Kapazität steigt, können Spot-Instances und -Tasks mit nur einer zweiminütigen Vorwarnung beendet werden. Nach dem Senden der Warnung sollten die Aufgaben gegebenenfalls korrekt heruntergefahren werden, bevor die Instance vollständig beendet wird. Dies trägt dazu bei, die Wahrscheinlichkeit von Fehlern zu minimieren. Weitere Informationen zu einem korrekten Herunterfahren finden Sie unter Korrektes Herunterfahren mit ECS.

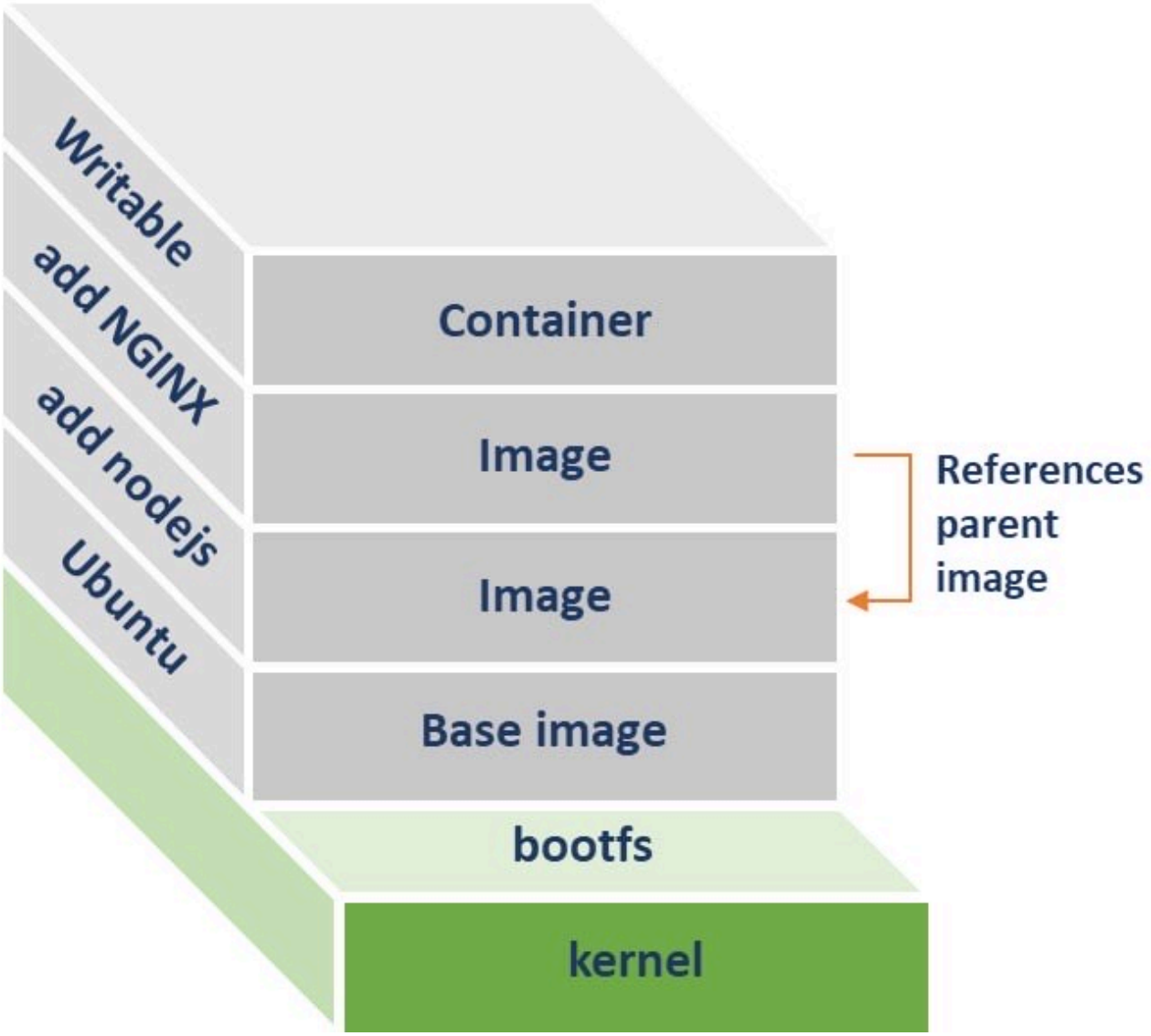
Beachten Sie die folgenden Empfehlungen, um Kapazitätsengpässe vor Ort zu minimieren:

- Verwenden Sie mehrere Regionen und Availability Zones. Die Spot-Kapazität variiert je nach Region und Verfügbarkeitszone. Sie können die Spot-Verfügbarkeit verbessern, indem Sie Ihre Workloads in mehreren Regionen und Availability Zones ausführen. Geben Sie nach Möglichkeit Subnetze in allen Availability Zones in den Regionen an, in denen Sie Ihre Tasks und Instances ausführen.
- Verwenden Sie mehrere Amazon EC2 EC2-Instance-Typen. Wenn Sie Mixed Instance Policies mit Amazon EC2 Auto Scaling verwenden, werden mehrere Instance-Typen in Ihrer Auto Scaling Scaling-Gruppe gestartet. Dadurch wird sichergestellt, dass eine Anfrage nach Spot-Kapazität bei Bedarf erfüllt werden kann. Um die Zuverlässigkeit zu maximieren und die Komplexität zu minimieren, sollten Sie in Ihrer Mixed Instances Policy Instance-Typen mit ungefähr der gleichen Menge an CPU und Arbeitsspeicher verwenden. Diese Instances können aus einer anderen Generation oder aus Varianten desselben Basis-Instance-Typs stammen. Beachten Sie, dass sie möglicherweise zusätzliche Features enthalten, die Sie möglicherweise nicht benötigen. Ein Beispiel für eine solche Liste könnte m4.large, m5.large, m5a.large, m5d.large, m5n.large, m5dn.large und m5ad.large enthalten. Weitere Informationen finden Sie unter Auto-Scaling-Gruppen mit mehreren Instance-Typen und Kaufoptionen im Amazon EC2 Auto Scaling-Benutzerhandbuch.
- Verwenden Sie die kapazitätsoptimierte Spot-Zuweisungsstrategie. Mit Amazon EC2 Spot können Sie zwischen kapazitäts- und kostenoptimierten Zuweisungsstrategien wählen. Wenn Sie beim Start einer neuen Instance die kapazitätsoptimierte Strategie wählen, wählt Amazon EC2 Spot den Instance-Typ mit der größten Verfügbarkeit in der ausgewählten Availability Zone aus. Dies trägt dazu bei, die Wahrscheinlichkeit zu verringern, dass die Instance kurz nach ihrem Start beendet wird.

Bewährte Methoden — Persistenter Speicher

Sie können Amazon ECS verwenden, um statusbehaftete containerisierte Anwendungen in großem Umfang auszuführen, indem Sie AWS Speicherdienste wie Amazon EFS, Amazon EBS oder FSx for Windows File Server verwenden, die Datenpersistenz für inhärent kurzlebige Container bereitstellen. Der Begriff Datenpersistenz bedeutet, dass die Daten selbst den Prozess überdauern, durch den sie erstellt wurden. Datenpersistenz AWS wird durch die Entkopplung von Rechen- und Speicherdiensten erreicht. Ähnlich wie Amazon EC2 können Sie auch Amazon ECS verwenden, um den Lebenszyklus Ihrer containerisierten Anwendungen von den Daten zu entkoppeln, die sie verbrauchen und produzieren. Mithilfe von AWS Speicherservices können Amazon ECS-Aufgaben Daten auch nach dem Beenden von Aufgaben beibehalten.

Standardmäßig speichern Container die von ihnen produzierten Daten nicht. Wenn ein Container beendet wird, werden die Daten, die er in seine beschreibbare Ebene geschrieben hat, zusammen mit dem Container zerstört. Dadurch eignen sich Container für zustandslose Anwendungen, die keine Daten lokal speichern müssen. Containerisierte Anwendungen, die Datenpersistenz erfordern, benötigen ein Speicher-Backend, das nicht zerstört wird, wenn der Container der Anwendung beendet wird.



Ein Container-Image besteht aus einer Reihe von Ebenen. Jede Ebene stellt eine Anweisung in der Dockerfile dar, aus der das Image erstellt wurde. Jede Ebene ist schreibgeschützt, mit Ausnahme des Containers. Das heißt, wenn Sie einen Container erstellen, wird eine beschreibbare Ebene über den darunterliegenden Ebenen hinzugefügt. Alle Dateien, die der Container erstellt, löscht oder ändert, werden in die beschreibbare Ebene geschrieben. Wenn der Container beendet wird, wird gleichzeitig auch die beschreibbare Ebene gelöscht. Ein neuer Container, der dasselbe Bild verwendet, hat eine eigene beschreibbare Ebene. Diese Ebene enthält keine Änderungen. Daher müssen die Daten eines Containers immer außerhalb der beschreibbaren Ebene des Containers gespeichert werden.

Mit Amazon ECS können Sie Stateful-Container mithilfe von Volumes auf vier Arten ausführen. Amazon ECS für Linux-Container ist nativ in Amazon EFS integriert. Amazon ECS für Windows-

und Linux-Aufgaben, die auf Container-Instances oder Fargate gehostet werden, sind nativ in Amazon EBS integriert. Amazon EBS-Volumes, die an eigenständige Aufgaben angehängt sind, können dauerhaft gespeichert werden, indem Sie auf setzen`deleteOnTermination`. `false` Sowohl Windows- als auch Linux-Amazon-EC2-Container-Instances können auch Docker-Volumes verwenden, die in Amazon EBS integriert sind. Für Windows-Container integriert sich Amazon ECS in FSx for Windows File Server, um persistenten Speicher bereitzustellen.

Themen

Wählen Sie den richtigen Speichertyp für Ihre Container

Anwendungen, die in einem Amazon ECS-Cluster ausgeführt werden, können eine Vielzahl von AWS Speicherdiensten und Drittanbieterprodukten verwenden, um persistenten Speicher für statusbehaftete Workloads bereitzustellen. Sie sollten Ihr Speicher-Backend für Ihre containerisierte Anwendung auf der Grundlage der Architektur und der Speicheranforderungen Ihrer Anwendung auswählen. Weitere Informationen zu AWS Speicherdiensten finden Sie unter Cloud Storage on. AWS

Für Amazon ECS-Cluster, die Linux-Instances oder Linux-Container enthalten, die mit Fargate verwendet werden, integriert sich Amazon ECS in Amazon EFS, um Containerspeicher bereitzustellen. Der deutlichste Unterschied zwischen Amazon EFS und Amazon EBS besteht darin, dass Sie ein Amazon EFS-Dateisystem gleichzeitig für Tausende von Amazon ECS-Aufgaben bereitstellen können. Im Gegensatz dazu unterstützen Amazon EBS-Volumes keinen gleichzeitigen Zugriff. Vor diesem Hintergrund ist Amazon EFS die empfohlene Speicheroption für containerisierte Anwendungen, die horizontal skaliert werden. Dies liegt daran, dass es Parallelität unterstützt. Amazon EFS speichert Ihre Daten redundant in mehreren Availability Zones und bietet unabhängig von der Availability Zone Zugriff mit geringer Latenz von Amazon ECS-Aufgaben aus.

Angenommen, Sie haben eine Anwendung wie eine Transaktionsdatenbank, die eine Latenz von unter einer Millisekunde erfordert und bei horizontaler Skalierung kein gemeinsam genutztes Dateisystem benötigt. Für eine solche Anwendung empfehlen wir die Verwendung von Amazon EBS-Volumes für persistenten Speicher. Amazon ECS unterstützt Amazon EBS-Volumes für Aufgaben,

die auf Amazon EC2 und Fargate gehostet werden. Bevor Sie Amazon EBS-Volumes mit Amazon ECS-Aufgaben verwenden, können Sie entweder Amazon EBS-Volumes an Container-Instances anhängen und Volumes getrennt vom Lebenszyklus der Aufgabe verwalten, oder Sie können ein Amazon EBS-Volume so konfigurieren, dass es während der Bereitstellung eines Service oder einer eigenständigen Aufgabe an eine Amazon ECS-Aufgabe angehängt wird.

Für Cluster, die Windows-Instanzen enthalten, bietet FSx for Windows File Server persistenten Speicher für Container. FSx for Windows File Server Server-Dateisysteme unterstützt Multi-AZ-Bereitstellungen. Durch diese Bereitstellungen können Sie ein Dateisystem gemeinsam mit Amazon ECS-Aufgaben nutzen, die in mehreren Availability Zones ausgeführt werden.

Sie können den Amazon EC2-Instance-Speicher auch für die Datenpersistenz für Amazon ECS-Aufgaben verwenden, die auf Amazon EC2 mithilfe von Bind-Mounts oder Docker-Volumes gehostet werden. Bei der Verwendung von Bind-Mounts oder Docker-Volumes speichern Container Daten im Container-Instance-Dateisystem. Eine Einschränkung bei der Verwendung eines Host-Dateisystems für den Containerspeicher besteht darin, dass die Daten jeweils nur auf einer einzelnen Container-Instance verfügbar sind. Das bedeutet, dass Container nur auf dem Host ausgeführt werden können, auf dem sich die Daten befinden. Daher wird die Verwendung von Hostspeicher nur in Szenarien empfohlen, in denen die Datenreplikation auf Anwendungsebene erfolgt.

Amazon EFS-Volumes

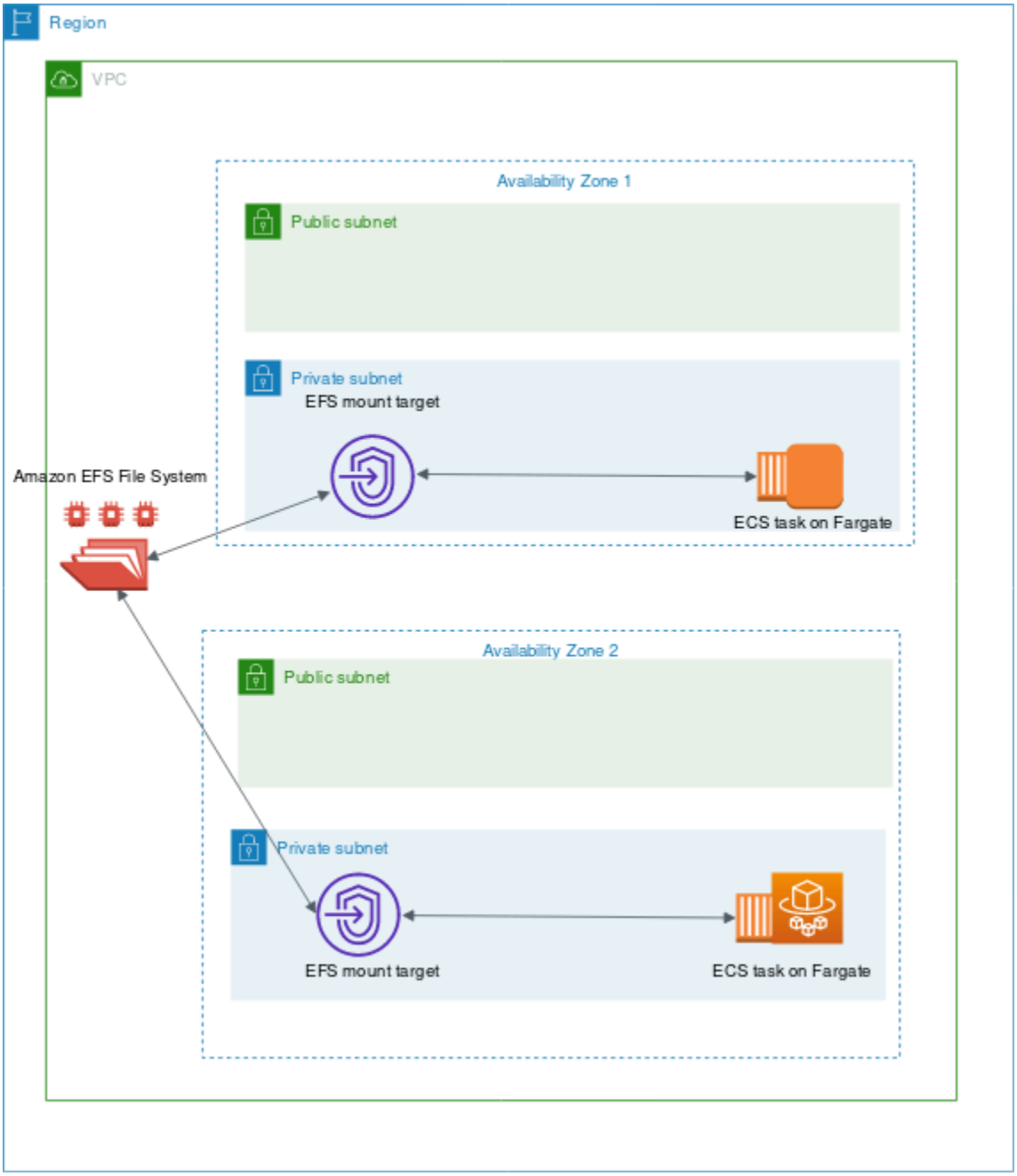
Amazon Elastic File System (Amazon EFS) bietet ein einfaches, skalierbares, vollständig verwaltetes elastisches NFS-Dateisystem. Es ist so konzipiert, dass es bei Bedarf auf Petabyte skaliert werden kann, ohne Anwendungen zu unterbrechen. Es kann beim Hinzufügen und Entfernen von Dateien vergrößert oder verkleinert werden.

Sie können Ihre statusbehafteten Anwendungen in Amazon ECS ausführen, indem Sie Amazon EFS-Volumes verwenden, um persistenten Speicher bereitzustellen. Amazon ECS-Aufgaben, die auf Amazon EC2 EC2-Instances oder Fargate mit Plattformversion `1.4.0` und höher ausgeführt werden, können ein vorhandenes Amazon EFS-Dateisystem mounten. Da mehrere Container gleichzeitig ein Amazon EFS-Dateisystem bereitstellen und darauf zugreifen können, haben Ihre Aufgaben Zugriff auf denselben Datensatz, unabhängig davon, wo sie gehostet werden.

Um ein Amazon EFS-Dateisystem in Ihrem Container zu mounten, können Sie in Ihrer Amazon ECS-Aufgabendefinition auf das Amazon EFS-Dateisystem und den Container-Einhängepunkt verweisen. Im Folgenden finden Sie einen Auszug aus einer Aufgabendefinition, die Amazon EFS für die Containerspeicherung verwendet.

```
...
"containerDefinitions": [
  {
    "mountPoints": [
      {
        "containerPath": "/opt/my-app",
        "sourceVolume": "Shared-EFS-Volume"
      }
    ]
  }
]
...
"volumes": [
    {
      "efsVolumeConfiguration": {
        "fileSystemId": "fs-1234",
        "transitEncryption": "DISABLED",
        "rootDirectory": ""
      },
      "name": "Shared-EFS-Volume"
    }
  ]
```

Amazon EFS speichert Daten redundant in mehreren Availability Zones innerhalb einer einzigen Region. Eine Amazon ECS-Task mountet das Amazon EFS-Dateisystem mithilfe eines Amazon EFS-Mount-Ziels in seiner Availability Zone. Eine Amazon ECS-Aufgabe kann ein Amazon EFS-Dateisystem nur bereitstellen, wenn das Amazon EFS-Dateisystem ein Mount-Ziel in der Availability Zone hat, in der die Aufgabe ausgeführt wird. Daher empfiehlt es sich, Amazon EFS-Mount-Ziele in allen Availability Zones zu erstellen, in denen Sie Amazon ECS-Aufgaben hosten möchten.



Weitere Informationen finden Sie unter Amazon-EFS-Volumes im Entwicklerhandbuch für Amazon Elastic Container Service.

Sicherheit und Zugriffskontrollen

Amazon EFS bietet Funktionen zur Zugriffskontrolle, mit denen Sie sicherstellen können, dass die in einem Amazon EFS-Dateisystem gespeicherten Daten sicher sind und nur für Anwendungen zugänglich sind, die sie benötigen. Sie können Daten schützen, indem Sie die Verschlüsselung

im Ruhezustand und bei der Übertragung aktivieren. Weitere Informationen finden Sie unter Datenverschlüsselung in Amazon EFS im Benutzerhandbuch zu Amazon Elastic File System.

Neben der Datenverschlüsselung können Sie Amazon EFS auch verwenden, um den Zugriff auf ein Dateisystem einzuschränken. Es gibt drei Möglichkeiten, die Zugriffskontrolle in EFS zu implementieren.

- Sicherheitsgruppen — Mit Amazon EFS-Mount-Zielen können Sie eine Sicherheitsgruppe konfigurieren, die verwendet wird, um Netzwerkverkehr zuzulassen und abzulehnen. Sie können die an Amazon EFS angehängte Sicherheitsgruppe so konfigurieren, dass NFS-Verkehr (Port 2049) von der Sicherheitsgruppe zugelassen wird, die Ihren Amazon ECS-Instances zugeordnet ist, oder, wenn Sie den `awsvpc` Netzwerkmodus verwenden, von der Amazon ECS-Task.
- IAM — Sie können den Zugriff auf ein Amazon EFS-Dateisystem mithilfe von IAM einschränken. Wenn sie konfiguriert sind, benötigen Amazon ECS-Aufgaben eine IAM-Rolle für den Dateisystemzugriff, um ein EFS-Dateisystem zu mounten. Weitere Informationen finden Sie unter Verwenden von IAM zur Steuerung des Dateisystemdatenzugriffs im Amazon Elastic File System-Benutzerhandbuch.

  IAM-Richtlinien können auch vordefinierte Bedingungen erzwingen, z. B. die Anforderung, dass ein Client TLS verwendet, wenn er eine Verbindung zu einem Amazon EFS-Dateisystem herstellt. Weitere Informationen finden Sie unter Amazon EFS-Bedingungsschlüssel für Clients im Amazon Elastic File System-Benutzerhandbuch.
- Amazon EFS-Zugriffspunkte — Amazon EFS-Zugriffspunkte sind anwendungsspezifische Einstiegspunkte in ein Amazon EFS-Dateisystem. Sie können Access Points verwenden, um eine Benutzeridentität, einschließlich der POSIX-Gruppen des Benutzers, für alle Dateisystemanfragen, die über den Access Point gestellt werden, durchzusetzen. Zugriffspunkte können auch ein anderes Stammverzeichnis für das Dateisystem erzwingen. Auf diese Weise können Clients nur auf Daten im angegebenen Verzeichnis oder seinen Unterverzeichnissen zugreifen.

Erwägen Sie die Implementierung aller drei Zugriffskontrollen in einem Amazon EFS-Dateisystem, um maximale Sicherheit zu gewährleisten. Sie können beispielsweise die Sicherheitsgruppe, die einem Amazon EFS-Bereitstellungspunkt zugeordnet ist, so konfigurieren, dass nur eingehender NFS-Verkehr von einer Sicherheitsgruppe zugelassen wird, die Ihrer Container-Instance oder Amazon ECS-Aufgabe zugeordnet ist. Darüber hinaus können Sie Amazon EFS so konfigurieren, dass für den Zugriff auf das Dateisystem eine IAM-Rolle erforderlich ist, auch wenn die Verbindung aus einer zugelassenen Sicherheitsgruppe stammt. Schließlich können Sie

Amazon EFS-Zugriffspunkte verwenden, um POSIX-Benutzerberechtigungen durchzusetzen und Stammverzeichnisse für Anwendungen anzugeben.

Der folgende Ausschnitt aus der Aufgabendefinition zeigt, wie ein Amazon EFS-Dateisystem mithilfe eines Access Points bereitgestellt wird.

```
"volumes": [
    {
      "efsVolumeConfiguration": {
        "fileSystemId": "fs-1234",
        "authorizationConfig": {
          "acessPointId": "fsap-1234",
          "iam": "ENABLED"
        },
        "transitEncryption": "ENABLED",
        "rootDirectory": ""
      },
      "name": "my-filesystem"
    }
]
```

Leistung

Amazon EFS bietet zwei Leistungsmodi: General Purpose und Max I/O. General Purpose eignet sich für latenzempfindliche Anwendungen wie Content-Management-Systeme und CI/CD-Tools. Im Gegensatz dazu eignen sich Max I/O-Dateisysteme für Workloads wie Datenanalyse, Medienverarbeitung und maschinelles Lernen. Diese Workloads müssen parallel Operationen von Hunderten oder sogar Tausenden von Containern aus ausführen und erfordern den höchstmöglichen Gesamtdurchsatz und die höchstmöglichen IOPS. Weitere Informationen finden Sie unter Amazon EFS-Leistungsmodi im Amazon Elastic File System-Benutzerhandbuch.

Einige latenzempfindliche Workloads erfordern sowohl die höheren I/O-Stufen, die durch den Modus Max I/O Performance bereitgestellt werden, als auch die niedrigere Latenz, die durch den Allzweck-Performance-Modus bereitgestellt wird. Für diese Art von Workload empfehlen wir, mehrere Allzweck-Leistungsmodus-Dateisysteme zu erstellen. Auf diese Weise können Sie die Arbeitslast Ihrer Anwendung auf all diese Dateisysteme verteilen, sofern die Arbeitslast und die Anwendungen sie unterstützen können.

Durchsatz

Allen Amazon EFS-Dateisystemen ist ein gemessener Durchsatz zugeordnet, der entweder durch die Menge des bereitgestellten Durchsatzes für Dateisysteme mit bereitgestelltem Durchsatz oder durch die Menge der in der EFS-Speicherklasse Standard oder One Zone gespeicherten Daten für Dateisysteme mit Bursting Throughput bestimmt wird. Weitere Informationen finden Sie unter Understanding Metered Throughput im Amazon Elastic File System-Benutzerhandbuch.

Der Standard-Durchsatzmodus für Amazon EFS-Dateisysteme ist der Bursting-Modus. Im Bursting-Modus wird der Durchsatz, der einem Dateisystem zur Verfügung steht, mit zunehmendem Dateisystem nach oben oder unten skaliert. Da dateibasierte Workloads in der Regel stark ansteigen und für bestimmte Zeiträume einen hohen Durchsatz und für den Rest der Zeit einen niedrigeren Durchsatz erfordern, ist Amazon EFS so konzipiert, dass es schnell geht, um hohe Durchsatzwerte für bestimmte Zeiträume zu ermöglichen. Da viele Workloads leseintensiv sind, werden Lesevorgänge außerdem im Verhältnis 1:3 zu anderen NFS-Vorgängen (wie Schreiben) gemessen.

Alle Amazon EFS-Dateisysteme bieten eine konsistente Basisleistung von 50 MB/s für jedes TB Amazon EFS Standard- oder Amazon EFS One Zone-Speicher. Alle Dateisysteme (unabhängig von ihrer Größe) können bis zu 100 MB/s erreichen. Dateisysteme mit mehr als 1 TB EFS Standard- oder EFS One Zone-Speicher können für jedes TB auf 100 MB/s ansteigen. Da Lesevorgänge im Verhältnis 1:3 gemessen werden, können Sie bis zu 300 MiBs /s pro TiB Lesedurchsatz erreichen. Wenn Sie Ihrem Dateisystem Daten hinzufügen, wird der maximale Durchsatz, der für das Dateisystem verfügbar ist, linear und automatisch mit Ihrem Speicher in der Amazon EFS-Standardspeicherklasse skaliert. Wenn Sie mehr Durchsatz benötigen, als Sie mit Ihrer gespeicherten Datenmenge erreichen können, können Sie den bereitgestellten Durchsatz auf die spezifische Menge konfigurieren, die Ihre Arbeitslast benötigt.

Der Dateisystemdurchsatz wird von allen Amazon EC2 EC2-Instances gemeinsam genutzt, die mit einem Dateisystem verbunden sind. Beispielsweise kann ein 1-TB-Dateisystem, das einen Durchsatz von 100 MB/s erreichen kann, 100 MB/s aus einer einzigen Amazon EC2-Instance erzeugen, die jeweils 10 MB/s erreichen kann. Weitere Informationen finden Sie unter Amazon EFS-Leistung im Amazon Elastic File System-Benutzerhandbuch.

Kostenoptimierung

Amazon EFS vereinfacht die Speicherskalierung für Sie. Amazon EFS-Dateisysteme wachsen automatisch, wenn Sie mehr Daten hinzufügen. Insbesondere im Amazon EFS Bursting Throughput-Modus skaliert der Durchsatz auf Amazon EFS, wenn die Größe Ihres Dateisystems in der

Standardspeicherklasse zunimmt. Um den Durchsatz zu verbessern, ohne zusätzliche Kosten für den bereitgestellten Durchsatz auf einem EFS-Dateisystem zu zahlen, können Sie ein Amazon EFS-Dateisystem mit mehreren Anwendungen gemeinsam nutzen. Mithilfe von Amazon EFS-Zugriffspunkten können Sie die Speicherisolierung in gemeinsam genutzten Amazon EFS-Dateisystemen implementieren. Auf diese Weise können die Anwendungen, obwohl sie immer noch dasselbe Dateisystem verwenden, nicht auf Daten zugreifen, es sei denn, Sie autorisieren sie.

Wenn Ihre Daten wachsen, hilft Ihnen Amazon EFS dabei, Dateien, auf die selten zugegriffen wird, automatisch in eine niedrigere Speicherklasse zu verschieben. Die Amazon EFS-Speicherklasse Standard-Infrequent Access (IA) reduziert die Speicherkosten für Dateien, auf die nicht täglich zugegriffen wird. Dies geschieht, ohne auf die hohe Verfügbarkeit, Haltbarkeit, Elastizität und den POSIX-Dateisystemzugriff zu verzichten, den Amazon EFS bietet. Weitere Informationen finden Sie unter Amazon EFS-Speicherklassen im Amazon Elastic File System-Benutzerhandbuch.

Erwägen Sie, Amazon EFS-Lebenszyklusrichtlinien zu verwenden, um automatisch Geld zu sparen, indem Sie selten aufgerufene Dateien in den Amazon EFS IA-Speicher verschieben. Weitere Informationen finden Sie unter Amazon-EFS-Lebenszyklusverwaltung im Benutzerhandbuch zu Amazon Elastic File System.

Bei der Erstellung eines Amazon EFS-Dateisystems können Sie wählen, ob Amazon EFS Ihre Daten über mehrere Availability Zones (Standard) hinweg repliziert oder Ihre Daten redundant in einer einzigen Availability Zone speichert. Die Amazon EFS One Zone-Speicherklasse kann die Speicherkosten im Vergleich zu Amazon EFS Standard-Speicherklassen erheblich senken. Erwägen Sie die Verwendung der Amazon EFS One Zone-Speicherklasse für Workloads, die keine Multi-AZ-Resilienz erfordern. Sie können die Kosten für Amazon EFS One Zone-Speicher weiter senken, indem Sie Dateien, auf die selten zugegriffen wird, nach Amazon EFS One Zone-Infrequent Access verschieben. Weitere Informationen finden Sie unter Amazon EFS Infrequent Access.

Datenschutz

Amazon EFS speichert Ihre Daten redundant in mehreren Availability Zones für Dateisysteme, die Standardspeicherklassen verwenden. Wenn Sie Amazon EFS One Zone-Speicherklassen auswählen, werden Ihre Daten redundant in einer einzigen Availability Zone gespeichert. Darüber hinaus ist Amazon EFS so konzipiert, dass es über ein bestimmtes Jahr eine Haltbarkeit von 99,999999999% (11 9) bietet.

Wie in jeder Umgebung ist es eine bewährte Methode, ein Backup zu erstellen und Schutzmaßnahmen gegen versehentliches Löschen zu treffen. Für Amazon EFS-Daten umfasst diese bewährte Methode ein funktionierendes, regelmäßig getestetes Backup mit AWS Backup.

Dateisysteme, die Amazon EFS One Zone-Speicherklassen verwenden, sind so konfiguriert, dass Dateien bei der Erstellung des Dateisystems standardmäßig automatisch gesichert werden, sofern Sie diese Funktion nicht deaktivieren. Weitere Informationen finden Sie unter Datenschutz für Amazon EFS im Amazon Elastic File System-Benutzerhandbuch.

Anwendungsfälle

Amazon EFS bietet parallel gemeinsamen Zugriff, der automatisch wächst und schrumpft, wenn Dateien hinzugefügt und entfernt werden. Daher eignet sich Amazon EFS für jede Anwendung, die einen Speicher mit Funktionen wie niedriger Latenz, hohem Durchsatz und read-after-write Konsistenz benötigt. Amazon EFS ist ein ideales Speicher-Backend für Anwendungen, die horizontal skaliert werden und ein gemeinsam genutztes Dateisystem benötigen. Workloads wie Datenanalyse, Medienverarbeitung, Inhaltsmanagement und Web-Serving sind einige der häufigsten Anwendungsfälle von Amazon EFS.

Ein Anwendungsfall, in dem Amazon EFS möglicherweise nicht geeignet ist, sind Anwendungen, die eine Latenz von unter einer Millisekunde erfordern. Dies ist im Allgemeinen eine Anforderung für transaktionale Datenbanksysteme. Wir empfehlen die Durchführung von Speicherleistungstests, um die Auswirkungen der Verwendung von Amazon EFS für latenzempfindliche Anwendungen zu ermitteln. Wenn die Anwendungsleistung bei der Verwendung von Amazon EFS nachlässt, sollten Sie Amazon EBS io2 Block Express in Betracht ziehen, das I/O-Latenz unter einer Millisekunde mit geringer Varianz auf Nitro-Instances bietet. Weitere Informationen finden Sie unter Amazon EBS-Volume-Typen in der Amazon EC2-Benutzerhandbuch für Linux-Instances.

Einige Anwendungen schlagen fehl, wenn ihr zugrundeliegender Speicher unerwartet geändert wird. Daher ist Amazon EFS nicht die beste Wahl für diese Anwendungen. Stattdessen ziehen Sie es vielleicht vor, ein Speichersystem zu verwenden, das keinen gleichzeitigen Zugriff von mehreren Orten aus ermöglicht.

Docker-Volumes

Docker-Volumes sind eine Funktion der Docker-Container-Laufzeit, die es Containern ermöglicht, Daten dauerhaft zu speichern, indem sie ein Verzeichnis aus dem Dateisystem des Hosts mounten. Docker-Volumetreiber (auch als Plugins bezeichnet) werden verwendet, um Container-Volumes in externe Speichersysteme wie Amazon EBS zu integrieren. Docker-Volumes werden nur unterstützt, wenn Amazon ECS-Aufgaben auf Amazon EC2 EC2-Instances gehostet werden.

Amazon ECS-Aufgaben können Docker-Volumes verwenden, um Daten mithilfe von Amazon EBS-Volumes dauerhaft zu speichern. Dazu wird ein Amazon EBS-Volume an eine Amazon EC2

EC2-Instance angehängt und das Volume anschließend mithilfe von Docker-Volumes in einer Aufgabe bereitgestellt. Ein Docker-Volume kann von mehreren Amazon ECS-Aufgaben auf dem Host gemeinsam genutzt werden.

Die Einschränkung von Docker-Volumes besteht darin, dass das von der Aufgabe verwendete Dateisystem an die spezifische Amazon EC2 EC2-Instance gebunden ist. Wenn die Instance aus irgendeinem Grund beendet wird und die Aufgabe auf eine andere Instance übertragen wird, gehen die Daten verloren. Sie können Instances Aufgaben zuweisen, um sicherzustellen, dass die zugehörigen EBS-Volumes immer für Aufgaben verfügbar sind.

Weitere Informationen finden Sie unter Docker-Volumes im Amazon Elastic Container Service Developer Guide.

Lebenszyklus eines Amazon EBS-Volumes

Es gibt zwei wichtige Nutzungsmuster bei Containerspeicher und Amazon EBS. Das erste ist der Fall, wenn eine Anwendung Daten persistieren und Datenverlust verhindern muss, wenn ihr Container beendet wird. Ein Beispiel für diese Art von Anwendung wäre eine Transaktionsdatenbank wie MySQL. Wenn eine MySQL-Aufgabe beendet wird, wird erwartet, dass sie durch eine andere Aufgabe ersetzt wird. In diesem Szenario ist der Lebenszyklus des Volumes vom Lebenszyklus der Aufgabe getrennt. Bei der Verwendung von EBS zur persistenten Speicherung von Containerdaten empfiehlt es sich, Beschränkungen für die Aufgabenplatzierung zu verwenden, um die Platzierung der Aufgabe auf einen einzelnen Host zu beschränken, an den das EBS-Volume angeschlossen ist.

Die zweite Möglichkeit besteht darin, dass der Lebenszyklus des Volumes unabhängig vom Task-Lebenszyklus ist. Dies ist besonders nützlich für Anwendungen, die einen leistungsstarken Speicher mit niedriger Latenz benötigen, aber keine Daten nach Abschluss der Aufgabe persistieren müssen. Beispielsweise kann ein ETL-Workload, der große Datenmengen verarbeitet, einen Speicher mit hohem Durchsatz erfordern. Amazon EBS ist für diese Art von Workload geeignet, da es leistungsstarke Volumes mit bis zu 256.000 IOPS bietet. Wenn die Aufgabe beendet ist, kann das Ersatzreplikat sicher auf allen Amazon EC2 EC2-Hosts im Cluster platziert werden. Solange die Aufgabe Zugriff auf ein Speicher-Backend hat, das ihre Leistungsanforderungen erfüllen kann, kann die Aufgabe ihre Funktion erfüllen. Daher sind in diesem Fall keine Einschränkungen bei der Aufgabenplatzierung erforderlich.

Wenn den Amazon EC2 EC2-Instances in Ihrem Cluster mehrere Typen von Amazon EBS-Volumes zugeordnet sind, können Sie Beschränkungen für die Aufgabenplatzierung verwenden, um sicherzustellen, dass Aufgaben auf Instances platziert werden, denen ein entsprechendes Amazon EBS-Volume zugewiesen ist. Nehmen wir zum Beispiel an, ein Cluster hat einige Instances

mit einem gp2 Volume, während andere Volumes verwenden. `io1` Sie können Instances mit `io1` Volumes benutzerdefinierte Attribute zuordnen und dann mithilfe von Beschränkungen für die Aufgabenplatzierung sicherstellen, dass Ihre I/O-intensiven Aufgaben immer auf Container-Instances mit `io1` Volumes platziert werden.

Der folgende AWS CLI Befehl wird verwendet, um Attribute auf einer Amazon ECS-Container-Instance zu platzieren.

```
aws ecs put-attributes \
     --attributes name=EBS,value=io1,targetId=<your-container-instance-arn>
```

Verfügbarkeit von Amazon EBS-Daten

Container sind in der Regel kurzlebig und werden häufig erstellt und beendet, wenn Anwendungen horizontal ein- und ausgeweitet werden. Es hat sich bewährt, Workloads in mehreren Availability Zones auszuführen, um die Verfügbarkeit Ihrer Anwendungen zu verbessern. Amazon ECS bietet Ihnen die Möglichkeit, die Aufgabenverteilung mithilfe von Aufgabenplatzierungsstrategien und Aufgabenplatzierungsbeschränkungen zu kontrollieren. Wenn ein Workload seine Daten mithilfe von Amazon EBS-Volumes persistiert, müssen seine Aufgaben in derselben Availability Zone platziert werden wie das Amazon EBS-Volume. Wir empfehlen außerdem, eine Platzierungsbeschränkung festzulegen, die die Availability Zone einschränkt, in der eine Aufgabe platziert werden kann. Dadurch wird sichergestellt, dass sich Ihre Aufgaben und die entsprechenden Volumes immer in derselben Availability Zone befinden.

Bei der Ausführung eigenständiger Aufgaben können Sie steuern, in welcher Availability Zone die Aufgabe platziert wird, indem Sie Platzierungsbeschränkungen mithilfe des Attributs Availability Zone festlegen.

```
attribute:ecs.availability-zone == us-east-1a
```

Wenn Sie Anwendungen ausführen, die von der Ausführung in mehreren Availability Zones profitieren würden, sollten Sie erwägen, für jede Availability Zone einen eigenen Amazon ECS-Service zu erstellen. Dadurch wird sichergestellt, dass Aufgaben, die ein Amazon EBS-Volume benötigen, immer in derselben Availability Zone platziert werden wie das zugehörige Volume.

Wir empfehlen, Container-Instances in jeder Availability Zone zu erstellen, Amazon EBS-Volumes mithilfe von Startvorlagen anzuhängen und den Instances benutzerdefinierte Attribute hinzuzufügen, um sie von anderen Container-Instances im Amazon ECS-Cluster zu unterscheiden. Konfigurieren Sie bei der Erstellung von Services Einschränkungen für die Aufgabenplatzierung, um

sicherzustellen, dass Amazon ECS Aufgaben in der richtigen Availability Zone und Instance platziert. Weitere Informationen finden Sie unter Beispiele für Einschränkungen bei der Aufgabenplatzierung im Amazon Elastic Container Service Developer Guide.

Docker-Volume-Plug-ins

Docker-Plugins wie Portworx bieten eine Abstraktion zwischen dem Docker-Volume und dem Amazon EBS-Volume. Diese Plug-ins können dynamisch ein Amazon EBS-Volume erstellen, wenn Ihre Aufgabe, die ein Volume benötigt, gestartet wird. Portworx kann auch ein Volume an einen neuen Host anhängen, wenn ein Container beendet wird und sein nachfolgendes Replikat auf einer anderen Container-Instance platziert wird. Es repliziert auch die Volumendaten jedes Containers zwischen Amazon ECS-Knoten und zwischen Availability Zones. Weitere Informationen finden Sie unter Portworx.

FSx für Windows File Server

FSx for Windows File Server bietet vollständig verwalteten, äußerst zuverlässigen und skalierbaren Dateispeicher, auf den über das branchenübliche Server Message Block (SMB) -Protokoll zugegriffen werden kann. Es basiert auf Windows Server und bietet eine Vielzahl von Verwaltungsfunktionen wie Benutzerkontingente, Dateiwiederherstellung für Endbenutzer und Microsoft Active Directory (AD) -Integration. Es bietet Single-AZ- und Multi-AZ-Bereitstellungsoptionen, vollständig verwaltete Backups und Verschlüsselung von Daten im Speicher und bei der Übertragung.

Amazon ECS unterstützt die Verwendung von FSx for Windows File Server in Amazon ECS Windows-Aufgabendefinitionen und ermöglicht persistenten Speicher als Bereitstellungspunkt über das SMBv3-Protokoll unter Verwendung einer SMB-Funktion namens. GlobalMappings

Um die Integration von FSx for Windows File Server und Amazon ECS einzurichten, muss die Windows-Container-Instance ein Domänenmitglied in einem Active Directory Domain Service (AD DS) sein, der von einem lokalen Active Directory oder einem AWS Directory Service for Microsoft Active Directory selbst gehosteten Active Directory auf Amazon EC2 gehostet wird. AWS Secrets Manager wird verwendet, um vertrauliche Daten wie den Benutzernamen und das Passwort einer Active Directory-Anmeldedaten zu speichern, die zur Zuordnung der Freigabe auf der Windows-Container-Instance verwendet werden.

Um FSx for Windows File Server-Dateisystem-Volumes für Ihre Container zu verwenden, müssen Sie die Volume- und Bereitstellungspunkt-Konfigurationen in der Aufgabendefinition angeben. Im Folgenden finden Sie einen Auszug aus einer Aufgabendefinition, die FSx for Windows File Server für den Containerspeicher verwendet.

```
{
 "containerDefinitions": [{
  "name": "container-using-fsx",
  "image": "iis:2",
  "entryPoint": [
   "powershell",
   "-command"
  ],
  "mountPoints": [{
   "sourceVolume": "myFsxVolume",
   "containerPath": "\\mount\\fsx",
   "readOnly": false
  }]
 }],
 "volumes": [{
  "fsxWindowsFileServerVolumeConfiguration": {
   "fileSystemId": "fs-ID",
   "authorizationConfig": {
    "domain": "ADDOMAIN.local",
    "credentialsParameter": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:SecretName"
   },
   "rootDirectory": "share"
  }
 }]
}
```

Weitere Informationen finden Sie unter Amazon FSx for Windows File Server Server-Volumes im Amazon Elastic Container Service Developer Guide.

Sicherheit und Zugriffskontrollen

FSx for Windows File Server bietet die folgenden Funktionen zur Zugriffskontrolle, mit denen Sie sicherstellen können, dass die in einem FSx for Windows File Server Server-Dateisystem gespeicherten Daten sicher sind und nur von Anwendungen aus zugänglich sind, die sie benötigen.

Datenverschlüsselung

FSx for Windows File Server unterstützt zwei Formen der Verschlüsselung für Dateisysteme. Dabei handelt es sich um die Verschlüsselung von Daten während der Übertragung und die Verschlüsselung im Ruhezustand. Die Verschlüsselung von Daten während der Übertragung wird auf Dateifreigaben unterstützt, die einer Container-Instance zugeordnet sind, die das SMB-Protokoll

3.0 oder höher unterstützt. Die Verschlüsselung von Daten im Ruhezustand wird automatisch aktiviert, wenn ein Amazon FSx-Dateisystem erstellt wird. Amazon FSx verschlüsselt Daten während der Übertragung automatisch mithilfe der SMB-Verschlüsselung, wenn Sie auf Ihr Dateisystem zugreifen, ohne dass Sie Ihre Anwendungen ändern müssen. Weitere Informationen finden Sie unter Datenverschlüsselung in Amazon FSx im Amazon FSx for Windows File Server Server-Benutzerhandbuch.

Zugriffskontrolle auf Ordnerebene mithilfe von Windows-ACLs

Die Windows Amazon EC2 EC2-Instance greift mithilfe von Active Directory-Anmeldeinformationen auf Amazon FSx-Dateifreigaben zu. Sie verwendet standardmäßige Windows-Zugriffskontrolllisten (ACLs) für eine differenzierte Zugriffskontrolle auf Datei- und Ordnerebene. Sie können mehrere Anmeldeinformationen erstellen, jeweils für einen bestimmten Ordner innerhalb der Freigabe, der einer bestimmten Aufgabe zugeordnet ist.

Im folgenden Beispiel hat die Aufgabe Zugriff auf den Ordner `App01` mithilfe von Anmeldeinformationen, die in Secrets Manager gespeichert sind. Sein Amazon-Ressourcenname (ARN) lautet`1234`.

```
"rootDirectory": "\\path\\to\\my\\data\App01",
"credentialsParameter": "arn-1234",
"domain": "corp.fullyqualified.com",
```

In einem anderen Beispiel hat eine Aufgabe Zugriff auf den Ordner `App02` mithilfe von Anmeldeinformationen, die im Secrets Manager gespeichert sind. Ihr ARN lautet `6789`.

```
"rootDirectory": "\\path\\to\\my\\data\App02",
"credentialsParameter": "arn-6789",
"domain": "corp.fullyqualified.com",
```

Anwendungsfälle

Container sind nicht darauf ausgelegt, Daten dauerhaft zu speichern. Einige containerisierte.NET-Anwendungen benötigen jedoch möglicherweise lokale Ordner als persistenten Speicher zum Speichern von Anwendungsausgaben. FSx for Windows File Server bietet einen lokalen Ordner im Container. Dadurch können mehrere Container auf demselben Dateisystem lesen und schreiben, das von einem SMB Share unterstützt wird.

Bewährte Methoden — Schnellerer Start von Aufgaben

Es gibt mehrere Verbesserungen, die Sie vornehmen können, um die Zeit zu verkürzen, die Amazon ECS benötigt, um Ihre Aufgaben zu starten.

Arbeitsablauf zum Starten von Amazon ECS-Aufgaben

Wenn Sie wissen, wie Amazon ECS Ihre Aufgaben bereitstellt, ist es hilfreich, über Optimierungen nachzudenken, mit denen Sie Ihre Aufgaben schneller starten können. Wenn Sie Amazon ECS-Aufgaben starten (eigenständige Aufgaben oder über Amazon ECS-Services), wird eine Aufgabe erstellt und zunächst in den `PROVISIONING` Status versetzt, bevor sie erfolgreich gestartet wird (Einzelheiten finden Sie unter Aufgabenlebenszyklus im Amazon ECS-Entwicklerhandbuch). `RUNNING` In diesem `PROVISIONING` Bundesstaat sind weder die Aufgabe noch die Container vorhanden, da Amazon ECS Rechenkapazität für die Platzierung der Aufgabe finden muss.

Amazon ECS wählt die passende Rechenkapazität für Ihre Aufgabe auf der Grundlage Ihres Starttyps oder der Konfiguration Ihres Kapazitätsanbieters aus. Die Starttypen sind AWS Fargate (Fargate) und Amazon EC2 aktiviert und der `EXTERNAL` Typ AWS, der mit Amazon ECS Anywhere verwendet wird. Kapazitätsanbieter und Kapazitätsanbieterstrategien können sowohl für die Starttypen Fargate als auch Amazon EC2 verwendet werden. Mit Fargate müssen Sie sich keine Gedanken über die Bereitstellung, Konfiguration und Skalierung Ihrer Clusterkapazität machen. Fargate kümmert sich um das gesamte Infrastrukturmanagement für Ihre Aufgaben. Für Amazon ECS mit Amazon EC2 können Sie entweder Ihre Clusterkapazität verwalten, indem Sie Amazon EC2 EC2-Instances in Ihrem Cluster registrieren, oder Sie können Amazon ECS Cluster Auto Scaling (CAS) verwenden, um Ihre Rechenkapazitätsverwaltung zu vereinfachen. CAS kümmert sich um die dynamische Skalierung Ihrer Clusterkapazität, sodass Sie sich auf die Ausführung von Aufgaben konzentrieren können. Amazon ECS bestimmt anhand der Anforderungen, die Sie in der Aufgabendefinition angeben, wie CPU und Arbeitsspeicher, sowie anhand Ihrer Platzierungsbeschränkungen und -strategien, wo die Aufgabe platziert werden soll. Weitere Informationen zur Aufgabenplatzierung finden Sie unter Amazon ECS-Aufgabenplatzierung.

Nachdem die Kapazität für die Platzierung Ihrer Aufgabe ermittelt wurde, stellt Amazon ECS die erforderlichen Anlagen bereit (z. B. Elastic Network Interfaces (ENIs) für Aufgaben im `awsvpc` Modus) und verwendet den Amazon ECS-Container-Agenten, um Ihre Container-Images abzurufen und Ihre Container zu starten. Sobald all dies abgeschlossen ist und die entsprechenden Container gestartet wurden, versetzt Amazon ECS die Aufgabe in den `RUNNING` Status.

Arbeitsablauf für Amazon ECS Service Scheduler

Amazon ECS bietet einen Service Scheduler zur Verwaltung des Status Ihrer Services. Der Service Scheduler stellt sicher, dass die von Ihnen angegebene Planungsstrategie eingehalten wird, und plant fehlgeschlagene Aufgaben neu. Wenn beispielsweise die zugrunde liegende Infrastruktur fehlschlägt, kann der Service-Scheduler Aufgaben neu planen. Eine Hauptaufgabe des Service Schedulers besteht darin, sicherzustellen, dass Ihre Anwendung immer die gewünschte Anzahl von Aufgaben ausführt — basierend auf der gewünschten Anzahl, die Sie in der Dienstkonfiguration angeben, oder der auto skalierten Anzahl von Aufgaben basierend auf der Anwendungslast, wenn Sie Service Autoscaling verwenden. Der Service Scheduler verwendet asynchrone Workflows, um Aufgaben stapelweise zu starten. Um zu verstehen, wie der Service Scheduler funktioniert, stellen Sie sich vor, Sie erstellen einen Amazon ECS-Service für eine große Web-API, die viel Verkehr empfängt. Sie erwarten, dass dieser Service eine Menge Web-Traffic abwickelt, und stellen fest, dass die gewünschte Anzahl für den Service 1.000 Aufgaben beträgt. Wenn Sie diesen Service bereitstellen, startet Amazon ECS Service Scheduler nicht alle 1.000 Aufgaben gleichzeitig. Stattdessen beginnt er mit der Ausführung von Workflow-Zyklen, um den aktuellen Status (0 Aufgaben) in den gewünschten Zustand (1.000 Aufgaben) zu bringen, wobei bei jedem Workflow-Zyklus ein Stapel neuer Aufgaben gestartet wird. Der Service Scheduler kann pro Service und Minute bis zu 500 Aufgaben für die Starttypen Fargate, Amazon EC2 und External bereitstellen. Weitere Informationen zu den zulässigen Tarifen und Kontingenten in Amazon ECS finden Sie unter Amazon ECS-Servicekontingente.

Nachdem Sie sich mit dem Arbeitsablauf beim Starten von Aufgaben in Amazon ECS vertraut gemacht haben, wollen wir nun besprechen, wie Sie einen Teil dieses Wissens nutzen können, um Ihre Aufgabenstarts zu beschleunigen.

Empfehlungen zur Beschleunigung des Starts von Aufgaben

Wie im vorherigen Abschnitt beschrieben, wird die Zeit zwischen dem Auslösen des Task-Starts (über Amazon ECS-APIs oder Service Scheduler) und dem erfolgreichen Start Ihrer Container von einer Vielzahl von Faktoren innerhalb von Amazon ECS, Ihren Konfigurationen und dem Container selbst beeinflusst. Beachten Sie die folgenden Empfehlungen, um den Start Ihrer Aufgaben zu beschleunigen.

- Container-Images und Binpack-Instances zwischenspeichern.

  Wenn Sie Amazon ECS auf Amazon EC2 ausführen, können Sie den Amazon ECS-Container-Agenten so konfigurieren, dass er zuvor verwendete Container-Images zwischenspeichert, um

die Abrufzeit von Images für nachfolgende Starts zu reduzieren. Der Effekt des Cachings ist noch größer, wenn Sie eine hohe Aufgabendichte in Ihren Container-Instances haben, die Sie mithilfe der Platzierungsstrategie konfigurieren können. `binpack` Das Zwischenspeichern von Container-Images ist besonders für Windows-basierte Workloads von Vorteil, die in der Regel große Container-Image-Größen (mehrere zehn GB) haben. Wenn Sie die `binpack` Platzierungsstrategie verwenden, können Sie auch erwägen, Elastic Network Interface (ENI) -Trunking zu verwenden, um mehr Aufgaben im `awsvpc` Netzwerkmodus auf jeder Container-Instance zu platzieren. ENI-Trunking erhöht die Anzahl der Aufgaben, die Sie im Modus ausführen `awsvpc` können. Beispielsweise kann eine c5.large-Instance, die möglicherweise nur die gleichzeitige Ausführung von 2 Aufgaben unterstützt, bis zu 10 Aufgaben mit ENI-Trunking ausführen.

- Wählen Sie einen optimalen Netzwerkmodus.

  Obwohl es viele Fälle gibt, in denen der `awsvpc` Netzwerkmodus ideal ist, kann dieser Netzwerkmodus von Natur aus die Latenz beim Starten von Aufgaben erhöhen. Für jede Aufgabe im `awsvpc` Modus müssen Amazon ECS-Workflows eine ENI bereitstellen und anhängen, indem sie Amazon EC2 EC2-APIs aufrufen, was Ihren Aufgabenstarts einen Mehraufwand von mehreren Sekunden hinzufügt. Im Gegensatz dazu besteht ein entscheidender Vorteil des `awsvpc` Netzwerkmodus darin, dass jede Aufgabe über eine Sicherheitsgruppe verfügt, die den Datenverkehr zulässt oder verweigert. Dies bedeutet, dass Sie flexibler sind, um die Kommunikation zwischen Aufgaben und Diensten detaillierter zu steuern. Wenn die Vorteile der schnellen Bereitstellung die Vorteile des Modus überwiegen, können Sie erwägen, den `awsvpc` Modus zu verwenden`bridge`, um das Starten von Aufgaben zu beschleunigen. Weitere Informationen zu den jeweiligen Vorteilen der einzelnen Netzwerkmodi finden Sie unter the section called “AWSVPC Modus” undthe section called “Bridge-Modus”.

- Verfolgen Sie Ihren Lebenszyklus beim Starten von Aufgaben, um Optimierungsmöglichkeiten zu finden.

  Es ist oft schwierig zu erkennen, wie lange es dauert, bis Ihre Anwendung gestartet wird. Das Starten Ihres Container-Images, das Ausführen von Startskripten und andere Konfigurationen während des Anwendungsstarts können überraschend viel Zeit in Anspruch nehmen. Sie können den ECS-Agent-Metadaten-Endpunkt verwenden, um Metriken zu veröffentlichen, mit denen Sie die Startzeit der Anwendung von `ContainerStartTime` bis zu dem Zeitpunkt verfolgen können, zu dem Ihre Anwendung bereit ist, Datenverkehr bereitzustellen. Anhand dieser Daten können Sie nachvollziehen, wie Ihre Anwendung zur Gesamtstartzeit beiträgt, und Bereiche ermitteln, in denen Sie unnötigen anwendungsspezifischen Aufwand reduzieren und Ihre Container-Images optimieren können.

- Wählen Sie einen optimalen Instance-Typ (bei Verwendung von Amazon ECS auf Amazon EC2).

  Die Auswahl des richtigen Instance-Typs basiert auf der Ressourcenreservierung (d. h. CPU, Arbeitsspeicher, ENI, GPU), die Sie für Ihre Aufgabe konfigurieren. Daher können Sie bei der Dimensionierung der Instance berechnen, wie viele Aufgaben auf einer einzelnen Instance platziert werden können. Ein einfaches Beispiel für eine gut platzierte Aufgabe ist das Hosten von 4 Aufgaben, die 0,5 vCPU und 2 GB an Speicherreservierungen in einer m5.large-Instance erfordern (unterstützt 2 vCPUs und 8 GB Arbeitsspeicher). Die Reservierungen dieser Aufgabendefinition nutzen die Ressourcen der Instanz voll aus.
- Verwenden Sie den Amazon ECS Service Scheduler, um Dienste gleichzeitig zu starten.

  Wie im vorherigen Abschnitt beschrieben, kann der Service Scheduler mithilfe asynchroner Workflows gleichzeitig Aufgaben für mehrere Services starten. Somit können Sie eine schnellere Bereitstellung erreichen, indem Sie Ihre Anwendungen als kleinere Dienste mit weniger Aufgaben anstatt als großen Dienst mit einer großen Anzahl von Aufgaben konzipieren. Anstatt beispielsweise einen einzigen Dienst mit 1.000 Aufgaben zu haben, führen 10 Dienste mit jeweils 100 Aufgaben zu einer viel schnelleren Bereitstellungsgeschwindigkeit, da der Service Scheduler die Aufgabenbereitstellung für alle Dienste parallel initiiert.

Bewährte Methoden — Schnellere Bereitstellungen

Sie können fortlaufende Updates für Ihren Amazon ECS-Service wählen. Bereitstellungen können länger dauern als erwartet, aber Sie können einige Optionen ändern, um Ihre Bereitstellungen zu beschleunigen. Zum besseren Kontext: Wenn Sie diesen Bereitstellungstyp wählen, weisen Sie den Amazon ECS-Service Scheduler an, alle aktuell laufenden Aufgaben durch neue Aufgaben zu ersetzen, wenn eine neue Servicebereitstellung gestartet wird. Die Bereitstellungskonfiguration bestimmt die spezifische Anzahl von Aufgaben, die Amazon ECS dem Service während eines fortlaufenden Updates hinzufügt oder daraus entfernt. Im Folgenden finden Sie einen Überblick über den Bereitstellungsprozess:

1. Der Scheduler startet Ihre Anwendung.
2. Der Scheduler entscheidet dann, ob Ihre Anwendung für den Web-Traffic bereit ist.
3. Wenn Sie die Anwendung herunterskalieren oder eine neue Version der Anwendung erstellen, entscheidet der Scheduler, ob Ihre Anwendung sicher beendet werden kann. Gleichzeitig muss die Verfügbarkeit der Anwendung während der fortlaufenden Bereitstellung aufrechterhalten werden.

Die Strategie, die Verfügbarkeit von Aufgaben aufrechtzuerhalten, kann dazu führen, dass Bereitstellungen länger dauern als erwartet.

Um die Bereitstellungszeiten zu beschleunigen, ändern Sie den standardmäßigen Load Balancer, den Amazon ECS-Agenten, den Service und die Aufgabendefinitionsoptionen. In den folgenden Abschnitten wird detailliert beschrieben, wie Sie all diese Optionen ändern können, um Bereitstellungen zu beschleunigen.

Themen

Parameter für die Integritätsprüfung des Load Balancers

Das folgende Diagramm beschreibt den Prozess zur Integritätsprüfung des Load Balancers. Der Load Balancer sendet regelmäßig Integritätsprüfungen an den Amazon ECS-Container. Der Amazon ECS-Agent überwacht den Zustand des Containers und wartet darauf, dass der Load Balancer Bericht erstattet. Dies geschieht, bevor der Container als fehlerfrei eingestuft wird.

Zwei Parameter für die Integritätsprüfung von Elastic Load Balancing wirken sich auf die Bereitstellungsgeschwindigkeit aus:

- Intervall Health Integritätsprüfungen: Bestimmt den ungefähren Zeitraum in Sekunden zwischen den Zustandsprüfungen eines einzelnen Containers. Standardmäßig überprüft der Load Balancer alle 30 Sekunden.

  Dieser Parameter heißt:

  - `HealthCheckIntervalSeconds`in der Elastic Load Balancing API
  - Intervall auf der Amazon EC2 EC2-Konsole
- Anzahl fehlerhafter Schwellenwerte: Bestimmt die Anzahl der aufeinanderfolgenden erfolgreichen Zustandsprüfungen, die erforderlich sind, bevor ein fehlerhafter Container als fehlerfrei eingestuft wird. Standardmäßig benötigt der Load Balancer fünf bestandene Zustandsprüfungen, bevor er meldet, dass der Zielcontainer fehlerfrei ist.

  Dieser Parameter heißt:

  - `HealthyThresholdCount`in der Elastic Load Balancing API
  - Fehlerfreier Schwellenwert auf der Amazon EC2 EC2-Konsole

Mit der Standardeinstellung beträgt die Gesamtzeit zur Bestimmung des Zustands eines Containers zwei Minuten und 30 Sekunden (`30 seconds * 5 = 150 seconds`).

Sie können den Integritätsprüfungsprozess beschleunigen, wenn Ihr Dienst in weniger als 10 Sekunden gestartet und stabilisiert wird. Um den Vorgang zu beschleunigen, reduzieren Sie die Anzahl der Prüfungen und das Intervall zwischen den Prüfungen.

- `HealthCheckIntervalSeconds`(Elastic Load Balancing API-Name) oder Interval (Name der Amazon EC2 EC2-Konsole): 5

- `HealthyThresholdCount`(Elastic Load Balancing API-Name) oder Health-Schwellenwert (Name der Amazon EC2 EC2-Konsole): 2

Mit dieser Einstellung dauert die Integritätsprüfung 10 Sekunden im Vergleich zur Standardeinstellung von zwei Minuten und 30 Sekunden.

Einstellung der Elastic Load Balancing Health Check-Parameter zur Beschleunigung der Bereitstellung

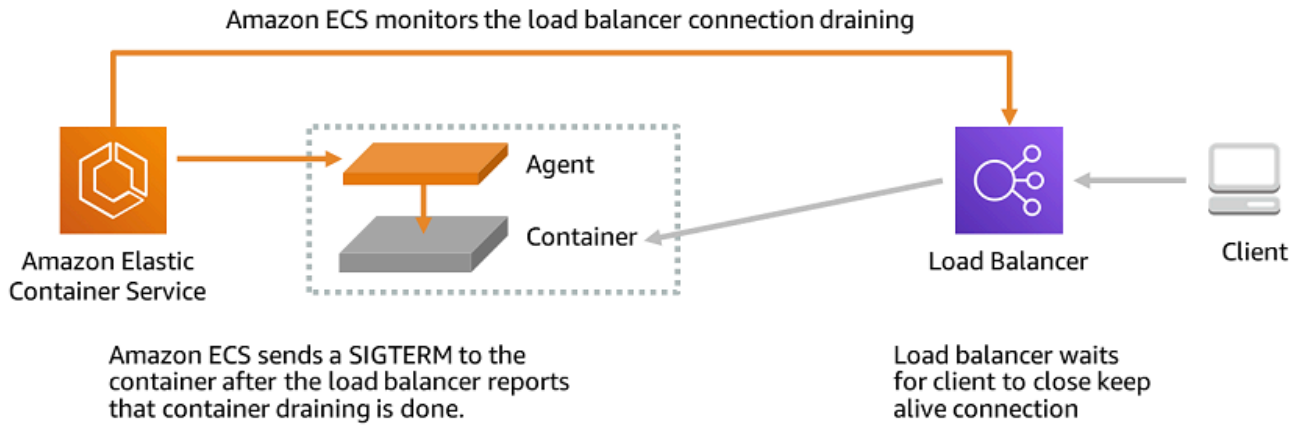
1. Gehen Sie zu https://console.aws.amazon.com/ec2/.
2. Wählen Sie in der linken Navigationsleiste unter Load Balancing die Option Zielgruppen aus.
3. Wählen Sie auf der Seite Zielgruppen die Zielgruppe aus.
4. Wählen Sie auf der Zielgruppenseite den Tab Gesundheitschecks aus.
5. Klicken Sie auf Bearbeiten.
6. Erweitern Sie Erweiterte Einstellungen für die Gesundheitsprüfung.
7. Stellen Sie die Parameter für die Integritätsprüfung ein.
8. Klicken Sie auf Änderungen speichern.

Weitere Informationen zu den Elastic Load Balancing Health Check-Parametern finden Sie TargetGroupin der Elastic Load Balancing API-Referenz.

Load Balancer-Verbindung wird entladen

Um eine Optimierung zu ermöglichen, halten die Clients eine permanente Verbindung zum Container-Service aufrecht. Auf diese Weise können nachfolgende Anfragen von diesem Client die bestehende Verbindung wiederverwenden. Wenn Sie den Verkehr zu einem Container unterbrechen möchten, benachrichtigen Sie den Load Balancer.

In der folgenden Abbildung wird der Verbindungsabbau beim Load Balancer beschrieben. Wenn Sie den Load Balancer anweisen, den Verkehr zum Container zu unterbrechen, überprüft er regelmäßig, ob der Client die Keep-Alive-Verbindung geschlossen hat. Der Amazon ECS-Agent überwacht den Load Balancer und wartet darauf, dass der Load Balancer meldet, dass die Keep-Alive-Verbindung geschlossen ist.



Die Zeit, die der Load Balancer wartet, entspricht der Verzögerung bei der Abmeldung. Sie können die folgende Load Balancer-Einstellung konfigurieren, um Ihre Bereitstellungen zu beschleunigen.

- `deregistration_delay.timeout_seconds`: 300 (Standard)

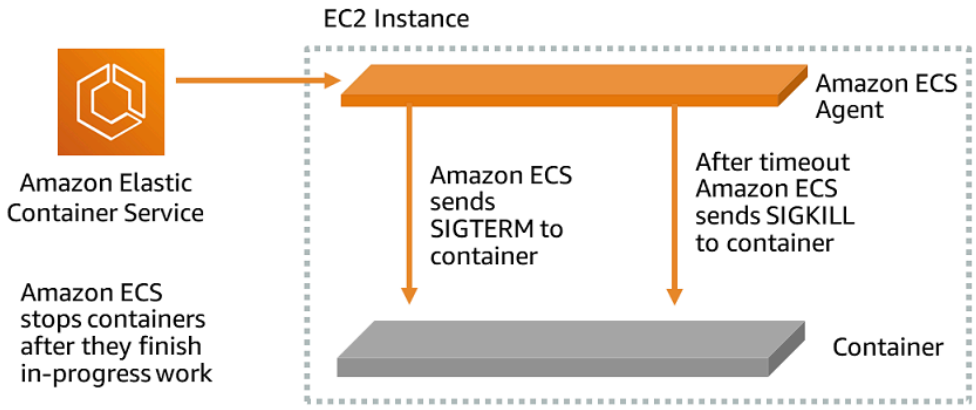
Wenn Sie einen Service mit einer Antwortzeit unter einer Sekunde haben, setzen Sie den Parameter auf den folgenden Wert, damit der Load Balancer nur fünf Sekunden wartet, bevor er die Verbindung zwischen dem Client und dem Back-End-Dienst unterbricht:

- `deregistration_delay.timeout_seconds`: 5

> **Note**
>
> Setzen Sie den Wert nicht auf 5 Sekunden, wenn Sie einen Dienst mit lang anhaltenden Anfragen haben, wie z. B. langsame Datei-Uploads oder Streaming-Verbindungen.

Reaktionsfähigkeit von SIGTERM

Das folgende Diagramm zeigt, wie Amazon ECS eine Aufgabe beendet. Amazon ECS sendet zunächst ein SIGTERM-Signal an die Aufgabe, um zu benachrichtigen, dass die Anwendung beendet und heruntergefahren werden muss. Anschließend sendet Amazon ECS eine SIGKILL-Nachricht. Wenn Anwendungen den SIGTERM ignorieren, muss der Amazon ECS-Service warten, bis er das SIGKILL-Signal sendet, um den Prozess zu beenden.



Die Dauer, die Amazon ECS wartet, wird durch die folgende Amazon ECS-Agentenoption bestimmt:

- `ECS_CONTAINER_STOP_TIMEOUT`: 30 (Standard)

  Weitere Informationen zum Container-Agent-Parameter finden Sie unter Konfiguration des Container-Agents im Amazon Elastic Container Service Developer Guide.

Um die Wartezeit zu verkürzen, setzen Sie die Amazon ECS-Agentenoption auf den folgenden Wert:

> **Note**
>
> Wenn Ihre Anwendung länger als eine Sekunde dauert, multiplizieren Sie den Wert mit zwei und verwenden Sie diese Zahl als Wert.

- `ECS_CONTAINER_STOP_TIMEOUT`: 2

In diesem Fall wartet Amazon ECS zwei Sekunden, bis der Container heruntergefahren wird, und Amazon ECS sendet dann eine SIGKILL-Nachricht, wenn die Anwendung nicht gestoppt wurde.

Sie können den Anwendungscode auch ändern, um das SIGTERM-Signal abzufangen und darauf zu reagieren. Das Folgende ist ein Beispiel in JavaScript:

```
process.on('SIGTERM', function() {
  server.close();
})
```

Dieser Code bewirkt, dass der HTTP-Server nicht mehr auf neue Anfragen wartet, alle laufenden Anfragen beantwortet, und dann wird der Prozess Node.js beendet. Das liegt daran, dass die zugehörige Ereignisschleife nichts mehr zu tun hat. Aus diesem Grund wird der Prozess vorzeitig beendet, wenn er nur 500 ms benötigt, um seine laufenden Anfragen zu beenden, ohne dass der Stop-Timeout abgewartet und ein SIGKILL gesendet werden muss.

Typ des Container-Images

Die Zeit, die ein Container zum Starten benötigt, hängt vom zugrunde liegenden Container-Image ab. Beispielsweise kann der Start eines dickeren Images (Vollversionen von Debian, Ubuntu und Amazon1/2) länger dauern, da in den Containern mehr Dienste ausgeführt werden als in ihren jeweiligen Slim-Versionen (Debian-Slim, Ubuntu-Slim und Amazon-Slim) oder kleineren Basis-Images (Alpine).

Verhalten beim Abrufen von Container-Images

Pull-Verhalten von Container-Images für Fargate-Starttypen

Fargate speichert keine Bilder im Cache, weshalb das gesamte Bild aus der Registrierung abgerufen wird, wenn eine Aufgabe ausgeführt wird. Im Folgenden finden Sie unsere Empfehlungen für Bilder, die für Fargate-Aufgaben verwendet werden:

- Verwenden Sie eine größere Aufgabengröße mit zusätzlichen vCPUs. Die größere Aufgabengröße kann dazu beitragen, die Zeit zu reduzieren, die zum Extrahieren des Images beim Start einer Aufgabe erforderlich ist.
- Verwenden Sie ein kleineres Basisimage.
- Stellen Sie sicher, dass sich das Repository, das das Bild speichert, in derselben Region wie die Aufgabe befindet.

Verhalten beim Abrufen von Container-Images für Fargate-Windows-Starttypen

Fargate Windows speichert das von Microsoft bereitgestellte Servercore-Basisimage des letzten Monats und des Vormonats im Cache. Diese Images entsprechen den Patches mit der KB/Build-Nummer, die an jedem Patch-Dienstag aktualisiert werden. Beispielsweise hat Microsoft am 8.8.2023 KB5029247 (17763.4737) für Windows Server 2019 veröffentlicht. Der vorherige Monat KB am 11.07.2023 war KB5028168 (17763.4645). Also wurden für die Plattformen und die folgenden Container-Images zwischengespeichert: `WINDOWS_SERVER_2019_CORE` `WINDOWS_SERVER_2019_FULL`

- `mcr.microsoft.com/windows/servercore:ltsc2019`
- `mcr.microsoft.com/windows/servercore:10.0.17763.4737`
- `mcr.microsoft.com/windows/servercore:10.0.17763.4645`

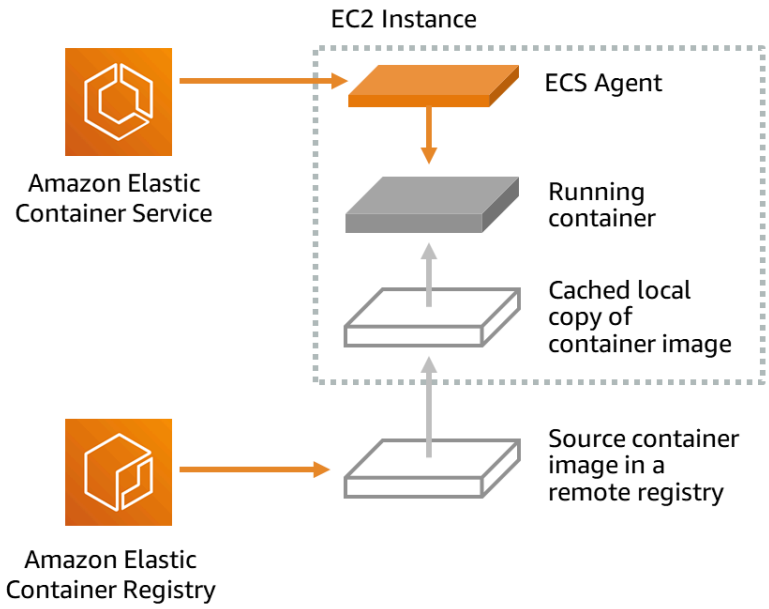
Darüber hinaus veröffentlichte Microsoft am 8.8.2023 KB5029250 (20348.1906) für Windows Server 2022. In den vergangenen Monaten war KB am 11.07.2023 KB5028171 (20348.1850). Also wurden für die Plattformen `WINDOWS_SERVER_2022_CORE` und die folgenden Container-Images zwischengespeichert: `WINDOWS_SERVER_2022_FULL`

- `mcr.microsoft.com/windows/servercore:ltsc2022`

- `mcr.microsoft.com/windows/servercore:10.0.20348.1906`
- `mcr.microsoft.com/windows/servercore:10.0.20348.1850`

Pull-Verhalten von Container-Images für Amazon EC2 EC2-Starttypen

Wenn der Amazon ECS-Agent eine Aufgabe startet, ruft er das Docker-Image aus seiner Remote-Registrierung ab und speichert dann eine lokale Kopie im Cache. Wenn Sie für jede Version Ihrer Anwendung ein neues Image-Tag verwenden, ist dieses Verhalten unnötig.



Der `ECS_IMAGE_PULL_BEHAVIOR` Agentenparameter bestimmt das Verhalten beim Abrufen von Bildern und hat die folgenden Werte:

- `ECS_IMAGE_PULL_BEHAVIOR`: `default`

  Das Bild wird remote abgerufen. Wenn der Abruf fehlschlägt, wird das zwischengespeicherte Bild in der Instanz verwendet.

- `ECS_IMAGE_PULL_BEHAVIOR`: `always`

  Das Bild wird remote abgerufen. Wenn der Abruf fehlschlägt, schlägt die Aufgabe fehl.

Um die Bereitstellung zu beschleunigen, setzen Sie den Amazon ECS-Agentenparameter auf einen der folgenden Werte:

- `ECS_IMAGE_PULL_BEHAVIOR`: `once`

  Das Image wird nur dann remote abgerufen, wenn es nicht durch eine vorherige Aufgabe auf derselben Container-Instance abgerufen wurde oder wenn das zwischengespeicherte Image durch den automatischen Image-Reinigungsprozess entfernt wurde. Andernfalls wird das zwischengespeicherte Image in der Instance verwendet. Dadurch wird sichergestellt, dass keine unnötigen Image-Abrufversuche durchgeführt werden.
- `ECS_IMAGE_PULL_BEHAVIOR`: `prefer-cached`

  Das Bild wird remote abgerufen, wenn kein zwischengespeichertes Bild vorhanden ist. Andernfalls wird das zwischengespeicherte Image in der Instance verwendet. Die automatische Bildbereinigung ist für den Container deaktiviert, um sicherzustellen, dass das zwischengespeicherte Bild nicht entfernt wird.

Wenn Sie den Parameter auf einen der vorherigen Werte setzen, kann dies Zeit sparen, da der Amazon ECS-Agent das vorhandene heruntergeladene Image verwendet. Bei größeren Docker-Images kann die Download-Zeit 10 bis 20 Sekunden dauern, bis sie über das Netzwerk abgerufen werden.

Bereitstellung von Aufgaben

Um sicherzustellen, dass es zu keinen Ausfallzeiten der Anwendung kommt, läuft der Bereitstellungsprozess wie folgt ab:

1. Starten Sie die neuen Anwendungscontainer, während die vorhandenen Container weiterlaufen.
2. Überprüfen Sie, ob die neuen Container fehlerfrei sind.
3. Stoppen Sie die alten Container.

Abhängig von Ihrer Bereitstellungskonfiguration und der Menge an freiem, nicht reserviertem Speicherplatz in Ihrem Cluster kann es mehrere Runden dauern, bis dieser Vorgang abgeschlossen ist und alle alten Aufgaben durch neue Aufgaben ersetzt sind.

Es gibt zwei Konfigurationsoptionen für den ECS-Service, mit denen Sie die Anzahl ändern können:

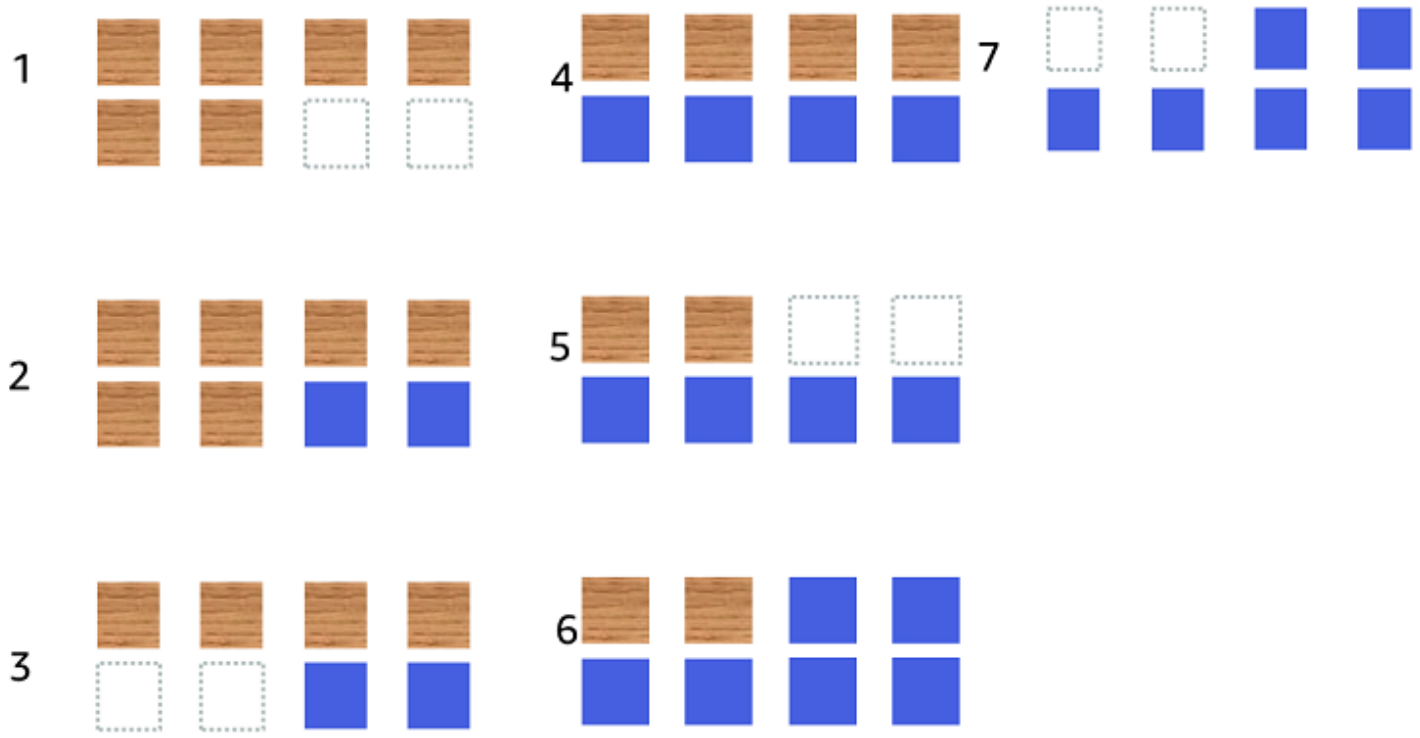
- `minimumHealthyPercent`: 100% (Standard)

  Die Untergrenze für die Anzahl der Aufgaben für Ihren Service, die während einer Bereitstellung den `RUNNING` Status beibehalten müssen. Dies wird als Prozentsatz von DesiredCount dargestellt. Es wird auf die nächste Ganzzahl aufgerundet. Mit diesem Parameter können Sie die Bereitstellung vornehmen, ohne zusätzliche Cluster-Kapazität zu nutzen.

- `maximumPercent`: 200% (Standard)

  Die Obergrenze für die Anzahl der Aufgaben für Ihren Service, die während einer Bereitstellung im `PENDING` Status `RUNNING` oder zulässig sind. Dies wird als Prozentsatz von DesiredCount dargestellt. Es wird auf die nächste Ganzzahl abgerundet.

Stellen Sie sich den folgenden Dienst mit sechs TAN-Aufgaben vor, die in einem Cluster bereitgestellt werden, der Platz für insgesamt acht Aufgaben bietet. Die standardmäßigen Amazon ECS-Servicekonfigurationsoptionen lassen nicht zu, dass die Bereitstellung unter 100% der sechs gewünschten Aufgaben fällt.



Der Bereitstellungsprozess sieht wie folgt aus:

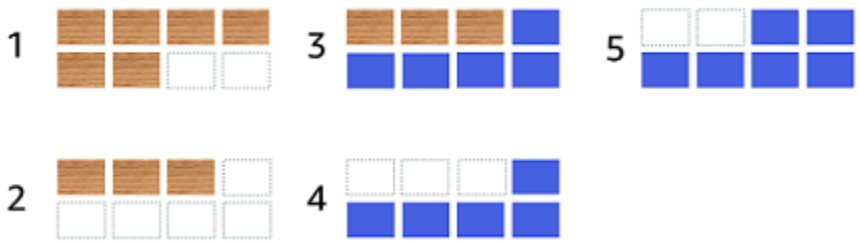
1. Ziel ist es, die hellbraunen Aufgaben durch die blauen Aufgaben zu ersetzen.

2. Der Scheduler startet zwei neue blaue Aufgaben, da die Standardeinstellungen voraussetzen, dass sechs Aufgaben ausgeführt werden.
3. Der Scheduler stoppt zwei der hellbraunen Aufgaben, da es insgesamt sechs Aufgaben geben wird (vier hellbraune und zwei blaue).
4. Der Scheduler startet zwei zusätzliche blaue Aufgaben.
5. Der Scheduler beendet zwei der Tan-Aufgaben.
6. Der Scheduler startet zwei zusätzliche blaue Aufgaben.
7. Der Scheduler beendet die letzten beiden hellbraunen Aufgaben.

Wenn Sie im obigen Beispiel die Standardwerte für die Optionen verwenden, gibt es eine Wartezeit von 2,5 Minuten für jede neue Aufgabe, die gestartet wird. Darüber hinaus muss der Load Balancer möglicherweise 5 Minuten warten, bis die alte Aufgabe beendet ist.

Sie können die Bereitstellung beschleunigen, indem Sie den `minimumHealthyPercent` Wert auf 50% setzen.

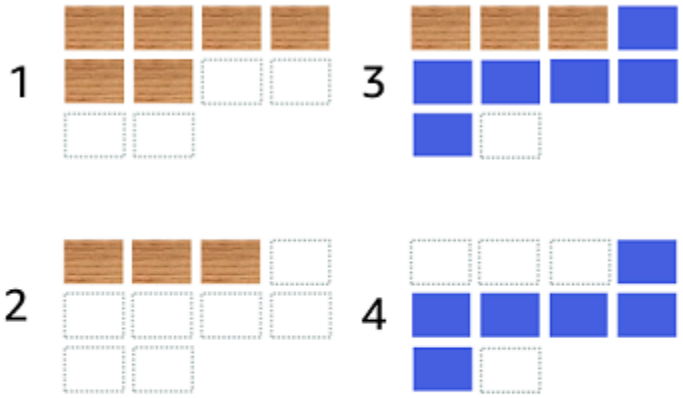
Stellen Sie sich den folgenden Dienst mit sechs Tan-Aufgaben vor, die in einem Cluster bereitgestellt werden, der Platz für insgesamt acht Aufgaben bietet.



Der Bereitstellungsprozess sieht wie folgt aus:

1. Ziel ist es, die hellbraunen Aufgaben durch die blauen Aufgaben zu ersetzen.
2. Der Scheduler stoppt drei der TAN-Aufgaben. Es werden immer noch drei TAN-Aufgaben ausgeführt, die dem `minimumHealthyPercent` Wert entsprechen.
3. Der Scheduler startet fünf blaue Aufgaben.
4. Der Scheduler stoppt die verbleibenden drei hellbraunen Aufgaben.
5. Der Scheduler startet die letzten blauen Aufgaben.

Sie können auch zusätzlichen freien Speicherplatz hinzufügen, sodass Sie zusätzliche Aufgaben ausführen können.



Der Bereitstellungsprozess sieht wie folgt aus:

1. Ziel ist es, die hellbraunen Aufgaben durch die blauen Aufgaben zu ersetzen.
2. Der Scheduler stoppt drei der TAN-Aufgaben
3. Der Scheduler startet sechs blaue Aufgaben
4. Der Scheduler stoppt die drei hellbraunen Aufgaben.

Verwenden Sie die folgenden Werte für die ECS-Servicekonfigurationsoptionen, wenn Ihre Aufgaben für einige Zeit inaktiv sind und keine hohe Auslastungsrate aufweisen.

- `minimumHealthyPercent`: 50%
- `maximumPercent`: 20%

Bewährte Methoden — Betrieb von Amazon ECS im großen Maßstab

Wenn Sie beginnen, Amazon ECS in großem Maßstab zu betreiben, sollten Sie sich überlegen, wie sich Servicekontingenten und API-Drosselungen für Amazon ECS und AWS-Services die Integration mit Amazon ECS auf Sie auswirken könnten. In diesem Thema werden die Servicekontingente und API-Drosselungen detailliert beschrieben und es werden auch andere wichtige Überlegungen zur Skalierung behandelt.

Themen

Servicekontingenten und API-Drosselungsgrenzen

Amazon ECS ist in mehrere Systeme integriert AWS-Services, darunter Elastic Load Balancing und Amazon EC2. AWS Cloud Map Durch diese enge Integration umfasst Amazon ECS mehrere Funktionen wie Service Load Balancing, Service Discovery, Task Networking und auto Clusterskalierung. Amazon ECS und das andere AWS-Services , das es in alle integriert, verwalten Service-Kontingente und API-Ratenbegrenzungen, um eine konsistente Leistung und Auslastung zu gewährleisten. Diese Servicekontingente verhindern auch, dass versehentlich mehr Ressourcen als benötigt bereitgestellt werden, und schützen vor böswilligen Aktionen, die Ihre Rechnung erhöhen könnten.

Wenn Sie sich mit Ihren Servicekontingenten und den AWS API-Ratenbegrenzungen vertraut machen, können Sie die Skalierung Ihrer Workloads planen, ohne sich Gedanken über unerwartete Leistungseinbußen machen zu müssen. Weitere Informationen finden Sie unter Amazon ECS-Servicekontingente und Anforderungsdrosselung für die Amazon ECS-API.

Wir empfehlen Ihnen, bei der Skalierung Ihrer Workloads auf Amazon ECS die folgende Servicequote zu berücksichtigen. Anweisungen, wie Sie eine Erhöhung der Servicekontingenten beantragen können, finden Sie unter Verwaltung Ihrer Amazon ECS- und AWS Fargate Servicekontingenten in der AWS Management Console.

- AWS Fargate verfügt über Kontingente, die jeweils AWS-Region die Anzahl der gleichzeitig ausgeführten Aufgaben begrenzen. Es gibt Kontingente sowohl für On-Demand-Aufgaben als

auch für Fargate Spot-Aufgaben auf Amazon ECS. Jedes Servicekontingent umfasst auch alle Amazon EKS-Pods, die Sie auf Fargate ausführen. Weitere Informationen zu den Fargate-Kontingenten finden Sie unter AWS Fargate Service-Kontingente im Amazon Elastic Container Service-Benutzerhandbuch für AWS Fargate.

- Für Aufgaben, die auf Amazon EC2 EC2-Instances ausgeführt werden, beträgt die maximale Anzahl von Amazon EC2 EC2-Instances, die Sie für jeden Cluster registrieren können, 5.000. Wenn Sie Amazon ECS-Cluster Auto Scaling mit einem Auto Scaling-Gruppenkapazitätsanbieter verwenden oder Amazon EC2 EC2-Instances für Ihren Cluster selbst verwalten, kann dieses Kontingent zu einem Engpass bei der Bereitstellung werden. Wenn Sie mehr Kapazität benötigen, können Sie mehr Cluster erstellen oder eine Erhöhung der Servicekontingenten beantragen.
- Wenn Sie Amazon ECS Cluster Auto Scaling mit einem Auto Scaling-Gruppenkapazitätsanbieter verwenden, berücksichtigen Sie bei der Skalierung Ihrer Services das `Tasks in the PROVISIONING state per cluster` Kontingent. Dieses Kontingent ist die maximale Anzahl von Aufgaben im `PROVISIONING` Bundesstaat für jeden Cluster, für die Kapazitätsanbieter die Kapazität erhöhen können. Wenn Sie eine große Anzahl von Aufgaben gleichzeitig starten, können Sie dieses Kontingent problemlos einhalten. Ein Beispiel ist die gleichzeitige Bereitstellung von Dutzenden von Diensten mit jeweils Hunderten von Aufgaben. In diesem Fall muss der Kapazitätsanbieter neue Container-Instances starten, um die Aufgaben zu platzieren, wenn der Cluster nicht genügend Kapazität hat. Während der Kapazitätsanbieter weitere Amazon EC2 EC2-Instances auf den Markt bringt, wird der Amazon ECS-Service Scheduler wahrscheinlich weiterhin Aufgaben parallel starten. Diese Aktivität kann jedoch aufgrund unzureichender Clusterkapazität gedrosselt werden. Der Amazon ECS Service Scheduler implementiert eine Back-off- und exponentielle Drosselungsstrategie, mit der versucht wird, Aufgaben erneut zu platzieren, wenn neue Container-Instances gestartet werden. Infolgedessen kann es zu langsameren Bereitstellungs- oder Skalierungszeiten kommen. Um diese Situation zu vermeiden, können Sie Ihre Servicebereitstellungen in einer der folgenden Arten planen. Stellen Sie entweder eine große Anzahl von Aufgaben bereit, ohne dass die Clusterkapazität erhöht werden muss, oder Sie behalten freie Clusterkapazität für den Start neuer Aufgaben.

Berücksichtigen Sie bei der Skalierung Ihrer Workloads nicht nur das Amazon ECS-Servicekontingent, sondern auch das Servicekontingent für die anderen AWS-Services , die in Amazon ECS integriert sind. Der folgende Abschnitt behandelt die wichtigsten Ratenlimits für jeden Service im Detail und enthält Empfehlungen zum Umgang mit potenziellen Drosselungsproblemen.

Elastic Load Balancing

Sie können Ihre Amazon ECS-Services so konfigurieren, dass Elastic Load Balancing verwendet wird, um den Traffic gleichmäßig auf die Aufgaben zu verteilen. Weitere Informationen und empfohlene Best Practices für die Auswahl eines Load Balancers finden Sie unter Überlegungen zum Service-Load-Balancing und Parameter für die Load Balancer-Zustandsprüfung.

Elastic Load Balancing Balancing-Dienstkontingente

Wenn Sie Ihre Workloads skalieren, sollten Sie die folgenden Elastic Load Balancing Service-Kontingente berücksichtigen. Die meisten Elastic Load Balancing Balancing-Dienstkontingente sind anpassbar, und Sie können eine Erhöhung in der Service Quotas-Konsole beantragen.

Application Load Balancer

Wenn Sie einen Application Load Balancer verwenden, müssen Sie je nach Anwendungsfall möglicherweise eine Erhöhung des Kontingents beantragen für:

- Das `Targets per Application Load Balancer` Kontingent, das die Anzahl der Ziele hinter Ihrem Application Load Balancer darstellt.
- Die `Targets per Target Group per Region` Quote, die der Anzahl der Ziele hinter Ihren Zielgruppen entspricht.

Weitere Informationen finden Sie unter Kontingente für Ihre Application Load Balancer.

Network Load Balancer

Die Anzahl der Ziele, die Sie bei einem Network Load Balancer registrieren können, ist strenger begrenzt. Wenn Sie einen Network Load Balancer verwenden, möchten Sie häufig die zonenübergreifende Unterstützung aktivieren, was zusätzliche Skalierungsbeschränkungen für `Targets per Availability Zone Per Network Load Balancer` die maximale Anzahl von Zielen pro Availability Zone für jeden Network Load Balancer mit sich bringt. Weitere Informationen finden Sie unter Kontingente für Ihre Network Load Balancer.

Drosselung der Elastic Load Balancing Balancing-API

Wenn Sie einen Amazon ECS-Service für die Verwendung eines Load Balancers konfigurieren, müssen die Zustandsprüfungen der Zielgruppe bestanden werden, bevor der Service als fehlerfrei eingestuft wird. Für die Durchführung dieser Zustandsprüfungen ruft Amazon ECS

in Ihrem Namen Elastic Load Balancing API-Operationen auf. Wenn Sie in Ihrem Konto eine große Anzahl von Diensten mit Load Balancern konfiguriert haben, verlangsamen Sie möglicherweise die Servicebereitstellung, da es zu einer möglichen Drosselung speziell für die API-Operationen`RegisterTarget`,`DeregisterTarget`, und `DescribeTargetHealth` Elastic Load Balancing kommen kann. Wenn eine Drosselung auftritt, treten Drosselungsfehler in Ihren Amazon ECS-Serviceereignismeldungen auf.

Wenn Sie eine AWS Cloud Map API-Drosselung feststellen, können Sie sich an uns wenden AWS Support , um Hilfe zu erhalten, wie Sie Ihre API-Drosselungsgrenzen erhöhen können. AWS Cloud Map Weitere Informationen zur Überwachung und Behebung solcher Drosselungsfehler finden Sie unter Umgang mit Drosselungsproblemen.

Elastic-Network-Schnittstelle

Wenn Ihre Aufgaben den `awsvpc` Netzwerkmodus verwenden, stellt Amazon ECS für jede Aufgabe eine eigene elastic network interface (ENI) bereit. Wenn Ihre Amazon ECS-Services einen Elastic Load Balancing Load Balancer verwenden, werden diese Netzwerkschnittstellen auch als Ziele für die entsprechende, im Service definierte Zielgruppe registriert.

Service-Kontingente für elastische Netzwerkschnittstellen

Wenn Sie Aufgaben ausführen, die den `awsvpc` Netzwerkmodus verwenden, wird jeder Aufgabe eine eindeutige elastic network interface zugewiesen. Wenn diese Aufgaben über das Internet erreicht werden müssen, weisen Sie der elastic network interface für diese Aufgaben eine öffentliche IP-Adresse zu. Wenn Sie Ihre Amazon ECS-Workloads skalieren, sollten Sie diese beiden wichtigen Kontingente berücksichtigen:

- Das `Network interfaces per Region` Kontingent, das die maximale Anzahl von Netzwerkschnittstellen in einem AWS-Region für Ihr Konto ist.
- Das `Elastic IP addresses per Region` Kontingent, das die maximale Anzahl elastischer IP-Adressen in einem darstellt AWS-Region.

Beide Service Quotas sind anpassbar, und Sie können für diese über Ihre Servicekontingenta-Konsole eine Erhöhung beantragen. Weitere Informationen finden Sie unter Amazon VPC Service Quotas.

Bei Amazon ECS-Workloads, die auf Amazon EC2 EC2-Instances gehostet werden, sollten Sie bei der Ausführung von Aufgaben, die den `awsvpc` Netzwerkmodus verwenden, das `Maximum`

`network interfaces` Service-Kontingent, die maximale Anzahl von Netzwerk-Instances für jede Amazon EC2 EC2-Instance, berücksichtigen. Dieses Kontingent begrenzt die Anzahl der Aufgaben, die Sie auf einer Instance platzieren können. Sie können das Kontingent nicht anpassen und es ist in der Service Quotas Quotas-Konsole nicht verfügbar. Weitere Informationen finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im Amazon EC2 EC2-Benutzerhandbuch.

Sie können zwar die Anzahl der Netzwerkschnittstellen, die an eine Amazon EC2 EC2-Instance angeschlossen werden können, nicht ändern, aber Sie können die elastic network interface Trunking-Funktion verwenden, um die Anzahl der verfügbaren Netzwerkschnittstellen zu erhöhen. Beispielsweise kann eine `c5.large` Instance standardmäßig bis zu drei Netzwerkschnittstellen haben. Die primäre Netzwerkschnittstelle für die Instance zählt dazu. Sie können also zwei weitere Netzwerkschnittstellen an die Instanz anhängen. Da für jede Aufgabe, die den `awsvpc` Netzwerkmodus verwendet, eine Netzwerkschnittstelle erforderlich ist, können Sie für diesen Instance-Typ in der Regel nur zwei solcher Aufgaben ausführen. Dies kann zu einer Unterauslastung Ihrer Clusterkapazität führen. Wenn Sie elastic network interface Trunking aktivieren, können Sie die Netzwerkschnittstellendichte erhöhen, um eine größere Anzahl von Aufgaben auf jeder Instance zu platzieren. Wenn Trunking aktiviert ist, kann eine `c5.large` Instance bis zu 12 Netzwerkschnittstellen haben. Die Instance hat die primäre Netzwerkschnittstelle und Amazon ECS erstellt und fügt der Instance eine „Trunk“ -Netzwerkschnittstelle hinzu. Daher können Sie mit dieser Konfiguration 10 Aufgaben statt der standardmäßigen zwei Aufgaben auf der Instance ausführen. Weitere Informationen finden Sie unter Elastic Network Interface Trunking.

Drosselung der API für elastische Netzwerkschnittstellen

Wenn Sie Aufgaben ausführen, die den `awsvpc` Netzwerkmodus verwenden, stützt sich Amazon ECS auf die folgenden Amazon EC2 EC2-APIs. Jede dieser APIs hat unterschiedliche API-Drosselungen. Weitere Informationen finden Sie unter Anforderungsdrosselung für die Amazon EC2 EC2-API.

- CreateNetworkInterface
- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces
- DescribeVpcs
- DescribeSubnets

- DescribeSecurityGroups
- DescribeInstances

Wenn die Amazon EC2 EC2-API-Aufrufe während der Workflows zur Bereitstellung elastic network interface Netzwerkschnittstellen gedrosselt werden, versucht der Amazon ECS-Service Scheduler es automatisch mit exponentiellen Back-offs erneut. Diese automatischen Stilllegungen können manchmal zu Verzögerungen beim Starten von Aufgaben führen, was wiederum zu langsameren Bereitstellungsgeschwindigkeiten führt. Wenn es zu einer API-Drosselung kommt, wird die entsprechende Meldung in Ihren Service-Event-Meldungen `Operations are being throttled. Will try again later.` angezeigt. Wenn Sie die Amazon EC2 EC2-API-Drosselungen regelmäßig einhalten, können Sie sich an uns wenden AWS Support , um Unterstützung zu erhalten, wie Sie Ihre API-Drosselungsgrenzwerte erhöhen können. Weitere Informationen zur Überwachung und Behebung von Drosselungsfehlern finden Sie unter Umgang mit Drosselungsproblemen.

AWS Cloud Map

Amazon ECS Service Discovery verwendet AWS Cloud Map APIs, um Namespaces für Ihre Amazon ECS-Services zu verwalten. Wenn Ihre Services eine große Anzahl von Aufgaben haben, sollten Sie die folgenden Empfehlungen berücksichtigen. Weitere Informationen finden Sie unter Überlegungen zur Amazon ECS Service Discovery.

AWS Cloud Map Service-Kontingente

Wenn Amazon ECS-Services für die Verwendung von Service Discovery konfiguriert sind, wird das `Tasks per service` Kontingent, das die maximale Anzahl von Aufgaben für den Service darstellt, von dem AWS Cloud Map `Instances per service` Service-Kontingent beeinflusst, das die maximale Anzahl von Instances für diesen Service darstellt. Insbesondere reduziert das AWS Cloud Map Service-Kontingent die Anzahl der Aufgaben, die Sie ausführen können, auf maximal 1.000 Serviceaufgaben. Sie können das AWS Cloud Map Kontingent nicht ändern. Weitere Informationen finden Sie unter AWS Cloud Map -Servicekontingente.

AWS Cloud Map API-Drosselung

Amazon ECS ruft die `DeregisterInstance` AWS Cloud Map APIs `ListInstancesGetInstancesHealthStatus`,`RegisterInstance`, und in Ihrem Namen auf. Sie helfen bei der Serviceerkennung und führen Integritätsprüfungen durch, wenn Sie eine Aufgabe starten. Wenn mehrere Dienste, die Service Discovery mit einer großen Anzahl von Aufgaben

verwenden, gleichzeitig bereitgestellt werden, kann dies zu einer Überschreitung der AWS Cloud Map API-Drosselungsgrenzen führen. In diesem Fall werden Sie wahrscheinlich die folgende Meldung sehen: `Operations are being throttled. Will try again later` in Ihren Amazon ECS-Serviceereignismeldungen und langsamere Bereitstellungs- und Taskstartgeschwindigkeit. AWS Cloud Map dokumentiert keine Drosselungsgrenzen für diese APIs. Wenn Sie aufgrund dieser Drosselungen feststellen, können Sie sich an uns wenden, um Hilfe AWS Support zur Erhöhung Ihrer API-Drosselungsgrenzen zu erhalten. Weitere Empfehlungen zur Überwachung und Behebung solcher Drosselungsfehler finden Sie unter Umgang mit Drosselungsproblemen.

Umgang mit Drosselungsproblemen

Dieser Abschnitt bietet einen detaillierten Überblick über einige Strategien, mit denen Sie API-Drosselungsfehler überwachen und beheben können. Drosselungsfehler lassen sich in zwei Hauptkategorien einteilen: synchrone Drosselung und asynchrone Drosselung.

Synchrone Drosselung

Wenn synchrone Drosselung auftritt, erhalten Sie sofort eine Fehlerantwort von Amazon ECS. Diese Kategorie der Drosselung tritt normalerweise auf, wenn Sie Amazon ECS-APIs aufrufen, während Sie Aufgaben ausführen oder Dienste erstellen. Weitere Informationen über die damit verbundene Drosselung und die entsprechenden Drosselungsgrenzen finden Sie unter Drosselung von Anfragen für die Amazon ECS-API.

Wenn Ihre Anwendung API-Anfragen initiiert, beispielsweise mithilfe des AWS CLI oder eines AWS SDK, können Sie die API-Drosselung beheben. Sie können dies tun, indem Sie entweder Ihre Anwendung so gestalten, dass sie die Fehler behandelt, oder indem Sie eine exponentielle Backoff- und Jitter-Strategie mit Wiederholungslogik für die API-Aufrufe implementieren. Weitere Informationen finden Sie unter Timeouts, Wiederholungen und Backoff mit Jitter.

Wenn Sie ein AWS SDK verwenden, ist die automatische Wiederholungslogik bereits integriert und konfigurierbar.

Asynchrone Drosselung

Asynchrone Drosselung ist auf asynchrone Workflows zurückzuführen, bei denen Amazon ECS oder AWS CloudFormation möglicherweise in Ihrem Namen APIs aufrufen, um Ressourcen bereitzustellen. Es ist wichtig zu wissen, welche AWS APIs Amazon ECS in Ihrem Namen aufruft. Die `CreateNetworkInterface` API wird beispielsweise für Aufgaben aufgerufen, die den

awsvpc Netzwerkmodus verwenden, und die DescribeTargetHealth API wird aufgerufen, wenn Zustandsprüfungen für Aufgaben durchgeführt werden, die bei einem Load Balancer registriert sind.

Wenn Ihre Workloads ein beträchtliches Ausmaß erreichen, werden diese API-Operationen möglicherweise gedrosselt. Das heißt, sie werden möglicherweise so stark gedrosselt, dass sie die von Amazon ECS oder dem AWS-Service aufgerufenen System durchgesetzten Grenzwerte überschreiten. Wenn Sie beispielsweise Hunderte von Services bereitstellen, die jeweils Hunderte von Aufgaben gleichzeitig ausführen und den awsvpc Netzwerkmodus verwenden, ruft Amazon ECS Amazon EC2 EC2-API-Operationen wie CreateNetworkInterface und Elastic Load Balancing Balancing-API-Operationen wie RegisterTarget oder auf, DescribeTargetHealth um die elastic network interface bzw. den Load Balancer zu registrieren. Diese API-Aufrufe können die API-Grenzwerte überschreiten, was zu Drosselungsfehlern führen kann. Im Folgenden finden Sie ein Beispiel für einen Elastic Load Balancing Balancing-Drosselungsfehler, der in der Service-Event-Meldung enthalten ist.

```
{
   "userIdentity":{
      "arn":"arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForECS/ecs-service-
scheduler",
      "eventTime":"2022-03-21T08:11:24Z",
      "eventSource":"elasticloadbalancing.amazonaws.com",
      "eventName":" DescribeTargetHealth ",
      "awsRegion":"us-east-1",
      "sourceIPAddress":"ecs.amazonaws.com",
      "userAgent":"ecs.amazonaws.com",
      "errorCode":"ThrottlingException",
      "errorMessage":"Rate exceeded",
      "eventID":"0aeb38fc-229b-4912-8b0d-2e8315193e9c"
   }
}
```

Wenn diese API-Aufrufe dieselben Grenzwerte wie anderer API-Traffic in Ihrem Konto haben, kann es schwierig sein, sie zu überwachen, obwohl sie als Service-Ereignisse ausgegeben werden.

Überwachung, Drosselung

Um die Drosselung zu überwachen, ist es wichtig zu ermitteln, welche API-Anfragen gedrosselt werden und wer diese Anfragen stellt. Sie können es verwenden AWS CloudTrail , um es zu tun. Dieser Service überwacht die Drosselung und kann in Amazon Athena und Amazon integriert werden. CloudWatch EventBridge Sie können so konfigurieren CloudTrail , dass

bestimmte Ereignisse an Logs gesendet werden. CloudWatch Diese Ereignisse werden dann mithilfe von CloudWatch Logs Log Insights analysiert und analysiert. Dadurch werden Details bei Drosselungsereignissen identifiziert, z. B. der Benutzer oder die IAM-Rolle, die den Anruf getätigt hat, und die Anzahl der getätigten API-Aufrufe. Weitere Informationen finden Sie unter Überwachen von CloudTrail Protokolldateien mit Protokollen. CloudWatch

Weitere Informationen zu CloudWatch Logs Insights und Anweisungen zum Abfragen von Protokolldateien finden Sie unter Logdaten mit CloudWatch Logs Insights analysieren.

Mit Amazon Athena können Sie Abfragen erstellen und Daten mit Standard-SQL analysieren. Sie können beispielsweise eine Athena-Tabelle erstellen, um Ereignisse zu analysieren CloudTrail . Weitere Informationen finden Sie unter Verwenden der CloudTrail Konsole zum Erstellen einer Athena-Tabelle für CloudTrail Protokolle.

Nachdem Sie eine Athena-Tabelle erstellt haben, können Sie einfache SQL-Abfragen wie die folgende verwenden, um `ThrottlingException` Fehler zu untersuchen.

```
select eventname, errorcode,eventsource,awsregion, useragent,COUNT(*) count
FROM cloudtrail-table-name
where errorcode = 'ThrottlingException'
AND eventtime between '2022-01-14T03:00:08Z' and '2022-01-23T07:15:08Z'
group by errorcode, awsregion, eventsource, username, eventname
order by count desc;
```

Amazon ECS sendet auch Ereignisbenachrichtigungen an Amazon EventBridge. Es gibt Ereignisse zur Änderung des Ressourcenstatus und Ereignisse bei Serviceaktionen. Dazu gehören API-Drosselungsereignisse wie und`ECS_OPERATION_THROTTLED`. `SERVICE_DISCOVERY_OPERATION_THROTTLED` Weitere Informationen finden Sie unter Amazon ECS Service Action Events.

Diese Ereignisse können von einem Service genutzt werden, AWS Lambda um beispielsweise als Reaktion darauf Aktionen auszuführen. Weitere Informationen finden Sie unter Behandlung von Ereignissen.

Wenn Sie eigenständige Aufgaben ausführen, `RunTask` sind einige API-Operationen, z. B. asynchron, und Wiederholungsvorgänge werden nicht automatisch ausgeführt. In solchen Fällen können Sie Dienste wie AWS Step Functions die EventBridge Integration verwenden, um gedrosselte oder fehlgeschlagene Operationen erneut zu versuchen. Weitere Informationen finden Sie unter Container-Aufgaben verwalten (Amazon ECS, Amazon SNS).

Wird CloudWatch zur Überwachung der Drosselung verwendet

CloudWatch bietet die Überwachung der API-Nutzung im `Usage` Namespace unter By AWS Resource. Diese Metriken werden mit dem Typ API und dem Metriknamen CallCountprotokolliert. Sie können Alarme erstellen, die immer dann ausgelöst werden, wenn diese Metriken einen bestimmten Schwellenwert erreichen. Weitere Informationen finden Sie unter Visualisieren Ihrer Servicequotas und Einstellen von Alarmen.

CloudWatch bietet auch eine Erkennung von Anomalien. Diese Funktion verwendet maschinelles Lernen, um Basiswerte zu analysieren und festzulegen, die auf dem spezifischen Verhalten der Metrik basieren, für die Sie sie aktiviert haben. Bei ungewöhnlichen API-Aktivitäten können Sie diese Funktion zusammen mit CloudWatch Alarmen verwenden. Weitere Informationen finden Sie unter Verwenden der CloudWatch Anomalieerkennung.

Durch die proaktive Überwachung von Drosselungsfehlern können Sie sich an uns wenden, AWS Support um die entsprechenden Drosselungsgrenzwerte zu erhöhen und Unterstützung für Ihre individuellen Anwendungsanforderungen zu erhalten.

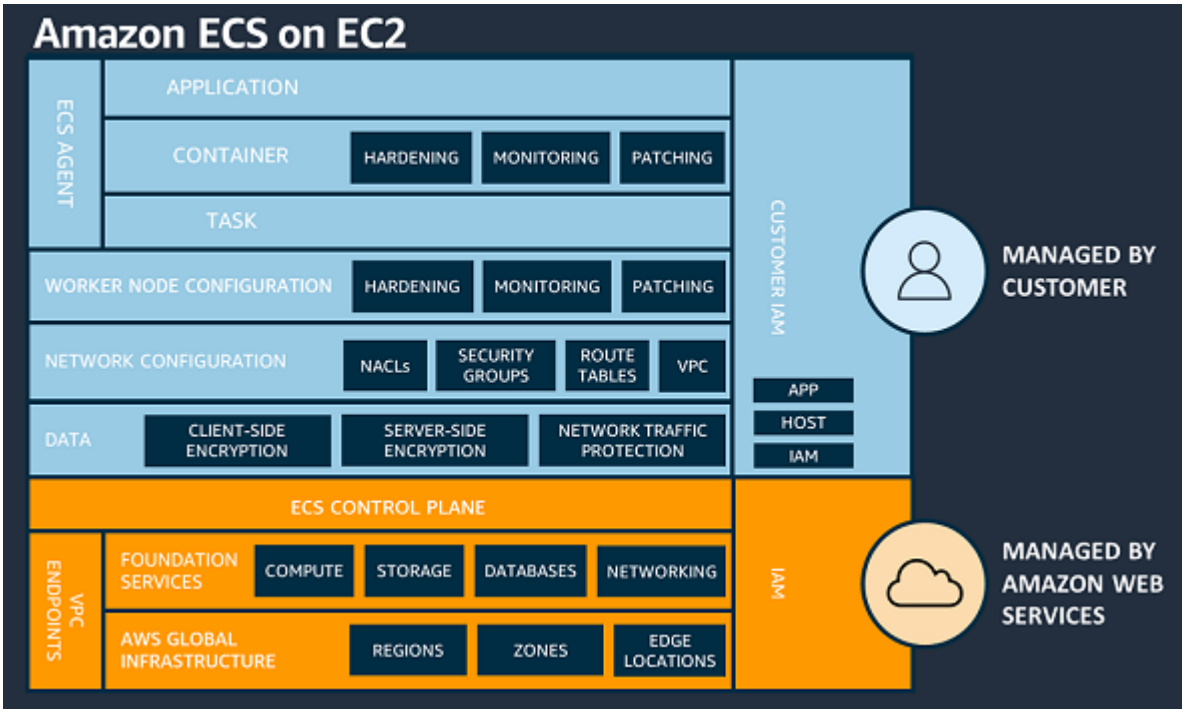
Bewährte Methoden — Sicherheit

Dieser Leitfaden enthält Sicherheits- und Compliance-Empfehlungen zum Schutz Ihrer Informationen, Systeme und anderen Ressourcen, die auf Amazon ECS angewiesen sind. Außerdem werden einige Risikobewertungen und Strategien zur Risikominderung vorgestellt, mit denen Sie die Sicherheitskontrollen, die für Amazon-ECS-Cluster entwickelt wurden, und die Workloads, die sie unterstützen, besser kontrollieren können. Jedes Thema in diesem Handbuch beginnt mit einem kurzen Überblick, gefolgt von einer Liste mit Empfehlungen und bewährten Methoden, mit denen Sie Ihre Amazon-ECS-Cluster schützen können.

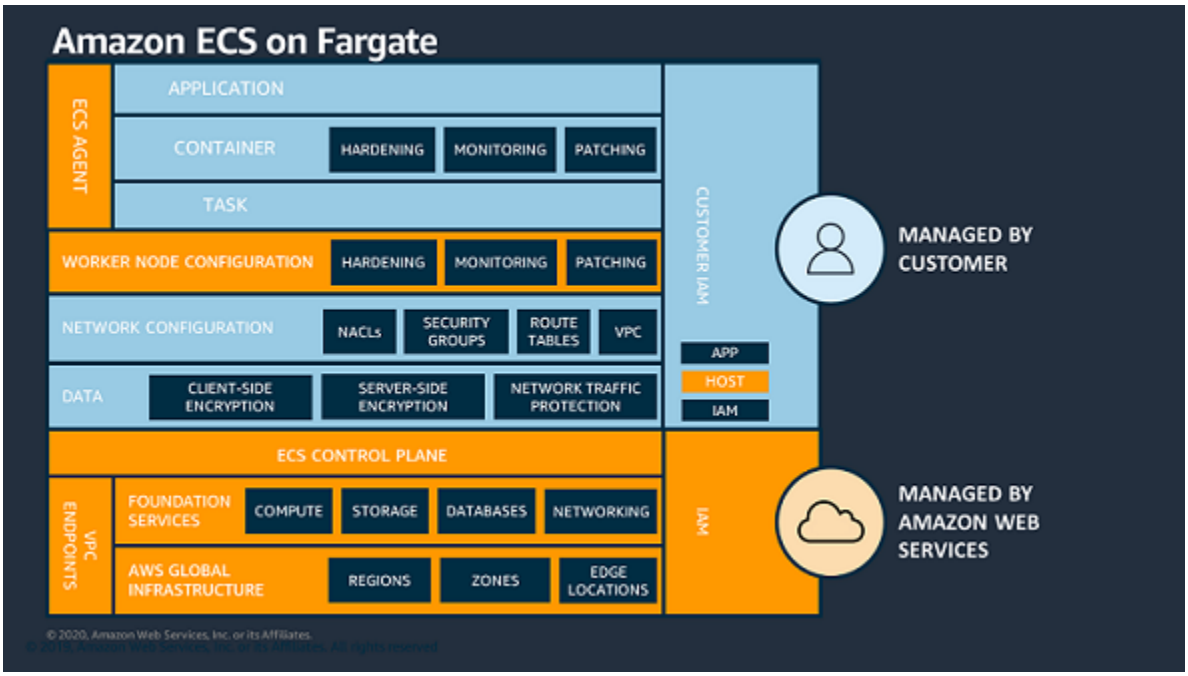
Themen

Modell der geteilten Verantwortung

Die Sicherheit und Konformität eines verwalteten Services wie Amazon ECS liegt in der gemeinsamen Verantwortung von Ihnen und AWS. Im Allgemeinen AWS ist Amazon für die Sicherheit „der“ Cloud verantwortlich, wohingegen Sie, der Kunde, für die Sicherheit „in“ der Cloud verantwortlich sind. AWS ist verantwortlich für die Verwaltung der Amazon ECS-Steuerebene, einschließlich der Infrastruktur, die für die Bereitstellung eines sicheren und zuverlässigen Dienstes erforderlich ist. Und Sie sind größtenteils für die Themen in diesem Handbuch verantwortlich. Dazu gehören Daten-, Netzwerk- und Laufzeitsicherheit sowie Protokollierung und Überwachung.



AWS Übernimmt in Bezug auf die Infrastruktursicherheit mehr Verantwortung für AWS Fargate Ressourcen als für andere selbstverwaltete Instances. AWS Verwaltet mit Fargate die Sicherheit der zugrunde liegenden Instanz in der Cloud und die Laufzeit, die für die Ausführung Ihrer Aufgaben verwendet wird. Fargate skaliert Ihre Infrastruktur auch automatisch in Ihrem Namen.



Bevor Sie Ihre Dienste auf die Cloud ausweiten, sollten Sie sich darüber im Klaren sein, für welche Sicherheits- und Compliance-Aspekte Sie verantwortlich sind.

Weitere Informationen zum Modell der gemeinsamen Verantwortung finden Sie unter Modell der gemeinsamen Verantwortung.

AWS Identity and Access Management

Sie können AWS Identity and Access Management (IAM) verwenden, um den Zugriff auf Ihre AWS Dienste und Ressourcen mithilfe von regelbasierten Richtlinien für Authentifizierungs- und Autorisierungszwecke zu verwalten und zu kontrollieren. Insbesondere kontrollieren Sie mit diesem Service den Zugriff auf Ihre AWS Ressourcen mithilfe von Richtlinien, die auf Benutzer, Gruppen oder Rollen angewendet werden. Unter diesen drei sind Benutzer die Konten, die Zugriff auf Ihre Ressourcen haben können. Und eine IAM-Rolle besteht aus einer Reihe von Berechtigungen, die von einer authentifizierten Identität übernommen werden können, die keiner bestimmten Identität außerhalb von IAM zugeordnet ist. Weitere Informationen finden Sie unter Übersicht über die Zugriffsverwaltung: Berechtigungen und Richtlinien.

Verwalten des Zugriffs auf Amazon ECS

Sie können den Zugriff auf Amazon ECS kontrollieren, indem Sie IAM-Richtlinien erstellen und anwenden. Diese Richtlinien bestehen aus einer Reihe von Aktionen, die für eine bestimmte Gruppe von Ressourcen gelten. Die Aktion einer Richtlinie definiert die Liste der Operationen (wie Amazon-ECS-APIs), die zugelassen oder verweigert werden, wohingegen die Ressource steuert, für welche Amazon-ECS-Objekte die Aktion gilt. Einer Richtlinie können Bedingungen hinzugefügt werden, um ihren Geltungsbereich einzugrenzen. Beispielsweise kann eine Richtlinie so geschrieben werden, dass eine Aktion nur für Aufgaben mit einem bestimmten Satz von Tags ausgeführt werden kann. Weitere Informationen finden Sie unter So arbeitet Amazon ECS mit IAM im Entwicklerhandbuch zum Amazon Elastic Container Service.

Empfehlungen

Es wird empfohlen, beim Einrichten Ihrer IAM-Rollen und -Richtlinien Folgendes auszuführen.

Sich an die Richtlinie mit dem Zugriff mit der geringsten Berechtigung halten

Erstellen Sie Richtlinien, die so begrenzt sind, dass Benutzer ihre vorgeschriebenen Aufgaben ausführen können. Wenn ein Entwickler beispielsweise eine Aufgabe regelmäßig beenden muss, erstellen Sie eine Richtlinie, die nur diese bestimmte Aktion zulässt. Das folgende Beispiel erlaubt es einem Benutzer nur, eine Aufgabe zu beenden, die zu einer bestimmten `task_family` auf einem Cluster mit einem bestimmten Amazon-Ressourcenname (ARN) gehört. Der Verweis auf einen ARN in einer Bedingung ist auch ein Beispiel für die Verwendung von Berechtigungen auf

Ressourcenebene. Sie können Berechtigungen auf Ressourcenebene verwenden, um die Ressource anzugeben, auf die eine Aktion angewendet werden soll.

> **Note**
>
> Wenn Sie in einer Richtlinie auf einen ARN verweisen, verwenden Sie das neue längere ARN-Format. Weitere Informationen finden Sie unter Amazon-Ressourcenname (ARN) und IDs im Entwicklerhandbuch für Amazon Elastic Container Service.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:StopTask"
      ],
      "Condition": {
        "ArnEquals": {
          "ecs:cluster": "arn:aws:ecs:region:account_id:cluster/cluster_name"
        }
      },
      "Resource": [
        "arn:aws:ecs:region:account_id:task-definition/task_family:*"
      ]
    }
  ]
}
```

Die Cluster-Ressource als administrative Grenze dienen lassen

Richtlinien, die zu eng gefasst sind, können zu einer Vielzahl von Rollen führen und den Verwaltungsaufwand erhöhen. Anstatt Rollen zu erstellen, die nur auf bestimmte Aufgaben oder Services beschränkt sind, sollten Sie Rollen erstellen, die auf Cluster zugeschnitten sind, und den Cluster als primäre administrative Grenze verwenden.

Endbenutzer von der Amazon-ECS-API durch automatisierte Pipelines isolieren

Sie können die Aktionen einschränken, die Benutzer verwenden können, indem Sie Pipelines erstellen, die Anwendungen automatisch verpacken und auf Amazon-ECS-Clustern bereitstellen.

Dadurch wird das Erstellen, Aktualisieren und Löschen von Aufgaben effektiv an die Pipeline delegiert. Weitere Informationen finden Sie unter Tutorial: Amazon ECS-Standardbereitstellung mit CodePipeline im AWS CodePipeline Benutzerhandbuch.

Richtlinienbedingungen für zusätzliche Sicherheit verwenden

Wenn Sie eine zusätzliche Sicherheitsebene benötigen, fügen Sie Ihrer Richtlinie eine Bedingung hinzu. Dies kann nützlich sein, wenn Sie einen privilegierten Vorgang ausführen oder wenn Sie die Anzahl der Aktionen einschränken müssen, die für bestimmte Ressourcen ausgeführt werden können. Die folgende Beispielrichtlinie erfordert eine mehrstufige Autorisierung beim Löschen eines Clusters.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:DeleteCluster"
      ],
      "Condition": {
        "Bool": {
          "aws:MultiFactorAuthPresent": "true"
        }
      },
    "Resource": ["*"]
    }
  ]
}
```

Auf Services angewendete Tags werden auf alle Aufgaben übertragen, die Teil dieses Services sind. Aus diesem Grund können Sie Rollen erstellen, die auf Amazon-ECS-Ressourcen mit bestimmten Tags beschränkt sind. In der folgenden Richtlinie startet und stoppt ein IAM-Prinzipal alle Aufgaben mit einem Tag-Schlüssel von `Department` und einem Tag-Wert von `Accounting`.

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

```
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Resource": "arn:aws:ecs:*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Department": "Accounting"}
            }
        }
    ]
}
```

Regelmäßig den Zugriff auf die Amazon-ECS-APIs überprüfen

Ein Benutzer könnte die Rollen wechseln. Nach einem Rollenwechsel gelten die Berechtigungen, die ihnen zuvor gewährt wurden, möglicherweise nicht mehr. Stellen Sie sicher, dass Sie prüfen, wer Zugriff auf die Amazon-ECS-APIs hat und ob dieser Zugriff weiterhin gewährleistet ist. Erwägen Sie die Integration von IAM mit einer Lösung für das Benutzerlebenszyklusmanagement, die den Zugriff automatisch widerruft, wenn ein Benutzer das Unternehmen verlässt. Weitere Informationen finden Sie in den Amazon-ECS-Sicherheitsaudit-Richtlinien im Allgemeine Amazon Web Services-Referenz.

Verwenden von IAM-Rollen mit Amazon-ECS-Aufgaben

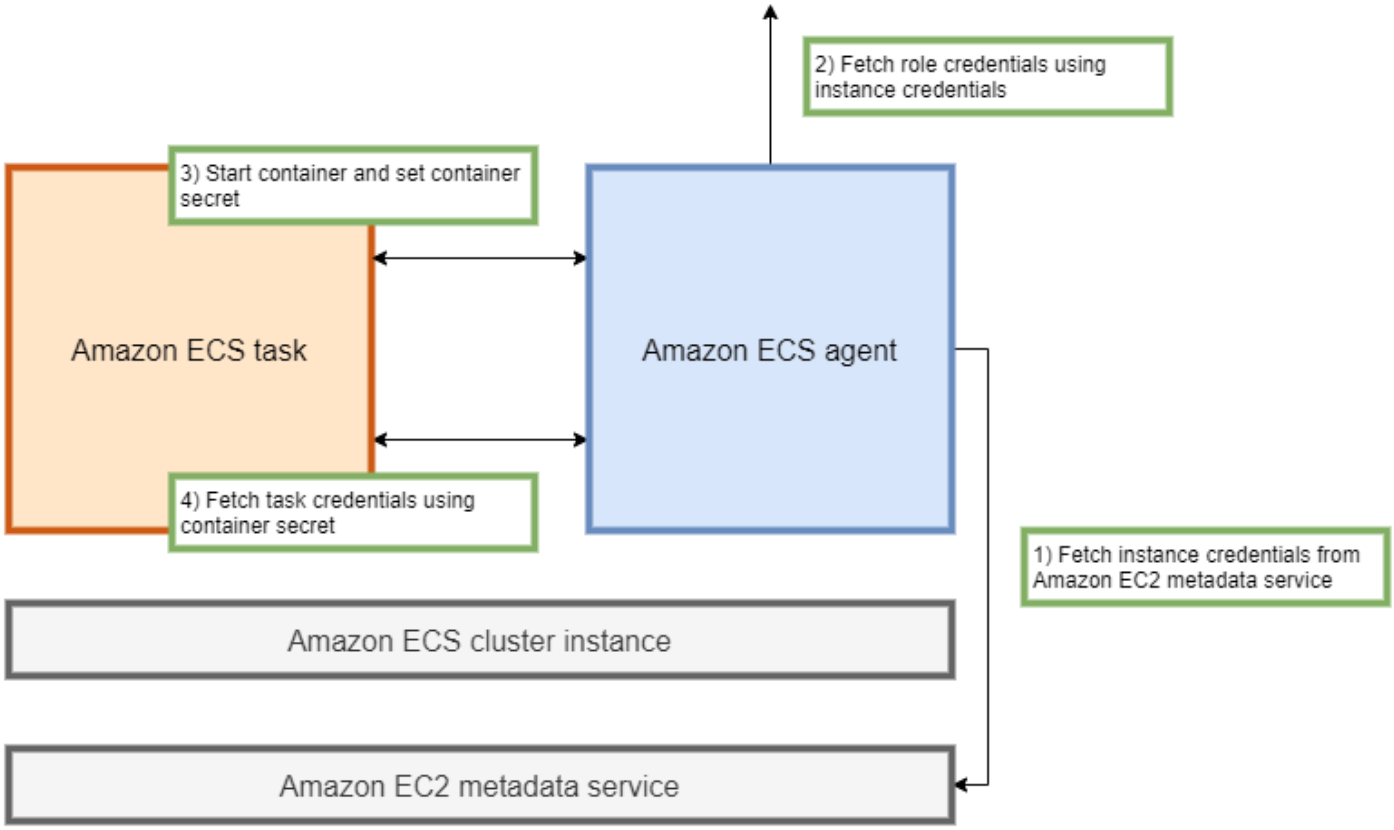
Wir empfehlen, einer Aufgabe eine IAM-Rolle zuzuweisen. Ihre Rolle kann von der Rolle der Amazon-EC2-Instance unterschieden werden, auf der sie ausgeführt werden. Die Zuweisung einer Rolle für jede Aufgabe entspricht dem Prinzip des Zugriffs mit der geringsten Berechtigung und ermöglicht eine differenzierte Kontrolle über Aktionen und Ressourcen.

Wenn Sie einer Aufgabe IAM-Rollen zuweisen, müssen Sie die folgende Vertrauensrichtlinie verwenden, damit jede Ihrer Aufgaben eine IAM-Rolle annehmen kann, die sich von der unterscheidet, die Ihre EC2-Instance verwendet. Auf diese Weise erbt Ihre Aufgabe nicht die Rolle Ihrer EC2-Instance.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Wenn Sie einer Aufgabendefinition eine Aufgabenrolle hinzufügen, erstellt der Amazon-ECS-Container-Agent automatisch ein Token mit einer eindeutigen Anmeldeinformations-ID (z. B. `12345678-90ab-cdef-1234-567890abcdef`) für die Aufgabe. Dieses Token und die Rollen-Anmeldeinformationen werden dann dem internen Cache des Agenten hinzugefügt. Der Agent füllt die Umgebungsvariable `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` im Container mit der URI der Anmeldeinformations-ID auf (z. B. `/v2/credentials/12345678-90ab-cdef-1234-567890abcdef`).



Sie können die temporären Rollenanmeldedaten manuell aus einem Container abrufen, indem Sie die Umgebungsvariable an die IP-Adresse des Amazon-ECS-Container-Agenten anhängen und den `curl`-Befehl für die resultierende Zeichenfolge ausführen.

```
curl 169.254.170.2$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI
```

Die erwartete Ausgabe sieht wie folgt aus:

```
{
 "RoleArn": "arn:aws:iam::123456789012:role/SSMTaskRole-SSMFargateTaskIAMRole-
DASWWSF2WGD6",
 "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
 "Token": "IQoJb3JpZ2luX2VjEEM/Example==",
 "Expiration": "2021-01-16T00:51:53Z"
}
```

Neuere Versionen der AWS SDKs rufen diese Anmeldeinformationen bei AWS API-Aufrufen automatisch aus der `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` Umgebungsvariablen ab.

Die Ausgabe enthält ein Zugriffsschlüsselpaar, das aus einer geheimen Zugriffsschlüssel-ID und einem geheimen Schlüssel besteht, das Ihre Anwendung für den Zugriff auf Ressourcen verwendet. AWS Es enthält auch ein Token, mit dem überprüft AWS wird, ob die Anmeldeinformationen gültig sind. Standardmäßig sind Anmeldeinformationen, die Aufgaben mithilfe von Aufgabenrollen zugewiesen wurden, sechs Stunden lang gültig. Danach werden sie automatisch vom Amazon-ECS-Container-Agenten rotiert.

Aufgabenausführungsrolle

Die Rolle „Aufgabenausführung“ wird verwendet, um dem Amazon ECS-Container-Agenten die Erlaubnis zu erteilen, bestimmte AWS API-Aktionen in Ihrem Namen aufzurufen. Wenn Sie beispielsweise verwenden AWS Fargate, benötigt Fargate eine IAM-Rolle, die es ermöglicht, Bilder aus Amazon ECR abzurufen und Protokolle in Logs zu schreiben. CloudWatch Eine IAM-Rolle ist auch erforderlich, wenn eine Aufgabe auf ein Geheimnis verweist, das in gespeichert ist AWS Secrets Manager, z. B. ein Image-Pull-Secret.

> **Note**
>
> Wenn Sie Images als authentifizierter Benutzer abrufen, ist es weniger wahrscheinlich, dass Sie von den Änderungen an den Pull-Rate-Limits von Docker Hub betroffen sind. Weitere Informationen finden Sie unter Private Registrierungsauthentifizierung für Container-Instances.
> Durch die Verwendung von Amazon ECR und Amazon ECR Public können Sie die von Docker auferlegten Beschränkungen umgehen. Wenn Sie Images von Amazon

ECR abrufen, trägt dies auch dazu bei, die Netzwerk-Pull-Zeiten zu verkürzen und Datenübertragungsänderungen zu reduzieren, wenn der Datenverkehr Ihre VPC verlässt.

Important

Wenn Sie Fargate verwenden, müssen Sie sich bei einer privaten Image-Registry mit `repositoryCredentials` authentifizieren. Es ist nicht möglich, die Umgebungsvariablen `ECS_ENGINE_AUTH_TYPE` oder `ECS_ENGINE_AUTH_DATA` für den Amazon-ECS-Container-Agenten festzulegen oder die `ecs.config`-Datei für auf Fargate gehostete Aufgaben zu ändern. Weitere Informationen finden Sie unter Private Registrierungsauthentifizierung für Aufgaben.

Container-Instance-Rolle von Amazon EC2

Der Amazon-ECS-Container-Agent ist ein Container, der auf jeder Amazon-EC2-Instance in einem Amazon-ECS-Cluster läuft. Sie wird außerhalb von Amazon ECS mithilfe des Befehls `init` initialisiert, der auf dem Betriebssystem verfügbar ist. Folglich können ihr keine Berechtigungen durch eine Aufgabenrolle erteilt werden. Stattdessen müssen die Berechtigungen den Amazon-EC2-Instances zugewiesen werden, auf denen die Agenten ausgeführt werden. Die Aktionsliste in der `AmazonEC2ContainerServiceforEC2Role`-Beispielrichtlinie muss der `ecsInstanceRole` erteilt werden. Wenn Sie dies nicht tun, können Ihre Instances dem Cluster nicht beitreten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeTags",
                "ecs:CreateCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:DiscoverPollEndpoint",
                "ecs:Poll",
                "ecs:RegisterContainerInstance",
                "ecs:StartTelemetrySession",
                "ecs:UpdateContainerInstancesState",
                "ecs:Submit*",
```

```
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

In dieser Richtlinie ermöglichen die `ecr` und `logs` API-Aktionen es den Containern, die auf Ihren Instances ausgeführt werden, Bilder von Amazon ECR abzurufen und Protokolle an Amazon CloudWatch zu schreiben. Die `ecs`-Aktionen ermöglichen es dem Agenten, Instances zu registrieren und deren Registrierung aufzuheben und mit der Amazon-ECS-Steuerebene zu kommunizieren. Von diesen ist die Aktion `ecs:CreateCluster` optional.

Service-verknüpfte Rollen

Sie können die serviceverknüpfte Rolle für Amazon ECS verwenden, um dem Amazon-ECS-Service die Berechtigung zu gewähren, andere Service-APIs in Ihrem Namen aufzurufen. Amazon ECS benötigt die Berechtigungen zum Erstellen und Löschen von Netzwerkschnittstellen sowie zum Registrieren und Abmelden von Zielen bei einer Zielgruppe. Außerdem benötigt es die erforderlichen Berechtigungen, um Skalierungsrichtlinien zu erstellen und zu löschen. Diese Berechtigungen werden durch die serviceverknüpfte Rolle erteilt. Diese Rolle wird in Ihrem Namen erstellt, wenn Sie den Service zum ersten Mal verwenden.

> **Note**
>
> Wenn Sie die serviceverknüpfte Rolle versehentlich löschen, können Sie sie neu erstellen. Anweisungen dazu finden Sie unter Die serviceverknüpfte Rolle erstellen.

Empfehlungen

Wir empfehlen Ihnen, bei der Einrichtung Ihrer IAM-Rollen und -Richtlinien für Aufgaben wie folgt vorzugehen.

Zugriff auf Amazon-EC2-Metadaten blockieren

Wenn Sie Ihre Aufgaben auf Amazon-EC2 Instances ausführen, empfehlen wir Ihnen dringend, den Zugriff auf Amazon-EC2-Metadaten zu blockieren, um zu verhindern, dass Ihre Container die diesen Instances zugewiesene Rolle erben. Wenn Ihre Anwendungen eine AWS API-Aktion aufrufen müssen, verwenden Sie stattdessen IAM-Rollen für Aufgaben.

Um zu verhindern, dass Aufgaben, die im Bridge-Modus ausgeführt werden, auf Amazon-EC2-Metadaten zugreifen, führen Sie den folgenden Befehl aus oder aktualisieren Sie die Benutzerdaten der Instance. Weitere Anweisungen zum Aktualisieren der Benutzerdaten einer Instance finden Sie in diesem AWS -Support-Artikel. Weitere Informationen zum Bridge-Modus für Aufgabendefinitionen finden Sie unter Netzwerkmodus für Aufgabendefinitionen.

```
sudo yum install -y iptables-services; sudo iptables --insert FORWARD 1 --in-interface
 docker+ --destination 192.0.2.0/32 --jump DROP
```

Damit diese Änderung nach einem Neustart bestehen bleibt, führen Sie den folgenden Befehl aus, der für Ihr Amazon Machine Image (AMI) spezifisch ist:

- Amazon Linux 2

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now
   iptables
  ```

- Amazon Linux

  ```
  sudo service iptables save
  ```

Für Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden, setzen Sie die Umgebungsvariable `ECS_AWSVPC_BLOCK_IMDS` in der `/etc/ecs/ecs.config`-Datei auf `true`.

Sie sollten die Variable `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` in der `ecs-agent config`-Datei auf `false` setzen, um zu verhindern, dass die Container, die im `host`-Netzwerk ausgeführt werden, auf die Amazon-EC2-Metadaten zugreifen.

Den **`awsvpc`**-Netzwerkmodus verwenden

Verwenden Sie den `awsvpc`-Netzwerkmodus, um den Verkehrsfluss zwischen verschiedenen Aufgaben oder zwischen Ihren Aufgaben und anderen Services, die in Ihrer Amazon VPC ausgeführt

werden, einzuschränken. Dies fügt eine zusätzliche Sicherheitsebene hinzu. Der `awsvpc`-Netzwerkmodus bietet Netzwerkisolierung auf Aufgabenebene für Aufgaben, die auf Amazon EC2 ausgeführt werden. Dies ist der Standardmodus aktiviert AWS Fargate. Es ist der einzige Netzwerkmodus, den Sie verwenden können, um Aufgaben eine Sicherheitsgruppe zuzuweisen.

IAM Access Advisor verwenden, um Rollen zu verfeinern

Wir empfehlen, alle Aktionen zu entfernen, die nie oder seit einiger Zeit nicht mehr verwendet wurden. Dadurch wird verhindert, dass unerwünschter Zugriff erfolgt. Überprüfen Sie dazu die von IAM Access Advisor erstellten Ergebnisse und entfernen Sie dann Aktionen, die nie oder in letzter Zeit nicht verwendet wurden. Gehen Sie dazu wie folgt vor.

Führen Sie den folgenden Befehl aus, um einen Bericht mit Informationen über den letzten Zugriff für die betreffende Richtlinie zu erstellen:

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Verwenden Sie die `JobId`, die in der Ausgabe enthalten war, um den folgenden Befehl auszuführen. Nachdem Sie dies getan haben, können Sie die Ergebnisse des Berichts einsehen.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Weitere Informationen finden Sie unter IAM Access Advisor.

Achten Sie AWS CloudTrail auf verdächtige Aktivitäten

Sie können AWS CloudTrail nach verdächtigen Aktivitäten Ausschau halten. Die meisten AWS API-Aufrufe werden AWS CloudTrail als Ereignisse protokolliert. Sie werden von AWS CloudTrail Insights analysiert, und Sie werden über jedes verdächtige Verhalten im Zusammenhang mit `write` API-Aufrufen informiert. Dies kann einen Anstieg des Anrufvolumens beinhalten. Diese Warnmeldungen enthalten Informationen wie den Zeitpunkt, zu dem die ungewöhnliche Aktivität aufgetreten ist, und den wichtigsten Identitäts-ARN, der zu den APIs beigetragen hat.

Sie können Aktionen identifizieren, die von Aufgaben mit einer IAM-Rolle in AWS CloudTrail ausgeführt werden, indem Sie sich die `userIdentity`-Eigenschaft des Ereignisses ansehen. Im folgenden Beispiel besteht das `arn` aus dem Namen der übernommenen Rolle, `s3-write-go-bucket-role`, gefolgt vom Namen der Aufgabe, `7e9894e088ad416eb5cab92afExample`.

```
"userIdentity": {
```

```
    "type": "AssumedRole",
    "principalId": "AROA36C6WWEJ2YEXAMPLE:7e9894e088ad416eb5cab92afExample",
    "arn": "arn:aws:sts::123456789012:assumed-role/s3-write-go-bucket-
role/7e9894e088ad416eb5cab92afExample",
    ...
}
```

> **Note**
>
> Wenn Aufgaben, die eine Rolle übernehmen, auf Amazon-EC2-Container-Instances ausgeführt werden, wird eine Anfrage vom Amazon ECS Container Agent im Audit-Protokoll des Agenten protokolliert, das sich an einer Adresse im `/var/log/ecs/audit.log.YYYY-MM-DD-HH`-Format befindet. Weitere Informationen finden Sie unter Aufgaben-IAM-Rollenprotokoll und Protokollieren von Insights-Ereignissen für Trails.

Netzwerksicherheit

Netzwerksicherheit ist ein breit gefächertes Thema, das mehrere Unterthemen umfasst. Dazu gehören encryption-in-transit Netzwerksegmentierung und -isolierung, Firewalling, Traffic-Routing und Beobachtbarkeit.

Verschlüsselung während der Übertragung

Die Verschlüsselung des Netzwerkverkehrs verhindert, dass unbefugte Benutzer Daten abfangen und lesen können, wenn diese Daten über ein Netzwerk übertragen werden. Mit Amazon ECS kann die Netzwerkverschlüsselung auf eine der folgenden Arten implementiert werden.

- Mit einem Service Mesh (TLS):

  Mit AWS App Mesh können Sie TLS-Verbindungen zwischen den Envoy-Proxys konfigurieren, die mit Mesh-Endpunkten bereitgestellt werden. Zwei Beispiele sind virtuelle Knoten und virtuelle Gateways. Die TLS-Zertifikate können von (ACM) stammen AWS Certificate Manager . Sie können auch von Ihrer eigenen privaten Zertifizierungsstelle stammen.

  - Transport Layer Security (TLS) aktivieren
  - Aktivieren Sie die Verschlüsselung des Datenverkehrs zwischen Diensten AWS App Mesh mithilfe von ACM-Zertifikaten oder vom Kunden bereitgestellten Zertifikaten
  - Schrittweise Anleitung für TLS ACM

  - Schrittweise Anleitung für TLS-Dateien
  - Envoy
- Verwenden von Nitro-Instances:

  Standardmäßig wird der Datenverkehr zwischen den folgenden Nitro-Instance-Typen automatisch verschlüsselt: C5n, G4, I3en, M5dn, M5n, P3dn, R5dn, und R5n. Der Datenverkehr wird nicht verschlüsselt, wenn er über ein Transit-Gateway, einen Load Balancer oder einen ähnlichen Vermittler geleitet wird.

  - Verschlüsselung während der Übertragung
  - Was ist die neue Ankündigung von 2019
  - Dieser Vortrag von Re:inForce 2019
- Server Name Indication (SNI) mit einem Application Load Balancer verwenden:

  Der Application Load Balancer (ALB) und der Network Load Balancer (NLB) unterstützen Server Name Indication (SNI). Mithilfe von SNI können Sie mehrere sichere Anwendungen hinter einem einzigen Listener platzieren. Hierfür hat jeder sein eigenes TLS-Zertifikat. Wir empfehlen, dass Sie Zertifikate für den Load Balancer mithilfe von AWS Certificate Manager (ACM) bereitstellen und sie dann zur Zertifikatsliste des Listeners hinzufügen. Der AWS Load Balancer verwendet einen intelligenten Algorithmus zur Zertifikatsauswahl mit SNI. Wenn der von einem Client bereitgestellte Hostname nur mit einem Zertifikat in der Zertifikatliste übereinstimmt, wählt der Load Balancer dieses Zertifikat aus. Wenn ein von einem Client bereitgestellter Hostname mit mehreren Zertifikaten in der Liste übereinstimmt, wählt der Load Balancer ein Zertifikat aus, das der Client unterstützen kann. Beispiele hierfür sind ein selbstsigniertes Zertifikat oder ein über das ACM generiertes Zertifikat.

  - SNI mit Application Load Balancer
  - SNI mit Network Load Balancer
- nd-to-end E-Verschlüsselung mit TLS-Zertifikaten:

  Dies beinhaltet die Bereitstellung eines TLS-Zertifikats für die Aufgabe. Dabei kann es sich entweder um ein selbstsigniertes Zertifikat oder um ein Zertifikat einer vertrauenswürdigen Zertifizierungsstelle handeln. Sie können das Zertifikat erhalten, indem Sie auf ein Secret für das Zertifikat verweisen. Andernfalls können Sie einen Container ausführen, der eine Certificate Signing Request (CSR) an ACM ausgibt und den resultierenden geheimen Schlüssel dann auf einem gemeinsam genutzten Volume mountet.

- Aufrechterhaltung der Sicherheit der Transportebene bis zu Ihren Containern unter Verwendung des Network Load Balancers mit Amazon ECS, Teil 1
- Aufrechterhaltung der Transport Layer Security (TLS) bis hin zu Ihrem Container Teil 2: Verwenden AWS Private Certificate Authority

Aufgabenvernetzung

Die folgenden Empfehlungen beziehen sich auf die Funktionsweise von Amazon ECS. Amazon ECS verwendet kein Overlay-Netzwerk. Stattdessen sind die Aufgaben so konfiguriert, dass sie in verschiedenen Netzwerkmodi arbeiten. Aufgaben, die für die Verwendung des `bridge`-Modus konfiguriert sind, erhalten beispielsweise eine nicht weiterleitbare IP-Adresse von einem Docker-Netzwerk, das auf jedem Host läuft. Aufgaben, die für die Verwendung des `awsvpc`-Netzwerkmodus konfiguriert sind, beziehen eine IP-Adresse aus dem Subnetz des Hosts. Aufgaben, die für `host`-Netzwerke konfiguriert sind, verwenden die Netzwerkschnittstelle des Hosts. `awsvpc` ist der bevorzugte Netzwerkmodus. Dies liegt daran, dass dies der einzige Modus ist, mit dem Sie Sicherheitsgruppen an Aufgaben zuweisen können. Es ist auch der einzige Modus, der für AWS Fargate Aufgaben auf Amazon ECS verfügbar ist.

Sicherheitsgruppen für Aufgaben

Es wird empfohlen, Ihre Aufgaben zur Verwendung des `awsvpc`-Netzwerkmodus zu konfigurieren. Nachdem Sie Ihre Aufgabe für die Verwendung dieses Modus konfiguriert haben, stellt der Amazon-ECS-Agent automatisch eine Elastic-Network-Schnittstelle (ENI) bereit und fügt sie der Aufgabe hinzu. Wenn die ENI bereitgestellt wird, wird die Aufgabe in eine AWS Sicherheitsgruppe aufgenommen. Die Sicherheitsgruppe dient als virtuelle Firewall, mit der Sie den ein- und ausgehenden Datenverkehr kontrollieren können.

Service Mesh und Mutual Transport Layer Security (mTLS)

Sie können ein Service Mesh verwenden, AWS App Mesh um beispielsweise den Netzwerkverkehr zu steuern. Standardmäßig kann ein virtueller Knoten nur mit seinen konfigurierten Service-Backends kommunizieren, z. B. mit den virtuellen Diensten, mit denen der virtuelle Knoten kommunizieren wird. Wenn ein virtueller Knoten mit einem Dienst außerhalb des Meshs kommunizieren muss, können Sie den `ALLOW_ALL` Ausgangsfilter verwenden oder einen virtuellen Knoten innerhalb des Meshs für den externen Dienst erstellen. Weitere Informationen finden Sie unter Anleitung zu Kubernetes Egress.

App Mesh bietet Ihnen auch die Möglichkeit, Mutual Transport Layer Security (mTLS) zu verwenden, bei der sowohl der Client als auch der Server mithilfe von Zertifikaten gegenseitig

authentifiziert werden. Die nachfolgende Kommunikation zwischen Client und Server wird dann mit TLS verschlüsselt. Indem Sie mTLS zwischen Diensten in einem Mesh vorschreiben, können Sie überprüfen, ob der Datenverkehr aus einer vertrauenswürdigen Quelle stammt. Weitere Informationen finden Sie unter den folgenden Themen:

- mTLS-Authentifizierung
- Exemplarische Vorgehensweise für den mTLS Secret Discovery Service (SDS)
- Exemplarische Vorgehensweise für die mTLS-Datei

AWS PrivateLink

AWS PrivateLink ist eine Netzwerktechnologie, mit der Sie private Endpunkte für verschiedene AWS Dienste, einschließlich Amazon ECS, einrichten können. Die Endpunkte sind in Sandbox-Umgebungen erforderlich, in denen kein Internet-Gateway (IGW) an die Amazon VPC angeschlossen ist und keine alternativen Routen zum Internet bestehen. Durch die Verwendung AWS PrivateLink wird sichergestellt, dass Aufrufe an den Amazon ECS-Service innerhalb der Amazon VPC bleiben und nicht das Internet durchqueren. Anweisungen zum Erstellen von AWS PrivateLink Endpunkten für Amazon ECS und andere verwandte Services finden Sie unter Amazon ECS-Schnittstelle Amazon VPC-Endpoints.

> **Important**
>
> AWS Fargate Aufgaben erfordern keinen AWS PrivateLink Endpunkt für Amazon ECS.

Sowohl Amazon ECR als auch Amazon ECS unterstützen Endpunktrichtlinien. Mit diesen Richtlinien können Sie den Zugriff auf die APIs eines Services verfeinern. Sie könnten beispielsweise eine Endpunktrichtlinie für Amazon ECR erstellen, nach der nur Images an Registrys bestimmter AWS -Konten gesendet werden können. Eine Richtlinie wie diese könnte verwendet werden, um zu verhindern, dass Daten durch Container-Images exfiltriert werden, während Benutzer weiterhin die Möglichkeit haben, an autorisierte Amazon-ECR-Registrys zu senden. Weitere Informationen finden Sie unter VPC-Endpunkt-Richtlinien verwenden.

Die folgende Richtlinie ermöglicht es allen AWS Principals in Ihrem Konto, alle Aktionen nur für Ihre Amazon ECR-Repositorys durchzuführen:

```
{
```

```
  "Statement": [
    {
      "Sid": "LimitECRAccess",
      "Principal": "*",
      "Action": "*",
      "Effect": "Allow",
      "Resource": "arn:aws:ecr:region:account_id:repository/*"
    },
  ]
}
```

Sie können dies weiter verbessern, indem Sie eine Bedingung festlegen, die die neue `PrincipalOrgID`-Eigenschaft verwendet. Dadurch wird verhindert, dass ein IAM-Principal, der nicht Teil Ihres IAM-Principals ist, das Push und Pull von Bildern durchführt. AWS Organizations Weitere Informationen finden Sie unter aws: PrincipalOrg ID.

Wir haben empfohlen, dieselbe Richtlinie sowohl auf die `com.amazonaws.region.ecr.dkr`- als auch die `com.amazonaws.region.ecr.api`-Endpunkte anzuwenden.

Einstellungen des Amazon-ECS-Container-Agenten

Die Konfigurationsdatei des Amazon ECS Container-Agenten enthält mehrere Umgebungsvariablen, die sich auf die Netzwerksicherheit beziehen. `ECS_AWSVPC_BLOCK_IMDS` und `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` werden verwendet, um den Zugriff einer Aufgabe zu Amazon EC2-Metadaten zu blockieren. `HTTP_PROXY` wird verwendet, um den Agenten so zu konfigurieren, dass er über einen HTTP-Proxy eine Verbindung zum Internet herstellt. Anweisungen zur Konfiguration des Agenten und der Docker-Laufzeit für die Weiterleitung über einen Proxy finden Sie unter HTTP-Proxykonfiguration.

> **Important**
>
> Diese Einstellungen sind nicht verfügbar, wenn Sie sie AWS Fargate verwenden.

Empfehlungen

Wir empfehlen Ihnen, bei der Einrichtung Ihrer Amazon VPC, Load Balancer und Ihres Netzwerks wie folgt vorzugehen.

Gegebenenfalls Netzwerkverschlüsselung verwenden

Sie sollten gegebenenfalls Netzwerkverschlüsselung verwenden. Bestimmte Compliance-Programme, wie PCI DSS, verlangen, dass Sie Daten während der Übertragung verschlüsseln, wenn die Daten Karteninhaberdaten enthalten. Wenn Ihr Workload ähnliche Anforderungen hat, konfigurieren Sie die Netzwerkverschlüsselung.

Moderne Browser warnen Benutzer, wenn sie eine Verbindung zu unsicheren Websites herstellen. Wenn Ihr Service von einem öffentlich zugänglichen Load Balancer unterstützt wird, verwenden Sie TLS/SSL, um den Datenverkehr vom Browser des Clients zum Load Balancer zu verschlüsseln, und verschlüsseln Sie ihn gegebenenfalls erneut zum Backend.

Verwenden Sie den **`awsvpc`**-Netzwerkmodus und Sicherheitsgruppen, wenn Sie den Verkehr zwischen Aufgaben oder solchen zwischen Aufgaben und anderen Netzwerkressourcen kontrollieren müssen

Sie sollten den `awsvpc`-Netzwerkmodus und Sicherheitsgruppen verwenden, wenn Sie den Verkehr zwischen Aufgaben sowie solchen zwischen Aufgaben und anderen Netzwerkressourcen kontrollieren müssen. Wenn sich Ihr Service hinter einem ALB befindet, verwenden Sie Sicherheitsgruppen, um nur eingehenden Datenverkehr von anderen Netzwerkressourcen zuzulassen, die die gleiche Sicherheitsgruppe wie Ihr ALB verwenden. Wenn sich Ihre Anwendung hinter einem NLB befindet, konfigurieren Sie die Sicherheitsgruppe der Aufgabe so, dass nur eingehender Datenverkehr aus dem Amazon-VPC-CIDR-Bereich und den statischen IP-Adressen, die dem NLB zugewiesen sind, zugelassen wird.

Sicherheitsgruppen sollten auch verwendet werden, um den Datenverkehr zwischen Aufgaben und anderen Ressourcen innerhalb der Amazon VPC wie beispielsweise Amazon-RDS-Datenbanken zu kontrollieren.

Erstellen von Clustern in separaten Amazon VPCs, wenn der Netzwerkverkehr strikt isoliert werden muss

Sie sollten Cluster in separaten Amazon VPCs erstellen, wenn der Netzwerkverkehr strikt isoliert werden muss. Vermeiden Sie es, Workloads mit strengen Sicherheitsanforderungen auf Clustern mit Workloads auszuführen, die diese Anforderungen nicht erfüllen müssen. Wenn eine strikte Netzwerkisolierung erforderlich ist, erstellen Sie Cluster in separaten Amazon VPCs und stellen Sie Services mithilfe von Amazon-VPC-Endpunkten selektiv anderen Amazon VPCs zur Verfügung. Weitere Informationen finden Sie unter Amazon-VPC-Endpunkte.

AWS PrivateLink -Endpunkten konfigurieren, wenn nötig

Sie sollten Endgeräte und AWS PrivateLink Endpunkte konfigurieren, sofern dies gerechtfertigt ist. Wenn Ihre Sicherheitsrichtlinie Sie daran hindert, ein Internet Gateway (IGW) an Ihre Amazon VPCs anzuhängen, konfigurieren Sie AWS PrivateLink Endpunkte für Amazon ECS und andere Dienste wie Amazon ECR, und Amazon. AWS Secrets Manager CloudWatch

Amazon-VPC-Flow-Protokolle verwenden, um den Verkehr zu und von lang andauernden Aufgaben zu analysieren

Sie sollten Amazon-VPC-Flow-Protokolle verwenden, um den Verkehr zu und von lang andauernden Aufgaben zu analysieren. Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden, erhalten ihre eigene ENI. Auf diese Weise können Sie mithilfe von Amazon-VPC-Flow-Protokollen den Verkehr überwachen, der zu und von einzelnen Aufgaben geht. Ein kürzlich veröffentlichtes Update von Amazon-VPC-Flow-Protokolle (v3) bereichert die Protokolle mit Verkehrsmetadaten wie der VPC-ID, der Subnetz-ID und der Instance-ID. Diese Metadaten können verwendet werden, um eine Untersuchung einzugrenzen. Weitere Informationen finden Sie unter Amazon-VPC-Flow-Protokolle.

> **Note**
>
> Aufgrund des temporären Charakters von Containern sind Flow-Protokolle möglicherweise nicht immer eine effektive Methode, um Verkehrsmuster zwischen verschiedenen Containern oder Containern und anderen Netzwerkressourcen zu analysieren.

Verwaltung von Secrets

Secrets, wie API-Schlüssel und Datenbankanmeldeinformationen, werden häufig von Anwendungen verwendet, um auf andere Systeme zuzugreifen. Sie bestehen häufig aus einem Benutzernamen und einem Passwort, einem Zertifikat oder einem API-Schlüssel. Der Zugriff auf diese Secrets sollte auf bestimmte IAM-Prinzipale beschränkt werden, die IAM verwenden und zur Laufzeit in Container eingespeist werden.

Secrets können nahtlos aus AWS Secrets Manager einem Amazon EC2 Systems Manager Parameter Store in Container eingefügt werden. Auf diese Secrets kann in Ihrer Aufgabe wie folgt verwiesen werden.

1. Sie werden als Umgebungsvariablen referenziert, die den `secrets`-Container-Definitionsparameter verwenden.

2. Sie werden als `secretOptions` bezeichnet, wenn Ihre Protokollierungsplattform eine Authentifizierung erfordert. Weitere Informationen finden Sie unter Konfigurationsoptionen für die Protokollierung.
3. Sie werden als Secrets bezeichnet, die von Images abgerufen werden, die den `repositoryCredentials`-Container-Definitionsparameter verwenden, wenn die Registrierung, aus der der Container abgerufen wird, eine Authentifizierung erfordert. Verwenden Sie diese Methode, wenn Sie Images aus Amazon ECR Public Gallery abrufen. Weitere Informationen finden Sie unter Private Registrierungsauthentifizierung für Aufgaben.

Empfehlungen

Wir empfehlen Ihnen, bei der Einrichtung der Verwaltung von Secrets wie folgt vorzugehen.

Verwenden Sie AWS Secrets Manager unseren Amazon EC2 Systems Manager Parameter Store zum Speichern geheimer Materialien

Sie sollten API-Schlüssel, Datenbankanmeldedaten und andere geheime Materialien sicher in AWS Secrets Manager oder als verschlüsselte Parameter im Amazon EC2 Systems Manager Parameter Store speichern. Diese Dienste ähneln sich, da es sich bei beiden um verwaltete Schlüsselwertspeicher handelt, die AWS KMS zur Verschlüsselung sensibler Daten verwendet werden. AWS Secrets Manager beinhaltet jedoch auch die Möglichkeit, Geheimnisse automatisch zu rotieren, zufällige Geheimnisse zu generieren und Geheimnisse zwischen Konten auszutauschen. AWS Wenn Sie diese Features als wichtig einstufen, verwenden Sie AWS Secrets Manager , ansonsten verwenden Sie verschlüsselte Parameter.

> **Note**
>
> Aufgaben, die auf ein Geheimnis aus AWS Secrets Manager oder dem Amazon EC2 Systems Manager Parameter Store verweisen, erfordern eine Aufgabenausführungsrolle mit einer Richtlinie, die Amazon ECS Zugriff auf das gewünschte Geheimnis und, falls zutreffend, auf den AWS KMS Schlüssel gewährt, der zum Verschlüsseln und Entschlüsseln dieses Geheimnisses verwendet wird.

> **Important**
>
> Secrets, auf die in Aufgaben verwiesen wird, werden nicht automatisch rotiert. Wenn sich Ihr Secret ändert, müssen Sie eine neue Bereitstellung erzwingen oder eine neue Aufgabe starten, um den neuesten Secret-Wert abzurufen. Weitere Informationen finden Sie unter den folgenden Themen:
>
> - AWS Secrets Manager: Daten als Umgebungsvariablen einfügen
> - Amazon EC2 Systems Manager Parameter Store: Daten als Umgebungsvariablen einfügen

Daten aus einem verschlüsselten Amazon-S3-Bucket abrufen

Da der Wert von Umgebungsvariablen versehentlich in Protokollen nach außen dringen kann und bei der Ausführung von `docker inspect` aufgedeckt wird, sollten Sie Secrets in einem verschlüsselten Amazon-S3-Bucket speichern und Aufgabenrollen verwenden, um den Zugriff auf diese Secrets zu beschränken. Wenn Sie dies tun, muss Ihre Anwendung so geschrieben werden, dass sie das Secret aus dem Amazon-S3-Bucket liest. Anweisungen dazu finden Sie unter Festlegen des standardmäßigen serverseitigen Verschlüsselungsverhaltens für Amazon-S3-Buckets.

Das Secret mit Hilfe eines Beiwagen-Containers in ein Volume mounten

Da bei Umgebungsvariablen ein erhöhtes Risiko von Datenlecks besteht, sollten Sie einen Sidecar-Container verwenden, der Ihre Secrets aus einem gemeinsam genutzten Volume liest AWS Secrets Manager und darauf schreibt. Dieser Container kann vor dem Anwendungscontainer ausgeführt und beendet werden, indem Sie Amazon-ECS-Container-Anordnungen verwenden. Wenn Sie dies tun, mountet der Anwendungscontainer anschließend das Volume, auf dem das Secret geschrieben wurde. Wie bei der Amazon-S3-Bucket-Methode muss Ihre Anwendung so geschrieben werden, dass sie das Secret aus dem gemeinsam genutzten Volume liest. Da das Volume auf die Aufgabe beschränkt ist, wird das Volume nach dem Beenden der Aufgabe automatisch gelöscht. Ein Beispiel für einen Sidecar-Container finden Sie im Projekt. aws-secret-sidecar-injector

> **Note**
>
> In Amazon EC2 kann das Volume, auf welches das Secret geschrieben wird, mit einem vom Kunden verwalteten AWS KMS -Schlüssel verschlüsselt werden. Ist diese Option aktiviert AWS Fargate, wird der Datenträgerspeicher automatisch mithilfe eines vom Service verwalteten Schlüssels verschlüsselt.

Weitere Ressourcen

- Weitergabe von Secrets an Container in einer Amazon-ECS-Aufgabe
- Chamber ist ein Wrapper zum Speichern von Secrets im Amazon EC2 Systems Manager Parameter Store

Verwenden von temporären Sicherheitsanmeldeinformationen mit API-Operationen

Wenn Sie direkte HTTPS-API-Anfragen an stellen AWS, können Sie diese Anfragen mit den temporären Sicherheitsanmeldedaten signieren, die Sie von der erhalten AWS Security Token Service. Weitere Informationen finden Sie unter Signieren von AWS-API-Anfragen in der Allgemeine AWS-Referenz.

Compliance und Sicherheit

Ihre Compliance-Verantwortung bei Verwendung von Amazon ECS hängt von der Vertraulichkeit der Daten, den Compliance-Zielen des Unternehmens und den geltenden Gesetzen und Vorschriften ab.

AWS bietet die folgenden Ressourcen zur Unterstützung bei der Einhaltung von Vorschriften:

- Schnellstartanleitungen zu Sicherheit und Compliance: In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Implementierung von sicherheits- und Compliance-orientierten Basisumgebungen beschrieben. AWS
- Whitepaper „Architecting for HIPAA Security and Compliance“: In diesem Whitepaper wird beschrieben, wie Unternehmen HIPAA-konforme Anwendungen entwickeln können. AWS
- AWS -Services im Umfang des Compliance Programms: Diese Liste enthält die AWS -Services, die zum Umfang bestimmter Compliance-Programme gehören. Weitere Informationen finden Sie unter AWS -Compliance-Programme.

Payment Card Industry Data Security Standards (PCI DSS)

Es ist wichtig, dass Sie den gesamten Fluss von Karteninhaberdaten (CHD) innerhalb der Umgebung verstehen, wenn Sie sich an PCI DSS halten. Der CHD-Ablauf bestimmt die Anwendbarkeit des PCI DSS, definiert die Grenzen und Komponenten einer Karteninhaberdatenumgebung (CDE) und damit den Umfang einer PCI-DSS-Bewertung. Die genaue Festlegung des PCI-DSS-

Umfangs ist entscheidend für die Festlegung der Sicherheitslage und letztlich für eine erfolgreiche Bewertung. Kunden müssen über ein Verfahren zur Festlegung des Umfangs verfügen, das dessen Vollständigkeit sicherstellt und Änderungen oder Abweichungen im Umfang erkennt.

Der temporäre Charakter containerisierter Anwendungen sorgt für zusätzliche Komplexität bei der Prüfung von Konfigurationen. Aus diesem Grund müssen Kunden stets über alle Parameter der Container-Konfiguration informiert sein, um sicherzustellen, dass die Compliance-Anforderungen in allen Phasen des Container-Lebenszyklus erfüllt werden.

Weitere Informationen zum Erreichen der PCI-DSS-Konformität mit Amazon ECS finden Sie in den folgenden Whitepapers.

- Architektur auf Amazon ECS für PCI-DSS-Konformität
- Architektur für PCI-DSS-Skalierung und -Segmentierung auf AWS

HIPAA (U.S. Health Insurance Portability and Accountability Act)

Die Verwendung von Amazon ECS mit Workloads, die geschützte Gesundheitsinformationen (PHI) verarbeiten, erfordert keine zusätzliche Konfiguration. Amazon ECS fungiert als Orchestrierungsservice, der den Start von Containern auf Amazon EC2 koordiniert. Er arbeitet nicht mit oder auf Daten innerhalb des zu orchestrierenden Workloads. Gemäß den HIPAA-Vorschriften und dem AWS Business Associate Addendum sollte PHI während der Übertragung und im Ruhezustand verschlüsselt werden, wenn Container, die mit Amazon ECS gestartet wurden, darauf zugreifen.

Für jede AWS Speicheroption sind verschiedene Mechanismen zur Verschlüsselung im Ruhezustand verfügbar, z. B. Amazon S3, Amazon EBS und. AWS KMS Sie können ein Overlay-Netzwerk (wie VNS3 oder Weave Net) einsetzen, um eine vollständige Verschlüsselung der zwischen Containern übertragenen PHI-Daten sicherzustellen oder um eine redundante Verschlüsselungsebene bereitzustellen. Die vollständige Protokollierung sollte ebenfalls aktiviert sein und alle Container-Protokolle sollten an Amazon weitergeleitet werden CloudWatch. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

AWS Security Hub

Wird verwendet AWS Security Hub , um Ihre Nutzung von Amazon ECS in Bezug auf bewährte Sicherheitsmethoden zu überwachen. Security Hub verwendet Kontrollen für die Bewertung von

Ressourcenkonfigurationen und Sicherheitsstandards, um Sie bei der Einhaltung verschiedener Compliance-Frameworks zu unterstützen. Weitere Informationen zur Verwendung von Security Hub zur Bewertung von Amazon-ECS-Ressourcen finden Sie unter Amazon-ECS-Steuerelemente im AWS Security Hub -Benutzerhandbuch.

Empfehlungen

Sie sollten die Eigentümer der Compliance-Programme in Ihrem Unternehmen frühzeitig einbinden und das AWS -Modell der geteilten Verantwortung nutzen, um die Verantwortlichkeiten für die Compliance-Kontrolle festzulegen, damit die entsprechenden Compliance-Programme erfolgreich durchgeführt werden können.

Protokollierung und Überwachung

Protokollierung und Überwachung sind ein wichtiger Aspekt zur Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon ECS und Ihren AWS Lösungen. AWS bietet verschiedene Tools zur Überwachung Ihrer Amazon ECS-Ressourcen und zur Reaktion auf potenzielle Vorfälle:

- CloudWatchAmazon-Alarme
- CloudWatch Amazon-Protokolle
- CloudWatch Amazon-Veranstaltungen
- AWS CloudTrail Protokolle

Sie können die Container in Ihren Aufgaben so konfigurieren, dass Protokollinformationen an Amazon CloudWatch Logs gesendet werden. Wenn Sie den AWS Fargate Starttyp für Ihre Aufgaben verwenden, können Sie die Protokolle Ihrer Container einsehen. Wenn Sie den Amazon-EC2-Starttyp verwenden, können Sie verschiedene Protokolle Ihrer Container bequem an einem Ort aufrufen. Dies verhindert auch, dass Ihre Container-Protokolle Speicherplatz auf Ihren Container-Instances belegen.

Weitere Informationen zu Amazon CloudWatch Logs finden Sie unter Überwachen von Protokollen von Amazon EC2 EC2-Instances im CloudWatch Amazon-Benutzerhandbuch. Anweisungen zum Senden von Container-Protokollen von Ihren Aufgaben an Amazon CloudWatch Logs finden Sie unter Den `awslogs` Protokolltreiber verwenden.

Container-Protokollierung mit Fluent Bit

AWS bietet ein Fluent Bit Bild mit Plugins für Amazon CloudWatch Logs und Amazon Data Firehose. Dieses Image bietet die Möglichkeit, Protokolle an Amazon CloudWatch - und Amazon Data Firehose-Ziele weiterzuleiten (zu denen Amazon S3, Amazon OpenSearch Service und Amazon Redshift gehören). Es wird empfohlen, Fluent Bit als Protokoll-Router zu verwenden, da es eine geringere Ressourcenauslastung aufweist als Fluentd. Weitere Informationen finden Sie unter Amazon CloudWatch Logs for Fluent Bit und Amazon Data Firehose for Fluent Bit.

Das AWS Fluent Bit For-Bild ist verfügbar auf:

- Amazon ECR in Amazon ECR Public Gallery
- Amazon-ECR-Repository (in den meisten Regionen mit hoher Verfügbarkeit)

Nachfolgend finden Sie die für die Docker-CLI zu verwendende Syntax.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Mit diesem Docker-CLI-Befehl können Sie beispielsweise das neueste AWS Fluent Bit For-Image abrufen:

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:latest
```

Weitere Informationen zu Fluent Bit und verwandten Features finden Sie auch in den folgenden Blogbeiträgen:

- Fluent Bitfür Amazon EKS auf AWS Fargate
- Zentralisierte Container-Protokollierung mit Fluent Bit
- Aufbau eines skalierbaren Aggregators für Protokolllösungen mit AWS Fargate Fluentd und Amazon Data Firehose

Benutzerdefiniertes Protokoll-Routing — FireLens für Amazon ECS

Mit FireLens for Amazon ECS können Sie Aufgabendefinitionsparameter verwenden, um Protokolle zur Protokollspeicherung und Analyse an ein AWS Service- oder AWS Partnernetzwerkziel (APN) weiterzuleiten. FireLens funktioniert mit Fluentd und. Fluent Bit Wir stellen das AWS Bild

zur Verfügung. Fluent Bit Oder Sie können alternativ Ihr eigenes Fluentd- oder Fluent Bit-Image verwenden.

Bei der Verwendung FireLens für Amazon ECS sollten Sie die folgenden Bedingungen und Überlegungen berücksichtigen:

- FireLens für Amazon ECS wird für Aufgaben unterstützt, die sowohl auf Amazon EC2 als auch auf AWS Fargate Amazon EC2 gehostet werden.
- FireLens für Amazon ECS wird in AWS CloudFormation Vorlagen unterstützt. Weitere Informationen finden Sie AWS::ECS::TaskDefinition FirelensConfigurationim AWS CloudFormation Benutzerhandbuch.
- Bei Aufgaben, die den `bridge` Netzwerkmodus verwenden, müssen Container mit der FireLens Konfiguration gestartet werden, bevor die Anwendungscontainer, die darauf angewiesen sind, gestartet werden. Um die Reihenfolge zu bestimmen, in der Ihre Container gestartet werden, verwenden Sie in Ihrer Aufgabendefinition Abhängigkeitsbedingungen. Weitere Informationen finden Sie unter Container-Abhängigkeit.

AWS Fargate Sicherheit

Wir empfehlen, dass Sie die folgenden bewährten Methoden berücksichtigen, wenn Sie AWS Fargate verwenden. Weitere Hinweise finden Sie unter Sicherheitsüberblick von AWS Fargate.

Wird verwendet AWS KMS , um kurzlebigen Speicher zu verschlüsseln

Sie sollten Ihren kurzlebigen Speicher mit verschlüsseln lassen. AWS KMS Für Amazon ECS-Aufgaben, die auf der AWS Fargate folgenden Plattformversion `1.4.0` oder höher gehostet werden, erhält jede Aufgabe 20 GiB flüchtigen Speicher. Sie können die Gesamtmenge des flüchtigen Speichers bis zu einem Maximum von 200 GiB erhöhen, indem Sie den `ephemeralStorage`-Parameter in Ihrer Aufgabendefinition angeben. Für Aufgaben, die am 28. Mai 2020 oder später gestartet wurden, wird der flüchtige Speicher mit einem AES-256-Verschlüsselungsalgorithmus unter Verwendung eines von AWS Fargate verwalteten Schlüssels verschlüsselt.

Weitere Informationen finden Sie unter Verwendung von Daten-Volumes in Aufgaben.

Beispiel: Starten einer Amazon ECS-Aufgabe auf der AWS Fargate Plattformversion 1.4.0 mit kurzlebiger Speicherverschlüsselung

Mit dem folgenden Befehl wird eine Amazon ECS-Aufgabe auf der AWS Fargate Plattformversion 1.4 gestartet. Da diese Aufgabe als Teil des Amazon-ECS-Clusters gestartet wird, verwendet sie den flüchtigen Speicher von 20 GiB, der automatisch verschlüsselt wird.

```
aws ecs run-task --cluster clustername \
  --task-definition taskdefinition:version \
  --count 1
  --launch-type "FARGATE" \
  --platform-version 1.4.0 \
  --network-configuration
 "awsvpcConfiguration={subnets=[subnetid],securityGroups=[securitygroupid]}" \
  --region region
```

SYS_PTRACE-Funktion für die Ablaufverfolgung von syscall im Kernel

Die Standardkonfiguration der Linux-Funktionen, die Ihrem Container hinzugefügt oder entfernt werden, wird von Docker bereitgestellt. Weitere Informationen zu den verfügbaren Funktionen finden Sie unter Laufzeitprivileg und Linux-Funktionen in der Docker-Run-Dokumentation.

Aufgaben, die am gestartet werden, unterstützen AWS Fargate nur das Hinzufügen der `SYS_PTRACE` Kernel-Fähigkeit.

Sehen Sie sich das folgende Tutorial-Video an, das zeigt, wie Sie dieses Feature im Sysdig-Falco-Projekt verwenden können.

ContainersFromTheCouch — Problembehandlung bei Ihrer AWS Fargate Aufgabe mithilfe der SYS_PTRACE-Funktion

Den im vorherigen Video besprochenen Code finden Sie hier. GitHub

AWS Fargate Überlegungen zur Sicherheit

Jede Aufgabe hat eine eigene Infrastrukturkapazität, da Fargate jeden Workload in einer isolierten virtuellen Umgebung ausführt. Workloads, die auf Fargate ausgeführt werden, teilen sich keine Netzwerkschnittstellen, flüchtigen Speicher, CPU oder Arbeitsspeicher mit anderen Aufgaben. Sie können mehrere Container innerhalb einer Aufgabe ausführen, darunter Anwendungs-Container und Beiwagen-Container oder einfach Beiwagen. Ein Beiwagen ist ein Container, der zusammen mit einem Anwendungs-Container in einer Amazon-ECS-Aufgabe ausgeführt wird. Während der Anwendungs-Container den Kernanwendungscode ausführt, können Prozesse,

die in Beiwagen ausgeführt werden, die Anwendung erweitern. Mithilfe von Beiwagen können Sie Anwendungsfunktionen in spezielle Container unterteilen, sodass Sie Teile Ihrer Anwendung einfacher aktualisieren können.

Container, die Teil derselben Aufgabe sind, teilen sich Ressourcen für den Fargate-Starttyp, da diese Container immer auf demselben Host laufen und Rechenressourcen gemeinsam nutzen. Diese Container teilen sich auch den von Fargate bereitgestellten flüchtigen Speicher. Linux-Container in einer Aufgabe teilen sich Netzwerk-Namespaces, einschließlich der IP-Adresse und der Netzwerkports. Innerhalb einer Aufgabe können Container, die zur Aufgabe gehören, über localhost miteinander kommunizieren.

Die Laufzeitumgebung in Fargate verhindert, dass Sie bestimmte Controller-Features verwenden, die auf EC2-Instances unterstützt werden. Berücksichtigen Sie Folgendes, wenn Sie Workloads erstellen, die auf Fargate ausgeführt werden:

- Keine privilegierten Container oder privilegierter Zugriff – Features wie privilegierte Container oder privilegierter Zugriff sind derzeit auf Fargate nicht verfügbar. Dies wird sich auf Anwendungsfälle wie die Ausführung von Docker in Docker auswirken.
- Eingeschränkter Zugriff auf Linux-Funktionen – Die Umgebung, in der Container auf Fargate laufen, ist gesperrt. Zusätzliche Linux-Funktionen, wie CAP_SYS_ADMIN und CAP_NET_ADMIN, sind eingeschränkt, um eine Rechteerweiterung zu verhindern. Fargate unterstützt das Hinzufügen der Linux-Funktion CAP_SYS_PTRACE zu Aufgaben, um innerhalb der Aufgabe bereitgestellte Beobachtbarkeits- und Sicherheitstools zur Überwachung der containerisierten Anwendung zu ermöglichen.
- Kein Zugriff auf den zugrunde liegenden Host — Weder Kunden noch AWS Betreiber können eine Verbindung zu einem Host herstellen, auf dem Kunden-Workloads ausgeführt werden. Sie können ECS Exec verwenden, um Befehle in einem auf Fargate laufenden Container auszuführen oder eine Shell für diesen zu erhalten. Sie können ECS Exec verwenden, um Diagnoseinformationen für das Debuggen zu sammeln. Fargate verhindert auch, dass Container auf die Ressourcen des zugrunde liegenden Hosts zugreifen, z. B. auf das Dateisystem, die Geräte, das Netzwerk und die Container-Laufzeit.
- Netzwerke – Sie können Sicherheitsgruppen und Netzwerk-ACLs verwenden, um den ein- und ausgehenden Datenverkehr zu steuern. Fargate-Aufgaben erhalten eine IP-Adresse aus dem konfigurierten Subnetz in Ihrer VPC.

Aufgaben- und Containersicherheit

Sie sollten das Container-Image als erste Verteidigungslinie gegen einen Angriff betrachten. Ein unsicheres, schlecht aufgebautes Image kann es einem Angreifer ermöglichen, die Grenzen des Containers zu umgehen und sich Zugriff auf den Host zu verschaffen. Sie sollten wie folgt vorgehen, um das Risiko eines solchen Vorfalls zu verringern.

Empfehlungen

Wir empfehlen Folgendes, wenn Sie Ihre Aufgaben und Container einrichten.

Minimale Images erstellen oder distroless-Images verwenden

Entfernen Sie zunächst alle überflüssigen Binärdateien aus dem Container-Image. Wenn Sie ein unbekanntes Image aus Amazon ECR Public Gallery verwenden, überprüfen Sie das Image, um auf den Inhalt der einzelnen Ebenen des Containers hinzuweisen. Sie können dafür eine Anwendung wie Dive verwenden.

Alternativ können Sie Images distroless verwenden, die nur Ihre Anwendung und ihre Laufzeitabhängigkeiten enthalten. Sie enthalten keine Paketmanager oder Shells. Distroless Images verbessern das „Signal-Rausch-Verhältnis von Scannern und reduzieren den Aufwand für den Nachweis der Herkunft auf das, was Sie brauchen.“ Weitere Informationen finden Sie in der GitHub Dokumentation zu distroless.

Docker verfügt über einen Mechanismus zum Erstellen von Images aus einem reservierten, minimalen Image, das als Scratch bezeichnet wird. Weitere Informationen finden Sie in der Docker-Dokumentation unter Erstellen eines einfachen übergeordneten Images mithilfe von Scratch. Mit Sprachen wie Go können Sie eine statische verknüpfte Binärdatei erstellen und in Ihrem Dockerfile darauf verweisen. Das folgende Beispiel zeigt, wie Sie dies erreichen können.

```
############################
# STEP 1 build executable binary
############################
FROM golang:alpine AS builder
# Install git.
# Git is required for fetching the dependencies.
RUN apk update && apk add --no-cache git
WORKDIR $GOPATH/src/mypackage/myapp/
COPY . .
# Fetch dependencies.
# Using go get.
```

```
RUN go get -d -v
# Build the binary.
RUN go build -o /go/bin/hello
############################
# STEP 2 build a small image
############################
FROM scratch
# Copy our static executable.
COPY --from=builder /go/bin/hello /go/bin/hello
# Run the hello binary.
ENTRYPOINT ["/go/bin/hello"]
This creates a container image that consists of your application and nothing else,
 making it extremely secure.
```

Das vorherige Beispiel ist auch ein Beispiel für einen mehrstufigen Build. Diese Art von Builds ist vom Standpunkt der Sicherheit aus attraktiv, da Sie damit die Größe des endgültigen Images, das in Ihre Container-Registrierung übertragen wird, minimieren können. Container-Images ohne Build-Tools und andere überflüssige Binärdateien verbessern Ihre Sicherheitslage, indem sie die Angriffsfläche des Images reduzieren. Weitere Informationen zu mehrstufigen Builds finden Sie unter Erstellen mehrstufiger Builds.

Ihre Images auf Schwachstellen scannen

Ähnlich wie ihre Pendants in virtuellen Maschinen können Container-Images Binärdateien und Anwendungsbibliotheken mit Schwachstellen enthalten oder mit der Zeit Schwachstellen entwickeln. Am besten schützen Sie sich vor Angriffen, indem Sie Ihre Images regelmäßig mit einem Image-Scanner überprüfen.

Images, die in Amazon ECR gespeichert sind, können per Push oder auf Abruf (einmal alle 24 Stunden) gescannt werden. Amazon ECR Basic Scanning verwendet Clair, eine Open-Source-Lösung zum Scannen von Images. Das erweiterte Scannen mit Amazon ECR verwendet Amazon Inspector. Nachdem ein Bild gescannt wurde, werden die Ergebnisse im Amazon ECR-Event-Stream in Amazon EventBridge protokolliert. Sie können die Ergebnisse eines Scans auch in der Amazon ECR-Konsole oder durch Aufrufen der DescribeImageScanFindingsAPI anzeigen. Images mit einer `HIGH`- oder `CRITICAL`-Schwachstelle sollten gelöscht oder neu erstellt werden. Wenn ein bereitgestelltes Image eine Schwachstelle aufweist, sollte es so schnell wie möglich ersetzt werden.

Docker Desktop Edge Version 2.3.6.0 oder höher kann lokale Images scannen. Die Scans werden von Snyk, einem Anwendungssicherheitsservice, unterstützt. Wenn Schwachstellen entdeckt werden, identifiziert Snyk die Ebenen und Abhängigkeiten mit der Sicherheitslücke im Dockerfile. Es

empfiehlt auch sichere Alternativen wie die Verwendung eines schlankeren Basis-Images mit weniger Schwachstellen oder die Aktualisierung eines bestimmten Pakets auf eine neuere Version. Mithilfe von Docker-Scan können Entwickler potenzielle Sicherheitsprobleme lösen, bevor sie ihre Images in die Registrierung übertragen.

- Automatisieren der Image-Compliance mithilfe von Amazon ECR und AWS Security Hub erklärt, wie Informationen zu Oberflächen-Schwachstellen aus Amazon ECR in AWS Security Hub abgerufen und deren Behebung automatisiert werden können, indem der Zugriff auf anfällige Images blockiert wird.

Sonderberechtigungen aus Ihren Images entfernen

Die Flags für Zugriffsrechte `setuid` und `setgid` erlauben die Ausführung einer ausführbaren Datei mit den Berechtigungen des Eigentümers oder der Gruppe der ausführbaren Datei. Entfernen Sie alle Binärdateien mit diesen Zugriffsrechten aus Ihrem Image, da diese Binärdateien zur Erweiterung von Rechten verwendet werden können. Erwägen Sie, alle Shells und Serviceprogramme wie `nc` und `curl`, die für böswillige Zwecke verwendet werden können, zu entfernen. Sie können die Dateien mit `setuid`- und `setgid`-Zugriffsrechten finden, indem Sie den folgenden Befehl verwenden.

```
find / -perm /6000 -type f -exec ls -ld {} \;
```

Um diese speziellen Berechtigungen aus diesen Dateien zu entfernen, fügen Sie Ihrem Container-Image die folgende Direktive hinzu.

```
RUN find / -xdev -perm /6000 -type f -exec chmod a-s {} \; || true
```

Eine Reihe von kuratierten Images erstellen

Anstatt es Entwicklern zu ermöglichen, ihre eigenen Images zu erstellen, sollten Sie eine Reihe von geprüften Images für die verschiedenen Anwendungsstapel in Ihrem Unternehmen erstellen. Auf diese Weise können Entwickler darauf verzichten, zu lernen, wie man Dockerfiles erstellt, und sich auf das Schreiben von Code konzentrieren. Wenn Änderungen in Ihrer Codebasis zusammengeführt werden, kann eine CI/CD-Pipeline das Asset automatisch kompilieren und dann in einem Artefakt-Repository speichern. Und zuletzt kopieren Sie das Artefakt in das entsprechende Image, bevor Sie es in eine Docker-Registry wie Amazon ECR übertragen. Zumindest sollten Sie eine Reihe von Basis-Images erstellen, aus denen Entwickler ihre eigenen Dockerfiles erstellen können. Sie sollten

es vermeiden, Bilder aus Docker Hub abzurufen. Sie wissen nicht immer, was sich im Image befindet, und etwa ein Fünftel der Top-1000 Images weist Schwachstellen auf. Eine Liste dieser Images und ihrer Schwachstellen finden Sie unter https://vulnerablecontainers.org/.

Anwendungspakete und Bibliotheken auf Schwachstellen scannen

Die Verwendung von Open-Source-Bibliotheken ist heute weit verbreitet. Wie bei Betriebssystemen und Betriebssystempaketen können diese Bibliotheken Schwachstellen aufweisen. Im Rahmen des Entwicklungszyklus sollten diese Bibliotheken gescannt und aktualisiert werden, wenn kritische Schwachstellen gefunden werden.

Docker Desktop führt lokale Scans mit Snyk durch. Es kann auch verwendet werden, um Schwachstellen und potenzielle Lizenzprobleme in Open-Source-Bibliotheken zu finden. Es kann direkt in Entwickler-Workflows integriert werden, sodass Sie die von Open-Source-Bibliotheken ausgehenden Risiken minimieren können. Weitere Informationen finden Sie unter den folgenden Themen:

- Die Open Source Application Security Tools enthalten eine Liste von Tools zur Erkennung von Schwachstellen in Anwendungen.

Eine statische Codeanalyse durchführen

Sie sollten eine statische Codeanalyse durchführen, bevor Sie ein Container-Image erstellen. Sie wird anhand Ihres Quellcodes durchgeführt und dient dazu, Codierungsfehler und Code zu identifizieren, der von böswilligen Akteuren ausgenutzt werden könnte, z. B. bei Fehlerinjektionen. SonarQubeist eine beliebte Option für statische Anwendungssicherheitstests (SAST) und unterstützt eine Vielzahl verschiedener Programmiersprachen.

Container als nicht Root-Benutzer ausführen

Sie sollten Container als Nicht-Root-Benutzer ausführen. Standardmäßig werden Container als der `root`-Benutzer ausgeführt, sofern die `USER`-Direktive nicht in Ihrem Dockerfile enthalten ist. Die standardmäßigen Linux-Funktionen, die von Docker zugewiesen werden, schränken die Aktionen ein, die als `root` ausgeführt werden können, aber nur geringfügig. Beispielsweise darf ein Container, der als `root` ausgeführt wird, immer noch nicht auf Geräte zugreifen.

Als Teil Ihrer CI/CD-Pipeline sollten Sie Dockerfiles so einrichten, dass es nach der `USER`-Direktive sucht und den Build fehlschlägt, falls sie fehlt. Weitere Informationen finden Sie unter den folgenden Themen:

- DockerFile-Lint ist ein Open-Source-Tool von RedHat , mit dem überprüft werden kann, ob die Datei den Best Practices entspricht.
- Hadolint ist ein weiteres Tool zum Erstellen von Docker-Images, die den bewährten Methoden entsprechen.

Ein schreibgeschütztes Root-Dateisystem verwenden

Sie sollten ein schreibgeschütztes Root-Dateisystem verwenden. Das Root-Dateisystem eines Containers ist standardmäßig beschreibbar. Wenn Sie einen Container mit einem `RO` (schreibgeschützten) Root-Dateisystem konfigurieren, müssen Sie explizit definieren, wo Daten persistent gespeichert werden können. Dadurch wird Ihre Angriffsfläche reduziert, da in das Dateisystem des Containers nur geschrieben werden kann, wenn ausdrückliche Berechtigungen erteilt wurden.

> **Note**
>
> Ein schreibgeschütztes Root-Dateisystem kann zu Problemen mit bestimmten Betriebssystempaketen führen, die erwarten, in das Dateisystem schreiben zu können. Wenn Sie vorhaben, schreibgeschützte Root-Dateisysteme zu verwenden, testen Sie es vorher gründlich.

Aufgaben mit CPU- und Speicherlimits konfigurieren (Amazon EC2)

Sie sollten Aufgaben mit CPU- und Speicherlimits konfigurieren, um das folgende Risiko zu minimieren. Die Ressourcenlimits einer Aufgabe legen eine Obergrenze für die Menge an CPU und Arbeitsspeicher fest, die von allen Containern innerhalb einer Aufgabe reserviert werden kann. Wenn keine Grenzwerte festgelegt sind, haben Aufgaben Zugriff auf die CPU und den Arbeitsspeicher des Hosts. Dies kann zu Problemen führen, bei denen Aufgaben, die auf einem gemeinsam genutzten Host bereitgestellt werden, anderen Aufgaben die Systemressourcen entziehen können.

> **Note**
>
> Bei Amazon ECS für AWS Fargate Aufgaben müssen Sie CPU- und Speicherlimits angeben, da diese Werte für Abrechnungszwecke verwendet werden. Eine Aufgabe, die alle Systemressourcen beansprucht, ist für Amazon ECS Fargate kein Problem, da jede Aufgabe auf einer eigenen Dedicated Instance ausgeführt wird. Wenn Sie kein Speicherlimit angeben,

weist Amazon ECS jedem Container mindestens 4 MB zu. Wenn für die Aufgabe kein CPU-Limit festgelegt ist, weist ihr der Amazon-ECS-Container-Agent ebenfalls mindestens 2 CPUs zu.

Unveränderliche Tags mit Amazon ECR verwenden

Mit Amazon ECR können und sollten Sie konfigurierte Images mit unveränderlichen Tags verwenden. Dadurch wird verhindert, dass eine geänderte oder aktualisierte Version eines Images mit einem identischen Tag in Ihr Image-Repository übertragen wird. Dies schützt davor, dass ein Angreifer eine kompromittierte Version eines Images über Ihr Image mit demselben Tag überträgt. Durch die Verwendung unveränderlicher Tags zwingen Sie sich quasi dazu, bei jeder Änderung ein neues Image mit einem anderen Tag zu übertragen.

Vermeiden, Container als privilegiert auszuführen (Amazon EC2)

Sie sollten es vermeiden, Container als privilegiert auszuführen. Für den Hintergrund werden Container als `privileged` mit erweiterten Rechten auf dem Host ausgeführt. Das bedeutet, dass der Container alle Linux-Funktionen erbt, die `root` auf dem Host zugewiesen sind. Seine Verwendung sollte stark eingeschränkt oder verboten werden. Wir empfehlen, die Amazon-ECS-Umgebungsvariable `ECS_DISABLE_PRIVILEGED` für den Container-Agenten auf `true` einzustellen, sodass Container nicht `privileged` auf bestimmten Hosts ausführen, wenn `privileged` nicht benötigt ist. Alternativ können Sie Ihre Aufgabendefinitionen AWS Lambda nach der Verwendung des `privileged` Parameters durchsuchen.

> **Note**
>
> Das Ausführen eines Containers als `privileged` wird auf Amazon ECS auf AWS Fargate nicht unterstützt.

Nicht benötigte Linux-Funktionen aus dem Container entfernen

Im Folgenden finden Sie eine Liste der Linux-Standardfunktionen, die Docker-Containern zugewiesen sind. Weitere Informationen zu den einzelnen Funktionen finden Sie unter Linux-Funktionen im Überblick.

```
CAP_CHOWN, CAP_DAC_OVERRIDE, CAP_FOWNER, CAP_FSETID, CAP_KILL,
CAP_SETGID, CAP_SETUID, CAP_SETPCAP, CAP_NET_BIND_SERVICE,
CAP_NET_RAW, CAP_SYS_CHROOT, CAP_MKNOD, CAP_AUDIT_WRITE,
CAP_SETFCAP
```

Wenn ein Container nicht alle der oben aufgeführten Docker-Kernelfunktionen benötigt, sollten Sie erwägen, sie aus dem Container zu löschen. Weitere Informationen zu den einzelnen Funktionen des Docker-Kernels finden Sie unter KernalCapabilities. Sie können herausfinden, welche Funktionen verwendet werden, indem Sie wie folgt vorgehen:

- Installieren Sie das Betriebssystempaket libcap-ng und führen Sie das `pscap`-Hilfsprogramm aus, um die Funktionen aufzulisten, die jeder Prozess verwendet.
- Sie können Capsh auch verwenden, um zu entziffern, welche Fähigkeiten ein Prozess verwendet.
- Weitere Informationen finden Sie unter Linux Capabilities 101.

Verwenden Sie einen kundenseitig verwalteten Schlüssel (CMK), um Images zu verschlüsseln, die an Amazon ECR übertragen werden

Sie sollten einen kundenseitig verwalteten Schlüssel (CMK) verwenden, um Images zu verschlüsseln, die an Amazon ECR übertragen werden. Bilder, die an Amazon ECR übertragen werden, werden im Ruhezustand automatisch mit einem AWS Key Management Service (AWS KMS) verwalteten Schlüssel verschlüsselt. Wenn Sie lieber Ihren eigenen Schlüssel verwenden möchten, unterstützt Amazon ECR jetzt die AWS KMS Verschlüsselung mit vom Kunden verwalteten Schlüsseln (CMK). Bevor Sie die serverseitige Verschlüsselung mit einem CMK aktivieren, sollten Sie die in der Dokumentation zur Verschlüsselung im Ruhezustand aufgeführten Überlegungen lesen.

Laufzeit-Sicherheit

Laufzeit-Sicherheit bietet aktiven Schutz für Ihre Container, während diese laufen. Die Idee besteht darin, böswillige Aktivitäten in Ihren Containern zu erkennen und zu verhindern. Die Konfiguration der Laufzeit-Sicherheit unterscheidet sich zwischen Windows- und Linux-Containern.

Informationen zum Sichern eines Microsoft-Windows-Containers finden Sie unter Windows-Container sichern.

Um einen Linux-Container zu sichern, können Sie Linux-Kernelfunktionen mithilfe von SELinux `labels` hinzufügen oder löschen oder ein AppArmor Profil mit dem anwendendockerSecurity0ptions, beides pro Container innerhalb einer Aufgabendefinition.

`linuxParameters` SELinux oder AppArmor müssen auf der Container-Instance konfiguriert werden, bevor sie verwendet werden können. SELinux und AppArmor sind nicht verfügbar in. AWS Fargate Weitere Informationen finden Sie unter `dockerSecurityOptions` in der API-Referenz zu Amazon Elastic Container Service und unter Sicherheitskonfiguration in der Docker-Run-Referenz.

AppArmor ist ein Linux-Sicherheitsmodul, das die Funktionen eines Containers einschränkt, einschließlich des Zugriffs auf Teile des Dateisystems. Es kann entweder `enforcement`- oder `complain`-Modus ausgeführt werden. Da das Erstellen von AppArmor Profilen eine Herausforderung sein kann, empfehlen wir Ihnen, ein Tool wie Bane zu verwenden. Weitere Informationen zu AppArmor finden Sie auf der offiziellen AppArmorSeite.

> **Important**
>
> AppArmor ist nur für Ubuntu- und Debian-Distributionen von Linux verfügbar.

Empfehlungen

Es wird empfohlen, die folgenden Aktionen auszuführen, wenn Sie Ihre Laufzeit-Sicherheit einrichten.

Eine Drittanbieter-Lösung für Laufzeit-Schutz verwenden

Verwenden Sie eine Drittanbieter-Lösung für Laufzeit-Schutz. Wenn Sie mit der Funktionsweise von Linux-Sicherheit vertraut sind, erstellen und verwalten Sie AppArmor Profile. Beide Projekte sind Open-Source-Projekte. Andernfalls sollten Sie erwägen, stattdessen einen anderen Service eines Drittanbieters zu verwenden. Die meisten nutzen Machine Learning, um verdächtige Aktivitäten zu blockieren oder davor zu warnen. Eine Liste der verfügbaren Drittanbieterlösungen finden Sie unter AWS Marketplace für Container.

AWS Partner

Sie können jedes der folgenden AWS Partnerprodukte verwenden, um Ihren Amazon ECS-Workloads zusätzliche Sicherheit und Funktionen hinzuzufügen. Weitere Informationen finden Sie unter Amazon-ECS-Partner.

Aqua Security

Sie können Aqua Security verwenden, um Ihre cloudnativen Anwendungen von der Entwicklung bis zur Produktion zu schützen. Die Aqua Cloud Native Security Platform lässt sich in Ihre cloudnativen

Ressourcen und Orchestrierungstools integrieren, um transparente und automatisierte Sicherheit zu gewährleisten. Sie kann verdächtige Aktivitäten und Angriffe in Echtzeit verhindern und dazu beitragen, Richtlinien durchzusetzen und die Einhaltung gesetzlicher Vorschriften zu vereinfachen.

Palo Alto Networks

Palo Alto Networks bietet Sicherheit und Schutz für Ihre Hosts, Container und Serverless-Infrastrukturen in der Cloud und während des gesamten Entwicklungs- und Softwarelebenszyklus.

Twistlock wird von Palo Alto Networks geliefert und kann in Amazon ECS integriert werden. FireLens Damit haben Sie Zugriff auf zuverlässige Sicherheitsprotokolle und Vorfälle, die nahtlos in mehrere Dienste zusammengefasst werden. AWS Dazu gehören Amazon CloudWatch, Amazon Athena und Amazon Kinesis. Twistlock sichert Workloads, die auf Container-Services bereitgestellt werden. AWS

Sysdig

Sie können Sysdig verwenden, um sichere und konforme cloudnative Workloads in Produktionsszenarien auszuführen. Die Sysdig Secure DevOps Platform verfügt über integrierte Sicherheits- und Compliance-Funktionen zum Schutz Ihrer Cloud-nativen Workloads und bietet außerdem Skalierbarkeit, Leistung und Anpassung auf Unternehmensniveau.

Dokumentenverlauf für den Amazon ECS Best Practices Guide

In der folgenden Tabelle werden die Dokumentationsversionen für den Amazon ECS Best Practices Guide beschrieben.

Änderung	Beschreibung	Datum
Bewährte Methoden für den Betrieb von Amazon ECS im großen Maßstab	Es wurden bewährte Methoden für den skalierbaren Betrieb von Amazon ECS hinzugefügt.	1. Juni 2022
Bewährte Methoden zur Beschleunigung des Starts von Aufgaben	Es wurden bewährte Methoden hinzugefügt, um den Start Ihrer Amazon ECS-Aufgaben zu beschleunigen.	1. April 2022
Bewährte Methoden für Anwendungen	Es wurden bewährte Methoden für die Implementierung Ihrer Anwendung mit Amazon ECS hinzugefügt.	28. Oktober 2021
Bewährte Methoden für die Bereitstellung	Es wurden bewährte Methoden zur Beschleunigung von Amazon ECS-Bereitstellungen hinzugefügt.	10. August 2021
Bewährte Methoden für die Gewährleistung der Sicherheit	Es wurden bewährte Methoden für das Sicherheitsmanagement für Amazon ECS-Workloads hinzugefügt.	26. Mai 2021
Bewährte Methoden für automatische Skalierung und Kapazitätsmanagement	Es wurden bewährte Methoden für auto Skalierung und Kapazitätsmanagement	14. Mai 2021

	für Amazon ECS-Workloads hinzugefügt.	
Bewährte Methoden für persistenten Speicher	Es wurden bewährte Methoden für persistenten Speicher für Amazon ECS-Workloads hinzugefügt.	7. Mai 2021
Bewährte Methoden für Netzwerke	Es wurden bewährte Methoden für das Netzwerkmanagement für Amazon ECS-Workloads hinzugefügt.	6. April 2021
Erstversion	Erste Veröffentlichung des Amazon ECS Best Practices Guide	6. April 2021

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.