aws

모범 사례

Amazon Elastic Container Service

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

Amazon Elastic Container Service: 모범 사례

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

Amazon의 상표 및 트레이드 드레스는 Amazon이 제공하지 않는 제품 또는 서비스와 관련하여 고객에게 혼동을 유발할 수 있는 방식 또는 Amazon을 폄하하거나 평판에 악영향을 주는 방식으로 사용될 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon과 제휴 관계이거나 관련이 있거나 후원 관계 여부에 관계없이 해당 소유자의 자산입니다.

Table of Contents

Introduction 1
네트워킹 2
인터넷에 연결하기 2
퍼블릭 서브넷 및 인터넷 게이트웨이 사용 3
프라이빗 서브넷 및 NAT 게이트웨이 사용 5
인터넷에서 인바운드 연결 수신 6
Application Load Balancer 6
Network Load Balancer 8
Amazon API Gateway HTTP API 9
네트워크 모드 선택 10
호스트 모드 10
브리지 모드 12
AWSVPC 모드 14
에 연결AWS서비스 19
NAT 게이트웨이 19
AWS PrivateLink 20
Amazon ECS 서비스 간 네트워킹 21
서비스 검색 사용 22
내부 로드 밸런서 사용 23
서비스 메시 사용 25
전반의 네트워킹 서비스AWS계정 및 VPC 26
최적화 및 문제 해결 27
CloudWatch 컨테이너 인사이트 27
AWS X-Ray 27
VPC 흐름 로그 29
네트워크 튜닝 팁 29
자동 확장 및 용량 관리 31
작업 크기 결정 31
상태 비저장 애플리케이션 32
기타 애플리케이션 32
Service Scaling 구성 32
애플리케이션 특성화 33
용량 및 가용성 37
속도 조정 속도 극대화 38

수요 충격 처리 40
클러스터 용량 41
클러스터 용량 모범 사례 42
Fargate 작업 크기 선택 42
Amazon EC2 인스턴스 유형 선택 42
Amazon EC2 스팟 및 퍼게이트_스팟 사용 43
영구 스토리지 45
올바른 스토리지 유형 선택 47
Amazon EFS 48
보안 및 액세스 제어 50
Performance 51
Throughput 51
비용 최적화 52
데이터 보호 53
사용 사례 53
도커 볼륨 53
Amazon EBS 볼륨 수명 주기 54
Amazon EBS 데이터 가용성 55
도커 볼륨 플러그인 55
Amazon FSx for Windows File Server 56
보안 및 액세스 제어 57
사용 사례 58
보안 59
공동 책임 모델 59
AWS Identity and Access Management 61
Amazon ECS에 대한 액세스 관리 61
Recommendations 62
Amazon ECS 작업에 IAM 역할 사용 64
작업 실행 역할 66
Amazon EC2 컨테이너 인스턴스 역할 67
서비스 연결 역할 68
Recommendations 68
네트워크 보안 70
전송 중 데이터 암호화 70
작업 네트워킹 72
서비스 메쉬 및 상호 전송 계층 보안 (MTL) 72

AWS PrivateLink 73
Amazon ECS 컨테이너 에이전트 설정 74
Recommendations 74
보안 관리 75
Recommendations 76
추가 리소스 77
Compliance 78
PCI DSS (지불 카드 산업 데이터 보안 표준) 78
HIPAA (미국 Health 보험 양도 및 책임에 관한 법) 78
Recommendations 79
로깅 및 모니터링 79
유창한 비트를 사용한 컨테이너 로깅 79
사용자 지정 로그 라우팅 - 아마존 ECS용 FireLens 80
AWS Fargate 보안 81
사용AWS KMS사용 후 삭제 스토리지를 암호화합니다. 81
커널 시스템 호출 추적을 위한 SYS_PSTRACE 기능 81
작업 및 컨테이너 보안 82
Recommendations 82
런타임 보안 87
Recommendations 88
AWS파트너 89
문서 기록 90
..... xci

Introduction

Amazon ECS () 는 클러스터에서 컨테이너를 손쉽게 실행, 중지 및 관리할 수 있게 해주는 컨테이너 관리 서비스로서 확장성과 속도가 뛰어납니다. 이 가이드에서는 가장 중요한 운영 모범 사례를 다룰 뿐만 아니라 Amazon ECS 기반 애플리케이션의 작동 방식을 뒷받침하는 핵심 주제를 설명합니다. 목표는 Amazon ECS 기반 애플리케이션의 운영 및 문제 해결을 위한 구체적이고 실행 가능한 접근 방식을 제공하는 것입니다.

이 가이드는 새로운 Amazon ECS 모범 사례를 통합하기 위해 정기적으로 개정됩니다. 이 가이드의 콘텐츠에 대해 궁금한 점 또는 의견이 있는 경우 GitHub 저장소에서 문제를 제기합니다. 자세한 내용은 단원을 참조하십시오.아마존 ECS 모범 사례 가이드(GitHub에 있음).

- 모범 사례 - 네트워킹
- 모범 사례 - 자동 확장 및 용량 관리
- 모범 사례 - 영구 스토리지
- 보안 모범 사례

모범 사례 - 네트워킹

최신 응용 프로그램은 일반적으로 서로 통신하는 여러 분산 구성 요소로 구축됩니다. 예를 들어 모바일 또는 웹 응용 프로그램은 API 끝점과 통신할 수 있으며 인터넷을 통해 통신하는 여러 마이크로 서비스에 의해 API가 구동될 수 있습니다.

이 가이드에서는 응용 프로그램의 구성 요소가 서로 안전하고 확장 가능한 방식으로 통신할 수 있는 네트워크를 구축하기 위한 모범 사례를 제공합니다.

주제

- 인터넷에 연결하기
- 인터넷에서 인바운드 연결 수신
- 네트워크 모드 선택
- 에 연결AWS서비스를 VPC 내부에서
- VPC Amazon ECS 서비스 간 네트워킹
- 전반의 네트워킹 서비스AWS계정 및 VPC
- 최적화 및 문제 해결

인터넷에 연결하기

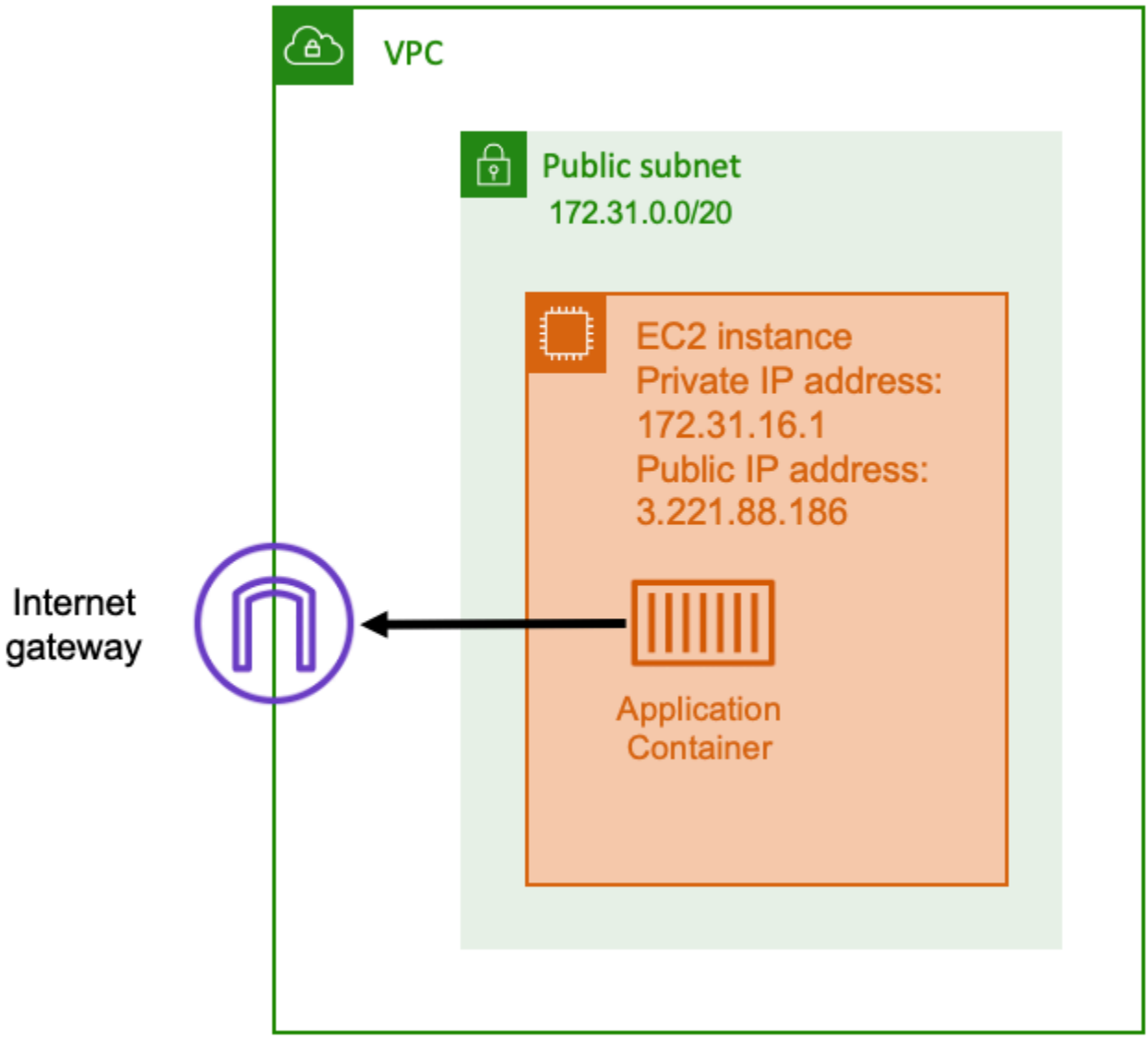
대부분의 컨테이너화된 응용 프로그램에는 인터넷에 대한 아웃바운드 액세스가 필요한 일부 구성 요소가 있습니다. 예를 들어 모바일 앱의 백엔드에서 푸시 알림에 대한 아웃바운드 액세스가 필요합니다.

Amazon Virtual Private Cloud 에는 VPC 와 인터넷 간의 통신을 촉진하는 두 가지 주요 방법이 있습니다.

주제

- 퍼블릭 서브넷 및 인터넷 게이트웨이 사용
- 프라이빗 서브넷 및 NAT 게이트웨이 사용

퍼블릭 서브넷 및 인터넷 게이트웨이 사용



인터넷 게이트웨이로 향하는 라우팅이 있는 퍼블릭 서브넷을 사용하면 컨테이너식 애플리케이션을 퍼블릭 서브넷의 VPC 내부의 호스트에서 실행할 수 있습니다. 컨테이너를 실행하는 호스트에는 퍼블릭 IP 주소가 할당됩니다. 이 공용 IP 주소는 인터넷에서 라우팅할 수 있습니다. 자세한 내용은 단원을 참조하십시오.인터넷 게이트웨이의Amazon VPC 사용 설명서.

이 네트워크 아키텍처는 응용 프로그램을 실행하는 호스트와 인터넷의 다른 호스트 간의 직접 통신을 용이하게합니다. 통신은 양방향입니다. 즉, 인터넷의 다른 호스트에 대한 아웃바운드 연결을 설정할 수 있을 뿐만 아니라 인터넷의 다른 호스트도 호스트에 연결을 시도할 수 있습니다. 따라서 보안 그룹 및

방화벽 규칙에 세심한 주의를 기울여야 합니다. 이는 인터넷의 다른 호스트가 열리지 않으려는 연결을 열 수 없도록 하기 위한 것입니다.

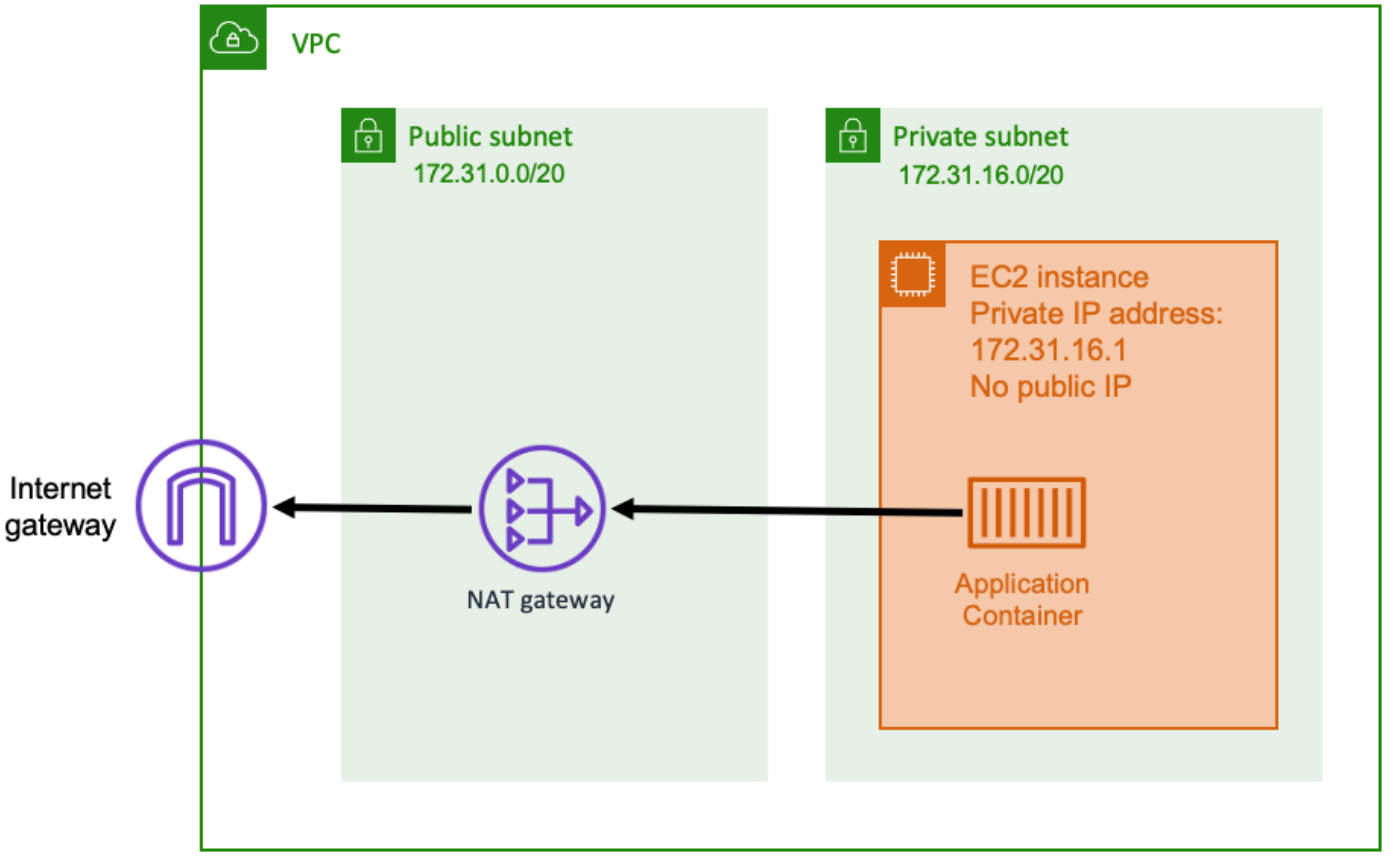
예를 들어 애플리케이션이 Amazon EC2 실행 중인 경우 SSH 액세스용 포트 22가 열려 있지 않은지 확인합니다. 그렇지 않으면 인스턴스가 인터넷상의 macilious 봇으로부터 지속적인 SSH 연결 시도를 수신할 수 있습니다. 이러한 봇은 공용 IP 주소를 통해 탐색합니다. 열린 SSH 포트를 찾은 후 인스턴스에 액세스하기 위해 무차별 대입 암호를 시도합니다. 이 때문에 많은 조직에서는 공용 서브넷의 사용을 제한하고 사설 서브넷 내에 리소스를 대부분 보유하는 것을 선호합니다.

네트워킹에 공용 서브넷을 사용하는 것은 많은 양의 대역폭이나 최소 대기 시간이 필요한 공용 응용 프로그램에 적합합니다. 적용 가능한 사용 사례에는 비디오 스트리밍 및 게임 서비스가 포함됩니다.

이 네트워킹 접근 방식은 Amazon ECAmazon EC2 ECS를 사용할 때와AWS Fargate.

- Amazon EC2 사용 — 퍼블릭 서브넷에서 EC2 인스턴스를 시작할 수 있습니다. Amazon ECS는 이러한 EC2 인스턴스를 클러스터 용량으로 사용하며 인스턴스에서 실행되는 모든 컨테이너는 아웃바운드 네트워킹에 호스트의 기본 퍼블릭 IP 주소를 사용할 수 있습니다. 이 두 적용 됩니다.`host`및`bridge`네트워크 모드. 그러나,`awsvpc`네트워크 모드는 작업 ENI에 퍼블릭 IP 주소를 제공하지 않습니다. 따라서 인터넷 게이트웨이를 직접 사용할 수 없습니다.
- Fargate 사용 — Amazon ECS 서비스를 생성할 때 서비스의 네트워킹 구성을 위한 퍼블릭 서브넷을 지정하고퍼블릭 IP 주소 배정옵션이 활성화되어 있는지 확인합니다. 각 Fargate 작업은 퍼블릭 서브넷에 네트워크로 연결되며 인터넷과 직접 통신할 수 있는 자체 퍼블릭 IP 주소가 있습니다.

프라이빗 서브넷 및 NAT 게이트웨이 사용



프라이빗 서브넷과 NAT 게이트웨이를 사용하면 프라이빗 서브넷에 있는 호스트에서 컨테이너화된 애플리케이션을 실행할 수 있습니다. 따라서 이 호스트에는 VPC 내부에서 라우팅할 수 있지만 인터넷에서 라우팅할 수 없는 프라이빗 IP 주소가 있습니다. 즉, VPC 내의 다른 호스트는 프라이빗 IP 주소를 사용하여 호스트에 연결할 수 있지만 인터넷의 다른 호스트는 호스트에 대한 인바운드 통신을 수행할 수 없습니다.

프라이빗 서브넷의 경우 NAT (네트워크 주소 변환) 게이트웨이를 사용하여 프라이빗 서브넷 내의 호스트를 인터넷에 연결하는 한편, 인터넷의 호스트는 퍼블릭 서브넷 내에 있는 NAT 게이트웨이의 퍼블릭 IP 주소에서 들어오는 것으로 보이는 인바운드 연결을 수신합니다. NAT 게이트웨이는 인터넷과 프라이빗 VPC 사이의 다리 역할을 담당합니다. 이 구성은 인터넷상의 공격자가 직접 액세스하지 못하도록 VPC 보호하므로 보안상의 이유로 선호되는 경우가 많습니다. 자세한 내용은 단원을 참조하십시오.NAT 게이트웨이의Amazon VPC 사용 설명서.

이 사설 네트워킹 접근 방식은 컨테이너를 직접 외부 액세스로부터 보호하려는 시나리오에 적합합니다. 적용 가능한 시나리오에는 사용자 데이터 및 암호를 저장하는 결제 처리 시스템 또는 컨테이너가 포함됩니다. 계정에서 NAT 게이트웨이 생성 및 사용에 대한 요금이 청구됩니다. NAT 게이트웨이 시간

당 사용 요금 및 데이터 처리 요금도 적용됩니다. 중복성을 위해 각 가용 영역에 NAT 게이트웨이가 있어야 합니다. 이렇게 하면 단일 가용 영역의 가용성 손실이 아웃바운드 연결을 손상시키지 않습니다. 따라서 워크로드가 적은 경우 사설 서브넷과 NAT 게이트웨이를 사용하는 것이 비용 효율적일 수 있습니다.

이 네트워킹 접근 방식은 Amazon ECAmazon EC2 ECS를 사용할 때와AWS Fargate.

- Amazon EC2 사용 — 프라이빗 서브넷에서 EC2 인스턴스를 시작할 수 있습니다. 이러한 EC2 호스트에서 실행되는 컨테이너는 기본 호스트 네트워킹을 사용하며 아웃바운드 요청은 NAT 게이트웨이를 통과합니다.
- Fargate 사용 — Amazon ECS 서비스를 생성할 때 서비스의 네트워킹 구성을 위한 프라이빗 서브넷을 지정하고퍼블릭 IP 주소 배정옵션을 선택합니다. 각 Fargate 작업은 프라이빗 서브넷에서 호스팅됩니다. 아웃바운드 트래픽은 해당 프라이빗 서브넷과 연결된 NAT 게이트웨이를 통해 라우팅됩니다.

인터넷에서 인바운드 연결 수신

공용 서비스를 실행하는 경우 인터넷으로부터의 인바운드 트래픽을 수락해야 합니다. 예를 들어 공용 웹 사이트는 브라우저의 인바운드 HTTP 요청을 수락해야 합니다. 이 경우 인터넷상의 다른 호스트도 애플리케이션 호스트에 대한 인바운드 연결을 시작해야 합니다.

이 문제에 대한 한 가지 방법은 퍼블릭 IP 주소를 사용하여 퍼블릭 서브넷에있는 호스트에서 컨테이너를 시작하는 것입니다. 그러나 대규모 응용 프로그램에서는 이 옵션을 권장하지 않습니다. 이를 위해 인터넷과 응용 프로그램 사이에 확장 가능한 입력 계층을 사용하는 것이 더 좋습니다. 이 방법을 사용하려면AWS이 섹션에 나열된 서비스를 입력으로 사용할 수 있습니다.

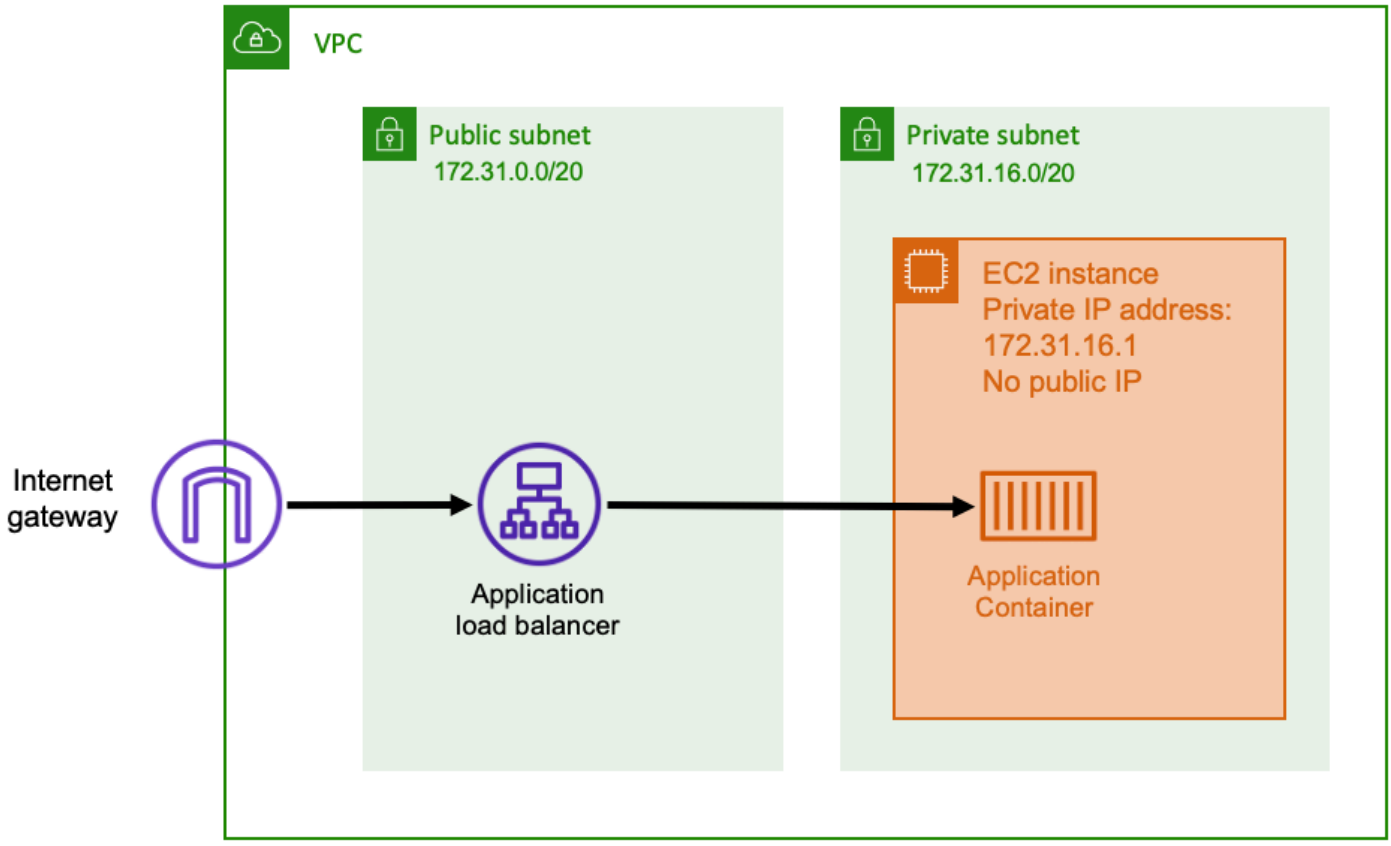
주제

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

Application Load Balancer

Application Load Balancer 애플리케이션 계층에서 작동합니다. 이는 개방형 시스템 간 상호 연결(OSI) 모델의 일곱 번째 계층입니다. 따라서 Application Load Balancer 공용 HTTP 서비스에 적합합니다. 웹 사이트 또는 HTTP REST API가 있는 경우 Application Load Balancer 이 워크로드에 적합

한 로드 밸런서입니다. 자세한 내용은 단원을 참조하십시오.Application Load Balancer 란 무엇입니까?의Application Load Balancer 사용 설명서.



이 아키텍처를 사용하면 퍼블릭 서브넷에 Application Load Balancer 서를 생성하여 퍼블릭 IP 주소를 가지며 인터넷에서 인바운드 연결을 수신할 수 있습니다. Application Load Balancer 인바운드 연결 또는 특히 HTTP 요청을 수신하면 프라이빗 IP 주소를 사용하여 애플리케이션에 대한 연결을 엽니다. 그런 다음 내부 연결을 통해 요청을 전달합니다.

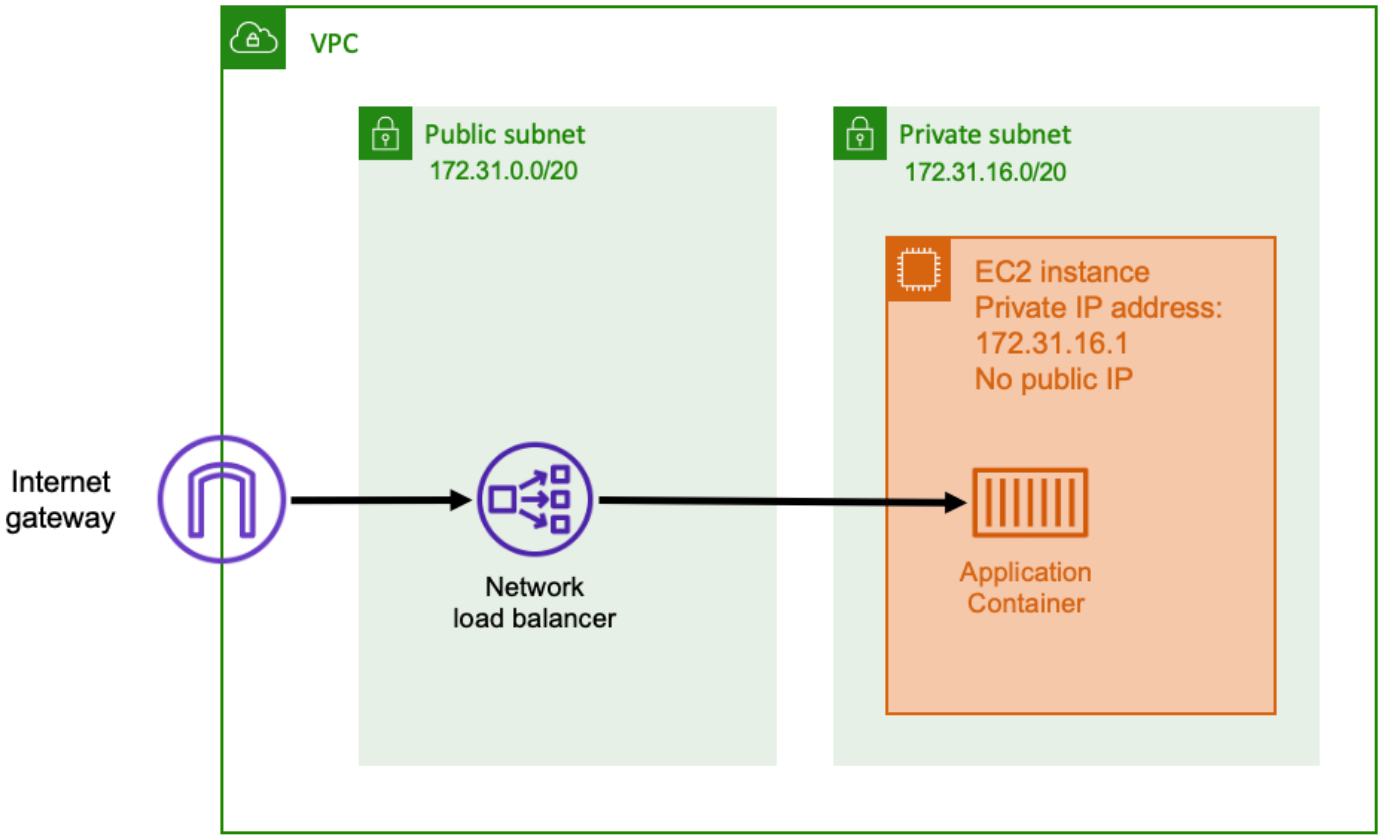
Application Load Balancer 의 장점은 다음과 같습니다.

- SSL/TLS 종료 — Application Load Balancer 클라이언트와의 통신을 위해 보안 HTTPS 통신 및 인증서를 유지할 수 있습니다. 필요에 따라 로드 밸런서 수준에서 SSL 연결을 종료할 수 있으므로 응용 프로그램에서 인증서를 처리할 필요가 없습니다.
- 고급 라우팅 — Application Load Balancer 여러 DNS 호스트 이름을 가질 수 있습니다. 또한 호스트 이름이나 요청 경로와 같은 메트릭을 기반으로 들어오는 HTTP 요청을 다른 대상으로 전송하는 고급 라우팅 기능이 있습니다. 즉, 단일 Application Load Balancer 여러 내부 서비스에 대한 입력으로 사용하거나 REST API의 다른 경로에 있는 마이크로서비스를 사용할 수 있습니다.

- gRPC 지원 및 웹 소켓 — Application Load Balancer 단순한 HTTP 이상의 기능을 처리할 수 있습니다. 또한 HTTP/2 지원을 통해 GRPC 및 websocket 기반 서비스를로드 밸런싱할 수 있습니다.
- 보안 — Application Load Balancer 사용하면 악성 트래픽으로부터 애플리케이션을 보호할 수 있습니다. 여기에는 HTTP 동기화 완화 기능과 같은 기능이 포함되어 있으며 AWS 웹 애플리케이션 방화벽 (AWS WAF).AWS WAF는 SQL 삽입 또는 교차 사이트 스크립팅과 같은 공격 패턴을 포함할 수 있는 악성 트래픽을 추가로 필터링할 수 있습니다.

Network Load Balancer

Network Load Balancer는 오픈 시스템 상호 연결(OSI) 모델의 네 번째 계층에서 작동합니다. 비 HTTP 프로토콜 또는 종단 간 암호화가 필요하지만 Application Load Balancer 동일한 HTTP 관련 기능을 갖지 않는 시나리오에 적합합니다. 따라서 Network Load Balancer HTTP를 사용하지 않는 응용 프로그램에 가장 적합합니다. 자세한 내용은 단원을 참조하십시오.Network Load Balancer 란 무엇입니까?의Network Load Balancer 사용 설명서.



Network Load Balancer 가 입력으로 사용되는 경우 Application Load Balancer 와 유사하게 작동합니다. 이는 퍼블릭 서브넷에서 생성되고 인터넷에서 액세스할 수 있는 퍼블릭 IP 주소를 갖고 있기 때문

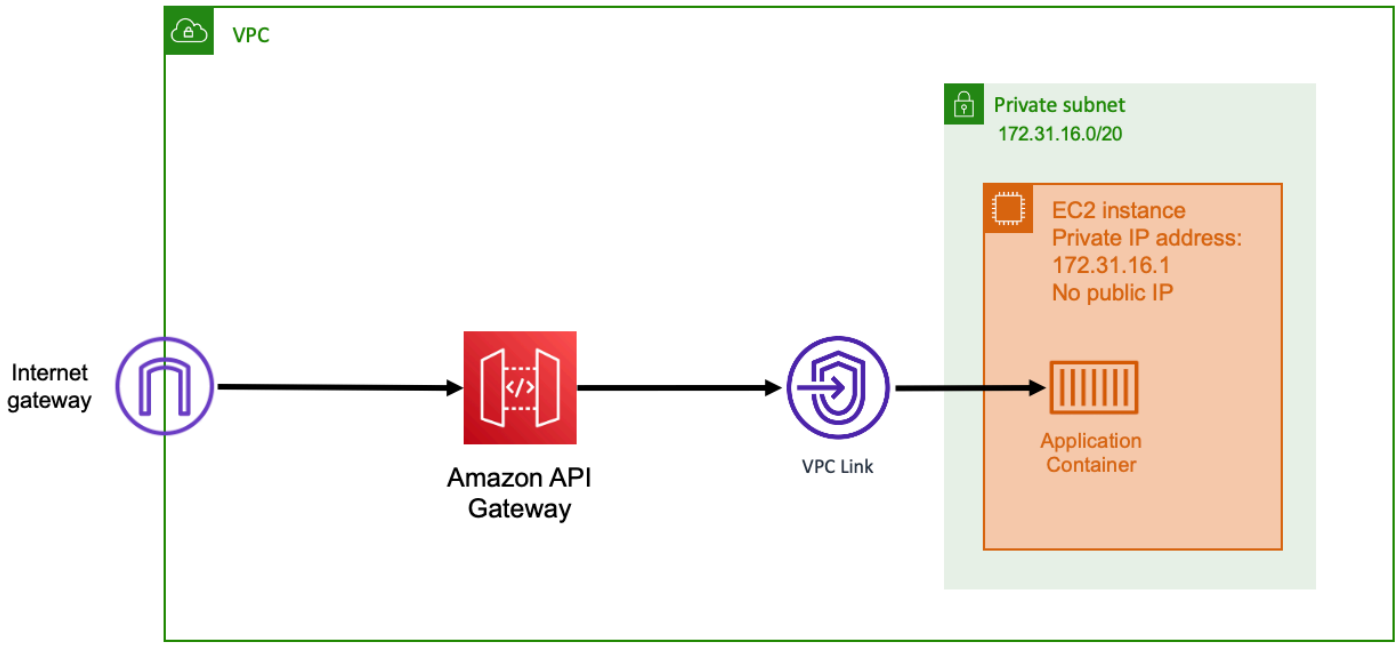
입니다. 그런 다음 Network Load Balancer 는 컨테이너를 실행하는 호스트의 프라이빗 IP 주소에 대한 연결을 열고 공개 측에서 개인 측으로 패킷을 보냅니다.

Network Load Balancer 낮은 수준의 네트워킹 스택에서 작동하므로 Application Load Balancer 동일한 기능 집합이 없습니다. 그러나 다음과 같은 중요한 기능이 있습니다.

- 종단 간 암호화 — Network Load Balancer OSI 모델의 네 번째 계층에서 작동하므로 패킷의 내용을 읽지 않습니다. 따라서 종단 간 암호화가 필요한 로드 밸런싱 통신에 적합합니다.
- TLS 암호화 — 종단 간 암호화 외에도 Network Load Balancer TLS 연결을 종료할 수도 있습니다. 이렇게 하면 백엔드 응용 프로그램에서 자체 TLS를 구현할 필요가 없습니다.
- UDP 지원 - Network Load Balancer OSI 모델의 네 번째 계층에서 작동하므로 TCP 이외의 비 HTTP 워크로드 및 프로토콜에 적합합니다.

Amazon API Gateway HTTP API

Amazon API Gateway HTTP API는 요청 볼륨이 급격히 증가하거나 요청 볼륨이 적은 HTTP 애플리케이션에 적합한 수신이 적은 서버입니다. 자세한 내용은 단원을 참조하십시오.Amazon API Gateway란 무엇입니까?의API Gateway 개발자 안내서.



애플리케이션 로드 밸런서와 네트워크 로드 밸런서의 가격 책정 모델에는 로드 밸런서를 항상 수신 연결을 수락할 수 있도록 하기 위한 시간당 가격이 포함되어 있습니다. 반대로 API Gateway 각 요청에 대해 별도로 요금을 부과합니다. 이로 인해 요청이 들어오지 않으면 요금이 부과되지 않습니다. 트래픽

로드가 많은 경우 Application Load Balancer 또는 Network Load Balancer API Gateway 보다 저렴한 요청당 가격으로 더 많은 양의 요청을 처리할 수 있습니다. 그러나 전체 요청 수가 적거나 트래픽이 짧은 기간인 경우 API Gateway 를 사용하는 누적 가격은 사용률이 낮은 로드 밸런서를 유지하기 위해 시간당 요금을 지불하는 것보다 비용 효율적이어야 합니다.

API Gateway VPC 링크를 사용하여AWS관리형 서비스를 사용하여 프라이빗 IP 주소를 사용하여 VPC 프라이빗 서브넷 내의 호스트에 연결할 수 있습니다. 이를 통해 이러한 개인 IP 주소를 감지할 수 있습니다.AWS Cloud MapAmazon ECS 서비스 검색으로 관리되는 서비스 검색 레코드입니다.

API Gateway 가 지원하는 기능은 다음과 같습니다.

- SSL/TLS 종료
- 다른 백엔드 마이크로 서비스로 서로 다른 HTTP 경로 라우팅

앞의 기능 외에도 API Gateway 는 무단 사용으로부터 API를 보호하는 데 사용할 수 있는 사용자 지정 Lambda 권한 부여자를 사용할 수도 있습니다. 자세한 내용은 단원을 참조하십시오.필드 참고: Amazon ECS와 아마존 API 게이트웨이를 사용하는 서버를 사용하지 않는 컨테이너 기반 API.

네트워크 모드 선택

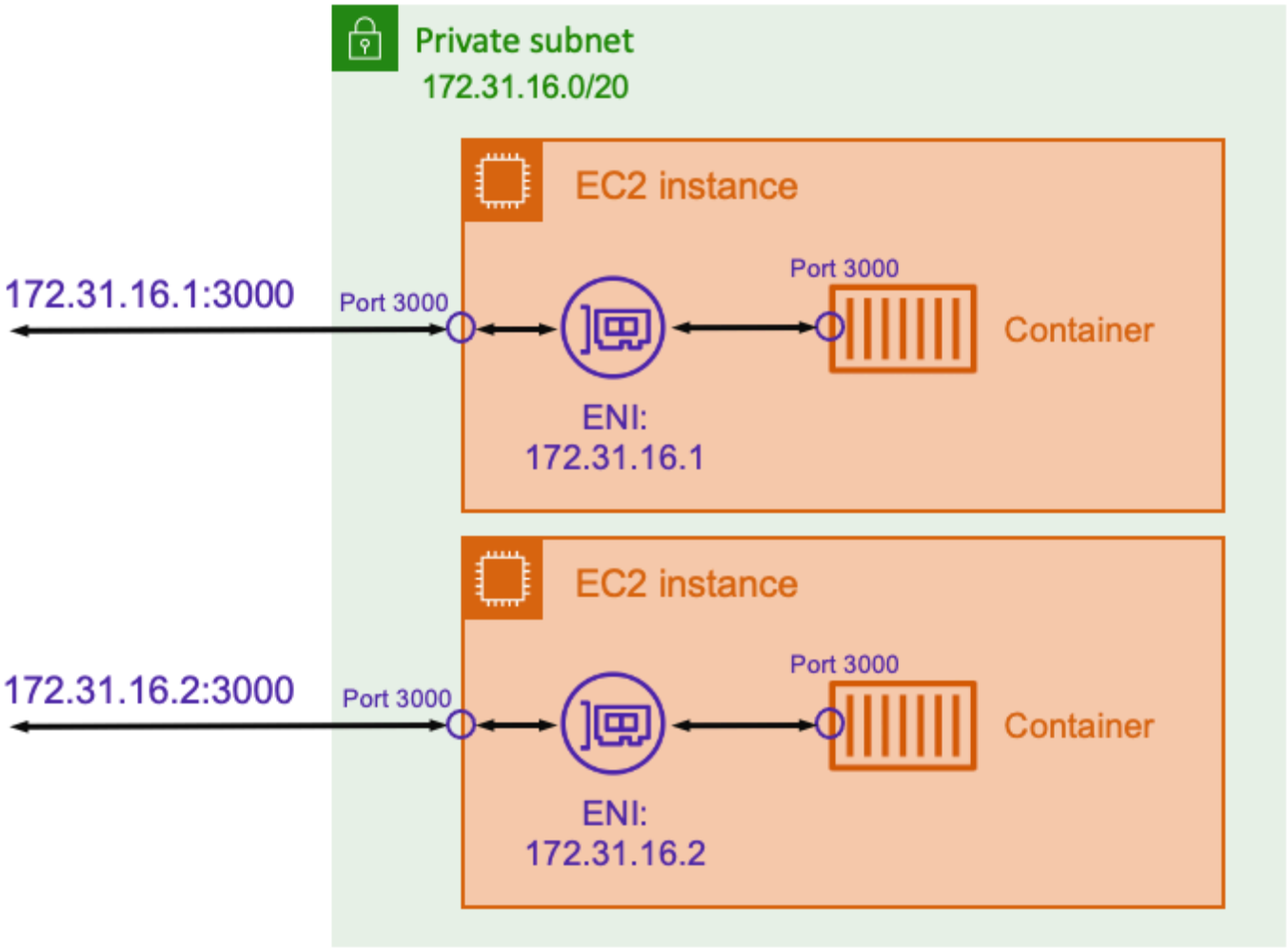
인바운드 및 아웃바운드 네트워크 연결을 설계하기 위해 앞서 언급한 접근 방식은AWS, 컨테이너 내부에 있지 않더라도. 컨테이너를 실행할 때AWS를 사용하려면 다른 수준의 네트워킹을 고려해야 합니다. 컨테이너를 사용하는 주요 이점 중 하나는 여러 컨테이너를 단일 호스트에 압축 할 수 있다는 것입니다. 이 작업을 수행 할 때 동일한 호스트에서 실행중인 컨테이너를 네트워크화하는 방법을 선택해야합니다. 다음은 선택할 수 있는 옵션입니다.

주제

- 호스트 모드
- 브리지 모드
- AWSVPC 모드

호스트 모드

이host네트워크 모드는 Amazon ECS에서 지원되는 가장 기본적인 네트워크 모드입니다. 호스트 모드를 사용하면 컨테이너의 네트워킹이 컨테이너를 실행하는 기본 호스트에 직접 연결됩니다.



포트에서 수신 대기 하는 Express 응용 프로그램을 사용 하 여 Node.js 컨테이너를 실행 하는 가정3000위 다이어그램에서 설명한 것과 유사합니다. 이 경우host네트워크 모드를 사용하는 경우 컨테이너는 기본 호스트 Amazon EC2 인스턴스의 IP 주소를 사용하여 포트 3000에서 트래픽을 수신합니다. 이 모드는 사용하지 않는 것이 좋습니다.

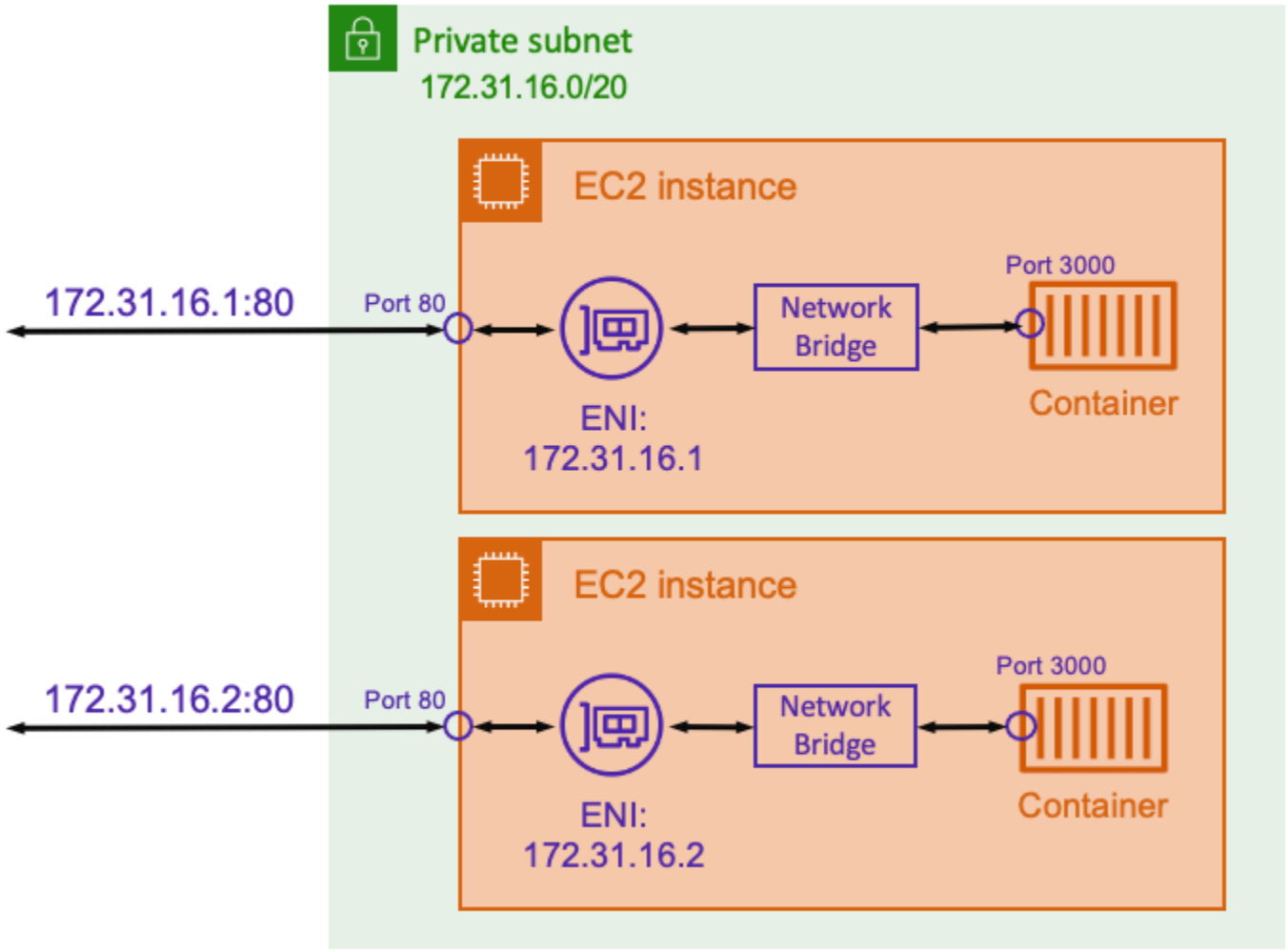
이 네트워크 모드를 사용하면 심각한 단점이 있습니다. 각 호스트에서 태스크의 인스턴스화를 하나 이상 실행할 수 없습니다. 이는 첫 번째 작업만 Amazon EC2 인스턴스의 필수 포트에 바인딩할 수 있기 때문입니다. 컨테이너 포트를 사용할 때 컨테이너 포트를 다시 매핑 할 수있는 방법도 없습니다.host네트워크 모드. 예를 들어 응용 프로그램이 특정 포트 번호에서 수신 대기해야하는 경우 포트 번호를 직접 다시 매핑 할 수 없습니다. 대신 응용 프로그램 구성을 변경하여 포트 충돌을 관리해야 합니다.

사용할 때 보안 영향도 있습니다host네트워크 모드. 이 모드를 사용하면 컨테이너가 호스트를 가장할 수 있으며 컨테이너가 호스트의 개인 루프백 네트워크 서비스에 연결할 수 있습니다.

이host네트워크 모드는 Amazon EC2 인스턴스에서 호스팅되는 Amazon ECS 작업에 대해서만 지원됩니다. Fargate 에서 Amazon ECS를 사용할 때는 지원되지 않습니다.

브리지 모드

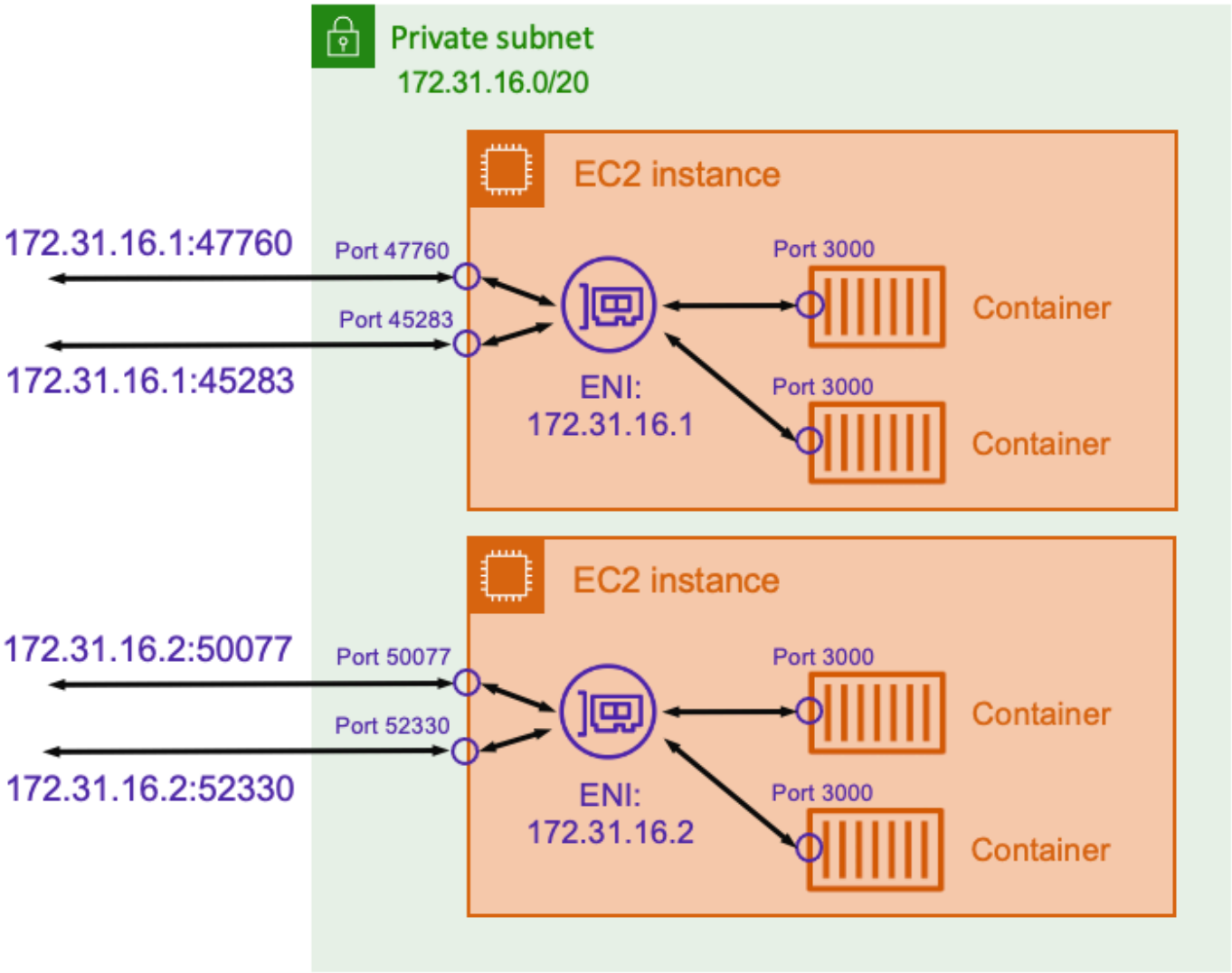
다음으로 바꿉니다.bridge모드에서 가상 네트워크 브리지를 사용하여 호스트와 컨테이너의 네트워킹 사이에 레이어를 만듭니다. 이렇게하면 호스트 포트를 컨테이너 포트에 다시 매핑하는 포트 매핑을 만들 수 있습니다. 매핑은 정적 또는 동적일 수 있습니다.



정적 포트 매핑을 사용하면 컨테이너 포트에 매핑 할 호스트 포트를 명시 적으로 정의 할 수 있습니다. 위의 예제를 사용하여 포트80호스트의 포트에 매핑되고 있습니다.3000컨테이너에 추가합니다. 컨테이너화된 응용 프로그램과 통신하려면 트래픽을 포트로 보냅니다.80Amazon EC2 인스턴스의 IP 주소로 설정합니다. 컨테이너화 된 응용 프로그램의 관점에서 포트의 인바운드 트래픽이3000.

트래픽 포트만 변경하려는 경우 정적 포트 매핑이 적합합니다. 그러나 이것은 여전히 사용하는 것과 동일한 단점이 있습니다.`host`네트워크 모드. 각 호스트에서 태스크의 인스턴스화를 하나 이상 실행할 수 없습니다. 이는 정적 포트 매핑이 단일 컨테이너 만 포트 80에 매핑 할 수 있기 때문입니다.

이 문제를 해결하려면`bridge`네트워크 모드를 다음 다이어그램에 표시된 대로 동적 포트 매핑으로 설정합니다.



포트 매핑에 호스트 포트를 지정하지 않으면 Docker가 임시 포트 범위에서 사용되지 않는 임의의 포트를 선택하여 컨테이너의 공용 호스트 포트로 할당하도록 할 수 있습니다. 예를 들어, 포트에서 수신 대기하는 Node.js 응용 프로그램`3000`에는 다음과 같은 임의의 높은 수의 포트가 할당 될 수 있습니다.`47760`Amazon EC2 호스트에서 해당 호스트를 선택합니다. 이렇게하면 호스트에서 해당 컨테이너의 여러 복사본을 실행할 수 있습니다. 또한 각 컨테이너는 호스트에 자체 포트를 할당 할 수 있습니다.

컨테이너의 각 복사본은 포트에서 트래픽을 수신합니다.`3000`. 그러나 이러한 컨테이너로 트래픽을 보내는 클라이언트는 임의로 할당된 호스트 포트를 사용합니다.

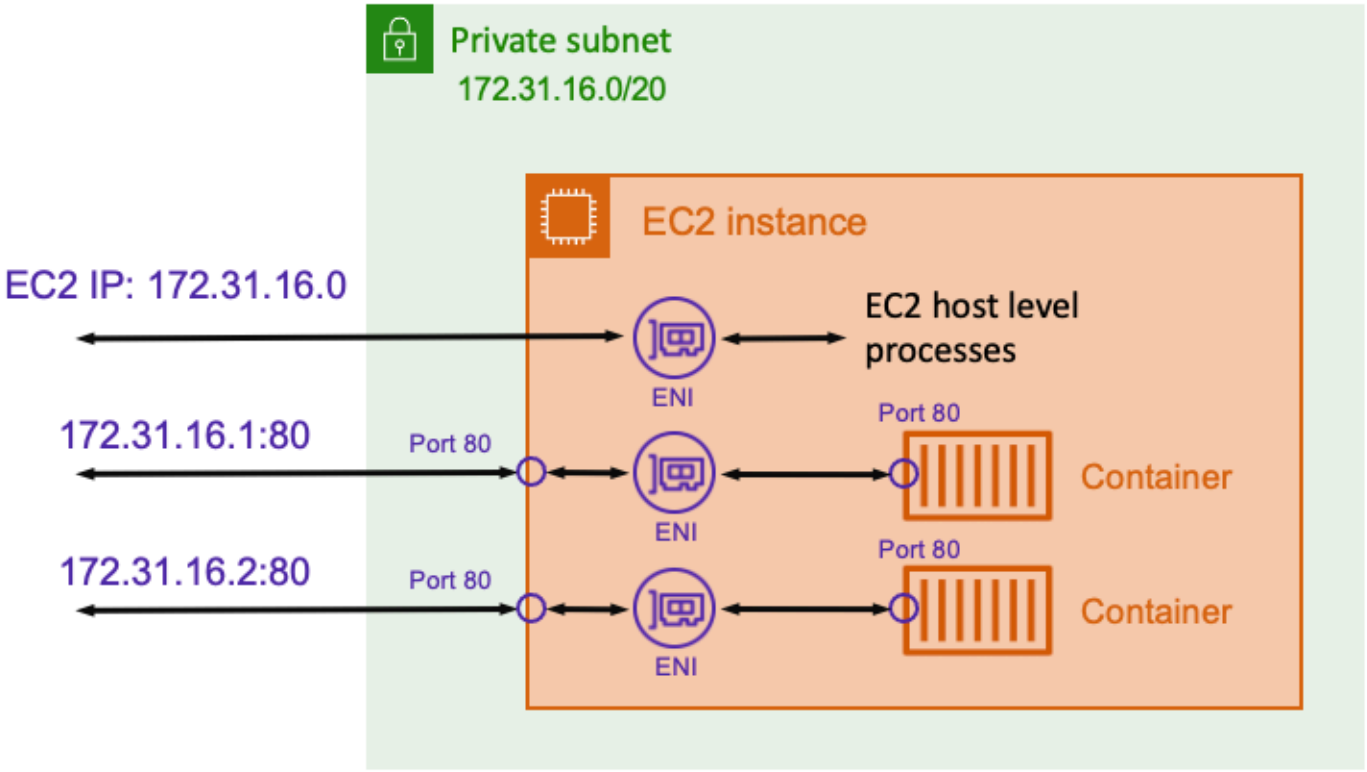
Amazon ECS를 사용하면 각 작업에 임의로 할당된 포트를 추적할 수 있습니다. 이 작업은 로드 밸런서 대상 그룹을 자동으로 업데이트하고AWS Cloud Map서비스 검색을 사용하여 작업 IP 주소 및 포트 목록을 가질 수 있습니다. 따라서 다음을 사용하여 작동하는 서비스를 더 쉽게 사용할 수 있습니다.`bridge`모드를 동적 포트와 함께 사용합니다.

그러나, 사용의 한 가지 단점은`bridge`네트워크 모드는 서비스 통신에 대한 서비스를 잠그기가 어렵다는 점입니다. 서비스는 사용되지 않는 임의의 포트에 할당될 수 있으므로 호스트 간에 광범위한 포트 범위를 열어야 합니다. 그러나 특정 서비스가 다른 특정 서비스와만 통신할 수 있도록 특정 규칙을 만드는 것은 쉽지 않습니다. 서비스에는 보안 그룹 네트워킹 규칙에 사용할 특정 포트가 없습니다.

이`bridge`네트워크 모드는 Amazon EC2 인스턴스에서 호스팅되는 Amazon ECS 작업에 대해서만 지원됩니다. Fargate 에서 Amazon ECS를 사용할 때는 지원되지 않습니다.

AWSVPC 모드

와 함께`awsvpc`네트워크 모드에서 Amazon ECS는 각 작업에 대해 ENI (엘라스틱 네트워크 인터페이스) 를 생성 및 관리하며, 각 작업은 VPC 내에서 자체 프라이빗 IP 주소를 받습니다. 이 ENI는 기본 호스트 ENI와 별개입니다. Amazon EC2 인스턴스에서 여러 작업을 실행하는 경우 각 작업의 ENI도 별개입니다.



앞의 예에서 Amazon EC2 인스턴스는 ENI에 할당됩니다. ENI는 호스트 수준에서 네트워크 통신에 사용되는 EC2 인스턴스의 IP 주소를 나타냅니다. 각 작업에는 해당 ENI 및 사설 IP 주소도 있습니다. 각 ENI는 분리되어 있으므로 각 컨테이너는 포트에 바인딩할 수 있습니다.80작업 ENI에. 따라서 포트 번호를 추적할 필요가 없습니다. 대신 포트로 트래픽을 보낼 수 있습니다.80작업 ENI의 IP 주소에서

사용의 장점은awsvpc네트워크 모드는 각 작업에 트래픽을 허용하거나 거부하는 별도의 보안 그룹이 있다는 것입니다. 즉, 보다 세분화된 수준에서 작업과 서비스 간의 통신을 제어할 수 있는 유연성이 향상됩니다. 동일한 호스트에 있는 다른 작업에서 들어오는 트래픽을 거부하도록 작업을 구성할 수도 있습니다.

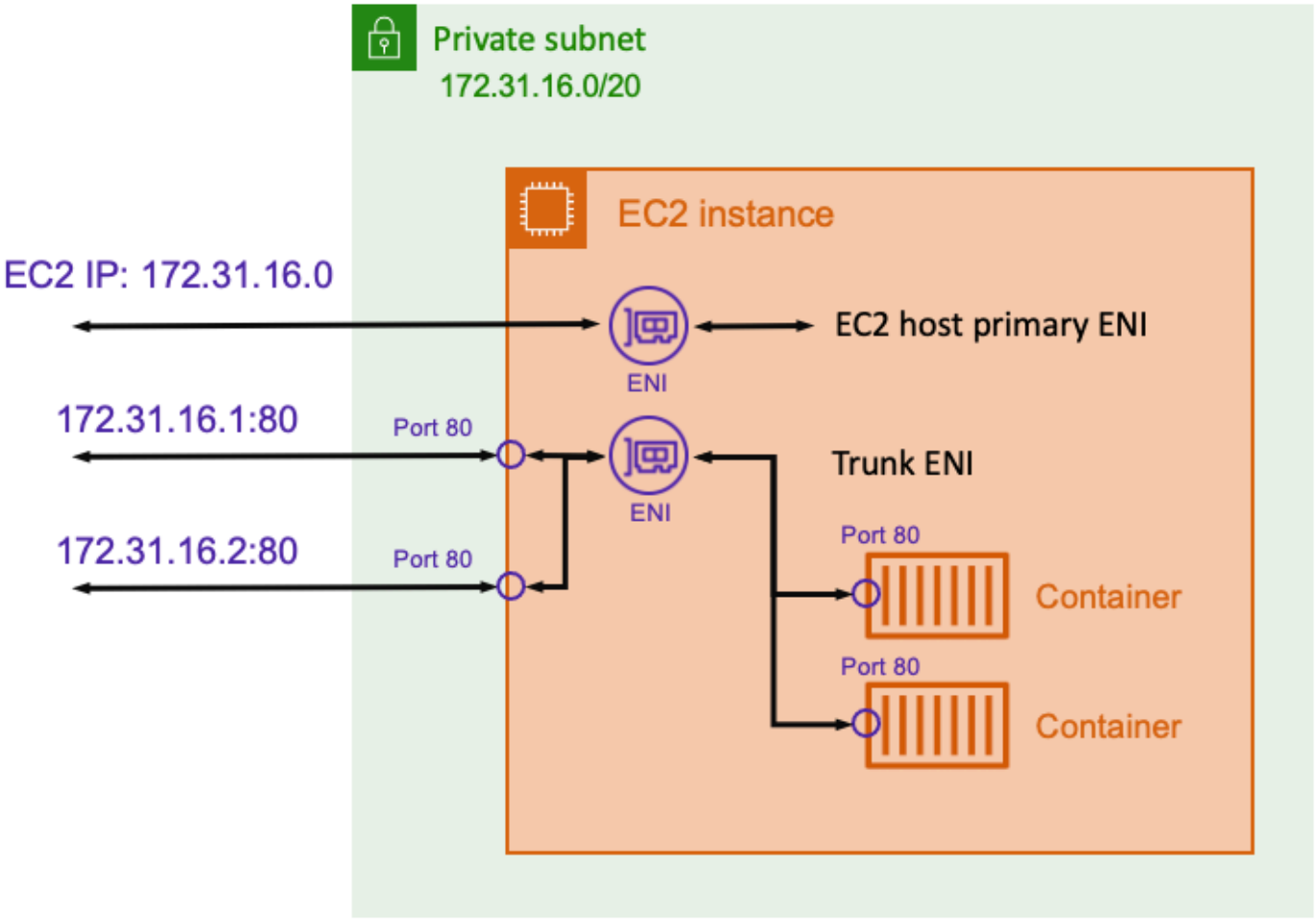
이awsvpc네트워크 모드는 Amazon EC2와 Fargate 에서 호스팅되는 Amazon ECS 작업에 대해 지원됩니다. Fargate 를 사용할 때awsvpc네트워크 모드가 필요합니다.

를 사용할 때awsvpc네트워크 모드에서는 몇 가지 문제를 염두에 두어야 합니다.

ENI 트렁킹으로 작업 밀도 향상

사용의 가장 큰 단점은awsvpc네트워크 모드를 Amazon EC2 인스턴스에서 호스팅하는 작업을 사용하는 경우 EC2 인스턴스에 연결할 수 있는 ENI 개수에는 한도가 있다는 것입니다. 이렇게 하면 각 인스

턴스에 배치할 수 있는 작업 수가 제한됩니다. Amazon ECS는 사용 가능한 ENI 수를 늘려 작업 밀도를 높일 수 있는 ENI 트렁킹 기능을 제공합니다.



ENI 트렁킹을 사용할 때 기본적으로 두 개의 ENI 첨부 파일이 사용됩니다. 첫 번째는 인스턴스의 기본 ENI이며 모든 호스트 수준 프로세스에 사용됩니다. 두 번째는 Amazon ECS가 생성하는 트렁크 ENI입니다. 이 기능은 특정 Amazon EC2 인스턴스 유형에서만 지원됩니다.

이 예를 생각해 보십시오. ENI 트렁킹이 없으면`c5.large`인스턴스는 두 개의 작업만 호스팅할 수 있습니다. 그러나 ENI 트렁킹을 사용하면`c5.large`인스턴스는 최대 10개의 작업을 호스팅할 수 있습니다. 각 작업마다 다른 IP 주소와 보안 그룹이 있습니다. 사용 가능한 인스턴스 유형 및 해당 밀도에 대한 자세한 내용은 단원을 참조하십시오.지원되는 Amazon EC2 인스턴스 유형의Amazon Elastic Container Service.

ENI 트렁킹은 지연 시간 또는 대역폭 측면에서 런타임 성능에 영향을 미치지 않습니다. 그러나 작업 시작 시간이 늘어납니다. ENI 트렁킹을 사용하는 경우 자동 크기 조정 규칙 및 작업 시작 시간에 의존하는 기타 워크로드가 예상대로 작동하는지 확인해야 합니다.

자세한 내용은 단원을 참조하십시오.탄력적 네트워크 인터페이스 트렁킹이 있음의Amazon Elastic Container Service.

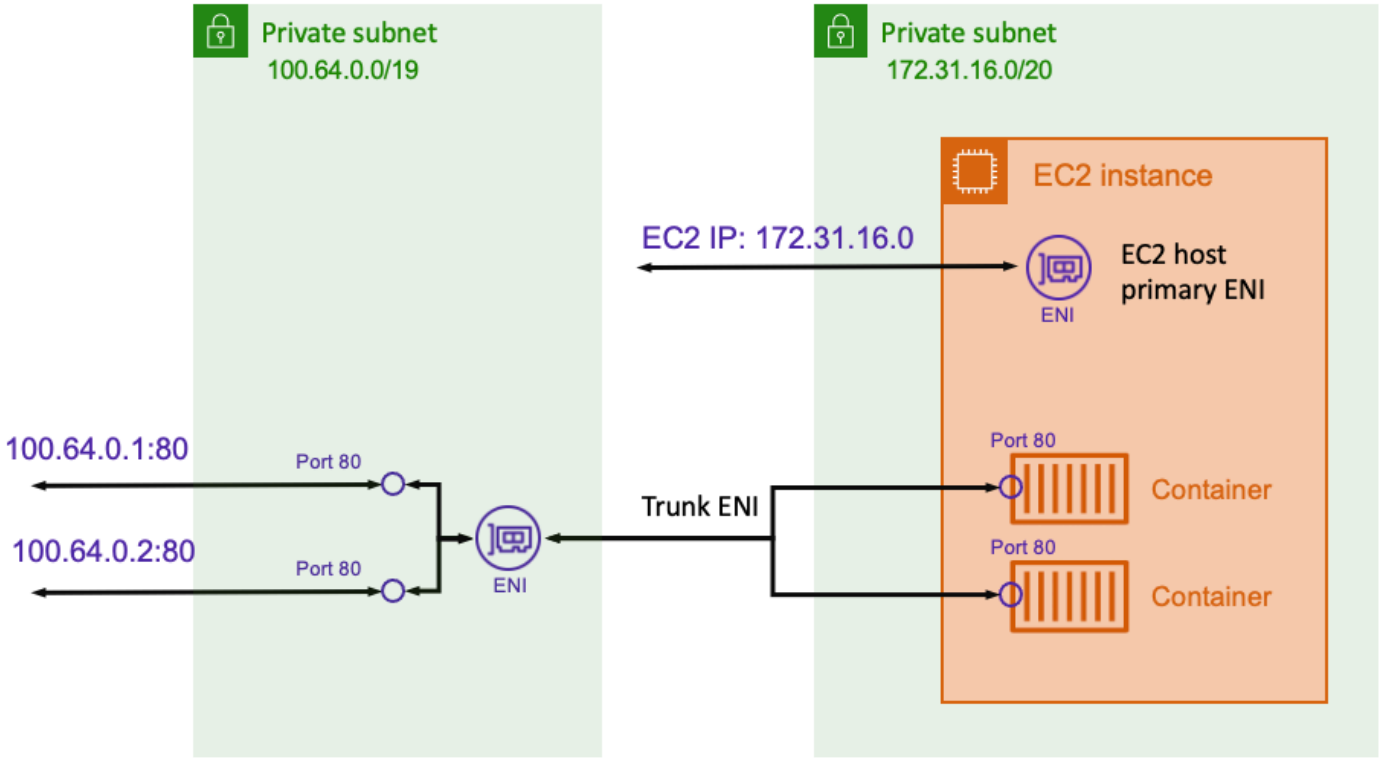
IP 주소 소모 방지

각 작업에 별도의 IP 주소를 할당하면 전체 인프라를 단순화하고 뛰어난 수준의 보안을 제공하는 보안 그룹을 유지 관리할 수 있습니다. 그러나 이 구성은 IP 소진으로 이어질 수 있습니다.

사용자의 기본 VPCAWS계정에 미리 프로비저닝된 서브넷이 있는/20CIDR 범위 즉, 각 서브넷에는 4,091개의 사용 가능한 IP 주소가 있습니다. 내 여러 IP 주소 참고/20범위는 AWS 전용 용도를 위해 예약되어 있습니다. 이 예를 생각해 보십시오. 고가용성을 위해 3개의 가용 영역에 있는 3개의 서브넷에 애플리케이션을 분산합니다. 이 경우 3개 서브넷에서 약 12,000개의 IP 주소를 사용할 수 있습니다.

ENI 트렁킹을 사용하면 시작하는 각 Amazon EC2 인스턴스에는 두 개의 IP 주소가 필요합니다. 하나의 IP 주소는 기본 ENI에 사용되고 다른 IP 주소는 트렁크 ENI에 사용됩니다. 인스턴스의 각 Amazon ECS 작업에는 하나의 IP 주소가 필요합니다. 매우 큰 워크로드를 시작하는 경우 사용 가능한 IP 주소가 부족할 수 있습니다. 이로 인해 Amazon EC2 시작 실패 또는 작업 시작 실패가 발생할 수 있습니다. 이러한 오류는 사용 가능한 IP 주소가 없는 경우 ENI가 VPC 내에 IP 주소를 추가할 수 없기 때문에 발생합니다.

를 사용할 때awsvpc네트워크 모드를 사용하려면 IP 주소 요구 사항을 평가하고 서브넷 CIDR 범위가 요구 사항을 충족하는지 확인해야 합니다. 서브넷이 작고 주소 공간이 부족하기 시작하는 VPC PC를 이미 사용하기 시작한 경우 보조 서브넷을 추가할 수 있습니다.



ENI 트렁킹을 사용하면 호스트와 다른 IP 주소 공간에서 ENI를 사용하도록 Amazon VPC CNI를 구성할 수 있습니다. 이렇게 하면 Amazon EC2 호스트와 작업에 중복되지 않는 다른 IP 주소 범위를 제공할 수 있습니다. 예제 다이어그램에서 EC2 호스트 IP 주소는`172.31.16.0/20`IP 범위 그러나 호스트에서 실행 중인 작업에는`100.64.0.0/19`범위. 두 개의 독립적인 IP 범위를 사용하면 너무 많은 IP 주소를 소비하고 인스턴스에 충분한 IP 주소를 남기지 않는 작업에 대해 걱정할 필요가 없습니다.

IPv6 이중 스택 모드 사용

이`awsvpc`네트워크 모드는 IPv6 이중 스택 모드에 대해 구성된 VPC와 호환됩니다. 듀얼 스택 모드를 사용하는 VPC 는 IPv4, IPv6 또는 둘 다를 통해 통신할 수 있습니다. VPC 각 서브넷은 IPv4 CIDR 범위와 IPv6 CIDR 범위를 둘 다 가질 수 있습니다. 자세한 내용은 단원을 참조하십시오.VPC IP 주소 지정의Amazon VPC 사용 설명서.

VPC 및 서브넷에 대한 IPv4 지원을 비활성화하여 IPv4 소모 문제를 해결할 수 없습니다. 그러나 IPv6 지원을 사용하면 외부 전용 인터넷 게이트웨이와 같은 몇 가지 새로운 기능을 사용할 수 있습니다. 외부 전용 인터넷 게이트웨이에서는 공개적으로 라우팅되는 IPv6 주소를 사용하여 인터넷에 대한 아웃바운드 연결을 시작할 수 있습니다. 그러나 외부 전용 인터넷 게이트웨이에서는 인터넷 연결을 허용하지 않습니다. 자세한 내용은 단원을 참조하십시오.외부 전용 인터넷 게이트웨이의Amazon VPC 사용 설명서.

에 연결AWS서비스를 VPC 내부에서

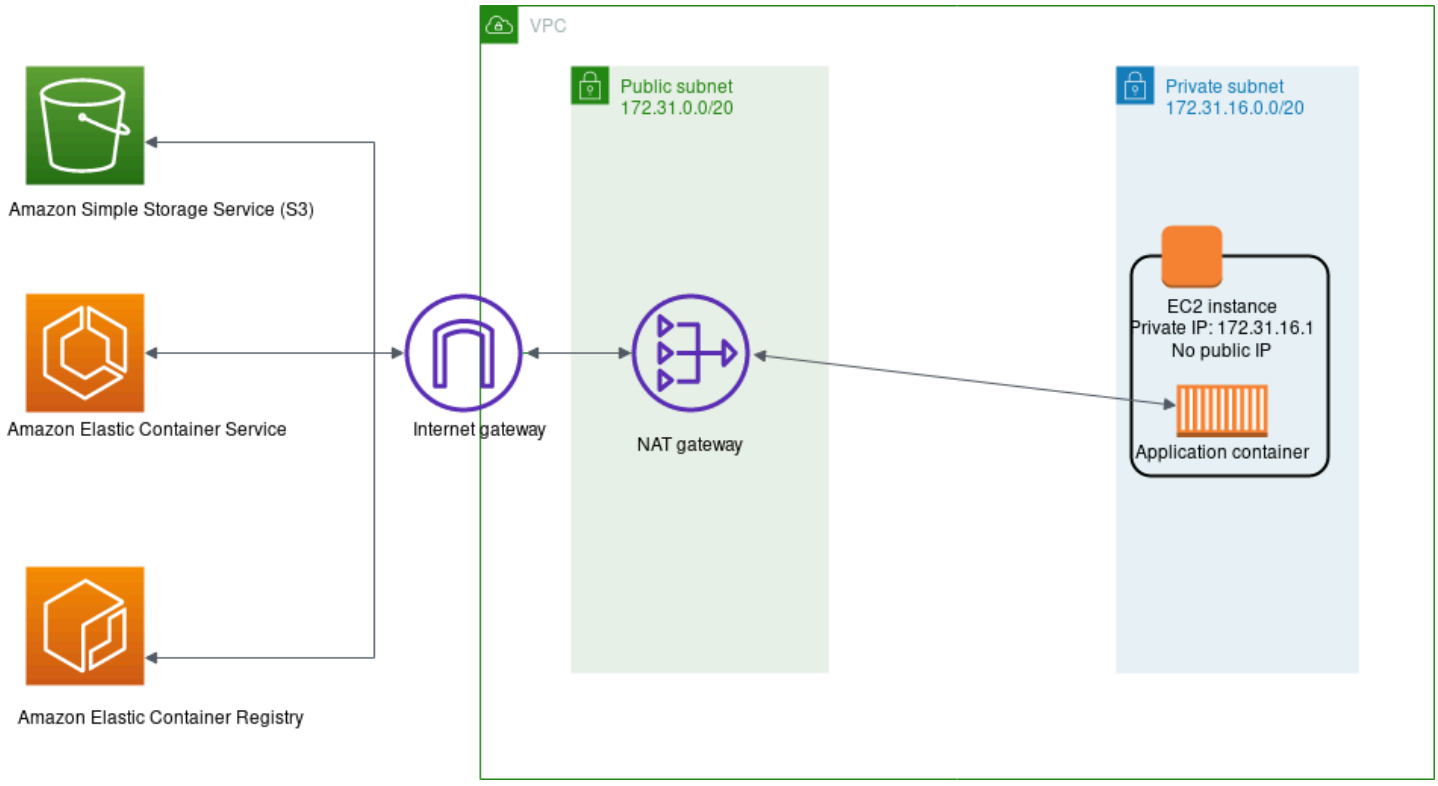
Amazon ECS가 제대로 작동하려면 각 호스트에서 실행되는 ECS 컨테이너 에이전트가 Amazon ECS 제어부와 통신해야 합니다. 컨테이너 이미지를 Amazon ECR에 저장하는 경우 Amazon EC2 호스트는 Amazon ECR 서비스 엔드포인트 및 이미지 레이어가 저장된 Amazon S3 와 통신해야 합니다. 다른 것을 사용하는 경우AWS서비스의 경우 (예: DynamoDB 에 저장된 데이터 유지) 이러한 서비스에 필요한 네트워킹 지원이 있는지 다시 확인합니다.

주제

- NAT 게이트웨이
- AWS PrivateLink

NAT 게이트웨이

NAT 게이트웨이를 사용하면 Amazon ECS 작업이 다른AWS서비스. 이 방법에 대한 자세한 내용은 단원을 참조하십시오.프라이빗 서브넷 및 NAT 게이트웨이 사용.



이 방법을 사용할 때의 단점은 다음과 같습니다.

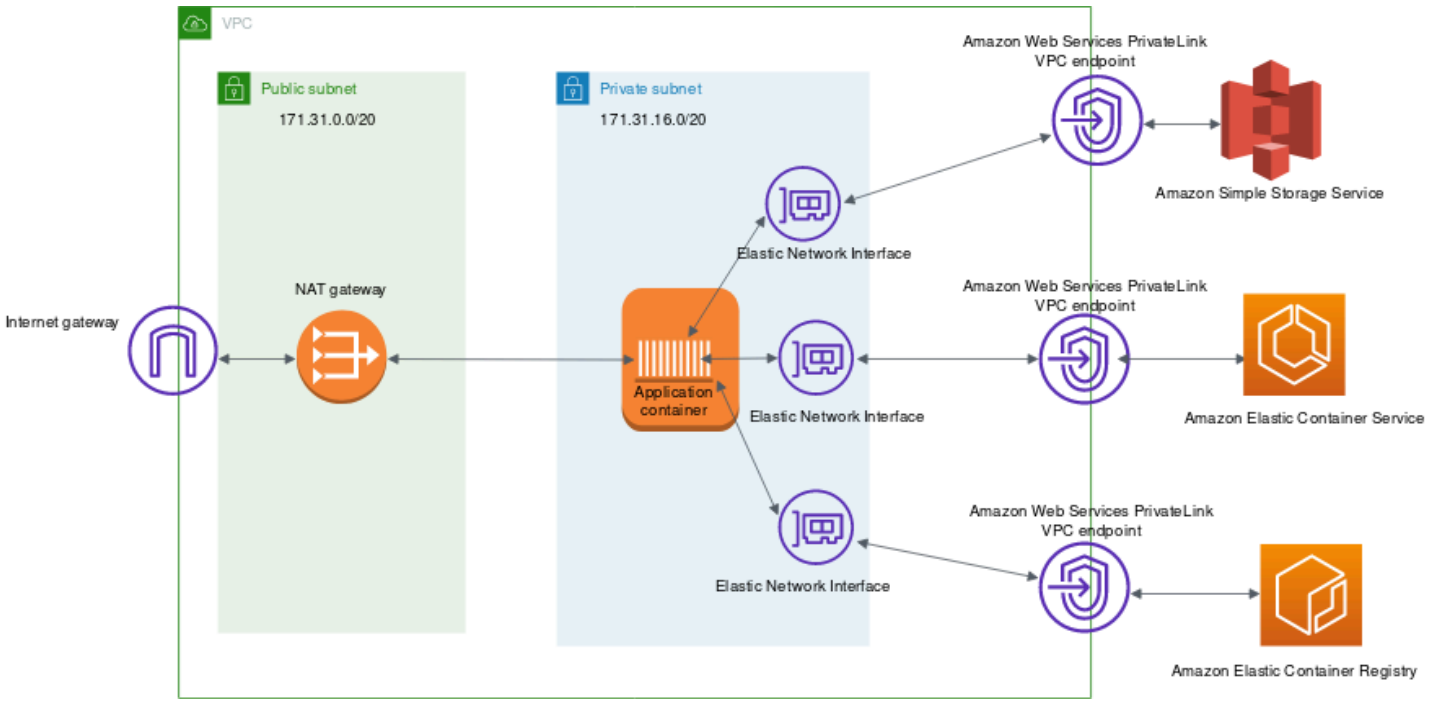
- NAT 게이트웨이가 통신할 수 있는 대상을 제한할 수 없습니다. 또한 VPC 로부터의 모든 아웃바운드 통신을 중단하지 않고 백엔드 타이어가 통신할 수 있는 대상을 제한할 수 없습니다.
- NAT 게이트웨이는 통과하는 모든 데이터 GB에 대해 요금을 청구합니다. Amazon S3 에서 대용량 파일을 다운로드하기 위해 NAT 게이트웨이를 사용하거나 DynamoDB 에 대량의 데이터베이스 쿼리를 수행하는 경우 대역폭 GB당 요금이 청구됩니다. 또한 NAT 게이트웨이는 5Gbps의 대역폭을 지원하며 최대 45Gbps까지 자동 확장합니다. 단일 NAT 게이트웨이를 통해 라우팅하는 경우 매우 높은 대역폭 연결이 필요한 응용 프로그램에는 네트워킹 제약이 발생할 수 있습니다. 이 문제를 해결하려면 워크로드를 여러 서브넷으로 나누고 각 서브넷에 고유한 NAT 게이트웨이를 제공할 수 있습니다.

AWS PrivateLink

AWS PrivateLink는 VPC 간 전용 연결을 제공하며,AWS서비스 및 온-프레미스 네트워크를 통해 트래픽을 공용 인터넷에 노출하지 않습니다.

이를 수행하는 데 사용되는 기술 중 하나는 VPC 엔드포인트입니다. VPC 엔드포인트는 VPC 엔드포인트와 VPC 간 프라이빗 연결을 활성화합니다.AWS서비스 및 VPC 엔드포인트 서비스를 구성합니다. VPC와 기타 서비스 간의 트래픽은 Amazon 네트워크를 벗어나지 않습니다. VPC 엔드포인트에는 인터넷 게이트웨이, 가상 프라이빗 게이트웨이, NAT 디바이스, VPN 연결 또는AWS Direct Connect연결. VPC Amazon EC2 인스턴스는 서비스의 리소스와 통신하는 데 퍼블릭 IP 주소를 필요로 하지 않습니다.

다음 다이어그램은 통신 방법을 보여 줍니다.AWS서비스는 인터넷 게이트웨이 대신 VPC 엔드포인트를 사용할 때 작동합니다.AWS PrivateLink는 서브넷 내부의 ENI (엘라스틱 네트워크 인터페이스) 를 프로비저닝하고 VPC 라우팅 규칙은 ENI를 통해 서비스 호스트 이름으로 모든 통신을 대상으로 직접 전송하는 데 사용됩니다.AWS서비스를 선택합니다. 이 트래픽은 더 이상 NAT 게이트웨이 또는 인터넷 게이트웨이를 사용할 필요가 없습니다.



다음은 Amazon ECS 서비스와 함께 사용되는 일반적인 VPC 엔드포인트 중 일부입니다.

- S3 게이트웨이 VPC 엔드포인트
- DynamoDB VPC 엔드포인트
- Amazon ECS VPC 엔드포인트
- Amazon ECR VPC 엔드포인트

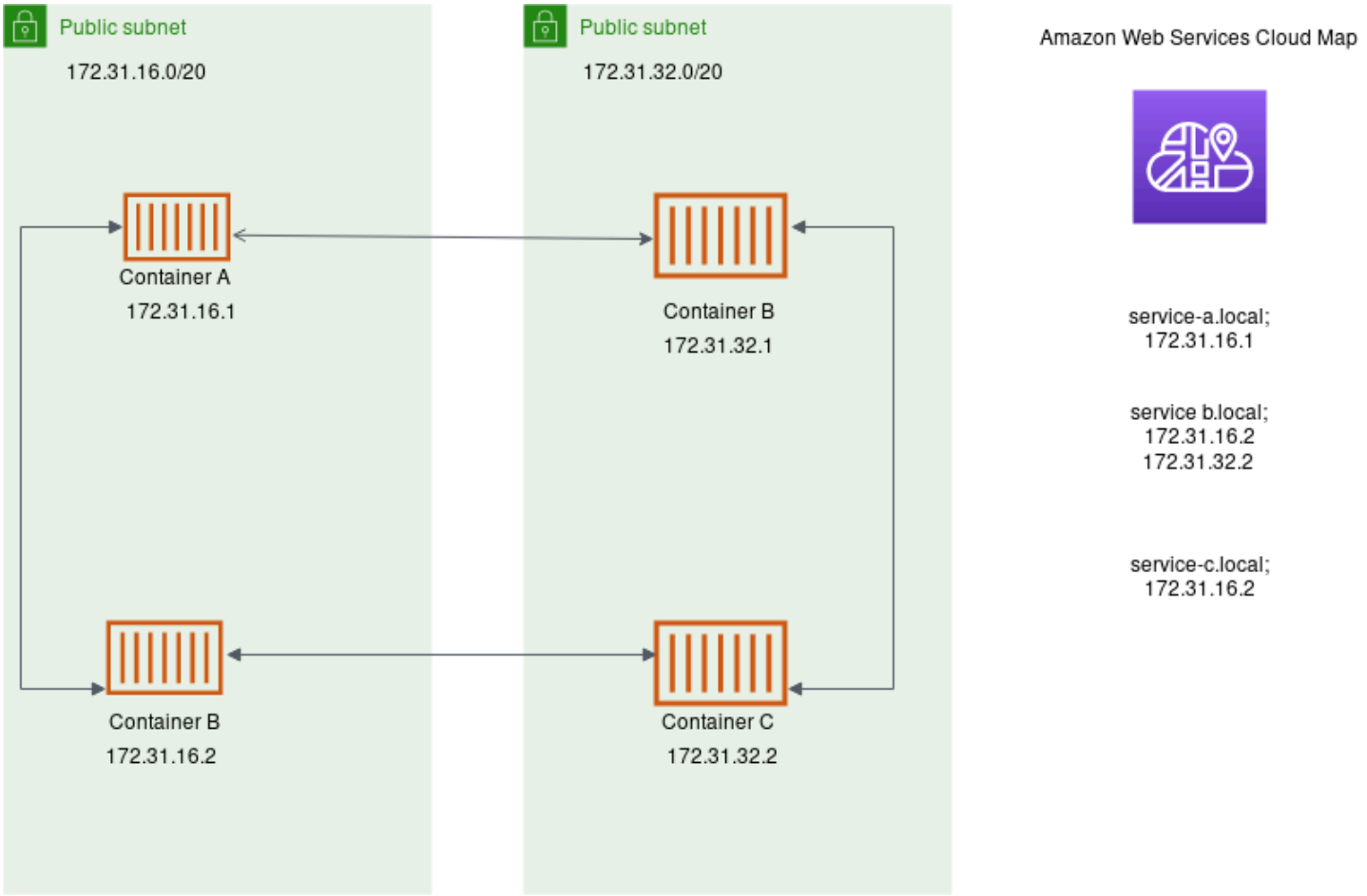
그 외 다수AWS서비스에서는 VPC 엔드포인트를 지원합니다. 당신이 어떤 것을 많이 사용한다면AWS서비스를 사용하려면 해당 서비스에 대한 특정 설명서와 해당 트래픽에 대한 VPC 엔드포인트를 만드는 방법을 찾아야 합니다.

VPC Amazon ECS 서비스 간 네트워킹

VPC Amazon ECS 컨테이너를 사용하면 모놀리식 애플리케이션을 별도의 부분으로 분산시켜 안전한 환경에서 독립적으로 배포 및 확장할 수 있습니다. 그러나 VPC 안팎에서 이러한 모든 부분이 서로 통신할 수 있는지 확인하는 것은 어려울 수 있습니다. 서로 다른 장점과 단점을 가진 의사 소통을 촉진하기위한 몇 가지 방법이 있습니다.

서비스 검색 사용

서비스 대 서비스 통신의 한 가지 방법은 서비스 검색을 사용한 직접 통신입니다. 이 방법에서는 사용할 수 있는AWS Cloud Map서비스 검색을 Amazon ECS와 통합합니다. Amazon ECS는 서비스 검색을 사용하여 시작된 작업 목록을AWS Cloud Map는 특정 서비스의 하나 이상의 작업에 대한 내부 IP 주소로 확인되는 DNS 호스트 이름을 유지 관리합니다. Amazon VPC 다른 서비스는 이 DNS 호스트 이름을 사용하여 내부 IP 주소를 사용하여 트래픽을 다른 컨테이너로 직접 전송할 수 있습니다. 자세한 내용은 단원을 참조하십시오.서비스 검색의Amazon Elastic Container Service.



위 다이어그램에서 세 가지 서비스가 있습니다.`serviceA`에는 하나의 컨테이너가 있고`serviceB`에는 두 개의 컨테이너가 있습니다.`serviceB`와 통신해야 합니다.`serviceC`에는 하나의 컨테이너가 있습니다. 이러한 세 가지 서비스 모두의 각 컨테이너는AWS Cloud Map를 사용하여 통신해야하는 다운스트림 서비스에서 컨테이너의 내부 IP 주소를 찾습니다.

서비스 간 통신에 대한 이러한 접근 방식은 짧은 대기 시간을 제공합니다. 언뜻보기에는 컨테이너 사이에 추가 구성 요소가 없으므로 간단합니다. 트래픽은 한 컨테이너에서 다른 컨테이너로 직접 이동합니다.

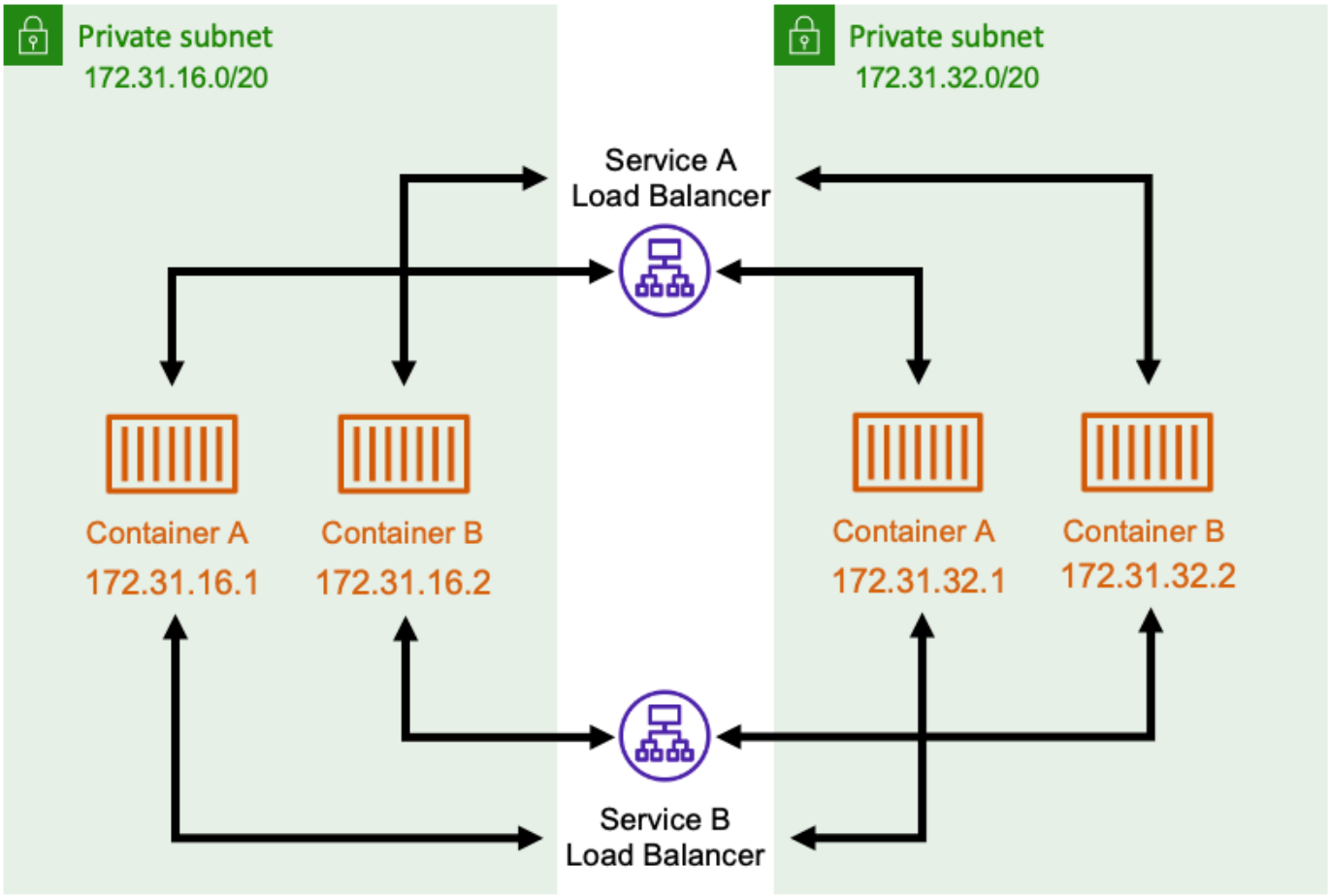
이 방법은 사용할 때 적합합니다awsvpc네트워크 모드입니다. 각 작업에는 고유 한 IP 주소가 있습니다. 대부분의 소프트웨어는 DNS 사용만을 지원합니다.A레코드는 IP 주소로 직접 해석됩니다. 를 사용할 때awsvpc네트워크 모드에서 각 작업의 IP 주소는A레코드. 그러나 를 사용하는 경우bridge네트워크 모드에서 여러 컨테이너가 동일한 IP 주소를 공유 할 수 있습니다. 또한 동적 포트 매핑으로 인해 컨테이너에 해당 단일 IP 주소에서 포트 번호가 임의로 할당됩니다. 이 시점에서A레코드가 더 이상 서비스 검색에 충분하지 않습니다. 또한 사용 해야 합니다.SRV레코드. 이러한 유형의 레코드는 IP 주소와 포트 번호를 모두 추적할 수 있지만 응용 프로그램을 적절히 구성해야 합니다. 사용하는 일부 사전 빌드된 응용 프로그램은SRV레코드.

의 또 다른 장점은awsvpc네트워크 모드는 각 서비스에 대해 고유한 보안 그룹을 가지고 있다는 것입니다. 해당 서비스와 통신해야 하는 특정 업스트림 서비스에서 들어오는 연결만 허용하도록 이 보안 그룹을 구성할 수 있습니다.

서비스 검색을 사용하는 직접적인 서비스 간 통신의 주된 단점은 재시도를 하고 연결 실패를 처리하기 위해 추가 논리를 구현해야 한다는 것입니다. DNS 레코드는 TTL (Time TTTTL) 기간을 통해 캐싱되는 기간을 제어할 수 있습니다. 애플리케이션이 DNS 레코드의 최신 버전을 선택할 수 있도록 DNS 레코드가 업데이트되고 캐시가 만료되려면 약간의 시간이 걸립니다. 따라서 응용 프로그램이 더 이상 존재하지 않는 다른 컨테이너를 가리 키도록 DNS 레코드를 해결 할 수 있습니다. 응용 프로그램은 재시도를 처리하고 잘못된 백엔드를 무시하는 논리가 있어야합니다.

내부 로드 밸런서 사용

서비스 간 통신에 대한 또 다른 방법은 내부 부하 분산 장치를 사용하는 것입니다. 내부 로드 밸런서는 전적으로 VPC 내부에 존재하며 VPC 내부의 서비스에서만 액세스할 수 있습니다.



로드 밸런서는 중복 리소스를 각 서브넷에 배포하여 고가용성을 유지합니다. 때 컨테이너에 서serviceA에서 컨테이너와 통신해야합니다.serviceB로 설정하면 로드 밸런서에 대한 연결을 엽니다. 그런 다음 로드 밸런서는 컨테이너에 대한 연결을service B. 로드 밸런서는 각 서비스 간의 모든 연결을 관리하기 위한 중앙 집중화된 장소 역할을 수행합니다.

에서 컨테이너의 경우serviceB가 중지되면 로드 밸런서가 풀에서 해당 컨테이너를 제거 할 수 있습니다. 또한 로드 밸런서는 풀의 각 다운스트림 대상에 대해 상태 확인을 수행하고 다시 정상 상태가 될 때까지 풀에서 잘못된 대상을 자동으로 제거할 수 있습니다. 응용 프로그램은 더 이상 다운스트림 컨테이너가 몇 개인지 알 필요가 없습니다. 로드 밸런서에 대한 연결을 열기만 하면 됩니다.

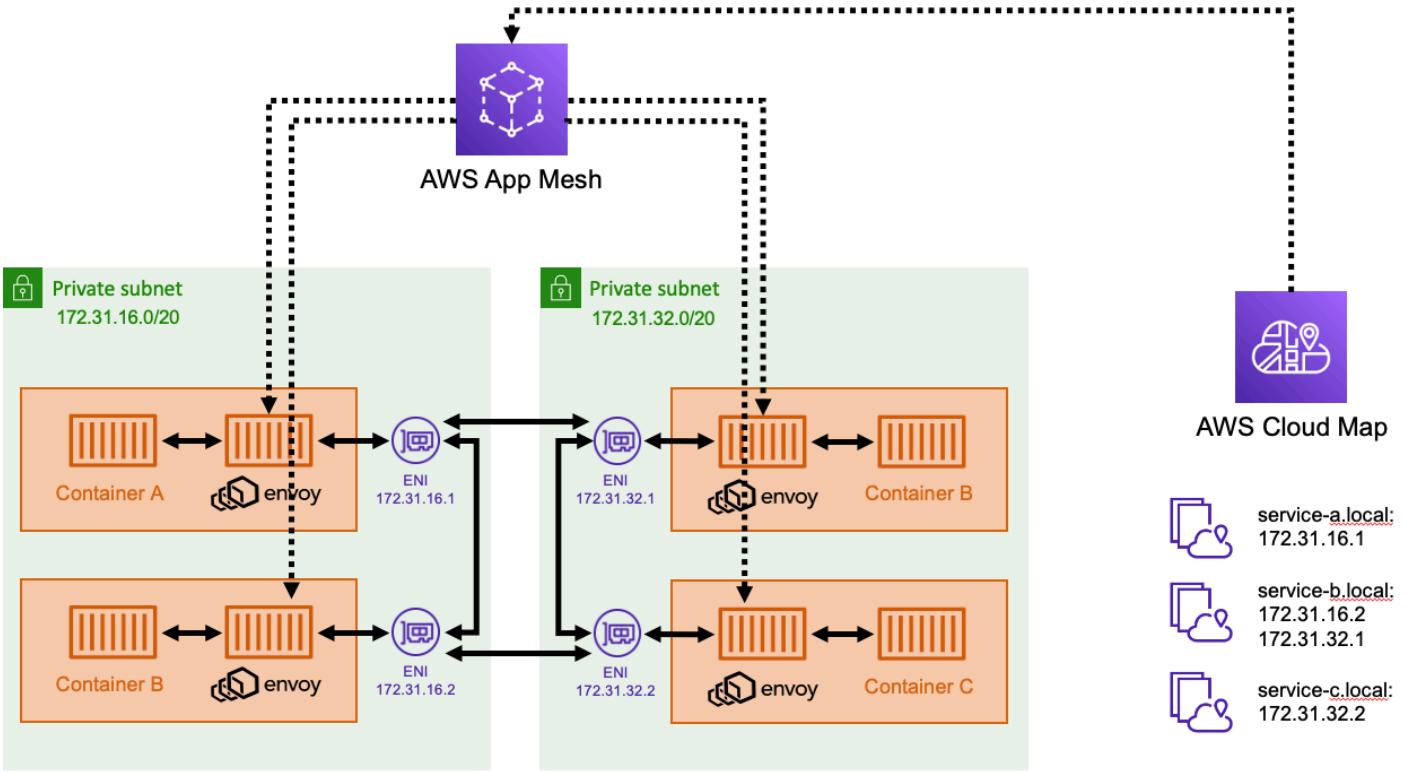
이 방법은 모든 네트워크 모드에 유리합니다. 로드 밸런서는 작업 IP 주소를 추적할 수 있습니다.awsvpc네트워크 모드를 사용할 때 IP 주소와 포트의 고급 조합뿐만 아니라bridge네트워크 모드. 여러 컨테이너가 실제로 동일한 Amazon EC2 인스턴스에서 다른 포트에서 호스팅되는 경우에도 모든 IP 주소와 포트 조합에 걸쳐 트래픽을 균등하게 분산합니다.

이 접근법의 한 가지 단점은 비용입니다. 고가용성을 유지하려면 로드 밸런서에 각 가용 영역에 리소스가 있어야 합니다. 이렇게 하면 로드 밸런서를 지불하는 오버헤드와 로드 밸런서를 통과하는 트래픽 양 때문에 추가 비용이 추가됩니다.

그러나 여러 서비스가 하나의 로드 밸런서를 공유하도록 함으로써 오버헤드 비용을 줄일 수 있습니다. 이는 Application Load Balancer 서를 사용하는 REST 서비스에 특히 적합합니다. 트래픽을 서로 다른 서비스로 라우팅하는 경로 기반 라우팅 규칙을 생성할 수 있습니다. 예,`/api/user/*`의 일부인 컨테이너로 라우팅 할 수 있습니다.`user`서비스, 반면`/api/order/*`로 라우팅할 수 있는`order`서비스를 선택합니다. 이 방법을 사용하면 하나의 Application Load Balancer 대해서만 비용을 지불하고 API에 대해 하나의 일관된 URL을 보유할 수 있습니다. 그러나 백엔드의 다양한 마이크로 서비스로 트래픽을 분할 할 수 있습니다.

서비스 메시 사용

AWS App Mesh는 많은 수의 서비스를 관리하고 서비스 간에 트래픽이 라우팅되는 방식을 더 잘 제어할 수 있는 서비스 메시입니다. App Mesh 는 기본 서비스 검색과 부하 분산 간의 중개자 역할을 합니다. App Mesh 사용하면 응용 프로그램이 서로 직접 상호 작용하지 않지만 중앙 집중식 부하 분산 장치도 사용하지 않습니다. 대신 작업의 각 복사본에는 Envoy 프록시 사이드카가 함께 제공됩니다. 자세한 내용은 단원을 참조하십시오.란 무엇입니까?AWS App Mesh의AWS App Mesh사용 설명서.



위 다이어그램에서 각 작업에는 Envoy 프록시 사이드카가 있습니다. 이 사이드카는 작업에 대한 모든 인바운드 및 아웃바운드 트래픽을 프록시합니다. App Mesh 컨트롤 플레인은AWS Cloud Map를 사용하여 사용 가능한 서비스 목록과 특정 작업의 IP 주소를 가져올 수 있습니다. 그런 다음 App Mesh 는 Envoy 프록시 사이드카에 구성을 전달합니다. 이 구성에는 연결할 수있는 사용 가능한 컨테이너 목록이 포함됩니다. 또한 Envoy 프록시 사이드카는 각 대상에 대해 상태 확인을 수행하여 사용 가능한지 확인합니다.

이 접근 방식은 관리 부하 분산 장치의 용이성과 함께 서비스 검색 기능을 제공합니다. Envoy 프록시 사이드카가 해당 부하 분산을 처리하므로 응용 프로그램은 코드 내에 많은 부하 분산 논리를 구현하지 않습니다. Envoy 프록시는 실패를 감지하고 실패한 요청을 다시 시도하도록 구성할 수 있습니다. 또한 MTL을 사용하여 전송 중인 트래픽을 암호화하고 응용 프로그램이 확인된 대상과 통신하고 있는지 확인하도록 구성할 수도 있습니다.

Envoy 프록시와 로드 밸런서 간에는 몇 가지 차이점이 있습니다. 즉, Envoy 프록시를 사용하면 자체 Envoy 프록시 사이드카를 배포하고 관리할 책임이 있습니다. Envoy 프록시 사이드카는 사용자가 Amazon ECS 작업에 할당하는 일부 CPU와 메모리를 사용합니다. 이렇게 하면 작업 리소스 소비에 약간의 오버헤드가 추가되고 필요할 때 프록시를 유지 관리하고 업데이트하기 위한 추가 운영 워크로드가 추가됩니다.

App Mesh 및 Envoy 프록시를 사용하면 작업 간 대기 시간이 매우 짧습니다. 이는 Envoy 프록시가 각 작업에 배치되기 때문입니다. 하나의 Envoy 프록시와 다른 Envoy 프록시 사이에는 네트워크 점프를 인스턴스화하는 인스턴스가 하나만 있습니다. 즉, 로드 밸런서를 사용할 때와 비교하여 네트워크 오버헤드가 적습니다. 로드 밸런서를 사용할 때 두 개의 네트워크 점프가 있습니다. 첫 번째는 업스트림 작업에서 로드 밸런서로, 두 번째는 로드 밸런서에서 다운스트림 작업으로 이어집니다.

전반의 네트워킹 서비스AWS계정 및 VPC

여러 팀과 부서가 있는 조직의 일원인 경우 서비스를 별도의 VPC에 독립적으로 배포하여 공유AWS계정 또는 여러 개인과 연결된 VPC로AWS계정. 서비스를 배포하는 방법에 관계없이 VPC 간에 트래픽을 라우팅하는 데 도움이 되도록 네트워킹 구성 요소를 보완하는 것이 좋습니다. 이를 위해 몇 가지AWS서비스를 사용하여 기존 네트워킹 구성 요소를 보완할 수 있습니다.

- AWS Transit Gateway — 먼저 이 네트워킹 서비스를 고려해야 합니다. 이 서비스는 Amazon VPC 간에 연결을 라우팅하기 위한 중앙 허브 역할을 합니다.AWS계정 및 온-프레미스 네트워크를 사용할 수 있습니다. 자세한 내용은 단원을 참조하십시오.전송 게이트웨이란 무엇입니까?의Amazon VPC 전송 게이트웨이 가이드.

- Amazon VPC 및 VPN 지원 — 이 서비스를 사용하여 온프레미스 네트워크를 VPC에 연결하기 위한 사이트 간 VPN 연결을 생성할 수 있습니다. 자세한 내용은 단원을 참조하십시오.란 무엇입니까? AWS Site-to-Site VPN?의AWS Site-to-Site VPN사용 설명서.
- Amazon VPC — Amazon VPC 피링을 사용하여 동일한 계정 또는 여러 계정에 있는 여러 VPC를 연결할 수 있습니다. 자세한 내용은 단원을 참조하십시오.VPC 피어링이란?의Amazon VPC 피어링 가이드.
- 공유 VPC — VPC 및 VPC 서브넷을 여러AWS계정. 자세한 내용은 단원을 참조하십시오.공유된 VPC에 대한 작업의Amazon VPC 사용 설명서.

최적화 및 문제 해결

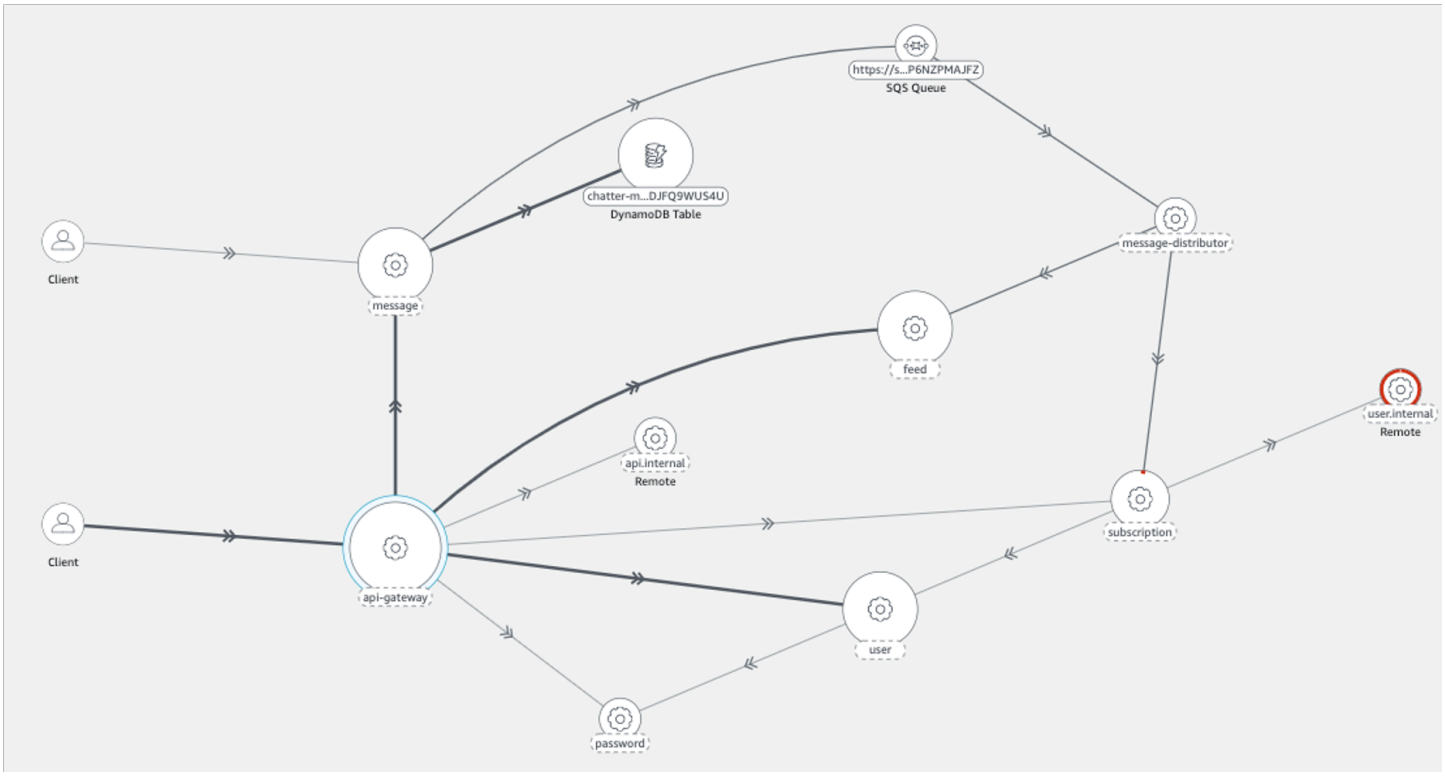
다음 서비스와 기능은 네트워크 및 서비스 구성에 대한 통찰력을 얻는 데 도움이 될 수 있습니다. 이 정보를 사용하여 네트워킹 문제를 해결하고 서비스를 최적화할 수 있습니다.

CloudWatch 컨테이너 인사이트

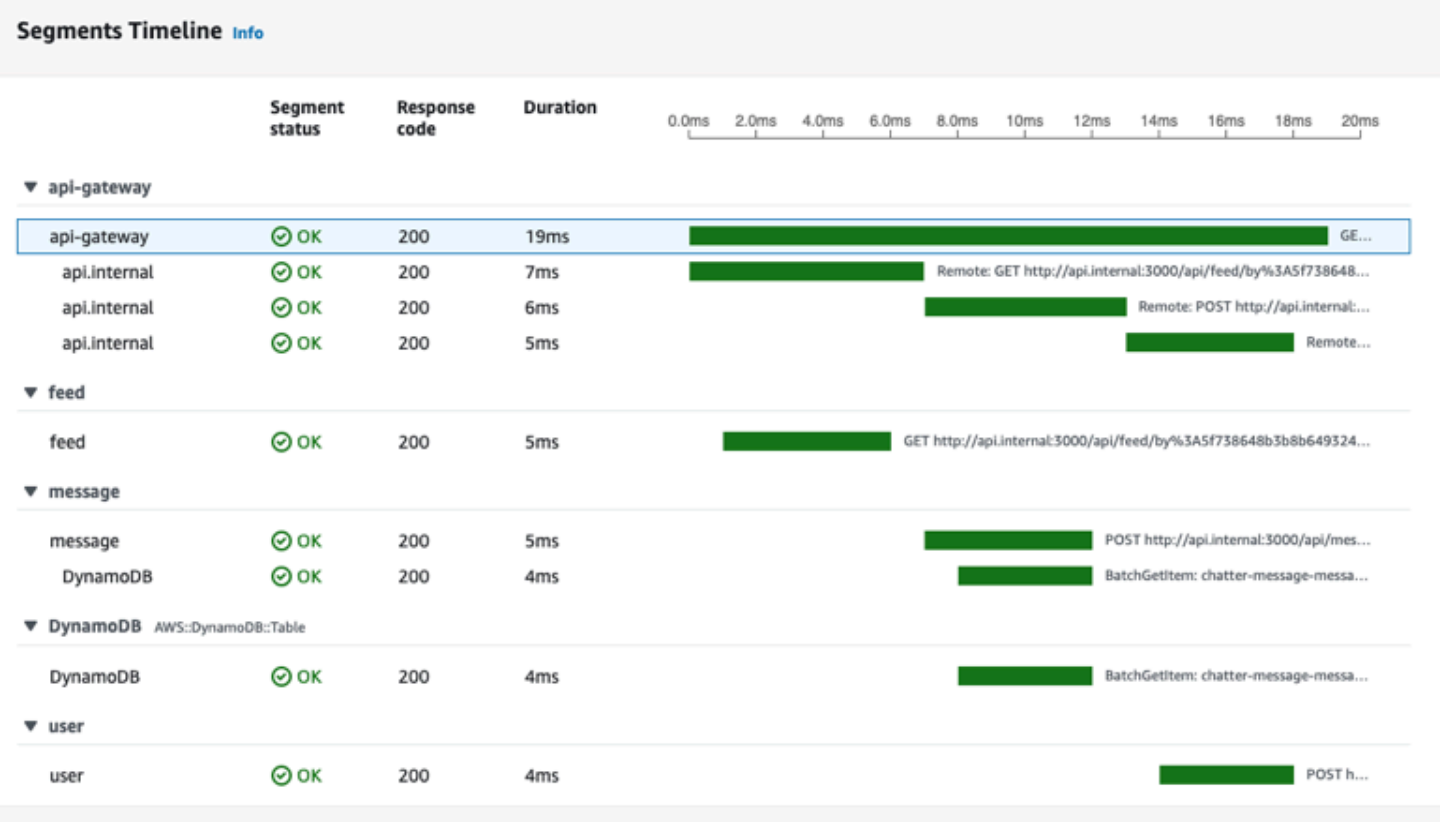
CloudWatch 컨테이너 인사이트는 컨테이너식 애플리케이션 및 마이크로서비스의 지표 및 로그를 수집하고 종합하며 요약합니다. 지표에는 CPU, 메모리, 디스크, 네트워크 같은 리소스 사용률이 포함되어 있습니다. CloudWatch 자동 대시보드에서 사용할 수 있습니다. 자세한 내용은 단원을 참조하십시오.Amazon ECS에서 컨테이너 인사이트 설정의Amazon CloudWatch 사용 설명서.

AWS X-Ray

AWS X-Ray는 응용 프로그램에서 수행하는 네트워크 요청에 대한 정보를 수집하는 데 사용할 수 있는 추적 서비스입니다. SDK를 사용하여 애플리케이션을 계측하고 서비스 간, 서비스 및 서비스 간 트래픽의 타이밍 및 응답 코드를 캡처할 수 있습니다.AWS서비스 엔드포인트를 선택합니다. 자세한 내용은 단원을 참조하십시오.란 무엇입니까?AWS X-Ray의AWS X-Ray개발자 안내서.



또한 탐색 할 수 있습니다AWS X-Ray서비스가 서로 네트워크로 연결되는 방식에 대한 그래프. 또는 이를 사용하여 각 서비스 간 링크가 수행되는 방식에 대한 집계 통계를 탐색할 수 있습니다. 마지막으로 특정 트랜잭션을 자세히 살펴보고 네트워크 호출을 나타내는 세그먼트가 특정 트랜잭션과 어떻게 연관되어 있는지 확인할 수 있습니다.



이러한 기능을 사용하여 네트워킹 병목 현상이 있는지 또는 네트워크 내의 특정 서비스가 예상대로 작동하지 않는지 확인할 수 있습니다.

VPC 흐름 로그

Amazon VPC 흐름 로그를 사용하여 네트워크 성능을 분석하고 연결 문제를 디버깅할 수 있습니다. VPC 흐름 로그를 활성화하면 VPC에 있는 모든 연결의 로그를 캡처할 수 있습니다. 여기에는 Elastic Load Balancing, Amazon RDS, NAT 게이트웨이 및 기타 키와 연결된 네트워킹 인터페이스에 대한 연결이 포함됩니다.AWS서비스를 사용할 수 있습니다. 자세한 내용은 단원을 참조하십시오.VPC 흐름 로그의Amazon VPC 사용 설명서.

네트워크 튜닝 팁

네트워킹을 개선하기 위해 미세 조정할 수 있는 몇 가지 설정이 있습니다.

Ulimit 없음

응용 프로그램의 트래픽이 많고 동시 연결을 많이 처리해야 하는 경우 허용되는 파일 수에 대한 시스템 할당량을 고려해야 합니다. 열린 네트워크 소켓이 많으면 각 소켓은 파일 설명자로 표시되어야합니다. 파일 설명자 할당량이 너무 낮으면 네트워크 소켓이 제한됩니다. 이로 인해 연결 실패 또는 오류가

발생합니다. Amazon ECS 작업 정의의 파일 수에 대한 컨테이너별 할당량을 업데이트할 수 있습니다. Amazon EC2 2에서 실행 중인 경우 (AWS Fargate) 를 사용하는 경우 기본 Amazon EC2 인스턴스에서 이러한 할당량을 조정해야 할 수도 있습니다.

sysctl 네트워크

조정 가능한 설정의 또 다른 범주는`sysctl`NET 설정을 선택합니다. 선택한 Linux 배포판에 대한 특정 설정을 참조해야 합니다. 이러한 설정의 대부분은 읽기 및 쓰기 버퍼의 크기를 조정합니다. 이는 컨테이너가 많은 대규모 Amazon EC2 인스턴스를 실행하는 경우에 도움이 될 수 있습니다.

모범 사례 - 자동 확장 및 용량 관리

Amazon ECS는 모든 규모의 컨테이너화된 애플리케이션 워크로드를 실행하는 데 사용됩니다. 여기에는 최소한의 테스트 환경과 글로벌 규모로 운영되는 대규모 프로덕션 환경이 모두 포함됩니다.

아마존 ECS를 사용하면 모든AWS서비스를 사용하면 사용한 만큼만 비용을 지불하는 요금제입니다. 적절하게 설계되면 애플리케이션이 필요한 시점에 필요한 리소스만 소비하도록 하여 비용을 절감할 수 있습니다. 이 모범 사례 가이드에서는 비용 효율적인 방식으로 운영하면서 서비스 수준 목표를 충족하는 방식으로 Amazon ECS 워크로드를 실행하는 방법을 보여 줍니다.

주제

- 작업 크기 결정
- Service Scaling 구성
- 용량 및 가용성
- 클러스터 용량
- Fargate 작업 크기 선택
- Amazon EC2 인스턴스 유형 선택
- Amazon EC2 스팟 및 퍼게이트_스팟 사용

작업 크기 결정

Amazon ECS에 컨테이너를 배포할 때 가장 중요한 선택 사항 중 하나는 컨테이너와 작업 크기입니다. 컨테이너와 작업 크기는 모두 확장과 용량 계획에 필수적입니다. Amazon ECS에는 용량에 사용되는 두 가지 리소스 메트릭이 있습니다. 메모리 CPU 및 메모리. CPU는 전체 vCPU CPU의 1/1024 단위로 측정됩니다. 여기서 1024단위는 전체 vCPU 1개와 같습니다. MB 단위의 메모리는 측정됩니다. 작업 정의에서 리소스 예약 및 제한을 선언할 수 있습니다.

예약을 선언할 때 작업에 필요한 최소 리소스 양을 선언합니다. 작업이 요청된 리소스의 양을 최소한으로 받습니다. 응용 프로그램에서 선언한 예약보다 더 많은 CPU 또는 메모리를 사용할 수 있습니다. 그러나 이것은 귀하가 선언한 모든 제한의 적용을 받습니다. 예약 금액 이상을 사용하는 것을 버스트(bursting) 라고 합니다. Amazon ECS에서는 예약이 보장됩니다. 예를 들어 Amazon EC2 인스턴스를 사용하여 용량을 제공하는 경우 Amazon ECS는 예약을 이행할 수 없는 인스턴스에 작업을 배치하지 않습니다.

제한은 컨테이너 또는 태스크에서 사용할 수 있는 CPU 단위 또는 메모리의 최대 양입니다. 이 모방보다 더 많은 CPU를 사용하려고하면 스로틀이 발생합니다. 더 많은 메모리를 사용하려고하면 컨테이너가 중지됩니다.

이러한 값을 선택하는 일은 쉽지 않습니다. 이는 응용 프로그램에 가장 적합한 값이 응용 프로그램의 리소스 요구 사항에 크게 의존하기 때문입니다. 응용 프로그램로드 테스트는 성공적인 리소스 요구 사항 계획과 응용 프로그램의 요구 사항을 더 잘 이해하는 데 핵심입니다.

상태 비저장 애플리케이션

부하 분산 장치 뒤의 응용 프로그램과 같이 수평으로 확장되는 상태 비저장 응용 프로그램의 경우 먼저 응용 프로그램이 요청을 처리할 때 사용하는 메모리 양을 결정하는 것이 좋습니다. 이렇게 하려면 다음과 같은 기존 도구를 사용할 수 있습니다.`ps`또는`top`또는 CloudWatch 컨테이너 인사이트와 같은 모니터링 솔루션을 사용할 수 있습니다.

CPU 예약을 결정할 때 비즈니스 요구 사항에 맞게 응용 프로그램을 확장하는 방법을 고려하십시오. 256개의 CPU 단위 (또는 1/4 vCPU) 와 같은 더 작은 CPU 예약을 사용하여 비용을 최소화하는 세분화된 방식으로 확장할 수 있습니다. 그러나 급격한 수요 급증에 대응할 수 있을 만큼 빠르게 확장되지 않을 수도 있습니다. 더 큰 CPU 예약을 사용하여 더 빠르게 확장 및 축소할 수 있으므로 수요 급증에 더 빠르게 대응할 수 있습니다. 그러나 CPU 예약이 클수록 비용이 많이 듭니다.

기타 애플리케이션

단일 작업자 또는 데이터베이스 서버와 같이 수평적으로 확장되지 않는 애플리케이션의 경우 사용 가능한 용량과 비용이 가장 중요한 고려 사항을 나타냅니다. 서비스 수준 목표를 충족하기 위해 트래픽을 처리해야 함을 나타내는 부하 테스트에 따라 메모리와 CPU의 양을 선택해야 합니다. Amazon ECS는 애플리케이션이 충분한 용량을 가진 호스트에 배치되도록 합니다.

Service Scaling 구성

Amazon ECS 서비스는 관리되는 작업 모음입니다. 각 서비스에는 연결된 작업 정의, 원하는 작업 수 및 선택적 배치 전략이 있습니다. Amazon ECS 서비스 Auto Scaling은 Application Auto Scaling 서비스를 통해 구현됩니다. Application Auto Scaling 은 CloudWatch 메트릭을 지표 확장의 소스로 사용합니다. 또한 CloudWatch 경보를 사용하여 서비스 확장 또는 축소 시기에 대한 임계값을 설정합니다. 측정 단위 대상을 설정하여 스케일링을 위한 임계값을 제공합니다.대상 추적 조정또는 임계 값을 지정하여단계 조정. Application Auto Scaling 이 구성된 후 서비스에 대해 원하는 적절한 작업 수를 지속적으로 계산합니다. 또한 원하는 작업 수가 변경되어야 하는 시기를 확장하거나 확장하여 Amazon ECS에 알립니다.

서비스 Auto Scaling을 효과적으로 사용하려면 적절한 조정 메트릭을 선택해야 합니다. 다음 단원에서는 지표를 선택하는 방법에 대해 설명합니다.

애플리케이션 특성화

응용 프로그램을 제대로 확장하려면 응용 프로그램을 확장해야 하는 조건과 축소 시기를 알고 있어야 합니다. 본질적으로 수요가 용량을 초과할 것으로 예측되는 경우 애플리케이션을 확장해야 합니다. 반대로, 리소스가 수요를 초과할 때 비용을 절약하기 위해 애플리케이션을 확장할 수 있습니다.

활용률 측정 단위 식별

효과적으로 확장하려면 사용률 또는 채도를 나타내는 메트릭을 식별하는 것이 중요합니다. 이 메트릭은 배율 조정에 유용하려면 다음 속성을 표시해야 합니다.

- 메트릭은 수요와 상관 관계가 있어야 합니다. 리소스가 일정하게 유지되지만 수요가 변경되면 메트릭 값도 변경되어야 합니다. 수요가 증가하거나 감소하면 메트릭이 증가하거나 감소해야 합니다.
- 메트릭 값은 용량에 비례하여 확장되어야 합니다. 수요가 일정하게 유지되는 경우 리소스를 더 추가하면 메트릭 값이 비례적으로 변경되어야 합니다. 따라서 작업 수를 두 배로 늘리면 메트릭이 50% 감소합니다.

사용률 메트릭을 식별하는 가장 좋은 방법은 스테이징 환경과 같은 사전 프로덕션 환경에서 로드 테스트를 수행하는 것입니다. 상용 및 오픈 소스 부하 테스트 솔루션이 널리 제공됩니다. 이러한 솔루션은 일반적으로 합성 부하를 생성하거나 실제 사용자 트래픽을 시뮬레이션할 수 있습니다.

로드 테스트 프로세스를 시작하려면 먼저 응용 프로그램의 사용률 메트릭에 대한 대시보드를 작성해야 합니다. 이러한 메트릭에는 CPU 사용률, 메모리 사용률, I/O 작업, I/O 대기열 길이 및 네트워크 처리량이 포함됩니다. CloudWatch 컨테이너 인사이트와 같은 서비스를 사용하여 이러한 지표를 수집할 수 있습니다. 또는 Grafana용 아마존 관리 서비스와 함께 Prometheus 용 아마존 관리 서비스를 사용하여 이를 수행할 수 있습니다. 이 프로세스 중에 애플리케이션의 응답 시간 또는 작업 완료율에 대한 메트릭을 수집하고 플롯해야 합니다.

부하 테스트 시 작은 요청 또는 작업 삽입 속도로 시작하십시오. 응용 프로그램이 워밍업될 수 있도록 이 속도를 몇 분 동안 계속 유지하십시오. 그런 다음 천천히 속도를 높이고 몇 분 동안 계속 유지하십시오. 이 주기를 반복하여 애플리케이션의 응답 또는 완료 시간이 너무 느려서 SLO (서비스 수준 목표)를 충족할 수 없을 때까지 매번 속도를 높입니다.

로드 테스트 중에 각 활용률 메트릭을 검사합니다. 부하와 함께 증가하는 메트릭은 최상의 활용률 메트릭으로 사용되는 최상위 후보입니다.

그런 다음 채도에 도달하는 리소스를 식별합니다. 동시에 활용률 메트릭을 검토하여 상위 수준에서 어느 것이 먼저 병합되는지 확인합니다. 또는 어느 것이 최고점에 도달하는지 확인한 다음 먼저 응용 프로그램을 충돌시킵니다. 예를 들어, 부하를 추가 할 때 CPU 사용률이 0% 에서 70 -80% 로 증가하면 더 많은 로드가 추가 된 후에도 해당 lebel에 머물러 있으면 CPU가 포화 상태라고 말할 수 있습니다. CPU 아키텍처에 따라 100% 에 도달하지 않을 수 있습니다. 예를 들어 로드를 추가할 때 메모리 사용률이 증가한 다음 애플리케이션이 작업 또는 Amazon EC2 인스턴스 메모리 제한에 도달하면 갑자기 중단된다고 가정합니다. 이 상황에서는 메모리가 완전히 소모되었을 가능성이 높습니다. 응용 프로그램에서 여러 리소스를 사용할 수 있습니다. 따라서 먼저 고갈되는 리소스를 나타내는 메트릭을 선택합니다.

마지막으로, 작업 또는 Amazon EC2 인스턴스의 수를 두 배로 늘린 후 로드 테스트를 다시 시도하십시오. 주요 지표가 이전과 같은 비율의 절반으로 증가 또는 감소한다고 가정합니다. 이 경우 메트릭은 용량에 비례합니다. 이것은 Auto Scaling에 적합한 사용률 메트릭입니다.

이제 이 가상의 시나리오를 생각해 보십시오. 응용 프로그램을 로드하고 CPU 사용률이 초당 100개의 요청에서 결국 80% 에 도달한다고 가정합니다. 로드가 더 추가되면 CPU 사용률이 더 이상 증가하지 않습니다. 그러나 응용 프로그램이 더 느리게 응답합니다. 그런 다음 부하 테스트를 다시 실행하여 작업 수를 두 배로 늘리고 이전 피크 값으로 속도를 유지합니다. 평균 CPU 사용률이 약 40% 로 떨어지면 평균 CPU 사용률이 확장 메트릭에 적합합니다. 반면에 작업 수를 늘린 후 CPU 사용률이 80% 로 유지되면 평균 CPU 사용률이 좋은 확장 메트릭이 아닙니다. 이 경우 적합한 메트릭을 찾기 위해 더 많은 연구가 필요합니다.

공통 애플리케이션 모델 및 확장 속성

모든 종류의 소프트웨어가AWS. 많은 워크로드는 자체 개발된 반면, 다른 워크로드는 널리 사용되는 오픈 소스 소프트웨어를 기반으로 합니다. 그들이 어디에서 시작했는지에 관계없이 우리는 서비스에 대한 몇 가지 일반적인 디자인 패턴을 관찰했습니다. 효과적으로 확장하는 방법은 패턴에 크게 좌우됩니다.

효율적인 CPU 바인딩 서버

효율적인 CPU 바인딩 서버는 CPU 및 네트워크 처리량 이외의 리소스를 거의 사용하지 않습니다. 각 요청은 응용 프로그램에서만 처리 할 수 있습니다. 요청은 데이터베이스와 같은 다른 서비스에 의존하지 않습니다. 이 응용 프로그램은 수십만 개의 동시 요청을 처리 할 수 있으며 여러 CPU를 효율적으로 활용할 수 있습니다. 각 요청은 메모리 오버 헤드가 적은 전용 스레드에서 처리되거나 요청을 처리하는 각 CPU에서 실행되는 비동기 이벤트 루프가 있습니다. 애플리케이션의 각 복제본은 요청을 동일하게 처리할 수 있습니다. CPU 전에 고갈 될 수있는 유일한 리소스는 네트워크 대역폭입니다. CPU 바운드 서비스에서 메모리 사용률은 최대 처리량에서도 사용 가능한 리소스의 일부분입니다.

이러한 유형의 응용 프로그램은 CPU 기반 Auto Scaling에 적합합니다. 이 응용 프로그램은 확장 측면에서 최대의 유연성을 누리고 있습니다. 더 큰 Amazon EC2 인스턴스 또는 Fargate vCPUs 를 제공하

여 수직으로 확장할 수 있습니다. 또한 더 많은 복제본을 추가하여 수평으로 확장 할 수도 있습니다. 복제본을 추가하거나 인스턴스 크기를 두 배로 늘리면 용량에 비해 평균 CPU 사용률이 절반씩 줄어 듭니다.

이 애플리케이션에 Amazon EC2 용량을 사용하는 경우 컴퓨팅 최적화 인스턴스 (예:`c5`또는`c6g`패밀리입니다.

효율적인 메모리 바인딩 서버

효율적인 메모리 바인딩 서버는 요청 당 상당한 양의 메모리를 할당합니다. 최대 동시성이지만 반드시 처리량은 아니지만 CPU 리소스가 고갈되기 전에 메모리가 고갈됩니다. 요청이 끝날 때 요청과 연관된 메모리는 해제됩니다. 사용 가능한 메모리가있는 한 추가 요청을 수락 할 수 있습니다.

이 유형의 응용 프로그램은 메모리 기반 Auto Scaling에 적합합니다. 이 응용 프로그램은 확장 측면에서 최대의 유연성을 누리고 있습니다. 더 큰 Amazon EC2 또는 Fargate 메모리 리소스를 제공하여 수직으로 확장할 수 있습니다. 또한 더 많은 복제본을 추가하여 수평으로 확장 할 수도 있습니다. 복제본을 추가하거나 인스턴스 크기를 두 배로 늘리면 용량에 비해 평균 메모리 사용률을 절반으로 줄일 수 있습니다.

이 애플리케이션에 Amazon EC2 용량을 사용하는 경우 메모리 최적화 인스턴스 (예:`r5`또는`r6g`패밀리입니다.

일부 메모리 바인딩 응용 프로그램은 요청이 끝날 때 요청과 관련된 메모리를 해제하지 않으므로 동시성이 감소해도 사용 된 메모리가 감소하지 않습니다. 이렇게 하려면 메모리 기반 크기 조정을 사용하지 않는 것이 좋습니다.

작업자 기반 서버

작업자 기반 서버는 각 개별 작업자 스레드에 대해 하나의 요청을 차례로 처리합니다. 작업자 스레드는 POSIX 스레드와 같은 경량 스레드가 될 수 있습니다. 또한 UNIX 프로세스와 같은 무거운 무게의 스레드가 될 수 있습니다. 어떤 스레드가 있더라도 응용 프로그램이 지원할 수있는 최대 동시성이 항상 있습니다. 일반적으로 동시성 제한은 사용 가능한 메모리 리소스에 비례하여 설정됩니다. 동시성 제한에 도달하면 추가 요청이 백로그 큐에 배치됩니다. 백로그 큐가 오버플로되면 추가 들어오는 요청이 즉시 거부됩니다. 이 패턴에 맞는 일반적인 응용 프로그램에는 Apache 웹 서버와 Gunicorn이 포함됩니다.

요청 동시성은 일반적으로이 응용 프로그램을 확장하기위한 최상의 메트릭입니다. 각 복제본에 대한 동시성 제한이 있으므로 평균 제한에 도달하기 전에 확장하는 것이 중요합니다.

요청 동시성 지표를 얻는 가장 좋은 방법은 애플리케이션에서 이를 CloudWatch 에 보고하도록 하는 것입니다. 애플리케이션의 각 복제본은 동시 요청 수를 높은 빈도로 사용자 지정 메트릭으로 게시할 수 있습니다. 주파수는 1분마다 한 번 이상 설정하는 것이 좋습니다. 여러 보고서를 수집한 후 평균 동시성

을 조정 메트릭으로 사용할 수 있습니다. 이 메트릭은 총 동시성을 취하여 복제본 수로 나누어 계산합니다. 예를 들어 총 동시성이 1000이고 복제본 수가 10이면 평균 동시성은 100입니다.

애플리케이션이 Application Load Balancer 뒤에 있는 경우`ActiveConnectionCount`메트릭을 확장 메트릭의 요인으로 사용합니다. 이`ActiveConnectionCount`메트릭은 평균 값을 얻기 위해 복제본 수로 나눠야 합니다. 평균 값은 원시 카운트 값과 달리 배율 조정에 사용해야 합니다.

이 설계가 가장 잘 작동하려면 응답 대기 시간의 표준 편차가 낮은 요청 속도로 작아야 합니다. 수요가 적은 기간에는 대부분의 요청이 짧은 시간 내에 응답되며 평균 응답 시간보다 훨씬 오래 걸리는 요청은 많지 않는 것이 좋습니다. 평균 반응 시간은 95번째 백분위수 응답 시간에 가까워야 합니다. 그렇지 않으면 대기열 오버플로가 결과로 발생할 수 있습니다. 이로 인해 오류가 발생합니다. 오버플로의 위험을 완화하기 위해 필요한 경우 추가 복제본을 제공하는 것이 좋습니다.

대기 서버

대기 서버는 각 요청에 대해 일부 처리를 수행하지만 작동하는 하나 이상의 다운 스트림 서비스에 크게 의존합니다. 컨테이너 응용 프로그램은 데이터베이스 및 기타 API 서비스와 같은 다운스트림 서비스를 많이 사용합니다. 특히 대용량 또는 높은 동시성 시나리오에서 이러한 서비스가 응답하는 데 약간의 시간이 걸릴 수 있습니다. 이러한 응용 프로그램은 사용 가능한 메모리 측면에서 CPU 리소스와 최대 동시성을 사용하는 경향이 있기 때문입니다.

대기 서비스는 응용 프로그램 설계 방식에 따라 메모리 바운드 서버 패턴 또는 작업자 기반 서버 패턴에 적합합니다. 응용 프로그램의 동시성이 메모리에 의해서만 제한되는 경우 평균 메모리 사용률을 조정 메트릭으로 사용해야 합니다. 응용 프로그램의 동시성이 작업자 제한을 기반으로 하는 경우 평균 동시성을 조정 메트릭으로 사용해야 합니다.

Java 기반 서버

Java 기반 서버가 CPU 바인딩이고 CPU 리소스에 비례하여 확장되는 경우 효율적인 CPU 바인딩 서버 패턴에 적합 할 수 있습니다. 이 경우 평균 CPU 사용률이 조정 메트릭으로 적절할 수 있습니다. 그러나 많은 Java 응용 프로그램은 CPU 바인딩이 아니기 때문에 확장하기가 어렵습니다.

최상의 성능을 위해 JVM (Java Virtual Machine) 힙에 가능한 한 많은 메모리를 할당하는 것이 좋습니다. Java 8 업데이트 191 이상을 포함한 JVM의 최신 버전은 자동으로 컨테이너 내에 맞게 힙 크기를 최대한 크게 설정합니다. 즉, Java에서 메모리 사용률은 응용 프로그램 사용률에 거의 비례하지 않습니다. 요청 속도와 동시성이 증가함에 따라 메모리 사용률은 일정하게 유지됩니다. 따라서 메모리 사용률에 따라 Java 기반 서버를 확장하지 않는 것이 좋습니다. 대신 CPU 사용률을 확장하는 것이 좋습니다.

경우에 따라 Java 기반 서버에서 CPU를 소모하기 전에 힙 소모가 발생합니다. 응용 프로그램이 높은 동시성으로 힙 소모가 발생하기 쉬운 경우 평균 연결이 최상의 확장 메트릭입니다. 응용 프로그램이 높은 처리량으로 힙 소모가 발생하기 쉬운 경우 평균 요청 속도가 가장 좋은 조정 메트릭입니다.

다른 가비지 수집 런타임을 사용하는 서버

많은 서버 응용 프로그램은.NET 및 Ruby와 같은 가비지 수집을 수행하는 런타임을 기반으로합니다. 이러한 서버 응용 프로그램은 앞에서 설명한 패턴 중 하나에 적합할 수 있습니다. 그러나 Java와 마찬가지로, 관찰된 평균 메모리 사용률이 처리량이나 동시성과 관련이 없는 경우가 많기 때문에 메모리를 기반으로 이러한 응용 프로그램을 확장하지 않는 것이 좋습니다.

이러한 응용 프로그램의 경우 응용 프로그램이 CPU 바인딩된 경우 CPU 사용률을 확장하는 것이 좋습니다. 그렇지 않으면 부하 테스트 결과에 따라 평균 처리량 또는 평균 동시성을 기준으로 확장하는 것이 좋습니다.

Job 처리자

많은 워크로드는 비동기 작업 처리를 포함합니다. 여기에는 실시간으로 요청을 수신하지 않고 작업 대기열을 구독하여 작업을 수신하는 응용 프로그램이 포함됩니다. 이러한 유형의 응용 프로그램에서 적절한 확장 메트릭은 거의 항상 대기열 길이입니다. 대기열 증가는 보류 중인 작업이 처리 용량을 능가하는 반면 빈 대기열은 수행할 작업보다 더 많은 용량이 있음을 나타냅니다.

AWS메시징 서비스 (예: Amazon SQS 및 Amazon Kinesis Data Streams) 는 확장에 사용할 수 있는 CloudWatch 지표를 제공합니다. Amazon SQS 경우,`ApproximateNumberOfMessagesVisible`가 최상의 지표입니다. KKinesis Data Streams 의 경우`MillisBehindLatest`메트릭을 사용하여 KCL (Kinesis Client Library) 에 의해 게시된 이 메트릭은 확장에 사용하기 전에 모든 소비자의 평균을 계산해야 합니다.

용량 및 가용성

애플리케이션 가용성은 오류 없는 환경을 제공하고 애플리케이션 대기 시간을 최소화하는 데 매우 중요합니다. 가용성은 액세스할 수 있고 수요를 충족할 수 있는 충분한 용량이 있는 리소스가 있는지 여부에 따라 달라집니다.AWS는 가용성을 관리하기 위한 몇 가지 메커니즘을 제공합니다. Amazon ECS에서 호스팅되는 애플리케이션의 경우 여기에는 Auto Scaling 및 가용 영역 (AZ) 이 포함됩니다. Auto Scaling은 사용자가 정의한 메트릭을 기반으로 작업 또는 인스턴스의 수를 관리하는 반면, 가용 영역을 사용하면 격리되었지만 지리적으로 가까운 위치에 애플리케이션을 호스팅할 수 있습니다.

작업 규모와 마찬가지로 용량과 가용성은 반드시 고려해야 할 몇 가지 절충안을 제시합니다. 이상적으로 용량은 수요와 완벽하게 일치할 수 있습니다. 짧은 대기 시간 및 오류율을 포함하여 SLO (서비스 수준 목표) 를 충족하기 위해 요청 및 프로세스 작업을 처리할 수 있는 충분한 용량만 있으면 됩니다. 용량이 너무 높지 않아 과도한 비용이 발생하거나 너무 낮지 않아 지연 시간과 오류율이 길어질 수 없습니다.

자동 크기 조정은 잠재 프로세스입니다. 첫째, 실시간 지표는 CloudWatch 에 전달되어야 합니다. 그런 다음 분석을 위해 집계해야 합니다. 이 작업은 메트릭의 세분성에 따라 최대 몇 분이 걸릴 수 있습니다. CloudWatch 는 메트릭을 경보 임계값과 비교하여 리소스 부족 또는 초과분을 식별합니다. 불안정성을 방지하려면 알람이 울리기 전에 몇 분 동안 설정된 임계값을 초과하도록 경보를 구성합니다. 또한 새 작업을 프로비저닝하고 더 이상 필요하지 않은 작업을 종료하는 데 시간이 걸립니다.

설명된 시스템에서 이러한 지연이 발생할 수 있으므로 오버프로비저닝을 통해 일부 여유 공간을 유지하는 것이 중요합니다. 이렇게 하면 단기적인 수요 급증을 수용할 수 있습니다. 또한 응용 프로그램이 포화 상태에 도달하지 않고 추가 요청을 처리하는 데 도움이됩니다. 스케일링 타겟을 사용률의 60-80% 사이로 설정하는 것이 좋습니다. 이렇게 하면 추가 용량이 프로비저닝되는 동안 애플리케이션이 폭증하는 추가 수요를 보다 효과적으로 처리할 수 있습니다.

과잉 프로비저닝을 권장하는 또 다른 이유는 가용 영역 장애에 신속하게 대응할 수 있기 때문입니다.AWS에서는 여러 가용 영역에서 프로덕션 워크로드를 처리할 것을 권장합니다. 이는 가용 영역 장애가 발생할 경우 나머지 가용 영역에서 실행 중인 작업이 여전히 수요를 처리할 수 있기 때문입니다. 애플리케이션이 두 개의 가용 영역에서 실행되는 경우 일반 작업 수를 두 배로 늘려야 합니다. 따라서 잠재적인 장애 발생 시 즉시 용량을 제공할 수 있습니다. 애플리케이션이 세 개의 가용 영역에서 실행되는 경우 일반 작업 수의 1.5배를 실행하는 것이 좋습니다. 즉, 일반 서빙에 필요한 두 사람마다 세 가지 작업을 실행하십시오.

속도 조정 속도 극대화

자동 크기 조정은 적용 시간이 걸리는 반응 프로세스입니다. 그러나 확장하는 데 필요한 시간을 최소화하는 데 도움이 되는 몇 가지 방법이 있습니다.

이미지 크기를 최소화합니다. 이미지가 클수록 이미지 저장소에서 다운로드하고 압축을 푸는 데 시간이 오래 걸립니다. 따라서 이미지 크기를 작게 유지하면 컨테이너를 시작하는 데 필요한 시간이 줄어듭니다. 이미지 크기를 줄이려면 다음과 같은 특정 권장 사항을 따르십시오.

- 정적 바이너리를 빌드하거나 Golang을 사용할 수 있다면 이미지를 빌드하십시오.`FROM`스크래치 및 결과 이미지에 이진 응용 프로그램 만 포함 할 수 있습니다.
- Amazon Linux 또는 Ubuntu와 같은 업스트림 배포판 공급업체의 최소화된 기본 이미지를 사용합니다.
- 최종 이미지에 빌드 아티팩트를 포함하지 마십시오. 다단계 빌드를 사용하면이 문제를 해결할 수 있습니다.
- 콤팩트`RUN`가능한 한 단계를 수행하십시오. 각`RUN`스테이지는 새 이미지 레이어를 생성하여 레이어를 다운로드할 수 있는 추가 왕복이 됩니다. 단일`RUN`에 의해 결합 된 여러 명령이있는`&&`여러 개의 레이어가 하나보다 적음`RUN`단계를 시작합니다.

- ML 추론 데이터와 같은 데이터를 최종 이미지에 포함하려는 경우 트래픽을 시작하고 시작하는 데 필요한 데이터만 포함합니다. 서비스에 영향을 주지 않고 Amazon S3 또는 다른 스토리지에서 온디맨드 방식으로 데이터를 가져오는 경우 데이터를 해당 위치에 저장하십시오.

이미지를 가까이 유지합니다. 네트워크 지연 시간이 길수록 이미지를 다운로드하는 데 더 오래 걸립니다. 동일한 저장소에 이미지를 호스팅하십시오.AWS워크로드가 있는 지역입니다. Amazon ECR은 Amazon ECS를 사용할 수 있는 모든 리전에서 사용할 수 있는 고성능 이미지 저장소입니다. 컨테이너 이미지를 다운로드하려면 인터넷이나 VPN 링크를 통과하지 마십시오. 동일한 지역에서 이미지를 호스팅하면 전반적인 안정성이 향상됩니다. 다른 지역의 네트워크 연결 문제 및 가용성 문제의 위험을 완화합니다. 또는 Amazon ECR 교차 리전 복제를 구현하여 이를 지원할 수도 있습니다.

로드 밸런서 상태 확인 임계값을 줄입니다. 로드 밸런서는 애플리케이션에 트래픽을 보내기 전에 상태 확인을 수행합니다. 대상 그룹에 대한 기본 상태 확인 구성은 90초 이상 걸릴 수 있습니다. 이 동안 상태 및 수신 요청을 확인합니다. 상태 확인 간격과 임계값 수를 줄이면 응용 프로그램이 트래픽을 더 빠르게 받아들이고 다른 작업에 대한 로드를 줄일 수 있습니다.

콜드 스타트 성능을 고려하십시오. 자바와 같은 일부 응용 프로그램은 JIT (저스트 인 타임) 컴파일을 수행합니다. 적어도 컴파일 프로세스가 시작되면 응용 프로그램 성능을 나타낼 수 있습니다. 해결 방법은 콜드 스타트 성능 저하를 부과하지 않는 언어로 워크로드의 대기 시간이 중요한 부분을 다시 작성하는 것입니다.

대상 추적 조정 정책이 아닌 단계 조정을 사용합니다. Amazon ECS 작업에는 여러 가지 Application Auto Scaling 옵션이 있습니다. 대상 추적은 가장 사용하기 쉬운 모드입니다. 이 기능을 사용하면 CPU 평균 활용률과 같은 측정 단위에 대한 대상 값을 설정하기만 하면 됩니다. 그런 다음 자동 스케일러는 해당 값을 얻는 데 필요한 작업 수를 자동으로 관리합니다. 그러나 수요 변화에 더 빠르게 대응할 수 있도록 단계 배율 조정을 대신 사용하는 것이 좋습니다. 단계 조정을 사용하여 조정 측정 단위에 대한 특정 임계값과 임계값을 초과할 때 추가 또는 제거할 작업 수를 정의합니다. 그리고 더 중요한 것은 임계값 경보가 위반되는 시간을 최소화하여 수요 변화에 매우 빠르게 대응할 수 있다는 것입니다. 자세한 내용은 단원을 참조하십시오.Service Auto Scaling의Amazon 탄력적 컨테이너 서비스 개발자 가이드.

Amazon EC2 인스턴스를 사용하여 클러스터 용량을 제공하는 경우 다음 권장 사항을 고려하십시오.

Amazon EC2 인스턴스와 더 빠른 Amazon EBS 볼륨을 사용합니다. 더 큰 Amazon EC2 인스턴스와 더 빠른 Amazon EBS 볼륨을 사용하여 이미지 다운로드 및 준비 속도를 향상시킬 수 있습니다. 지정된 Amazon EC2 인스턴스 제품군 내에서 인스턴스 크기가 증가함에 따라 네트워크 및 Amazon EBS의 최대 처리량이 증가합니다 (예:`m5.xlarge`to`m5.2xlarge`). 또한 Amazon EBS 볼륨을 사용자 지정하여 처리량과 IOPS를 늘릴 수도 있습니다. 예를 들어, 사용 중인 경우`gp2`볼륨의 경우 더 많은 기본 처

리량을 제공하는 더 큰 볼륨을 사용합니다. 사용 중인 경우gp3볼륨의 경우 볼륨을 생성할 때 처리량과 IOPS를 지정합니다.

Amazon EC2 인스턴스에서 실행되는 작업에 브리지 네트워크 모드를 사용합니다. 사용하는 작업bridge네트워크 모드를 사용하는 작업보다 빠르게 시작할 수 있습니다.awsvpc네트워크 모드로 전환됩니다. 일시awsvpc네트워크 모드가 사용되는 경우 Amazon Eelastic network interface (ENI) 를 사용하여 작업을 시작하기 전에 인스턴스에 연결합니다. 이로 인해 대기 시간이 추가로 발생합니다. 브리지 네트워킹을 사용하기위한 몇 가지 절충점이 있습니다. 이러한 작업은 자체 보안 그룹을 가져오지 않으며 로드 밸런싱에 대한 몇 가지 의미가 있습니다. 자세한 내용은 단원을 참조하십시오.로드 밸런서 대상 그룹의Elastic Load Balancing 사용자 가이드.

수요 충격 처리

일부 응용 프로그램은 갑작스런 큰 충격을 경험합니다. 이것은 다양한 이유로 발생합니다: 뉴스 이벤트, 대규모 판매, 미디어 이벤트 또는 바이러스가 발생하여 트래픽이 매우 짧은 시간 내에 신속하고 크게 증가하는 기타 이벤트. 계획되지 않은 경우 수요가 가용 자원을 빠르게 초과할 수 있습니다.

수요 충격을 처리하는 가장 좋은 방법은 수요 충격을 예측하고 그에 따라 계획하는 것입니다. 자동 크기 조정에는 시간이 걸릴 수 있으므로 수요 충격을 시작하기 전에 응용 프로그램을 확장하는 것이 좋습니다. 최상의 결과를 얻으려면 공유 캘린더를 사용하는 팀 간의 긴밀한 협업을 포함하는 비즈니스 계획을 세우는 것이 좋습니다. 이벤트를 계획하고있는 팀은 사전에 응용 프로그램을 담당하는 팀과 긴밀하게 협력해야합니다. 이렇게 하면 팀이 명확한 일정 계획을 수립할 수 있는 충분한 시간을 확보할 수 있습니다. 이벤트 전에 스케일아웃하고 이벤트 후에 확장하도록 용량을 예약할 수 있습니다. 자세한 내용은 단원을 참조하십시오.예약 조정의Auto Scaling 애플리케이션 사용 설명서.

엔터프라이즈 Support 플랜이 있는 경우 TAM (기술 계정 관리자) 과 협력해야 합니다. TAM은 서비스 할당량을 확인하고 이벤트가 시작되기 전에 필요한 할당량이 발생하는지 확인할 수 있습니다. 이렇게 하면 실수로 서비스 할당량에 도달하지 않습니다. 또한 로드 밸런서와 같은 서비스를 미리 워밍하여 이벤트가 원활하게 진행되는지 확인할 수 있습니다.

예정되지 않은 수요 충격을 처리하는 것은 더 어려운 문제입니다. 진폭이 충분히 클 경우 예기치 않은 충격으로 인해 수요가 신속하게 용량을 초과할 수 있습니다. 또한 자동 크기 조정이 반응하는 기능보다 앞설 수 있습니다. 예정되지 않은 충격을 대비하는 가장 좋은 방법은 자원을 과도하게 프로비저닝하는 것입니다. 언제든지 최대 예상 트래픽 수요를 처리할 수 있는 충분한 리소스가 있어야 합니다.

예기치 않은 수요 충격을 예상하여 최대 용량을 유지하는 것은 비용이 많이 듭니다. 비용 영향을 완화하기 위해 큰 수요 충격이 임박할 것으로 예측하는 주요 지표 지표 또는 이벤트를 찾습니다. 메트릭 또는 이벤트가 상당한 사전 알림을 안정적으로 제공하는 경우 이벤트가 발생하거나 메트릭이 설정한 특정 임계값을 초과할 때 즉시 확장 프로세스를 시작합니다.

애플리케이션에 갑작스러운 수요 충동이 발생하기 쉬운 경우 중요하지 않은 기능은 그대로 유지하면서 고객에게 중요한 기능을 유지하는 고성능 모드를 애플리케이션에 추가하는 것이 좋습니다. 예를 들어 응용 프로그램이 고가의 사용자 지정 응답 생성에서 정적 응답 페이지 제공으로 전환할 수 있다고 가정합니다. 이 시나리오에서는 응용 프로그램을 전혀 확장하지 않고 처리량을 크게 늘릴 수 있습니다.

마지막으로, 수요 충격을 더 잘 처리하기 위해 모 놀리 식 서비스를 분리하는 것을 고려할 수 있습니다. 응용 프로그램이 실행 비용이 많이 들고 확장 속도가 느린 모 놀리 식 서비스인 경우 성능에 중요한 부분을 추출하거나 다시 작성하여 별도의 서비스로 실행할 수 있습니다. 그런 다음 이러한 새로운 서비스는 덜 중요한 구성 요소와 독립적으로 확장 할 수 있습니다. 성능에 중요한 기능을 애플리케이션의 다른 부분과 별도로 확장할 수 있는 유연성을 갖추면 용량을 추가하는 데 걸리는 시간을 줄이고 비용을 절약할 수 있습니다.

클러스터 용량

이 항목의 앞부분에서 배율 조정 메트릭을 사용하여 의 복제본 계정을 확장하는 방법에 대해 설명했습니다. CPU 및 메모리 리소스를 포함한 리소스에서도 작업을 실행해야 합니다. 이것은 다시 용량 주제와 관련이 있습니다. Amazon ECS에서 용량은 다음 두 가지 기본 공급자를 통해 제공됩니다.AWS FargateAmazon EC2 및 Amazon EC2.

여러 가지 방법으로 Amazon ECS 클러스터에 용량을 제공할 수 있습니다. 예를 들어 Amazon EC2 인스턴스를 시작하고 시작 시 Amazon ECS 컨테이너 에이전트를 사용하여 클러스터에 등록할 수 있습니다. 그러나 이 방법은 자체적으로 확장을 관리해야 하기 때문에 어려울 수 있습니다. 따라서 Amazon ECS 용량 공급자를 사용하는 것이 좋습니다. 리소스 확장을 관리합니다. 다음과 같은 세 가지 종류의 용량 공급자가 있습니다. Amazon EC2, Fargate 및 Fargate 스팟.

Fargate 및 Fargate 스팟 용량 공급자는 Fargate 작업의 수명 주기를 처리합니다. Fargate는 온디맨드 용량을 제공하며 Fargate 스팟은 스팟 용량을 제공합니다. 작업이 시작되면 ECS에서 Fargate 리소스를 프로비저닝합니다. 이 Fargate 리소스는 작업 정의에서 선언한 작업 수준 제한에 직접 해당하는 메모리 및 CPU 단위와 함께 제공됩니다. 각 작업은 자체 Fargate 리소스를 수신하여 작업과 계산 iresources 간에 1:1 관계를 만듭니다.

Fargate 스팟에서 실행되는 작업은 중단될 수 있습니다. 중단은 2 분 경고 후에 발생합니다. 이러한 수요가 많은 기간 동안 발생합니다. Fargate Spot은 배치 작업, 개발 또는 스테이징 환경과 같은 중단 방지 워크로드에 가장 적합합니다. 높은 가용성과 낮은 대기 시간이 요구 사항이 아닌 다른 시나리오에도 적합합니다.

Fargate 스팟 작업은 Fargate 주문형 작업과 함께 실행할 수 있습니다. 이를 함께 사용하면 더 낮은 비용으로 “버스트” 용량을 프로비저닝할 수 있습니다.

또한 ECS는 작업에 대한 Amazon EC2 인스턴스 용량을 관리할 수 있습니다. 각 Amazon EC2 용량 공급자는 사용자가 지정한 Amazon EC2 Auto Scaling 그룹과 연결됩니다. Amazon EC2 용량 공급자를 사용하는 경우 ECS 클러스터 Auto Scaling은 Amazon EC2 Auto Scaling 그룹의 크기를 유지하여 모든 예약된 작업을 배치할 수 있도록 합니다.

클러스터 용량 모범 사례

용량 제공자가 아닌 서비스에 헤드룸을 추가합니다. Amazon EC2 용량 공급자는 목표 용량 값을 제공합니다. 이 값을 100% 보다 낮게 설정하면 ECS는 작업을 수용하는 데 필요한 것보다 많은 Amazon EC2 인스턴스를 프로비저닝합니다. 여러 Amazon EC2 인스턴스가 작업을 수락할 준비가 되어 있으면 도움이 될 수 있습니다. 그러나 Amazon Virtual Private Cloud 사용하는 경우 새 작업을 시작하려면 이미지를 다운로드하고 네트워크 인터페이스를 연결하는 데 추가 시간이 필요합니다. 이 추가 대기 시간은 수익에 악영향을 줄 수 있습니다.

따라서 다음을 수행하는 것이 좋습니다. 용량 제공자의 대상 용량을 줄이는 대신 대상 추적 조정 메트릭 또는 서비스의 단계 조정 임계값을 수정하여 서비스의 복제본 수를 늘립니다. 관련 조정 정책에 대한 자세한 내용은 단원을 참조하세요.대상 추적 조정 정책또는단계 조정 정책의Amazon 탄력적 컨테이너 서비스 개발자 가이드. Amazon EC2 용량 공급자는 Auto Scaling 그룹에 인스턴스를 추가하여 추가 작업에 필요한 용량을 프로비저닝합니다. 이렇게 하면 필요할 때 컴퓨팅 리소스와 애플리케이션 리소스를 모두 사용할 수 있습니다. 예를 들어 ECS 서비스의 작업 수를 두 배로 늘려 즉시 100% 급증할 수 있습니다.

Fargate 작업 크기 선택

에서 작업을 실행하는 경우AWS Fargate작업 정의에서 작업 CPU 및 메모리 제한을 선언해야 합니다. ECS는 이러한 제한을 사용하여 작업을 실행할 Fargate 인스턴스 유형을 결정합니다. 사용자가 결정하는 제한은 선언한 예약보다 크거나 이와 동일해야 합니다. 대부분의 경우 작업 정의에 선언된 각 컨테이너의 예약 합계로 설정할 수 있습니다. 그런 다음 숫자를 가장 가까운 Fargate 인스턴스 크기로 반올림합니다. 사용 가능한 크기에 대한 자세한 내용은 단원을 참조하세요.작업 CPU 및 메모리의Amazon 탄력적 컨테이너 서비스 개발자 가이드.

Amazon EC2 인스턴스 유형 선택

Amazon EC2 사용하여 ECS 클러스터에 용량을 제공하는 경우 다양한 인스턴스 유형 중에서 선택할 수 있습니다. 모든 Amazon EC2 인스턴스 유형 및 패밀리는 ECS와 호환됩니다.

사용할 수 있는 인스턴스 유형을 확인하려면 먼저 애플리케이션의 특정 요구 사항을 충족하지 않는 인스턴스 유형 또는 인스턴스 패밀리를 제거합니다. 예를 들어 애플리케이션에 GPU가 필요한 경우 GPU

가 없는 인스턴스 유형을 제외할 수 있습니다. 그러나 다른 요구 사항도 고려해야합니다. 예를 들어, CPU 아키텍처, 네트워크 처리량 및 인스턴스 스토리지가 요구 사항인 경우를 고려합니다. 그런 다음 각 인스턴스 유형에서 제공하는 CPU 및 메모리 양을 검사합니다. 일반적으로 CPU와 메모리는 실행할 작업의 복제본을 하나 이상 보관할 수 있을 만큼 충분히 커야 합니다.

애플리케이션과 호환되는 인스턴스 유형 중에서 선택할 수 있습니다. 인스턴스 크기가 클수록 더 많은 작업을 동시에 시작할 수 있습니다. 또한 인스턴스 크기가 작을수록 보다 세밀한 방식으로 확장하여 비용을 절감할 수 있습니다. 클러스터의 모든 애플리케이션에 적합한 단일 Amazon EC2 인스턴스 유형을 선택할 필요가 없습니다. 대신 여러 개의 Auto Scaling 그룹 을 생성할 수 있습니다. 각 그룹은 다른 인스턴스 유형을 가질 수 있습니다. 그런 다음 각 그룹에 대해 Amazon EC2 용량 공급자를 생성할 수 있습니다. 마지막으로 서비스 및 작업의 용량 제공자 전략에서 요구 사항에 가장 적합한 용량 제공자를 선택할 수 있습니다.

Amazon EC2 스팟 및 퍼게이트_스팟 사용

스팟 용량은 온 디맨드 인스턴스에 비해 상당한 비용을 절감할 수 있습니다. 스팟 용량은 온 디맨드 용량이나 예약 용량보다 가격이 상당히 낮은 초과 용량입니다. 스팟 용량은 배치 프로세싱 및 머신 러닝 워크로드, 개발 및 스테이징 환경에 적합합니다. 보다 일반적으로 일시적인 가동 중지 시간을 허용하는 모든 워크로드에 적합합니다.

스팟 용량을 항상 사용할 수 없기 때문에 다음과 같은 결과를 이해합니다.

- 첫째, 수요가 매우 많은 기간에는 스팟 용량을 사용하지 못할 수 있습니다. 이로 인해 Fargate 스팟 작업과 Amazon EC2 스팟 인스턴스 시작이 지연될 수 있습니다. 이러한 이벤트에서 ECS 서비스는 작업을 다시 시작하고 필요한 용량을 사용할 수 있을 때까지 Amazon EC2 Auto Scaling 그룹도 인스턴스 시작을 다시 시도합니다. Fargate 와 Amazon EC2 스팟 용량을 온 디맨드 용량으로 대체하지 않습니다.
- 둘째, 용량에 대한 전반적인 수요가 증가하면 스팟 인스턴스와 작업이 2분 경고만 표시되면서 종료될 수 있습니다. 경고가 전송된 후 인스턴스가 완전히 종료되기 전에 필요한 경우 작업을 순서대로 종료해야 합니다. 이렇게 하면 오류 가능성을 최소화할 수 있습니다. 정상 종료에 대한 자세한 내용은 단원을 참조하세요.ECS를 통한 정상 종료.

스팟 용량 부족을 최소화하려면 다음 권장 사항을 고려하십시오.

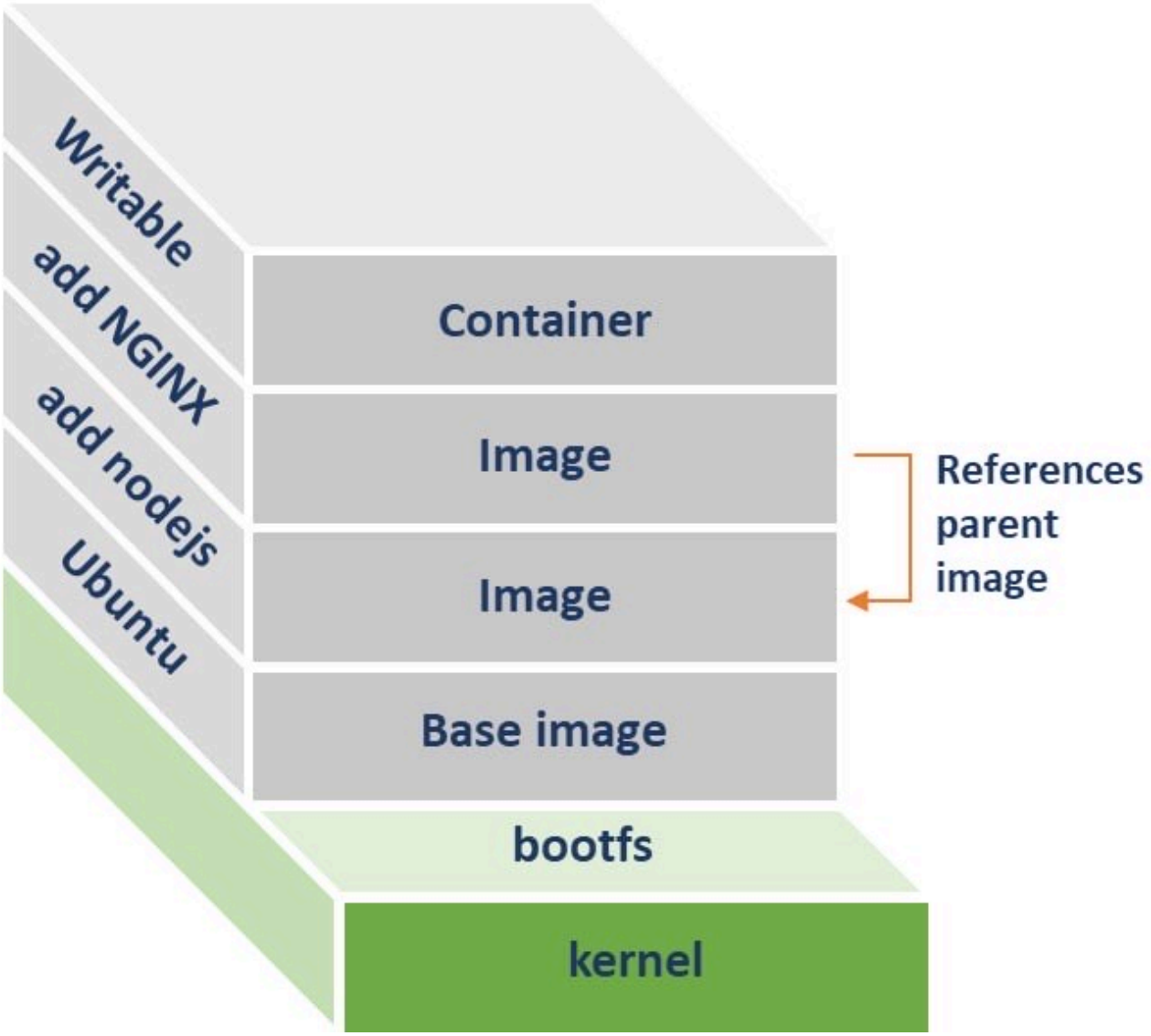
- 여러 리전 및 가용 영역을 사용합니다. 스팟 용량은 지역 및 가용 영역에 따라 다릅니다. 워크로드를 여러 리전 및 가용 영역에서 실행하여 스팟 가용성을 향상시킬 수 있습니다. 가능한 경우 작업 및 인스턴스를 실행하는 리전의 모든 가용 영역에 서브넷을 지정합니다.

- 여러 Amazon EC2 인스턴스 유형을 사용합니다. Amazon EC2 Auto Scaling에서 혼합 인스턴스 정책을 사용하면 Auto Scaling 그룹에 여러 인스턴스 유형이 시작됩니다. 이렇게 하면 필요할 때 스팟 용량 요청을 이행할 수 있습니다. 안정성을 극대화하고 복잡성을 최소화하려면 혼합 인스턴스 정책에서 CPU 및 메모리 양이 거의 같은 인스턴스 유형을 사용합니다. 이러한 인스턴스는 다른 세대 또는 동일한 기본 인스턴스 유형의 변형일 수 있습니다. 여기에는 필요하지 않은 추가 기능이 함께 제공될 수 있습니다. 이러한 목록의 예는 m4.라지, m5.라지, m5a.라지, m5d.라지, m5n.라지, m5dn.라지 및 m5ad.large 등이 포함될 수 있습니다. 자세한 내용은 단원을 참조하십시오.여러 인스턴스 유형과 구매 옵션이 있는 Auto Scaling 그룹의Amazon EC2 Auto Scaling 사용 설명서.
- 용량 최적화된 스팟 할당 전략을 사용합니다. Amazon EC2 스팟을 사용하면 용량 및 비용 최적화된 할당 전략 중에서 선택할 수 있습니다. 새 인스턴스를 시작할 때 용량 최적화 전략을 선택하면 Amazon EC2 스팟은 선택한 가용 영역에서 가용성이 가장 높은 인스턴스 유형을 선택합니다. 이렇게 하면 인스턴스가 시작된 직후에 인스턴스가 종료될 가능성을 줄일 수 있습니다.

모범 사례 - 영구 스토리지

Amazon ECS를 사용하여 상태 저장 컨테이너화된 애플리케이션을 대규모로 실행할 수 있습니다.AWS기본적으로 임시 컨테이너에 데이터 지속성을 제공하는 Amazon EFS, Amazon EBS 또는 Windows 파일 서버용 Amazon FSX와 같은 스토리지 서비스를 제공합니다. 용어데이터 지속성는 데이터 자체가 데이터를 생성한 프로세스보다 오래 지속됨을 의미합니다. 데이터 지속성AWS는 컴퓨팅과 스토리지 서비스를 결합함으로써 달성됩니다. Amazon EC2 와 마찬가지로 Amazon ECS를 사용하여 컨테이너화된 애플리케이션의 수명 주기를 해당 애플리케이션에서 사용하고 생성하는 데이터와 분리할 수도 있습니다. 사용AWS스토리지 서비스를 사용하는 경우 Amazon ECS 작업은 작업이 종료된 후에도 데이터를 유지할 수 있습니다.

기본적으로 컨테이너는 생성 된 데이터를 유지하지 않습니다. 컨테이너가 종료되면 쓰기 가능한 레이어에 쓴 데이터가 컨테이너로 파괴됩니다. 따라서 로컬에 데이터를 저장할 필요가없는 상태 비 저장 응용 프로그램에 컨테이너가 적합합니다. 데이터 지속성이 필요한 컨테이너화된 애플리케이션에는 애플리케이션의 컨테이너가 종료될 때 삭제되지 않는 스토리지 백엔드가 필요합니다.



컨테이너 이미지는 일련의 레이어에서 만들어집니다. 각 레이어는 이미지가 빌드 된 Dockerfile의 명령을 나타냅니다. 컨테이너를 제외하고 각 레이어는 읽기 전용입니다. 즉, 컨테이너를 만들 때 쓰기 가능한 레이어가 기본 레이어 위에 추가됩니다. 컨테이너가 생성, 삭제 또는 수정하는 모든 파일은 쓰기 가능한 레이어에 기록됩니다. 컨테이너가 종료되면 쓰기 가능한 레이어도 동시에 삭제됩니다. 동일한 이미지를 사용하는 새 컨테이너에는 자체 쓰기 가능한 레이어가 있습니다. 이 레이어에는 변경 사항이 포함되어 있지 않습니다. 따라서 컨테이너의 데이터는 항상 컨테이너 쓰기 가능 계층 외부에 저장되어야 합니다.

Amazon ECS를 사용하면 볼륨을 사용하여 상태 저장 컨테이너를 실행할 수 있습니다. Amazon ECS는 기본적으로 Amazon EFS S와 통합되며 Amazon EBS와 통합된 볼륨을 사용합니다. 윈도우 컨테이너의 경우 Amazon ECS는 윈도우 파일 서버용 아마존 FSX와 통합하여 영구 스토리지를 제공합니다.

주제

- 컨테이너에 적합한 스토리지 유형 선택
- Amazon EFS 볼륨
- 도커 볼륨
- Amazon FSx for Windows File Server

컨테이너에 적합한 스토리지 유형 선택

Amazon ECS 클러스터에서 실행 중인 애플리케이션은 다양한AWS스토리지 서비스 및 타사 제품을 사용하여 상태 저장 워크로드를 위한 영구 스토리지를 제공합니다. 애플리케이션의 아키텍처 및 스토리지 요구 사항에 따라 컨테이너화된 애플리케이션의 스토리지 백엔드를 선택해야 합니다. 에 대한 자세한 내용AWS스토리지 서비스에 대한 자세한 내용은on 클라우드 스토리지AWS.

Linux 인스턴스가 포함되어 있거나 Fargate 와 함께 사용되는 Amazon ECS 클러스터의 경우 Amazon ECS는 Amazon EFS 및 Amazon EBS와 통합하여 컨테이너 스토리지를 제공합니다. Amazon EFS와 Amazon EBS의 가장 독특한 차이점은 수천 개의 Amazon ECS 작업에 Amazon EFS 파일 시스템을 동시에 마운트할 수 있다는 점입니다. 반대로 Amazon EBS 볼륨은 동시 액세스를 지원하지 않습니다. 따라서 Amazon EFS 수평으로 확장되는 컨테이너화된 애플리케이션에 권장되는 스토리지 옵션입니다. 이것은 동시성을 지원하기 때문입니다. Amazon EFS 여러 가용 영역에 데이터를 중복 저장하며 가용 영역과 상관없이 Amazon ECS 작업에서 지연 시간이 짧은 액세스를 제공합니다. Amazon EFS Amazon EC2 와 Fargate 트에서 모두 실행되는 작업을 지원합니다.

밀리 초 미만의 대기 시간이 필요하고 수평으로 확장 될 때 공유 파일 시스템이 필요하지 않은 트랜잭션 데이터베이스와 같은 응용 프로그램이 있다고 가정합니다. 이러한 애플리케이션의 경우 영구 스토리지에 Amazon EBS 볼륨을 사용하는 것이 좋습니다. 현재 Amazon ECS는 Amazon EC2에서 호스팅되는 작업에 대해서만 Amazon EBS 볼륨을 지원합니다. Fargate 의 작업에는 Amazon EBS 볼륨에 대한 Support 사용할 수 없습니다. Amazon ECS 작업에 Amazon EBS 볼륨을 사용하기 전에 먼저 Amazon EBS 볼륨을 컨테이너 인스턴스에 연결하고 작업 수명 주기와 별도로 볼륨을 관리해야 합니다.

Windows 인스턴스가 포함된 클러스터의 경우 Windows 파일 서버용 Amazon FSx는 컨테이너에 영구 스토리지를 제공합니다. Amazon FSx for Windows File Server 파일 시스템은 다중 AZ 배포를 지원합니다. 이러한 배포를 통해 여러 가용 영역에서 실행되는 Amazon ECS 작업과 파일 시스템을 공유할 수 있습니다.

또한 바인드 마운트 또는 Docker 볼륨을 사용하여 Amazon EC2에서 호스팅되는 Amazon ECS 작업의 데이터 지속성을 위해 Amazon EC2 인스턴스 스토리지를 사용할 수 있습니다. 바인드 마운트 또는

Docker 볼륨을 사용할 때 컨테이너는 컨테이너 인스턴스 파일 시스템에 데이터를 저장합니다. 컨테이너 저장을 위해 호스트 파일 시스템을 사용하는 한 가지 제한은 데이터가 한 번에 단일 컨테이너 인스턴스에서만 사용할 수 있다는 것입니다. 즉, 컨테이너는 데이터가있는 호스트에서만 실행될 수 있습니다. 따라서 데이터 복제가 응용 프로그램 수준에서 처리되는 경우에만 호스트 저장소를 사용하는 것이 좋습니다.

Amazon EFS 볼륨

Amazon Elastic File System (Amazon EFS) 은 단순하고 확장 가능하며 탄력적 NFS 파일 시스템을 제공합니다. 애플리케이션을 중단하지 않고 필요에 따라 페타바이트까지 확장할 수 있도록 설계되었습니다. 파일을 추가 및 제거하면 확장 또는 축소할 수 있습니다.

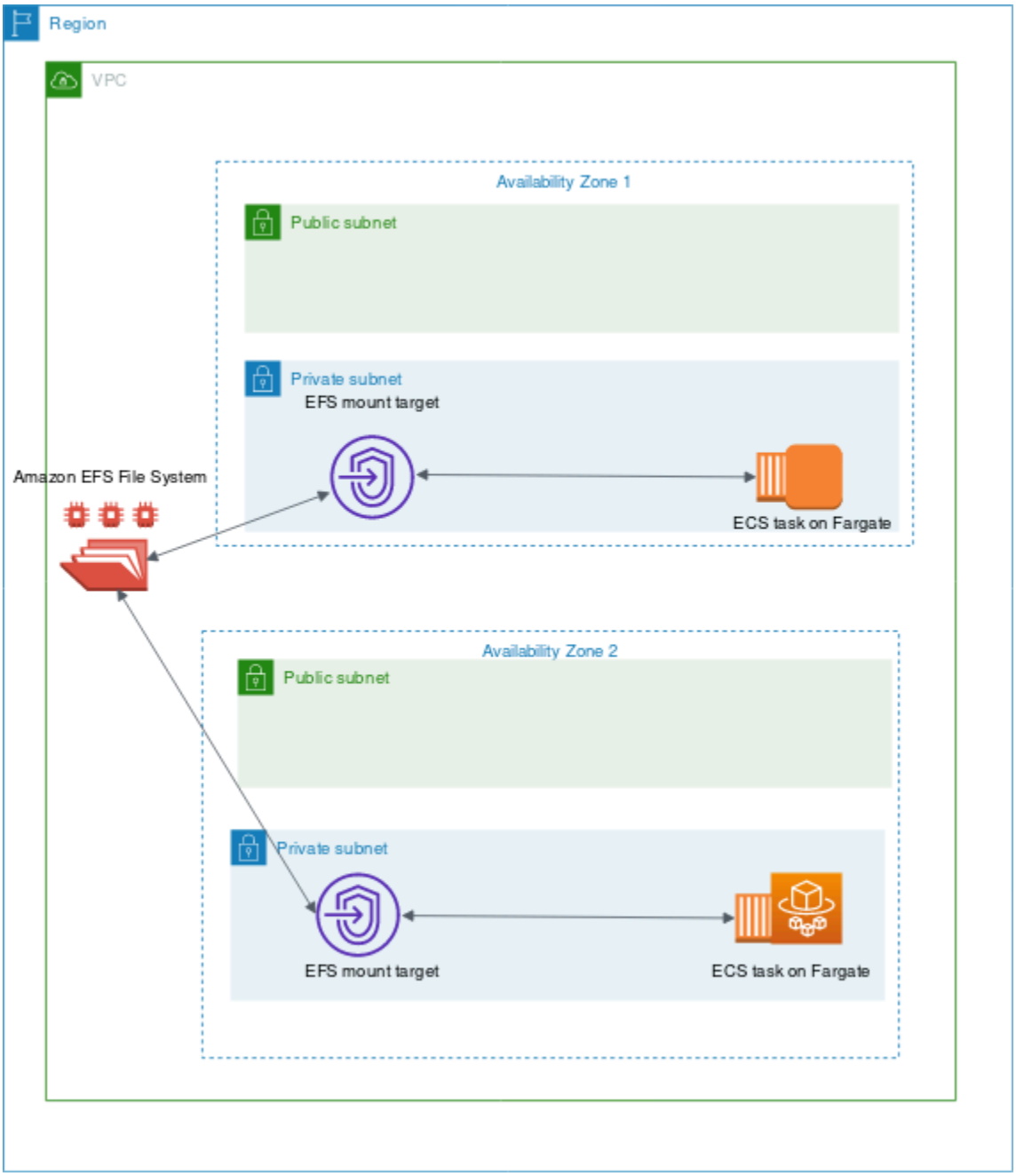
Amazon EFS 볼륨을 사용하여 영구 스토리지를 제공함으로써 Amazon ECS에서 상태 저장 애플리케이션을 실행할 수 있습니다. Amazon EC2 인스턴스 또는 플랫폼 버전을 사용하는 Fargate 에서 실행되는 Amazon ECS 작업`1.4.0`및 이후 버전에서는 기존 Amazon EFS 파일 시스템을 마운트할 수 있습니다. 여러 컨테이너가 Amazon EFS 파일 시스템을 동시에 마운트하고 액세스할 수 있다는 점을 감안할 때, 태스크는 호스팅 위치에 관계없이 동일한 데이터 세트에 액세스할 수 있습니다.

Amazon EFS 파일 시스템을 컨테이너에 탑재하려면 Amazon EFS 파일 시스템 및 컨테이너 탑재 지점을 참조할 수 있습니다. 다음은 Amazon EFS 컨테이너 스토리지에 사용하는 작업 정의의 조각입니다.

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

```
]
```

Amazon EFS 단일 리전 내 여러 가용 영역에 걸쳐 데이터를 중복 저장합니다. Amazon ECS 작업은 가용 영역에서 Amazon EFS 마운트 대상을 사용하여 Amazon EFS 파일 시스템을 마운트합니다. Amazon ECS 작업은 Amazon EFS 파일 시스템에 작업이 실행되는 가용 영역에 마운트 대상이 있는 경우에만 Amazon EFS 파일 시스템을 마운트할 수 있습니다. 따라서 Amazon ECS 작업을 호스팅하려는 모든 가용 영역에 Amazon EFS 마운트 대상을 생성하는 것이 가장 좋습니다.



자세한 내용은 단원을 참조하십시오.Amazon EFS 볼륨의Amazon Elastic Container Service.

보안 및 액세스 제어

Amazon EFS는 Amazon EFS 파일 시스템에 저장된 데이터를 안전하게 보호하고 필요한 애플리케이션에서만 액세스할 수 있도록 하는 데 사용할 수 있는 액세스 제어 기능을 제공합니다. 유휴 데이터 암호화와 전송 중 암호화를 사용하도록 설정하여 데이터를 보호할 수 있습니다. 자세한 내용은 단원을 참조하십시오.Amazon EFS 에서의 데이터 암호화의Amazon Elastic File System 사용자.

데이터 암호화 외에도 Amazon EFS 사용하여 파일 시스템에 대한 액세스를 제한할 수도 있습니다. EFS S에서 액세스 제어를 구현하는 방법에는 세 가지가 있습니다.

- 보안 그룹- Amazon EFS 마운트 대상을 사용하면 네트워크 트래픽을 허용 및 거부하는 데 사용되는 보안 그룹을 구성할 수 있습니다. Amazon EFS FS에 연결된 보안 그룹을 구성하여 Amazon ECS 인스턴스에 연결된 보안 그룹에서 NFS 트래픽 (포트 2049) 을 허용하거나`awsvpc`네트워크 모드의 경우 Amazon ECS 작업을 수행해야 합니다.
- IAM— IAM을 사용하여 Amazon EFS 파일 시스템에 대한 액세스를 제한할 수 있습니다. Amazon ECS 작업을 구성하면 EFS 파일 시스템을 마운트하려면 파일 시스템 액세스를 위해 IAM 역할이 필요합니다. 자세한 내용은 단원을 참조하십시오.IAM을 사용하여 파일 시스템 데이터 액세스 제어의Amazon Elastic File System 사용자.

  또한 클라이언트가 Amazon EFS 파일 시스템에 연결할 때 TLS를 사용하도록 요구하는 등의 사전 정의된 조건을 적용할 수 있습니다. 자세한 내용은 단원을 참조하십시오.클라이언트용 아마존 EFS 조건 키의Amazon Elastic File System 사용자.
- Amazon EFS 액세스 포인트— Amazon EFS 액세스 포인트는 Amazon EFS 파일 시스템에 대한 애플리케이션별 진입점입니다. 액세스 포인트를 사용하여 액세스 포인트를 통해 이루어지는 모든 파일 시스템 요청에 대해 사용자의 POSIX 그룹을 포함한 사용자 자격 증명을 적용할 수 있습니다. 액세스 포인트는 또한 파일 시스템에 대해 다른 루트 디렉터리를 적용할 수 있습니다. 클라이언트가 지정된 디렉터리 또는 하위 디렉터리의 데이터에만 액세스할 수 있도록 합니다.

보안을 극대화하기 위해 Amazon EFS 파일 시스템에 세 가지 액세스 제어를 모두 구현하는 것이 좋습니다. 예를 들어 컨테이너 인스턴스 또는 Amazon ECS 작업과 연결된 보안 그룹의 수신 NFS 트래픽만 허용하도록 Amazon EFS 마운트 지점에 연결된 보안 그룹을 구성할 수 있습니다. 또한 연결이 허용된 보안 그룹에서 시작된 경우에도 IAM 역할이 파일 시스템에 액세스하도록 요구하도록 Amazon EFS 구성할 수 있습니다. 마지막으로 Amazon EFS 액세스 포인트를 사용하여 POSIX 사용자 권한을 적용하고 애플리케이션에 대한 루트 디렉토리를 지정할 수 있습니다.

다음 작업 정의 조각은 액세스 포인트를 사용하여 Amazon EFS 파일 시스템을 탑재하는 방법을 보여줍니다.

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

Performance

Amazon EFS 두 가지 성능 모드를 제공합니다. 범용 및 최대 I/O 범용 범용 컨텐츠 관리 시스템 및 CI/CD 툴과 같은 지연 시간에 민감한 애플리케이션에 적합합니다. 반면, Max I/O 파일 시스템은 데이터 분석, 미디어 처리 및 기계 학습과 같은 워크로드에 적합합니다. 이러한 워크로드는 수백 또는 수천 개의 컨테이너에서 병렬 작업을 수행해야 하며 가능한 가장 높은 집계 처리량과 IOPS를 필요로 합니다. 자세한 내용은 단원을 참조하십시오.Amazon EFS 성능 모드의Amazon Elastic File System 사용자.

지연 시간에 민감한 일부 워크로드는 최대 I/O 성능 모드에서 제공하는 더 높은 I/O 수준과 범용 성능 모드에서 제공하는 더 낮은 지연 시간이 모두 필요합니다. 이러한 유형의 워크로드에는 여러 범용 성능 모드 파일 시스템을 생성하는 것이 좋습니다. 이렇게 하면 워크로드 및 애플리케이션이 지원할 수 있는 한, 모든 파일 시스템에 애플리케이션 워크로드를 분산시킬 수 있습니다.

Throughput

모든 Amazon EFS 파일 시스템에는 연결된 측정된 처리량이 있으며, 이 처리량은프로비저닝된 처리량를 사용하는 파일 시스템의 경우 EFS 표준 또는 One Zone 저장소 클래스에 저장된 데이터의 양버스트 처리량. 자세한 내용은 단원을 참조하십시오.측정된 처리량 이해의Amazon Elastic File System 사용자.

Amazon EFS 파일 시스템의 기본 처리량 모드는 버스트 모드입니다. 버스트 모드를 사용하면 파일 시스템이 증가함에 따라 파일 시스템에서 사용할 수 있는 처리량이 확장되거나 확장됩니다. 파일 기반 워크로드는 일반적으로 급증하기 때문에 일정 기간 동안 처리량이 높고 나머지 기간에는 처리량이 낮게 요구되기 때문에 Amazon EFS 일정 기간 동안 높은 처리량을 제공하도록 설계되었습니다. 또한 많은 워크로드가 읽기가 많기 때문에 읽기 작업은 다른 NFS 작업 (예: 쓰기) 에 대해 1:3 비율로 측정됩니다.

모든 Amazon EFS 파일 시스템은 Amazon EFS 표준 또는 Amazon EFS 원 존 스토리지의 각 TB에 대해 50MB/s의 일관된 기본 성능을 제공합니다. 모든 파일 시스템 (크기에 관계없이) 은 100MB/s로 버스트할 수 있습니다. EFS 표준 또는 EFS 원 존 스토리지가 1TB 이상인 파일 시스템은 각 TB에 대해 100MB/s로 버스트할 수 있습니다. 읽기 작업은 1:3 비율로 측정되므로 읽기 처리량의 각 TiB에 대해 최대 300 MIBS/s까지 구동할 수 있습니다. 파일 시스템에 데이터를 추가하면 파일 시스템에서 사용할 수 있는 최대 처리량이 Amazon EFS Standard 스토리지 클래스의 스토리지에 따라 선형적으로 자동으로 확장됩니다. 저장된 데이터 양보다 더 많은 처리량이 필요한 경우 프로비저닝된 처리량을 워크로드에 필요한 특정 양으로 구성할 수 있습니다.

파일 시스템 처리량은 파일 시스템에 연결된 모든 Amazon EC2 인스턴스에서 공유됩니다. 예를 들어 100MB/s의 처리량으로 버스트할 수 있는 1TB 파일 시스템은 단일 Amazon EC2 인스턴스에서 100MB/s로 구동할 수 있습니다. 자세한 내용은 단원을 참조하십시오.Amazon EFS 성능의Amazon Elastic File System 사용자.

비용 최적화

Amazon EFS 스토리지 확장을 간소화합니다. Amazon EFS 파일 시스템은 데이터를 추가함에 따라 자동으로 확장됩니다. 특히 Amazon EFS버스트 처리량모드에서 Amazon EFS 의 처리량은 표준 스토리지 클래스의 파일 시스템 크기가 커짐에 따라 확장됩니다. EFS 파일 시스템에서 프로비저닝된 처리량에 대한 추가 비용을 지불하지 않고 처리량을 향상시키기 위해 Amazon EFS 파일 시스템을 여러 애플리케이션과 공유할 수 있습니다. Amazon EFS 액세스 포인트를 사용하면 공유 Amazon EFS 파일 시스템에서 스토리지 격리를 구현할 수 있습니다. 이렇게 하면 응용 프로그램이 여전히 동일한 파일 시스템을 공유하더라도 사용자가 권한을 부여하지 않으면 데이터에 액세스할 수 없습니다.

데이터가 증가함에 따라 Amazon EFS 사용하면 자주 액세스하지 않는 파일을 하위 스토리지 클래스로 자동 이동할 수 있습니다. Amazon EFS 표준 자주 액세스 (IA) 스토리지 클래스는 매일 액세스하지 않는 파일의 스토리지 비용을 줄여줍니다. Amazon EFS 가 제공하는 고가용성, 높은 내구성, 탄력성 및 POSIX 파일 시스템 액세스에 영향을 주지 않습니다. 자세한 내용은 단원을 참조하십시오.Amazon EFS 스토리지 클래스의Amazon Elastic File System 사용자.

Amazon EFS 수명 주기 정책을 사용하여 액세스 빈도가 낮은 파일을 Amazon EFS IA 스토리지로 이동하여 비용을 자동으로 절감할 수 있습니다. 자세한 내용은 단원을 참조하십시오.Amazon EFS 수명 주기 관리의Amazon Elastic File System 사용자.

Amazon EFS 파일 시스템을 생성할 때 Amazon EFS가 데이터를 여러 가용 영역 (표준) 에 복제할지 아니면 데이터를 단일 가용 영역 내에 중복 저장할지를 선택할 수 있습니다. Amazon EFS One Zone 스토리지 클래스는 Amazon EFS 표준 스토리지 클래스에 비해 스토리지 비용을 크게 절감할 수 있습니다. 다중 AZ 복원력이 필요하지 않은 워크로드에 Amazon EFS One Zone 스토리지 클래스를 사용하는 것이 좋습니다. 액세스 빈도가 낮은 파일을 Amazon EFS One Zone 액세스 영역으로 이동하여

Amazon EFS One Zone 스토리지의 비용을 더 절감할 수 있습니다. 자세한 내용은 단원을 참조하십시오.Amazon EFS 액세스.

데이터 보호

Amazon EFS 표준 스토리지 클래스를 사용하여 파일 시스템의 여러 가용 영역에 데이터를 중복 저장합니다. Amazon EFS One Zone 스토리지 클래스를 선택하면 데이터가 단일 가용 영역 내에 중복 저장됩니다. 또한 Amazon EFS 1년 동안 99.999999999% (11 9) 의 내구성을 제공하도록 설계되었습니다.

다른 환경과 마찬가지로 백업을 수행하고 실수로 삭제하지 않도록 보호하는 것이 가장 좋습니다. Amazon EFS 데이터의 경우 이러한 모범 사례에는AWS Backup. Amazon EFS One Zone 스토리지 클래스를 사용하는 파일 시스템은 사용자가 이 기능을 비활성화하도록 선택하지 않는 한 파일 시스템 생성 시 기본적으로 파일을 자동으로 백업하도록 구성됩니다. 자세한 내용은 단원을 참조하십시오.Amazon EFS 데이터 보호의Amazon Elastic File System 사용자.

사용 사례

Amazon EFS 파일이 추가 및 제거될 때 자동으로 확장 및 축소되는 병렬 공유 액세스를 제공합니다. 따라서 Amazon EFS 짧은 지연 시간, 높은 처리량, 읽기 후 쓰기 일관성 등의 기능을 갖춘 스토리지가 필요한 모든 애플리케이션에 적합합니다. Amazon EFS 수평으로 확장되고 공유 파일 시스템이 필요한 애플리케이션에 이상적인 스토리지 백엔드입니다. 데이터 분석, 미디어 처리, 콘텐츠 관리 및 웹 서비스 등의 워크로드는 일반적인 Amazon EFS 사용 사례의 일부입니다.

Amazon EFS S가 적합하지 않을 수 있는 한 가지 사용 사례는 밀리초 미만의 지연 시간이 필요한 애플리케이션에 적합합니다. 이것은 일반적으로 트랜잭션 데이터베이스 시스템에 대한 요구 사항입니다. 지연 시간에 민감한 애플리케이션에 Amazon EFS 사용할 때 미치는 영향을 확인하려면 스토리지 성능 테스트를 실행하는 것이 좋습니다. Amazon EFS 사용할 때 애플리케이션 성능이 저하될 경우 Nitro 인스턴스에서 밀리초 미만의 낮은 I/O 지연 시간을 제공하는 Amazon EBS io2 블록 Express를 고려하십시오. 자세한 내용은 단원을 참조하십시오.Amazon EBS 볼륨 유형의Linux 인스턴스용 Amazon EC2 사용 설명서.

일부 응용 프로그램은 기본 저장소가 예기치 않게 변경되면 실패합니다. 따라서 Amazon EFS 이러한 애플리케이션에 가장 적합한 선택이 아닙니다. 대신 여러 위치에서 동시에 액세스할 수 없는 스토리지 시스템을 사용하는 것이 좋습니다.

도커 볼륨

Docker 볼륨은 컨테이너가 호스트의 파일 시스템에서 디렉토리를 마운트하여 데이터를 유지할 수있게 하는 Docker 컨테이너 런타임의 기능입니다. 도커 볼륨 드라이버 (플러그인이라고도 함) 는 컨테이너

볼륨을 Amazon EBS와 같은 외부 스토리지 시스템과 통합하는 데 사용됩니다. 도커 볼륨은 Amazon EC2 인스턴스에서 Amazon ECS 작업을 호스팅하는 경우에만 지원됩니다.

Amazon ECS 작업은 Docker 볼륨을 사용하여 Amazon EBS 볼륨을 사용하여 데이터를 유지할 수 있습니다. 이 작업은 Amazon EBS 볼륨을 Amazon EC2 인스턴스에 연결한 다음 Docker 볼륨을 사용하여 작업에 볼륨을 마운트하는 방식으로 수행됩니다. Docker 볼륨은 호스트의 여러 Amazon ECS 작업 간에 공유할 수 있습니다.

Docker 볼륨의 한계는 작업에서 사용하는 파일 시스템이 특정 Amazon EC2 인스턴스에 연결되어 있다는 것입니다. 어떤 이유로든 인스턴스가 중지되고 작업이 다른 인스턴스에 배치되면 데이터가 손실됩니다. 인스턴스에 작업을 할당하여 관련 EBS 볼륨을 항상 작업에 사용할 수 있도록 할 수 있습니다.

자세한 내용은 단원을 참조하십시오.도커 볼륨의Amazon Elastic Container Service.

Amazon EBS 볼륨 수명 주기

컨테이너 스토리지와 Amazon EBS에는 두 가지 주요 사용 패턴이 있습니다. 첫 번째는 응용 프로그램이 데이터를 유지하고 컨테이너가 종료 될 때 데이터 손실을 방지해야하는 경우입니다. 이러한 유형의 응용 프로그램의 예는 MySQL 과 같은 트랜잭션 데이터베이스입니다. MySQL 작업이 종료되면 다른 작업은이를 대체 할 것으로 예상됩니다. 이 시나리오에서는 볼륨의 수명 주기가 작업의 수명 주기에서 분리됩니다. EBS를 사용하여 컨테이너 데이터를 유지하는 경우 작업 배치 제약 조건을 사용하여 EBS 볼륨이 연결된 단일 호스트로 작업 배치를 제한하는 것이 좋습니다.

두 번째는 볼륨의 수명주기가 작업 수명주기와 독립적일 때입니다. 이 기능은 성능이 높고 대기 시간이 짧은 스토리지가 필요하지만 작업이 종료된 후에도 데이터를 유지할 필요가 없는 애플리케이션에 특히 유용합니다. 예를 들어 대량의 데이터를 처리하는 ETL 워크로드의 경우 처리량이 높은 스토리지가 필요할 수 있습니다. Amazon EBS는 256,000 IOPS까지 프로비저닝하는 고성능 볼륨을 제공하므로 이러한 유형의 워크로드에 적합합니다. 작업이 종료되면 대체 복제본을 클러스터의 모든 Amazon EC2 호스트에 안전하게 배치할 수 있습니다. 작업에 성능 요구 사항을 충족할 수 있는 스토리지 백엔드에 액세스할 수 있는 한 작업이 해당 기능을 수행할 수 있습니다. 따라서 이 경우에는 작업 배치 구속이 필요하지 않습니다.

클러스터의 Amazon EC2 인스턴스에 여러 유형의 Amazon EBS 볼륨이 연결되어 있는 경우 작업 배치 제약 조건을 사용하여 적절한 Amazon EBS 볼륨이 연결된 인스턴스에 작업이 배치되도록 할 수 있습니다. 예를 들어 클러스터에 일부 인스턴스가 있는 경우를 예로 들어 보겠습니다.gp2볼륨을 사용하고 다른 볼륨은io1볼륨을 참조하십시오. 당신은 인스턴스에 사용자 정의 속성을 첨부 할 수 있습니다io1볼륨을 확장한 다음 작업 배치 제약 조건을 사용하여 I/O 집약적 인 작업이 항상io1볼륨을 참조하십시오.

다음AWS CLI명령은 Amazon ECS 컨테이너 인스턴스에 속성을 배치하는 데 사용됩니다.

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

Amazon EBS 데이터 가용성

컨테이너는 일반적으로 수명이 짧고 자주 생성되며 응용 프로그램이 수평으로 확장 및 축소됨에 따라 종료됩니다. 여러 가용 영역에서 워크로드를 실행하여 애플리케이션의 가용성을 높이는 것이 좋습니다. Amazon ECS는 작업 배치 전략 및 작업 배치 제약 조건을 사용하여 작업 배치를 제어할 수 있는 방법을 제공합니다. 워크로드가 Amazon EBS 볼륨을 사용하여 데이터를 유지하는 경우 작업을 Amazon EBS 볼륨과 동일한 가용 영역에 배치해야 합니다. 또한 작업을 배치할 수 있는 가용 영역을 제한하는 배치 제약 조건을 설정하는 것이 좋습니다. 이렇게 하면 작업 및 해당 볼륨이 항상 동일한 가용 영역에 위치할 수 있습니다.

독립 실행형 작업을 실행할 때 가용 영역 속성을 사용하여 배치 제약 조건을 설정하여 작업이 배치되는 가용 영역을 제어할 수 있습니다.

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

여러 가용 영역에서 실행할 수 있는 애플리케이션을 실행할 때는 각 가용 영역에 대해 다른 Amazon ECS 서비스를 생성하는 것이 좋습니다. Amazon EBS 볼륨이 필요한 작업은 항상 관련 볼륨과 동일한 가용 영역에 배치됩니다.

각 가용 영역에 컨테이너 인스턴스를 생성하고 Amazon EBS 볼륨을시작 템플릿및사용자 지정 속성를 인스턴스에 추가하여 Amazon ECS 클러스터의 다른 컨테이너 인스턴스와 구별할 수 있습니다. 서비스를 생성할 때 Amazon ECS가 적절한 가용 영역과 인스턴스에 작업을 배치하도록 작업 배치 제약 조건을 구성합니다. 자세한 내용은 단원을 참조하십시오.작업 배치 제약의Amazon Elastic Container Service.

도커 볼륨 플러그인

Portworx와 같은 Docker 플러그인은 Docker 볼륨과 Amazon EBS 볼륨 간의 추상화를 제공합니다. 이러한 플러그인은 볼륨이 필요한 작업이 시작될 때 Amazon EBS 볼륨을 동적으로 생성할 수 있습니다. Portworx는 컨테이너가 종료되면 볼륨을 새 호스트에 연결할 수 있으며 후속 복제본은 다른 컨테이너 인스턴스에 배치됩니다. 또한 각 컨테이너의 볼륨 데이터를 Amazon ECS 노드와 가용 영역 간에 복제합니다. 자세한 내용은 단원을 참조하십시오.포트워스.

Amazon FSx for Windows File Server

Windows 파일 서버용 Amazon FSX는 업계 표준 SMB (서버 메시지 블록) 프로토콜을 통해 액세스할 수 있는 완전 관리형, 안정성 및 확장 가능한 파일 스토리지를 제공합니다. Windows Server를 기반으로 하여 사용자 할당량, 최종 사용자 파일 복원 및 Microsoft Active Directory (AD) 통합과 같은 광범위한 관리 기능을 제공합니다. 단일 AZ 및 다중 AZ 배포 옵션, 완전 관리형 백업, 저장 및 전송 중인 데이터 암호화를 제공합니다.

Amazon ECS는 Amazon ECS 윈도우 작업 정의에서 윈도우 파일 서버용 Amazon FSX를 사용할 수 있도록 지원하므로 글로벌 매핑이라는 SMB 기능을 사용하여 SMBv3 프로토콜을 통해 영구 스토리지를 마운트 지점으로 사용할 수 있습니다.

Windows 파일 서버용 Amazon FSX 및 Amazon ECS 통합을 설정하려면 Windows 컨테이너 인스턴스가 AD DS (액티브 디렉터리 도메인 서비스) 의 도메인 구성원이어야 하며AWS Directory Service for Microsoft Active Directory, 온프레미스 액티브 디렉토리 또는 Amazon EC2 자체 호스팅 액티브 디렉토리.AWS Secrets Manager는 Windows 컨테이너 인스턴스에서 공유를 매핑하는 데 사용되는 Active Directory 자격 증명의 사용자 이름 및 암호와 같은 중요한 데이터를 저장하는 데 사용됩니다.

컨테이너에 Amazon FSx for Windows File Server 파일 시스템 볼륨을 사용하려면 작업 정의에 볼륨 및 탑재 지점 구성을 지정해야 합니다. 다음은 컨테이너 스토리지에 Amazon FSx for Windows File Server (Windows File Server) 를 사용하는 작업 정의의 조각입니다.

```
{
 "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"
  }
 }]
}
```

자세한 내용은 단원을 참조하십시오.Amazon FSx for Windows File Server x의Amazon Elastic Container Service.

보안 및 액세스 제어

Amazon FSx for Windows File Server 파일 시스템에 저장된 데이터를 안전하게 보호하고 필요한 애플리케이션에서만 액세스할 수 있도록 하는 데 사용할 수 있는 다음과 같은 액세스 제어 기능을 제공합니다.

데이터 암호화

Amazon FSx for Windows File Server 는 두 가지 파일 시스템 암호화를 지원합니다. 그들은 전송 중 데이터의 암호화와 유휴 데이터 암호화입니다. 전송 중인 데이터의 암호화는 SMB 프로토콜 3.0 이상을 지원하는 컨테이너 인스턴스에 매핑된 파일 공유에서 지원됩니다. Amazon FSx 파일 시스템을 생성할 때 사용 중인 데이터의 암호화가 자동으로 활성화됩니다. Amazon FSx 는 사용자가 애플리케이션을 수정할 필요 없이 파일 시스템에 액세스할 때 SMB 암호화를 사용하여 전송 중인 데이터를 자동으로 암호화합니다. 자세한 내용은 단원을 참조하십시오.Amazon FSx 에서의 데이터 암호화의Amazon FSx for Windows File Server x 사용자 설명서.

Windows ACL을 사용한 폴더 수준 액세스 제어

윈도우 Amazon EC2 인스턴스는 액티브 디렉토리 자격 증명을 사용하여 Amazon FSX 파일 공유에 액세스합니다. 세분화된 파일 및 폴더 수준의 액세스 제어를 위해 표준 ACL (Windows ACL) 을 사용합니다. 특정 작업에 매핑되는 공유 내의 특정 폴더에 대해 각각 자격 증명을 여러 개 만들 수 있습니다.

다음 예에서는 작업에 폴더에 대한 액세스 권한이 있습니다.App01Secrets Manager 저장된 자격 증명을 사용합니다. Amazon 리소스 이름 (ARN)1234.

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

또 다른 예에서는 작업에 폴더에 대한 액세스 권한이 있습니다.App02Secrets Manager 저장된 자격 증명을 사용합니다. ARN은 6789입니다.

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

사용 사례

컨테이너는 데이터를 유지하도록 설계되지 않았습니다. 그러나 일부 컨테이너화된 .NET 응용 프로그램은 응용 프로그램 출력을 저장하기 위해 로컬 폴더를 영구 저장소로 필요로 할 수 있습니다. Amazon FSx for Windows File Server for Windows File Server 는 컨테이너에 로컬 폴더를 제공합니다. 이를 통해 SMB 공유에 의해 지원되는 동일한 파일 시스템에서 여러 컨테이너가 읽기/쓰기를 할 수 있습니다.

보안 모범 사례

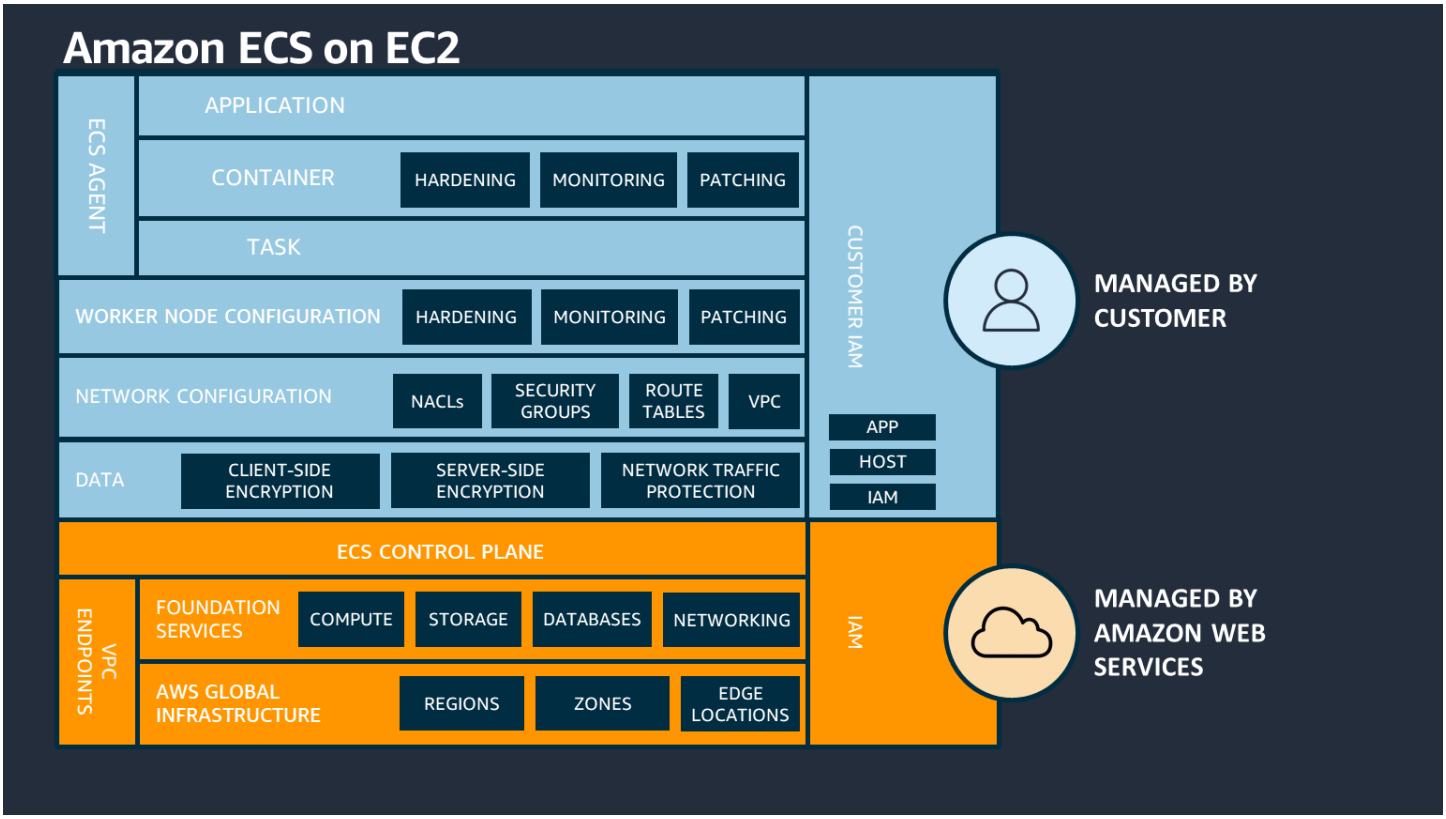
이 가이드에서는 정보, 시스템 및 Amazon ECS에 의존하는 기타 자산을 보호하기 위한 보안 및 규정 준수 권장 사항을 제공합니다. 또한 Amazon ECS 클러스터용으로 구축된 보안 제어 및 해당 클러스터가 지원하는 워크로드에 대해 더 잘 파악하기 위해 사용할 수 있는 몇 가지 위험 평가 및 완화 전략을 소개합니다. 이 가이드의 각 주제는 간략한 개요로 시작되며, 그 다음에는 Amazon ECS 클러스터를 보호하는 데 사용할 수 있는 권장 사항 및 모범 사례 목록이 나와 있습니다.

주제

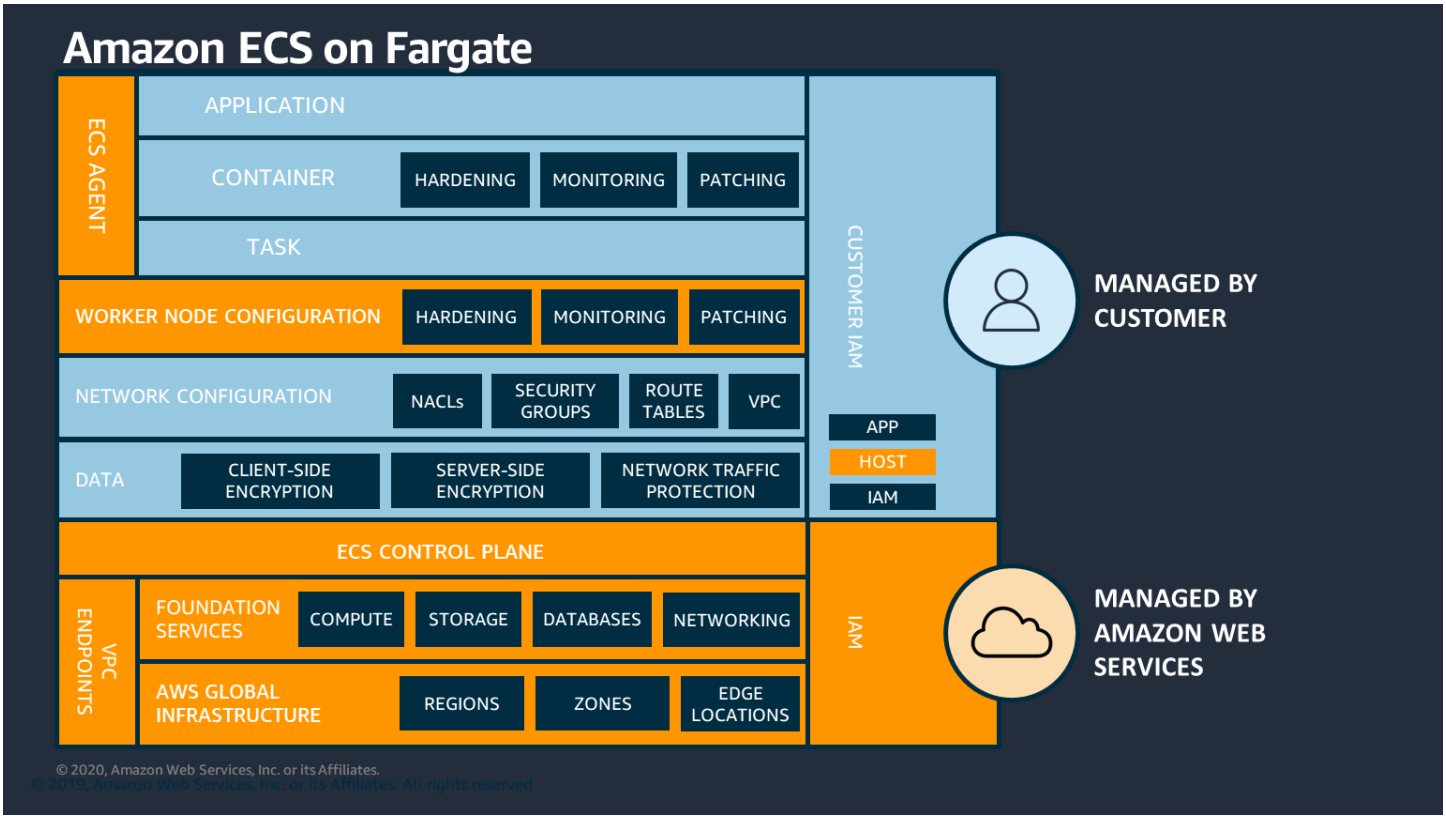
- 공동 책임 모델
- AWS Identity and Access Management
- Amazon ECS 작업에 IAM 역할 사용
- 네트워크 보안
- 보안 관리
- Compliance
- 로깅 및 모니터링
- AWS Fargate 보안
- 작업 및 컨테이너 보안
- 런타임 보안
- AWS파트너

공동 책임 모델

Amazon ECS와 같은 관리형 서비스의 보안 및 규정 준수는 사용자와AWS. 일반적으로 말하자면AWS는 클라우드의 “보안”을 담당하는 반면 고객은 클라우드의 “내부” 보안에 대한 책임이 있습니다.AWS는 안전하고 안정적인 서비스를 제공하는 데 필요한 인프라를 포함하여 Amazon ECS 제어부 관리를 담당합니다. 그리고, 이 가이드의 주제에 대한 책임은 크게 있습니다. 여기에는 데이터, 네트워크 및 런타임 보안뿐만 아니라 로깅 및 모니터링이 포함됩니다.



인프라 보안과 관련하여AWS에 대한 더 많은 책임을 지지AWS Fargate리소스를 다른 자체 관리 인스턴스보다 효율적으로 사용할 수 있습니다. Fargate,AWS는 클라우드의 기본 인스턴스 보안과 작업을 실행하는 데 사용되는 런타임을 관리합니다. 또한 Fargate 는 고객을 대신하여 인프라를 자동으로 확장합니다.



서비스를 클라우드로 확장하기 전에 책임지고 있는 보안 및 규정 준수의 측면을 이해해야 합니다.

공유 책임 모델에 대한 자세한 내용은 단원을 참조하십시오.공동 책임 모델.

AWS Identity and Access Management

다음을 수행할 수 있습니다.AWS Identity and Access Management(IAM) 을 사용하여AWS서비스 및 리소스를 인증 및 권한 부여 목적으로 규칙 기반 정책을 통해 사용할 수 있습니다. 보다 구체적으로 말하자면, 이 서비스를 통해AWSIAM 사용자, 그룹 또는 역할에 적용되는 정책을 사용하여 리소스를 관리할 수 있습니다. 이 세 가지 중에서 IAM 사용자는 리소스에 액세스할 수 있는 계정입니다. 그리고 IAM 역할은 IAM 외부의 특정 자격 증명과 연결되지 않은 인증된 자격 증명에서 가정할 수 있는 권한 집합입니다. 자세한 내용은 단원을 참조하십시오.IAM의 정책 및 권한?.

Amazon ECS에 대한 액세스 관리

IAM 정책을 생성하고 적용하여 Amazon ECS에 대한 액세스를 제어할 수 있습니다. 이러한 정책은 특정 리소스 집합에 적용되는 일련의 작업으로 구성됩니다. 정책 작업은 허용되거나 거부되는 작업 (예: Amazon ECS API) 목록을 정의하는 반면 리소스는 작업이 적용되는 Amazon ECS 객체를 제어합니다. 조건을 정책에 추가하여 범위를 좁힐 수 있습니다. 예를 들어 특정 태그 집합이 있는 작업에 대해서

만 작업을 수행할 수 있도록 정책을 작성할 수 있습니다. 자세한 내용은 단원을 참조하십시오.Amazon ECS가 IAM과 작동하는 방식의Amazon Elastic Container Service.

Recommendations

IAM 역할 및 정책을 설정할 때 다음을 수행하는 것이 좋습니다.

최소 권한 액세스 정책 따르기

사용자가 지정된 작업을 수행할 수 있도록 범위가 지정된 정책을 만듭니다. 예를 들어 개발자가 작업을 주기적으로 중지해야 하는 경우 해당 특정 작업만 허용하는 정책을 만듭니다. 다음 예제에서는 사용자가 특정`task_family`특정 Amazon 리소스 이름 (ARN) 이 있는 클러스터에 있습니다. 조건에서 ARN 참조하는 것도 리소스 수준 권한을 사용하는 예입니다. 리소스 수준 권한을 사용하여 작업을 적용할 리소스를 지정할 수 있습니다.

> **Note**
>
> 정책에서 ARN 참조할 때는 더 긴 새 ARN 형식을 사용합니다. 자세한 내용은 단원을 참조하십시오.Amazon 리소스 이름(ARN) 및 ID의Amazon Elastic Container Service.

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

클러스터 리소스가 관리 경계 역할을 하도록 합니다.

범위가 너무 좁으면 역할이 확산되고 관리 오버헤드가 증가할 수 있습니다. 특정 작업 또는 서비스로만 범위가 지정된 역할을 만드는 대신 클러스터로 범위가 지정된 역할을 만들고 해당 클러스터를 기본 관리 경계로 사용합니다.

자동화된 파이프라인을 생성하여 Amazon ECS API에서 최종 사용자 격리

Amazon ECS 클러스터에 애플리케이션을 자동으로 패키징하고 배포하는 파이프라인을 생성하여 사용자가 사용할 수 있는 작업을 제한할 수 있습니다. 이렇게 하면 작업 생성, 업데이트 및 삭제 작업을 파이프라인에 효과적으로 위임할 수 있습니다. 자세한 내용은 단원을 참조하십시오.자습서: CodePipeline 사용한 Amazon ECS 표준 배포의AWS CodePipeline사용 설명서.

추가된 보안 계층에 정책 조건 사용

추가 보안 계층이 필요한 경우 정책에 조건을 추가합니다. 이 기능은 권한이 있는 작업을 수행 중이거나 특정 리소스에 대해 수행할 수 있는 작업 집합을 제한해야 하는 경우에 유용할 수 있습니다. 다음 예제 정책에서는 클러스터를 삭제할 때 다중 요소 권한 부여가 필요합니다.

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

서비스에 적용된 태그는 해당 서비스의 일부인 모든 작업에 전파됩니다. 따라서 특정 태그를 사용하여 Amazon ECS 리소스로 범위가 지정된 역할을 생성할 수 있습니다. 다음 정책에서 IAM 보안 주체는 태그 키가`Department`의 태그 값`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"}
            }
        }
    ]
}
```

Amazon ECS API에 대한 액세스를 주기적으로 감사

사용자가 역할을 변경할 수 있습니다. 역할을 변경한 후에는 이전에 부여된 권한이 더 이상 적용되지 않을 수 있습니다. Amazon ECS API에 대한 액세스 권한이 있는 사용자와 해당 액세스가 여전히 보장되는지 여부를 감사해야 합니다. IAM을 사용자가 퇴사할 때 자동으로 액세스를 취소하는 사용자 수명 주기 관리 솔루션과 통합하는 것이 좋습니다. 자세한 내용은 단원을 참조하십시오.Amazon ECS 보안 감사 지침의Amazon Web Services 일반 참조.

Amazon ECS 작업에 IAM 역할 사용

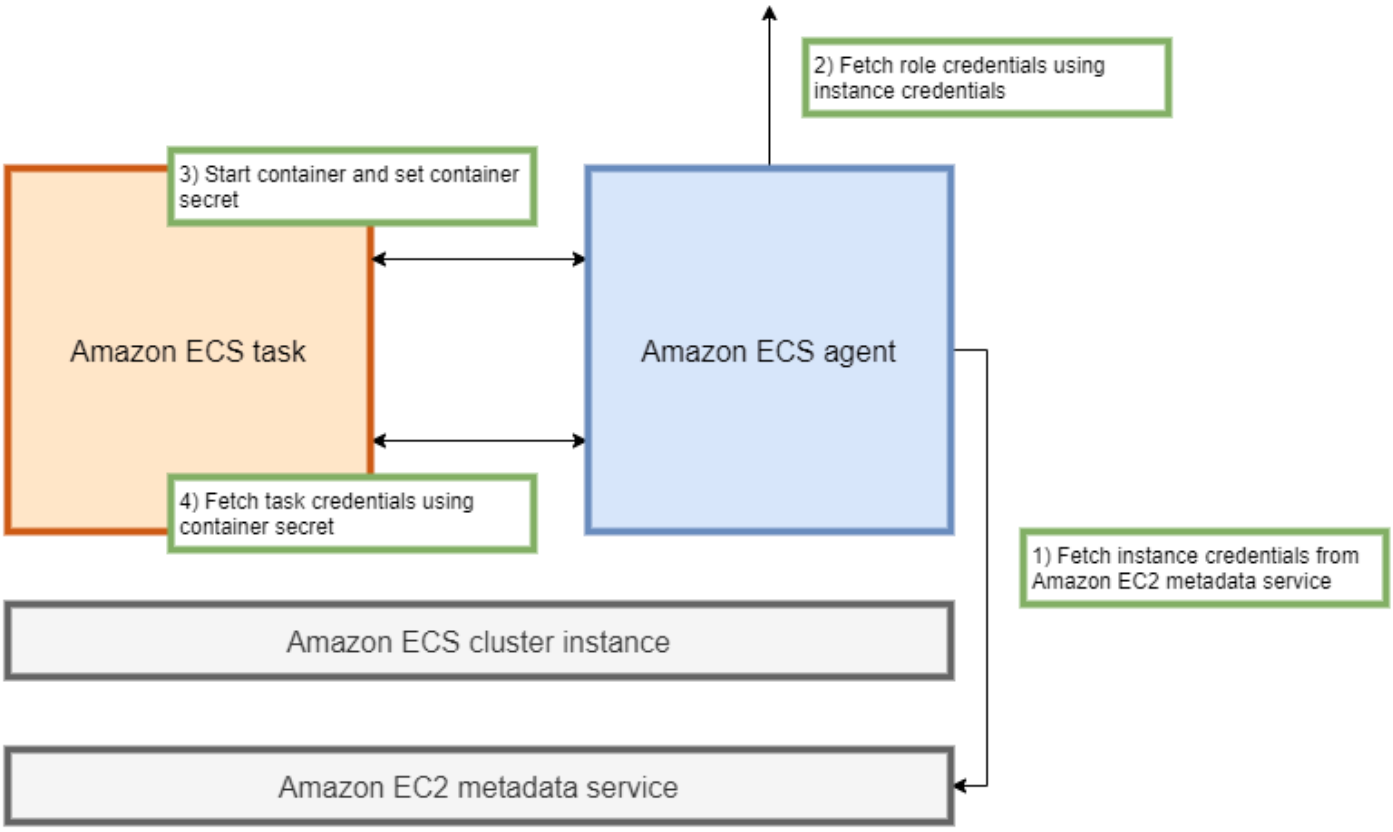
작업에 IAM 역할을 할당하는 것이 좋습니다. 그 역할은 실행 중인 Amazon EC2 인스턴스의 역할과 구별될 수 있습니다. 각 작업에 역할을 할당하면 권한이 부여된 액세스 원칙과 일치하며 작업 및 리소스를 보다 세부적으로 제어할 수 있습니다.

작업에 IAM 역할을 할당할 때 각 작업이 EC2 인스턴스에서 사용하는 것과 다른 IAM 역할을 가정할 수 있도록 다음 신뢰 정책을 사용해야 합니다. 이렇게 하면 작업이 EC2 인스턴스의 역할을 상속하지 않습니다.

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

```
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

작업 정의에 작업 역할을 추가하면 Amazon ECS 컨테이너 에이전트는 고유한 자격 증명 ID (예:12345678-90ab-cdef-1234-567890abcdef) 를 사용하여 작업을 수행할 수 있습니다. 그러면 이 토큰과 역할 자격 증명이 에이전트의 내부 캐시에 추가됩니다. 에이전트는 환경 변수를 채웁니다.AWS_CONTAINER_CREDENTIALS_RELATIVE_URI를 자격 증명 ID의 URI가 있는 컨테이너 (예:/v2/credentials/12345678-90ab-cdef-1234-567890abcdef).



Amazon ECS 컨테이너 에이전트의 IP 주소에 환경 변수를 추가하고curl명령을 결과 문자열에.

```
curl 192.0.2.0$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI
```

예상 출력은 다음과 같습니다.

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

최신 버전의AWSSDK는 이러한 자격 증명을AWS_CONTAINER_CREDENTIALS_RELATIVE_URI만들 때 환경 가변AWSAPI 호출입니다.

출력에는 보안 액세스 키 ID와 애플리케이션이 액세스하는 데 사용하는 보안 키로 구성된 액세스 키 쌍이 포함됩니다.AWS있습니다. 여기에는 토큰도 포함되어 있습니다.AWS에서 자격 증명이 유효한지 확인하는 데 사용하는 것입니다. 기본적으로 작업 역할을 사용하여 작업에 할당된 자격 증명은 6시간 동안 유효합니다. 그러면 Amazon ECS 컨테이너 에이전트에 의해 자동으로 교체됩니다.

작업 실행 역할

작업 실행 역할은 Amazon ECS 컨테이너 에이전트에 특정AWS사용자를 대신하여 API 작업을 실행합니다. 예를 들어, 를 사용할 때AWS Fargate에 따르면 Fargate 는 Amazon ECR에서 이미지를 가져오고 CloudWatch Logs 로그에 로그를 기록할 수 있는 IAM 역할이 필요합니다. IAM 역할은 작업이AWS Secrets Manager, 이미지 풀 비밀과 같은.

> **Note**
>
> 인증 된 사용자로 이미지를 가져 오는 경우도커 허브의 끌어오기 속도 제한. 자세한 내용은 단원을 참조하십시오.컨테이너 인스턴스에 대한 프라이빗 레지스트리 인증.
> Amazon ECR 및 Amazon ECR 공용을 사용하면 Docker가 부과하는 제한을 피할 수 있습니다. Amazon ECR에서 이미지를 가져오는 경우 네트워크 풀 시간을 단축하고 트래픽이 VPC 벗어날 때 데이터 전송 변경을 줄이는 데도 도움이 됩니다.

> **Important**
>
> Fargate 를 사용하는 경우 다음을 사용하여 개인 이미지 레지스트리에 인증해야합니다.repositoryCredentials. Amazon ECS 컨테이너 에이전트 환경 변수를 설정할 수 없습니다.ECS_ENGINE_AUTH_TYPE또는ECS_ENGINE_AUTH_DATA를 수정하거

나ecs.config파일을 사용하여 Fargate 에서 호스팅되는 작업을 확인하십시오. 자세한 내용은 단원을 참조하십시오.작업에 대한 개인 레지스트리 인증.

Amazon EC2 컨테이너 인스턴스 역할

Amazon ECS 컨테이너 에이전트는 Amazon ECS 클러스터의 각 Amazon EC2 인스턴스에서 실행되는 컨테이너입니다. Amazon ECS를 사용하여 초기화됩니다.init명령을 사용할 수 있습니다. 따라서 작업 역할을 통해 권한을 부여받을 수 없습니다. 대신 에이전트가 실행되는 Amazon EC2 인스턴스에 권한을 할당해야 합니다. 예제의 작업 목록AmazonEC2ContainerServiceforEC2Role정책을 부여해야 하는ecsInstanceRole. 이렇게 하지 않으면 인스턴스가 클러스터에 가입할 수 없습니다.

```
{
    "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": "*"
        }
    ]
}
```

이 정책에서ecr및logsAPI 작업을 사용하면 인스턴스에서 실행 중인 컨테이너가 Amazon ECR에서 이미지를 가져와 Amazon CloudWatch 에 로그를 작성할 수 있습니다. 이ecs작업을 통해

에이전트는 인스턴스를 등록 및 등록 취소하고 Amazon ECS 제어부와 통신할 수 있습니다. 이 중ecs:CreateCluster작업은 선택 사항입니다.

서비스 연결 역할

Amazon ECS의 서비스 연결 역할을 사용하여 사용자를 대신해 다른 서비스 API를 호출할 수 있는 권한을 Amazon ECS 서비스에 부여할 수 있습니다. Amazon ECS는 네트워크 인터페이스를 생성 및 삭제하고 대상 그룹에 대상을 등록 및 등록 취소할 수 있는 권한이 필요합니다. 또한 조정 정책을 만들고 삭제하는 데 필요한 권한이 필요합니다. 이러한 권한은 서비스 연결 역할을 통해 부여됩니다. 이 역할은 서비스를 처음 사용할 때 사용자를 대신하여 만들어집니다.

> **Note**
>
> 서비스 연결 역할을 실수로 삭제하면 해당 역할을 다시 만들 수 있습니다. 지침은 단원을 참조하십시오.서비스 연결 역할 생성.

Recommendations

IAM 역할 및 정책을 설정할 때 다음을 수행하는 것이 좋습니다.

Amazon EC2 메타데이터에 대한 액세스 차단

Amazon EC2 인스턴스에서 작업을 실행할 때는 컨테이너가 해당 인스턴스에 할당된 역할을 상속하지 못하도록 Amazon EC2 메타데이터에 대한 액세스를 차단하는 것이 좋습니다. 응용 프로그램이 호출해야하는 경우AWSAPI 작업을 사용하려면 작업에 IAM 역할을 대신 사용합니다.

에서 실행되는 작업을 방지하려면브릿지모드가 Amazon EC2 메타데이터에 액세스하지 못하도록 하려면 다음 명령을 실행하거나 인스턴스의 사용자 데이터를 업데이트합니다. 인스턴스의 사용자 데이터 업데이트에 대한 자세한 지침은 이AWSSupport 기사. 작업 정의 브리지 모드에 대한 자세한 내용은작업 정의.

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

재부팅 후에도 이 변경 사항을 유지하려면 Amazon 머신 이미지 (AMI) 에 해당하는 다음 명령을 실행합니다.

- Amazon Linux 2

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

- Amazon Linux

```
sudo service iptables save
```

사용하는 작업의 경우awsvpc네트워크 모드에서 환경 변수ECS_AWSVPC_BLOCK_IMDStotrue의/etc/ecs/ecs.config파일을 참조하십시오.

설정 해야 합니다.ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST변수를false에서 실행 중인 컨테이너를 방지하기 위해host네트워크가 Amazon EC2 메타데이터에 액세스하지 못하도록 합니다.

사용awsvpc네트워크 모드

네트워크 사용awsvpc네트워크 모드를 사용하여 서로 다른 작업 간 또는 작업과 Amazon VPC 내에서 실행되는 기타 서비스 간의 트래픽 흐름을 제한할 수 있습니다. 이렇게 하면 추가 보안 계층이 추가됩니다. 이awsvpc네트워크 모드는 Amazon EC2 실행되는 작업에 대한 작업 수준 네트워크 격리를 제공합니다. 그것은의 기본 모드입니다AWS Fargate. 보안 그룹을 작업에 할당하는 데 사용할 수 있는 유일한 네트워크 모드입니다.

IAM 액세스 관리자를 사용하여 역할 구체화

사용되지 않았거나 얼마 동안 사용되지 않은 작업은 모두 제거하는 것이 좋습니다. 이렇게하면 원치 않는 액세스가 발생하지 않습니다. 이렇게 하려면 IAM Access Advisor에서 생성된 결과를 검토한 다음 사용되지 않았거나 최근에 사용되지 않은 작업을 제거합니다. 이 작업은 다음 단계에 따라 수행할 수 있습니다.

다음 명령을 실행하여 참조된 정책에 대한 마지막 액세스 정보를 보여 주는 보고서를 생성합니다.

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

를 사용하려면JobId다음 명령을 실행할 수 있습니다. 이렇게 하면 보고서의 결과를 볼 수 있습니다.

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

자세한 내용은 단원을 참조하십시오.IAM 액세스 관리자.

모니터링AWS CloudTrail의심스러운 활동

모니터링할 수 있습니다.AWS CloudTrail의심스러운 활동에 대한. MoSTAWSAPI 호출을 에 로깅합니다.AWS CloudTrail를 이벤트로 사용합니다. 그들은에 의해 분석된다AWS CloudTrail통찰력, 그리고 관련된 의심스러운 행동에 대한 경고를 받게 됩니다.`write`API 호출입니다. 통화 볼륨에 스파이크가 포함될 수 있습니다. 이러한 알림에는 비정상적인 활동이 발생한 시간 및 API에 기여한 최상위 자격 증명 ARN 과 같은 정보가 포함됩니다.

IAM 역할을 가진 작업에 의해 수행되는 작업은AWS CloudTrail이벤트의`userIdentity`속성입니다. 다음 예제에서`arn`가정된 역할의 이름을 포함하며,`s3-write-go-bucket-role`, 작업 이름 뒤에`7e9894e088ad416eb5cab92afExample`.

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

> **Note**
>
> 역할을 가정하는 작업이 Amazon EC2 컨테이너 인스턴스에서 실행되는 경우 Amazon ECS 컨테이너 에이전트에 의해 요청이`/var/log/ecs/audit.log.YYYY-MM-DD-HH`형식 자세한 내용은 단원을 참조하십시오.작업 IAM 역할 로그및트레일에 대한 인사이트 이벤트 로깅.

네트워크 보안

네트워크 보안은 여러 하위 항목을 포괄하는 광범위한 주제입니다. 여기에는 전송 중 암호화, 네트워크 세분화 및 격리, 방화벽, 트래픽 라우팅 및 관찰 가능성이 포함됩니다.

전송 중 데이터 암호화

네트워크 트래픽을 암호화하면 권한이 없는 사용자가 네트워크를 통해 데이터가 전송될 때 데이터를 가로채거나 읽을 수 없습니다. Amazon ECS를 사용하면 다음 방법 중 하나를 사용하여 네트워크 암호화를 구현할 수 있습니다.

- 서비스 메시 (TLS) 사용:

다음으로 바꿉니다.AWS App Mesh에서 메시 끝점과 함께 배포되는 Envoy 프록시 간에 TLS 연결을 구성할 수 있습니다. 두 가지 예로는 가상 노드와 가상 게이트웨이가 있습니다. TLS 인증서는AWS Certificate Manager(ACM). 또는 자신의 개인 인증 기관에서 가져올 수 있습니다.

- TLS (전송 계층 보안)
- 서비스 간 트래픽 암호화 사용AWS App MeshACM 인증서 또는 고객 제공 인증서 사용
- TLS ACM 연습
- TLS 파일 연습
- 사절기

- Nitro 인스턴스 사용:

  기본적으로 트래픽은 다음 Nitro 인스턴스 유형 간에 자동으로 암호화됩니다. C5n, Gn, I3en, M5dn, M5dn, M5dn, M5dn, M5dn, M5dn, M5dn, M5dn, M 트래픽은 전송 게이트웨이, 로드 밸런서 또는 유사한 중개자를 통해 라우팅될 때 암호화되지 않습니다.

  - 전송 중 데이터 암호화
  - 2019년의 새로운 계정은 무엇입니까?
  - 이 강연: 인포스 2019

- Application Load Balancer SNI (서버 이름 표시) 사용

  ALB (Application Load Balancer) 및 NLB (Network Load Balancer) 는 SNI (서버 이름 표시) 를 지원합니다. SNI를 사용하면 여러 보안 응용 프로그램을 단일 수신기 뒤에 배치할 수 있습니다. 이를 위해 각각에는 자체 TLS 인증서가 있습니다. 다음을 사용하여 로드 밸런서에 대한 인증서를 프로비저닝하는 것이 좋습니다.AWS Certificate Manager(ACM) 을 클릭한 다음 리스너의 인증서 목록에 추가합니다. 이AWS로드 밸런서는 SNI와 함께 스마트 인증서 선택 알고리즘을 사용합니다. 클라이언트가 제공한 호스트 이름이 인증서 목록의 단일 인증서와 일치하면 로드 밸런서는 해당 인증서를 선택합니다. 클라이언트가 제공한 호스트 이름이 목록의 여러 인증서와 일치하면 로드 밸런서는 클라이언트가 지원할 수 있는 인증서를 선택합니다. 예를 들어 자체 서명된 인증서 또는 ACM을 통해 생성된 인증서가 있습니다.

  - SNI (Application Load Balancer 포함)
  - Network Load Balancer 포함

- TLS 인증서를 사용한 종단 간 암호화

  여기에는 작업과 함께 TLS 인증서를 배포하는 작업이 포함됩니다. 이 인증서란 자체 서명된 인증서이거나 신뢰할 수 있는 인증 기관의 인증서일 수 있습니다. 인증서에 대한 암호를 참조하여 인증서를

얻을 수 있습니다. 그렇지 않으면 ACM에 CSR (인증서 서명 요청) 을 발급한 다음 결과 암호를 공유 볼륨에 마운트하는 컨테이너를 실행하도록 선택할 수 있습니다.

- Amazon ECS와 함께 Network Load Balancer 사용하여 컨테이너에 대한 전송 계층 보안 유지
- TLS (전송 계층 보안) 를 컨테이너 2부까지 유지 관리합니다. AWS Private Certificate Authority 사용

작업 네트워킹

다음은 Amazon ECS의 작동 방식을 고려한 권장 사항입니다. Amazon ECS는 오버레이 네트워크를 사용하지 않습니다. 대신 작업이 서로 다른 네트워크 모드에서 작동하도록 구성됩니다. 예를 들어, 사용 하도록 구성 된 작업`bridge`모드는 각 호스트에서 실행되는 Docker 네트워크에서 라우팅 할 수 없는 IP 주소를 얻습니다. 사용 하도록 구성 된 작업`awsvpc`네트워크 모드는 호스트의 서브넷에서 IP 주소를 획득합니다. 로 구성된 작업`host`네트워킹은 호스트의 네트워크 인터페이스를 사용합니다.`awsvpc`은 기본 설정 네트워크 모드입니다. 이는 보안 그룹을 작업에 할당하는 데 사용할 수 있는 유일한 모드이기 때문입니다. 또한 사용할 수 있는 유일한 모드이기도 합니다.AWS Fargate아마존 ECS에서 작업을 수행할 수 있습니다.

작업에 대한 보안 그룹

사용 하도록 작업을 구성하는 것이 좋습니다.`awsvpc`네트워크 모드입니다. 이 모드를 사용하도록 작업을 구성하면 Amazon ECS 에이전트가 ENI (엘라스틱 네트워크 인터페이스) 를 자동으로 프로비저닝하고 작업에 연결합니다. ENI가 프로비저닝되면 작업은AWS보안 그룹입니다. 보안 그룹은 인바운드 및 아웃바운드 트래픽을 제어하는 데 사용할 수 있는 가상 방화벽 역할을 합니다.

서비스 메쉬 및 상호 전송 계층 보안 (MTL)

다음과 같은 서비스 메쉬를 사용할 수 있습니다.AWS App Mesh를 사용하여 네트워크 트래픽을 제어할 수 있습니다. 기본적으로 가상 노드는 가상 노드가 통신할 가상 서비스와 같이 구성된 서비스 백엔드와만 통신할 수 있습니다. 가상 노드가 메시 외부의 서비스와 통신해야 하는 경우`ALLOW_ALL`아웃바운드 필터를 사용하거나 외부 서비스에 대한 메시 내에 가상 노드를 만들어 자세한 내용은 단원을 참조하십시오.쿠버네테스 송신 방법 연습.

또한 App Mesh 는 클라이언트와 서버가 인증서를 사용하여 상호 인증되는 MTL (상호 전송 계층 보안) 을 사용할 수 있는 기능을 제공합니다. 그런 다음 클라이언트와 서버 간의 후속 통신은 TLS를 사용하여 암호화됩니다. 메시의 서비스 간에 MTL을 요구하면 트래픽이 신뢰할 수 있는 원본에서 발생하는지 확인할 수 있습니다. 자세한 내용은 다음 항목을 참조하십시오.

- MTLS 인증

- MTL 비밀 검색 서비스 (SDS) 연습
- MTLS 파일 연습

AWS PrivateLink

AWS PrivateLink는 네트워킹 기술로서 서로 다른AWS서비스 (아마존 ECS를 포함). 엔드포인트는 Amazon VPC 연결된 Internet Gateway (IGW) 가 없고 인터넷에 대한 대체 경로가 없는 샌드박스 환경에서 필요합니다. 사용AWS PrivateLink는 Amazon ECS 서비스에 대한 호출이 Amazon VPC 내에 유지되고 인터넷을 통과하지 않도록 합니다. 만드는 방법에 대한 지침은 다음과 같습니다.AWS PrivateLink엔드포인트에 대한 자세한 내용은Amazon ECS 인터페이스 Amazon VPC 엔드포인트.

> **Important**
>
> AWS Fargate작업에는AWS PrivateLink엔드포인트를 사용할 수 있습니다.

Amazon ECR과 Amazon ECS는 모두 엔드포인트 정책을 지원합니다. 이러한 정책을 통해 서비스의 API에 대한 액세스를 세분화할 수 있습니다. 예를 들어, 이미지를 레지스트리에 푸시하는 것만 허용하는 Amazon ECR에 대한 엔드포인트 정책을 생성할 수 있습니다.AWS계정을 참조하십시오. 이와 같은 정책을 사용하여 컨테이너 이미지를 통해 데이터가 유출되는 것을 방지하고 사용자가 승인된 Amazon ECR 레지스트리로 푸시할 수 있습니다. 자세한 내용은 단원을 참조하십시오.VPC 엔드포인트 정책 사용.

다음 정책은 모든AWS사용자 계정의 보안 주체를 사용하여 Amazon ECR 리포지토리에 대해서만 모든 작업을 수행할 수 있습니다.

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

당신은 더 새로운 사용하는 조건을 설정하여이를 향상시킬 수 있습니다`PrincipalOrgID`속성입니다. 이렇게 하면 사용자의 일부가 아닌 IAM 보안 주체가 이미지를 푸시하고 당기는 것을 방지할 수 있습니다.AWS Organizations. 자세한 내용은 단원을 참조하십시오.aws:PrincipalOrgID.

우리는 두 가지 모두에 동일한 정책을 적용하는 것이 좋습니다`com.amazonaws.region.ecr.dkr`및`com.amazonaws.region.ecr.api`엔드포인트.

Amazon ECS 컨테이너 에이전트 설정

Amazon ECS 컨테이너 에이전트 구성 파일에는 네트워크 보안과 관련된 몇 가지 환경 변수가 포함되어 있습니다.`ECS_AWSVPC_BLOCK_IMDS`및`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST`는 Amazon EC2 메타데이터에 대한 작업의 액세스를 차단하는 데 사용됩니다.`HTTP_PROXY`는 HTTP 프록시를 통해 라우팅하여 인터넷에 연결하도록 에이전트를 구성하는 데 사용됩니다. 프록시를 통해 라우팅되도록 에이전트 및 Docker 런타임을 구성하는 방법은HTTP 프록시 구성.

> **Important**
>
> 이러한 설정을 사용할 때 사용할 수 없습니다.AWS Fargate.

Recommendations

Amazon VPC, 로드 밸런서 및 네트워크를 설정할 때 다음을 수행하는 것이 좋습니다.

해당하는 경우 네트워크 암호화 사용

해당하는 경우 네트워크 암호화를 사용해야 합니다. PCI DSS와 같은 특정 규정 준수 프로그램은 데이터에 카드 소지자 데이터가 포함된 경우 전송 중인 데이터를 암호화해야 합니다. 워크로드에 비슷한 요구 사항이 있는 경우 네트워크 암호화를 구성합니다.

최신 브라우저는 안전하지 않은 사이트에 연결할 때 사용자에게 경고합니다. 서비스가 공용 부하 분산 장치에 의해 선두되는 경우 TLS/SSL을 사용하여 클라이언트 브라우저에서 로드 밸런서로의 트래픽을 암호화하고 보증이 필요한 경우 백엔드로 다시 암호화합니다.

사용`awsvpc`작업 간 또는 작업과 다른 네트워크 리소스 간의 트래픽을 제어해야 하는 경우 네트워크 모드 및 보안 그룹

를 사용해야 합니다.`awsvpc`네트워크 모드 및 보안 그룹을 사용하여 작업 간, 작업 및 기타 네트워크 리소스 간의 트래픽을 제어해야 하는 경우 ALB 뒤에있는 서비스의 경우 보안 그룹을 사용하여 ALB와 동일한 보안 그룹을 사용하는 다른 네트워크 리소스의 인바운드 트래픽만 허용하십시오. 애플리케이

션이 NLB 뒤에 있는 경우 Amazon VPC CIDR 범위의 인바운드 트래픽과 NLB에 할당된 고정 IP 주소만 허용하도록 작업의 보안 그룹을 구성합니다.

또한 Amazon RDS 데이터베이스와 같은 Amazon VPC 내의 다른 리소스와 태스크 간의 트래픽을 제어하는 데에도 보안 그룹을 사용해야 합니다.

네트워크 트래픽을 엄격하게 격리해야 하는 경우 별도의 Amazon VPC에 클러스터를 생성합니다.

네트워크 트래픽을 엄격하게 격리해야 하는 경우 별도의 Amazon VPC에 클러스터를 생성해야 합니다. 이러한 요구 사항을 준수하지 않아도 되는 워크로드가 있는 클러스터에서 엄격한 보안 요구 사항이 있는 워크로드를 실행하지 마십시오. 엄격한 네트워크 격리가 필요한 경우 별도의 Amazon VPC에 클러스터를 생성하고 Amazon VPC 엔드포인트를 사용하여 다른 Amazon VPC에 서비스를 선택적으로 노출합니다. 자세한 내용은 단원을 참조하십시오.Amazon VPC 엔드포인트.

구성AWS PrivateLink무상수리 시 엔드포인트

다음을 구성해야 합니다.AWS PrivateLink엔드포인트 엔드포인트 (보증 대상 경우) 보안 정책으로 인해 IGW (Internet Gateway) 를 Amazon VPC에 연결할 수 없는 경우AWS PrivateLink엔드포인트 및 기타 서비스 (예: Amazon ECR)AWS Secrets Manager및 Amazon CloudWatch 를 참조하십시오.

Amazon VPC 흐름 로그를 사용하여 장기 실행 작업으로 들어오고 나가는 트래픽 분석

Amazon VPC 흐름 로그를 사용하여 장기 실행 작업으로 들어오고 나가는 트래픽을 분석해야 합니다. 사용하는 작업`awsvpc`네트워크 모드는 자체 ENI를 얻습니다. 이렇게 하면 Amazon VPC 흐름 로그를 사용하여 개별 작업으로 들어오고 나가는 트래픽을 모니터링할 수 있습니다. Amazon VPC 흐름 로그 (v3) 에 대한 최근 업데이트로, vpc ID, 서브넷 ID, 인스턴스 ID 등의 트래픽 메타데이터로 로그를 보강합니다. 이 메타데이터는 조사 범위를 좁히는 데 사용할 수 있습니다. 자세한 내용은 단원을 참조하십시오.Amazon VPC 흐름 로그.

> **Note**
>
> 컨테이너의 일시적인 특성으로 인해 흐름 로그가 항상 다른 컨테이너 또는 컨테이너와 다른 네트워크 리소스 간의 트래픽 패턴을 분석하는 효과적인 방법은 아닐 수 있습니다.

보안 관리

API 키 및 데이터베이스 자격 증명과 같은 비밀은 응용 프로그램에서 다른 시스템에 액세스하는 데 자주 사용됩니다. 사용자 이름과 암호, 인증서 또는 API 키로 구성되는 경우가 많습니다. 이러한 비밀에

대한 액세스는 IAM을 사용하고 런타임 시 컨테이너에 삽입되는 특정 IAM 보안 주체로 제한되어야 합니다.

비밀은 완벽하게 컨테이너에 주입 할 수 있습니다.AWS Secrets Manager및 Amazon EC2 Systems Manager 파라미터 스토어를 사용합니다. 이러한 비밀은 작업에서 다음 중 하나와 같이 참조할 수 있습니다.

1. 그들은 사용하는 환경 변수로 참조됩니다`secrets`컨테이너 정의 매개 변수입니다.
2. 그들은 다음과 같이 참조됩니다.`secretOptions`로깅 플랫폼에 인증이 필요한 경우 자세한 내용은 단원을 참조하십시오.로깅 구성 옵션.
3. 그들은 사용하는 이미지에 의해 가져온 비밀로 참조됩니다`repositoryCredentials`컨테이너 정의 매개 변수는 컨테이너를 가져올 레지스트리에 인증이 필요한 경우입니다. Docker Hub에서 이미지를 가져올 때이 방법을 사용하십시오. 자세한 내용은 단원을 참조하십시오.작업에 대한 개인 레지스트리 인증.

Recommendations

비밀 관리를 설정할 때 다음을 수행하는 것이 좋습니다.

사용AWS Secrets Manager보안 자료를 저장하기 위한 Amazon EC2 Systems Manager 파라미터 스토어

API 키, 데이터베이스 자격 증명 및 기타 비밀 자료를AWS Secrets Manager를 암호화된 파라미터 Amazon EC2 Systems Manager 있습니다. 이러한 서비스는 둘 다 관리되는 키 - 값 저장소이기 때문에 AWS KMS를 사용하여 기밀 데이터를 암호화합니다.AWS Secrets Manager에는 비밀을 자동으로 회전하고, 임의의 비밀을 생성하고, 비밀을 공유하는 기능도 포함되어 있습니다.AWS계정을 참조하십시오. 이러한 중요한 기능을 생각하면AWS Secrets Manager그렇지 않으면 암호화된 매개 변수를 사용합니다.

> **Note**
>
> 의 암호를 참조하는 작업AWS Secrets Manager또는 Amazon EC2 Systems Manager 파라미터 스토어를 사용하려면작업 실행 역할를 Amazon ECS에게 원하는 비밀에 대한 액세스 권한을 부여하는 정책과 (해당되는 경우)AWS KMS키를 사용하여 해당 암호를 암호화하고 해독합니다.

> **Important**
>
> 작업에서 참조되는 암호는 자동으로 회전되지 않습니다. 암호가 변경되면 새 배포를 강제 실행하거나 새 작업을 시작하여 최신 암호 값을 검색해야 합니다. 자세한 내용은 다음 항목을 참조하십시오.
>
> - AWS Secrets Manager: 데이터를 환경 변수로 주입
> - Amazon EC2 Systems Manager 파라미터 스토어: 환경 변수로 데이터 주입

암호화된 Amazon S3 버킷에서 데이터 검색

환경 변수의 값이 실수로 로그에서 누수 될 수 있으며`docker inspect`를 사용하는 경우 암호화된 Amazon S3 버킷에 암호를 저장하고 작업 역할을 사용하여 해당 비밀에 대한 액세스를 제한해야 합니다. 이렇게 하면 Amazon S3 버킷의 암호를 읽기 위해 애플리케이션을 작성해야 합니다. 지침은 단원을 참조하십시오.Amazon S3 버킷에 대한 기본 서버 측 암호화 동작 설정.

사이드카 컨테이너를 사용하여 비밀을 볼륨에 마운트

환경 변수로 인해 데이터 유출의 위험이 높아지기 때문에AWS Secrets Manager를 사용하여 공유 볼륨에 쓸 수 있습니다. 이 컨테이너를 사용하여 응용 프로그램 컨테이너 전에 실행하고 종료 할 수 있습니다Amazon ECS 컨테이너 주문. 이렇게 하면 응용 프로그램 컨테이너는 이후에 비밀이 기록된 볼륨을 마운트합니다. Amazon S3 버킷 메서드와 마찬가지로 공유 볼륨에서 암호를 읽도록 애플리케이션을 작성해야 합니다. 볼륨은 작업 범위가 지정되기 때문에 작업이 중지된 후 볼륨이 자동으로 삭제됩니다. 사이드카 컨테이너의 예는aws-secret-sidecar-Injector프로젝트를 참조하십시오.

> **Note**
>
> Amazon EC2 비밀이 기록되는 볼륨은AWS KMS고객 관리형 키 :AWS Fargate를 선택하면 볼륨 저장소가 서비스 관리 키를 사용하여 자동으로 암호화됩니다.

추가 리소스

- Amazon ECS 작업에서 컨테이너에 비밀 전달
- 챔버는 Amazon EC2 Systems Manager 파라미터 스토어에 비밀을 저장하기 위한 래퍼입니다.

Compliance

Amazon ECS를 사용할 때 귀하의 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표, 관련 법률과 규정에 따라 결정됩니다.

AWS규정 준수에 도움이 되도록 다음 리소스를 제공합니다.

- 보안 및 컴플라이언스 퀵 스타트 가이드: 이 배포 안내서에서는 아키텍처 고려 사항에 대해 설명하고 보안 및 규정 준수에 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.AWS.
- HIPAA Security and Compliance: 이 백서에서는 회사가AWS를 사용하여 HIPAA 규정 준수 애플리케이션을 만들 수 있습니다.
- AWS규정 준수 프로그램별 서비스: 이 목록에는AWS특정 규정 준수 프로그램의 범위에서 서비스를 제공합니다. 자세한 내용은 단원을 참조하십시오.AWS규정 준수 프로그램.

PCI DSS (지불 카드 산업 데이터 보안 표준)

PCI DSS를 준수할 때는 환경 내에서 CHD (카드 소지자 데이터) 의 전체 흐름을 이해하는 것이 중요합니다. CHD 흐름은 PCI DSS의 적용 가능성을 결정하고, 카드 소지자 데이터 환경 (CDE) 의 경계와 구성 요소를 정의하며, 따라서 PCI DSS 평가의 범위를 정의합니다. PCI DSS 범위의 정확한 결정은 보안 상태를 정의하고 궁극적으로 성공적인 평가를 위한 핵심입니다. 고객은 완전성을 보장하고 범위에서의 변경 또는 편차를 감지하는 범위 결정 절차를 거쳐야 합니다.

컨테이너화된 애플리케이션의 일시적인 특성은 구성을 감사할 때 추가적인 복잡성을 제공합니다. 따라서 고객은 컨테이너 수명 주기의 모든 단계에서 규정 준수 요구 사항을 충족할 수 있도록 모든 컨테이너 구성 매개 변수에 대한 인식을 유지해야 합니다.

Amazon ECS에서 PCI DSS 규정 준수를 달성하는 방법에 대한 자세한 내용은 다음 백서를 참조하십시오.

- Amazon ECS를 기반으로 PCI DSS 규정 준수
- PCI DSS 범위 지정 및 세분화를 위한 아키텍처 AWS

HIPAA (미국 Health 보험 양도 및 책임에 관한 법)

보호된 상태 정보 (PHI) 를 처리하는 워크로드에 Amazon ECS를 사용하려면 추가 구성이 필요하지 않습니다. Amazon ECS는 Amazon ECAmazon EC2 컨테이너 시작을 조정하는 오케스트레이션 서비스 역할을 합니다. 오케스트레이션되는 워크로드 내의 데이터와 함께 또는 데이터에서는 작동하지 않습

니다. HIPAA 규정 및AWS비즈니스 어소시에이트 부록, PHI는 Amazon ECS에서 시작한 컨테이너에서 액세스할 때 전송 중에 암호화되어야 합니다.

미사용 시 암호화를위한 다양한 메커니즘은 각AWS스토리지 옵션 (예: Amazon S3, Amazon EBS 및 AWS KMS. 오버레이 네트워크 (예: VNS3 또는 Wave Net) 를 배포하여 컨테이너간에 전송된 PHI의 완전한 암호화를 보장하거나 중복 암호화 계층을 제공할 수 있습니다. 전체 로깅도 활성화되어야 하며 모든 컨테이너 로그는 Amazon CloudWatch 로 전달되어야 합니다. 자세한 내용은 단원을 참조하십시오.HIPAA Security and Compliance.

Recommendations

규정 준수 프로그램 소유자를 비즈니스 조기에 참여시키고AWS공동 책임 모델를 사용하여 관련 규정 준수 프로그램의 성공을 위한 규정 준수 관리 소유권을 식별합니다.

로깅 및 모니터링

로깅 및 모니터링은 Amazon ECS의 안정성, 가용성 및 성능을 유지하는 데 중요한 부분입니다.AWS 솔루션을 참조하십시오.AWS는 다음과 같이 Amazon ECS 리소스를 모니터링하고 잠재적 인시던트에 대응할 수 있는 몇 가지 도구를 제공합니다.

- Amazon CloudWatch 경보
- Amazon CloudWatch Logs
- Amazon CloudWatch Events
- AWS CloudTrail 로그

태스크의 컨테이너를 구성하여 로그 정보를 Amazon CloudWatch Logs 로 전송할 수 있습니다. 를 사용하고 있는 경우AWS Fargate시작 유형을 사용하여 컨테이너의 로그를 볼 수 있습니다. Amazon EC2 시작 유형을 사용하는 경우 하나의 편리한 위치에서 컨테이너의 다양한 로그를 볼 수 있습니다. 또한 컨테이너 로그가 컨테이너 인스턴스에서 디스크 공간을 차지하는 것이 방지됩니다.

Amazon CloudWatch Logs s에 대한 자세한 내용은 단원을 참조하십시오.Amazon EC2 인스턴스의 로그 모니터링의Amazon CloudWatch 사용 설명서. 태스크의 컨테이너 로그를 Amazon CloudWatch Logs 로 전송하는 방법은 단원을 참조하십시오.사용awslogs로그 드라이버.

유창한 비트를 사용한 컨테이너 로깅

AWS는 Amazon CloudWatch 로그와 Amazon Kinesis Data Firehose 용 플러그인과 함께 Fluent Bit 이미지를 제공합니다. 이 이미지는 로그를 Amazon CloudWatch 및 Amazon Kinesis 데이터 파이어호

스 대상 (Amazon S3, Amazon Elasticsearch Service 및 Amazon Redshift 포함) 으로 라우팅하는 기능을 제공합니다. Fluent Bit가 Fluentd보다 리소스 사용률이 낮으므로 Fluent Bit를 로그 라우터로 사용하는 것이 좋습니다. 자세한 내용은 단원을 참조하십시오.Amazon CloudWatch Logs및Amazon Kinesis Data Firehose.

이AWS이미지는 다음 에서 얻을 수 있습니다.

- 아마존 ECR 공개 갤러리에 아마존 ECR
- Amazon ECR 저장소(대부분의 고가용성 지역에서)
- Docker Hub

다음은 Docker CLI에 사용할 구문을 나타낸 것입니다.

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

예를 들어, 최신AWS이 Docker CLI 명령을 사용하는 유창한 비트 이미지:

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

또한 Fluent Bit 및 관련 기능에 대한 자세한 내용은 다음 블로그 게시물을 참조하십시오.

- 아마존 EKS에 대한 유창한 비트 AWS Fargate
- 유창한 비트로 중앙 집중식 컨테이너 로깅
- 다음과 같이 확장 가능한 로그 솔루션 애그리게이터 구축AWS Fargate, Fluentd, Amazon Kinesis Data Firehose

사용자 지정 로그 라우팅 - 아마존 ECS용 FireLens

Amazon ECS용 FireLens 를 사용하면 작업 정의 파라미터를 사용하여 로그를AWS서비스 또는AWS 파트너 네트워크 (APN) 대상입니다. FireLens는 Fluentd 및 Fluent Bit와 함께 작동합니다. 우리는 제공AWS유창한 비트 이미지. 또는 자체 Fluentd 또는 Fluent Bit 이미지를 사용할 수 있습니다.

Amazon ECS용 FireLens 를 사용할 때는 다음 조건 및 고려 사항을 고려해야 합니다.

- Amazon ECS용 파이어렌즈 (FireLens) 는AWS Fargate및 Amazon EC2.
- Amazon ECS용 FireLens는AWS CloudFormation템플릿을 참조하십시오. 자세한 내용은 단원을 참조하십시오.AWS::ECS::TaskDefinition FirelensConfiguration의AWS CloudFormation사용 설명서.

- 사용 하는 작업에 대 한`bridge`네트워크 모드를 사용하는 경우 FireLens 구성이 포함된 컨테이너는 해당 컨테이너에 의존하는 모든 애플리케이션 컨테이너가 시작되기 전에 시작해야 합니다. 컨테이너가 시작되는 순서를 제어하려면 작업 정의에서 종속성 조건을 사용하십시오. 자세한 내용은 단원을 참조하십시오.컨테이너 종속성.

AWS Fargate 보안

를 사용할 때 다음 모범 사례를 고려하는 것이 좋습니다.AWS Fargate.

사용AWS KMS사용 후 삭제 스토리지를 암호화합니다.

임시 저장소는 다음과 같이 암호화해야합니다.AWS KMS. 에서 호스팅되는 Amazon ECS 작업의 경우 AWS Fargate플랫폼 버전 사용`1.4.0`이상 버전에서 각 작업은 20GB의 휘발성 스토리지를 수신합니다. 스토리지 용량은 조정할 수 없습니다. 2020202020205월 28일 이후 시작된 태스크의 경우, 관리형 암호화 키와 AES-256 암호화 알고리즘으로 휘발성 스토리지가 암호화됩니다.AWS Fargate.

예: Amazon ECS 작업 시작AWS Fargate플랫폼 버전 1.4.0과 휘발성 스토리지 암호화

다음 명령은 Amazon ECS 작업을 시작합니다.AWS Fargate플랫폼 버전 1.4로 업데이트 이 작업은 Amazon ECS 클러스터의 일부로 시작되므로 자동으로 암호화된 20GB의 휘발성 스토리지를 사용합니다.

```
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
```

커널 시스템 호출 추적을 위한 SYS_PSTRACE 기능

컨테이너에서 추가하거나 제거 된 Linux 기능의 기본 구성은 Docker에서 제공합니다. 사용 가능한 기능에 대한 자세한 내용은런타임 권한 및 Linux 기능의도커를 실행하려면설명서를 참조하십시오.

에서 실행되는 작업AWS Fargate는 추가 기능만 지원합니다.`SYS_PTRACE`커널 기능을 실행합니다.

Sysdig를 통해 이 기능을 사용하는 방법을 보여주는 아래의 자습서 비디오를 참조하십시오.팔코프로젝트를 참조하십시오.

#ContainersFromTheCouch - 문제 해결AWS FargateSYS_PSTRACE 기능을 사용하는 작업

이전 비디오에서 논의 된 코드는 GitHub 에서 찾을 수 있습니다.여기를.

작업 및 컨테이너 보안

컨테이너 이미지를 공격에 대한 1차 방어선으로 고려해야 합니다. 안전하지 않고 제대로 구성되지 않은 이미지는 공격자가 컨테이너의 경계를 벗어나 호스트에 대한 액세스 권한을 얻을 수 있습니다. 이러한 현상의 위험을 완화하기 위해 다음을 수행해야 합니다.

Recommendations

태스크와 컨테이너를 설정할 때 다음을 수행하는 것이 좋습니다.

최소한의 이미지 만들기 또는 디트로리스 이미지 사용

먼저 컨테이너 이미지에서 관계없는 모든 바이너리를 제거합니다. Docker Hub에서 익숙하지 않은 이미지를 사용하는 경우 이미지를 검사하여 각 컨테이너 레이어의 내용을 참조하십시오. 다음과 같은 응용 프로그램을 사용할 수 있습니다.다이브를 실행하여 주의해야 합니다.

또는 를 사용할 수 있습니다.비참한응용 프로그램 및 런타임 종속성 만 포함하는 이미지. 패키지 관리자 또는 셸이 포함되어 있지 않습니다. 디트로리스 이미지는 “스캐너의 잡음에 대한 신호”를 향상시키고 필요한 것만 얻을 수 있는 출처를 확립하는 부담을 줄여줍니다. 자세한 내용은 GitHub 설명서를 참조하십시오.비참한.

Docker에는 예약 된 최소 이미지에서 이미지를 만드는 메커니즘이 있습니다.스크래치. 자세한 내용은 단원을 참조하십시오.사용하여 간단한 부모 이미지 만들기스크래치Docker 설명서에 나와 있습니다. Go와 같은 langage를 사용하면 정적 링크 된 바이너리를 만들어 Dockerfile에서 참조 할 수 있습니다. 다음 예에서는 이를 수행하는 방법을 보여 줍니다.

```
############################
# 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.
```

앞의 예는 다단계 빌드의 예입니다. 이러한 유형의 빌드는 컨테이너 레지스트리에 푸시된 최종 이미지의 크기를 최소화하는 데 사용할 수 있기 때문에 보안 측면에서 매력적입니다. 빌드 도구 및 기타 관계없는 바이너리가 없는 컨테이너 이미지는 이미지의 공격 영역을 줄임으로써 보안 상태를 개선합니다. 다단계 빌드에 대한 자세한 내용은다단계 빌드 생성.

이미지에 취약성 검사

가상 시스템과 마찬가지로 컨테이너 이미지에는 취약점이 있는 바이너리 및 애플리케이션 라이브러리가 포함되거나 시간이 지남에 따라 취약점이 발생할 수 있습니다. 악용으로부터 보호하는 가장 좋은 방법은 이미지 스캐너를 사용하여 이미지를 정기적으로 스캔하는 것입니다. Amazon ECR에 저장된 이미지는 푸시 또는 온디맨드 (24시간마다 한 번) 스캔할 수 있습니다. Amazon ECR은 현재클레어는 오픈 소스 이미지 스캔 솔루션입니다. 이미지를 스캔한 후 결과는 Amazon EAmazon EventBridge 의 Amazon ECR 이벤트 스트림에 기록됩니다. 또한 Amazon ECR 콘솔 내에서 또는설명이미지검색 결과API. 및 이미지HIGH또는CRITICAL취약점이 삭제되거나 다시 구축되어야 합니다. 배포된 이미지에 취약점이 발생하는 경우 가능한 한 빨리 교체해야 합니다.

Docker Desktop Edge 버전 2.3.6.0이상스캔로컬 이미지. 스캔에 의해 전원이 공급됩니다.스나이크, 응용 프로그램 보안 서비스. 취약점이 발견되면 Snyk는 Dockerfile의 취약점과의 계층 및 종속성을 식별합니다. 또한 취약점이 적은 얇은 기본 이미지를 사용하거나 특정 패키지를 최신 버전으로 업그레이드하는 것과 같은 안전한 대안을 권장합니다. Docker 스캔을 사용하여 개발자는 이미지를 레지스트리로 푸시하기 전에 잠재적 인 보안 문제를 해결할 수 있습니다.

- Amazon ECR 및AWS Security Hub에서 Amazon ECR의 취약성 정보를 표시하는 방법을 설명합니다.AWS Security Hub취약한 이미지에 대한 액세스를 차단하여 문제 해결을 자동화할 수 있습니다.

이미지에서 특수 권한 제거

액세스 권한 플래그`setuid`및`setgid`는 실행 파일의 소유자 또는 그룹의 권한으로 실행 파일을 실행할 수 있도록 허용합니다. 이러한 바이너리를 사용하여 권한을 에스컬레이션할 수 있으므로 이미지에서 이러한 액세스 권한이 있는 모든 바이너리를 제거합니다. 다음과 같은 모든 쉘과 유틸리티를 제거하는 것이 좋습니다.`nc`및`curl`악의적 인 목적으로 사용할 수 있습니다. 를 사용하여 파일을 찾을 수 있습니다.`setuid`및`setgid`다음 명령을 사용하여 액세스 권한을 확인합니다.

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

이러한 파일에서 이러한 특수 권한을 제거하려면 컨테이너 이미지에 다음 지시문을 추가합니다.

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

선별된 이미지 세트 만들기

개발자가 자신의 이미지를 만들 수 있도록 허용하는 대신 조직의 여러 응용 프로그램 스택에 대해 심사된 이미지 집합을 만듭니다. 그렇게함으로써 개발자는 Dockerfiles를 작성하는 방법을 배우지 않고 코드 작성에 집중할 수 있습니다. 변경 사항이 코드베이스에 병합되면 CI/CD 파이프라인이 자산을 자동으로 컴파일한 다음 아티팩트 리포지토리에 저장할 수 있습니다. 마지막으로 아티팩트를 Amazon ECR과 같은 Docker 레지스트리로 푸시하기 전에 해당 이미지에 복사합니다. 최소한 모자 개발자가 자신의 Dockerfiles를 만들 수있는 기본 이미지 세트를 만들어야합니다. Docker 허브에서 이미지를 가져오는 것을 피해야합니다. 이미지에 무엇이 있는지 항상 알지 못하고 상위 1000 개 이미지 중 약 5 분의 1에는 취약성이 있습니다. 이러한 이미지 및 해당 취약점의 목록은https://vulnerablecontainers.org/.

응용 프로그램 패키지 및 라이브러리의 취약성 검사

이제 오픈 소스 라이브러리를 사용하는 것이 일반적입니다. 운영 체제 및 OS 패키지와 마찬가지로 이러한 라이브러리에는 취약점이 있을 수 있습니다. 개발 수명 주기의 일부로 중요한 취약점이 발견되면 이러한 라이브러리를 검사하고 업데이트해야 합니다.

Docker 데스크톱은 Snyk를 사용하여 로컬 스캔을 수행합니다. 또한 오픈 소스 라이브러리에서 취약성 및 잠재적 라이센스 문제를 찾는 데 사용할 수 있습니다. 개발자 워크플로에 직접 통합되어 오픈 소스 라이브러리에서 발생하는 위험을 완화할 수 있습니다. 자세한 내용은 다음 항목을 참조하십시오.

- 오픈 소스 애플리케이션 보안 도구에는 응용 프로그램의 취약점을 탐지하기 위한 도구 목록이 포함되어 있습니다.

- Docker 스캔 치트 시트

정적 코드 분석 수행

컨테이너 이미지를 작성하기 전에 정적 코드 분석을 수행해야 합니다. 소스 코드에 대해 수행되며 오류 주입과 같은 악의적인 행위자가 악용할 수 있는 코딩 오류 및 코드를 식별하는 데 사용됩니다. SonarQube는 다양한 프로그래밍 언어를 지원하는 정적 응용 프로그램 보안 테스트 (SAAST) 에 널리 사용되는 옵션입니다.

비 루트 사용자로 컨테이너 실행

컨테이너를 루트 이외 사용자로 실행해야 합니다. 기본적으로 컨테이너는`root`를 사용하지 않는 한`USER`지시문이 Dockerfile에 포함되어 있습니다. Docker에 의해 할당 된 기본 Linux 기능은 다음과 같이 실행할 수있는 작업을 제한합니다.`root`, 하지만 소외로 만. 예를 들어, 다음과 같이 실행되는 컨테이너`root`는 여전히 장치에 액세스할 수 없습니다.

CI/CD 파이프 라인의 일부로 Dockerfiles를 보풀어서`USER`지시문을 사용하고 누락 된 경우 빌드를 실패합니다. 자세한 내용은 다음 항목을 참조하십시오.

- Dockerfile 린트는 파일이 모범 사례를 준수하는지 확인하는 데 사용할 수있는 RedHat 의 오픈 소스 도구입니다.
- 하돌린트는 모범 사례를 준수하는 Docker 이미지를 작성하기위한 또 다른 도구입니다.

읽기 전용 루트 파일 시스템 사용

읽기 전용 루트 파일 시스템을 사용해야 합니다. 컨테이너의 루트 파일 시스템은 기본적으로 쓰기 가능합니다. 당신은과 컨테이너를 구성 할 때RO(읽기 전용) 루트 파일 시스템을 사용하면 데이터를 유지할 수있는 위치를 명시 적으로 정의해야합니다. 이렇게 하면 권한이 특별히 부여되지 않는 한 컨테이너의 파일 시스템을 쓸 수 없기 때문에 공격 대상이 줄어듭니다.

> **Note**
>
> 읽기 전용 루트 파일 시스템이 있으면 파일 시스템에 쓸 수 있을 것으로 예상되는 특정 OS 패키지에 문제가 발생할 수 있습니다. 읽기 전용 루트 파일 시스템을 사용하려는 경우 사전에 철저히 테스트하십시오.

CPU 및 메모리 제한을 사용하여 작업 구성 (Amazon EC2)

다음과 같은 위험을 최소화하려면 CPU 및 메모리 제한을 사용하여 작업을 구성해야 합니다. 작업의 리소스 제한은 작업 내의 모든 컨테이너에서 예약 할 수있는 CPU 및 메모리의 양에 대한 상한을 설정합니다. 제한이 설정되지 않은 경우 작업은 호스트의 CPU 및 메모리에 액세스할 수 있습니다. 이로 인해 공유 호스트에 배포된 작업이 시스템 리소스의 다른 작업을 중단시킬 수 있는 문제가 발생할 수 있습니다.

> **Note**
>
> Amazon ECSAWS Fargate작업에서는 이러한 값을 청구 목적으로 사용하기 때문에 CPU 및 메모리 제한을 지정해야 합니다. Amazon ECS Fargate 에서는 각 작업이 자체 전용 인스턴스에서 실행되기 때문에 모든 시스템 리소스를 호깅하는 작업 중 하나가 문제가 되지 않습니다. 메모리 제한을 지정하지 않으면 Amazon ECS가 각 컨테이너에 최소 4MB를 할당합니다. 마찬가지로 작업에 대해 설정된 CPU 제한이 없는 경우 Amazon ECS 컨테이너 에이전트는 최소 2 개의 CPU를 할당합니다.

Amazon ECR에서 불변 태그 사용

Amazon ECR을 사용하면 변경 불가능한 태그가 있는 구성 이미지를 사용할 수 있고 사용해야 합니다. 이렇게하면 이미지의 변경되거나 업데이트 된 버전을 동일한 태그로 이미지 저장소로 푸시하는 것을 방지 할 수 있습니다. 이렇게하면 공격자가 동일한 태그로 이미지 위에 손상된 버전의 이미지를 푸시하는 것을 방지 할 수 있습니다. 변경 불가능한 태그를 사용하면 각 변경 사항에 대해 다른 태그가있는 새 이미지를 효과적으로 푸시해야합니다.

컨테이너를 권한 있는 상태로 실행하지 마십시오 (Amazon EC2).

특권으로 컨테이너를 실행하지 않아야합니다. 백그라운드의 경우 컨테이너는`privileged`는 호스트에서 확장된 권한으로 실행됩니다. 이것은 컨테이너가 할당 된 모든 Linux 기능을 상속한다는 것을 의미합니다.`root`를 호스트에 설치합니다. 사용은 심각하게 제한되거나 금지되어야합니다. Amazon ECS 컨테이너 에이전트 환경 변수를 설정하는 것이 좋습니다.`ECS_DISABLE_PRIVILEGED`to`true`를 사용하여 컨테이너가`privileged`경우 특정 호스트에서`privileged`는 필요하지 않습니다. 또는 다음을 사용할 수 있습니다.AWS Lambda를 사용하여 작업 정의를 검사하여`privileged`파라미터.

> **Note**
>
> 컨테이너를 다음과 같이 실행`privileged`는 Amazon ECS에서 지원되지 않습니다.AWS Fargate.

컨테이너에서 불필요한 Linux 기능 제거

다음은 Docker 컨테이너에 할당 된 기본 Linux 기능 목록입니다. 각 기능에 대한 자세한 내용은Linux 기능 개요.

```
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
```

컨테이너에 위에 나열된 Docker 커널 기능이 모두 필요하지 않은 경우 컨테이너에서 삭제하는 것이 좋습니다. 각 Docker 커널 기능에 대한 자세한 내용은커널 기능. 다음을 수행하여 사용 중인 기능을 확인할 수 있습니다.

- OS 패키지를 설치합니다.libcap-ng를 실행하고`pscap`유틸리티를 사용하여 각 프로세스에서 사용 중인 기능을 나열할 수 있습니다.
- 다음을 사용할 수도 있습니다.캡쉬를 사용하여 프로세스가 사용하고있는 기능을 해독합니다.
- 자세한 내용은Linux 기능자세한 내용은 단원을 참조하십시오.

CMK (고객 관리 키) 를 사용하여 Amazon ECR로 푸시된 이미지를 암호화합니다.

CMK (고객 관리 키) 를 사용하여 Amazon ECR로 푸시되는 이미지를 압축해야 합니다. Amazon ECR로 푸시되는 이미지는 저장 시AWS Key Management Service(AWS KMS) 관리 키입니다. 사용자 고유의 키를 사용하려는 경우 Amazon ECR은 이제AWS KMS고객 관리 키 (CMK) 를 사용하여 암호화를 수행할 수 있습니다. CMK를 사용하여 서버 측 암호화를 활성화하기 전에저장된 암호화.

런타임 보안

런타임 보안은 컨테이너가 실행되는 동안 컨테이너에 대한 적극적인 보호를 제공합니다. 아이디어는 악의적 인 활동이 컨테이너에서 발생하는 것을 감지하고 방지하는 것입니다.

보안 컴퓨팅을 통해 (seccomp) 를 사용하면 컨테이너화 된 응용 프로그램이 특정 syscall을 기본 호스트 운영 체제의 커널로 만들지 못하게 할 수 있습니다. Linux 운영 체제에는 수백 개의 시스템 호출이 있지만 대부분의 시스템 호출은 컨테이너를 실행하는 데 필요하지 않습니다. 컨테이너에서 만들 수 있는 syscall을 제한하면 응용 프로그램의 공격 영역을 효과적으로 줄일 수 있습니다.

seccomp를 시작하려면 다음을 사용할 수 있습니다.`strace`를 사용하여 스택 추적을 생성하여 응용 프로그램이 수행하는 시스템 호출을 확인할 수 있습니다. 다음과 같은 도구를 사용할 수 있습니다.`syscall2seccomp`을 클릭하여 스택 추적에서 수집된 데이터에서 seccomp 프로필을 만듭니다. 자세한 내용은 단원을 참조하십시오.스트레이스및syscall2seccomp.

SELinux 보안 모듈과 달리 seccomp은 컨테이너를 서로 분리 할 수 없습니다. 그러나 호스트 커널을 승인되지 않은 syscalls로부터 보호합니다. 그것은 syscalls를 가로 채고 허용 나열된 항목 만 통과 할 수 있도록하여 작동합니다. Docker에는기본값대부분의 범용 워크로드에 적합한 seccomp 프로필입니다.

> **Note**
>
> 추가 권한이 필요한 경우 프로필을 생성할 수도 있습니다.

AppArmor 는 seccomp과 유사한 리눅스 보안 모듈이지만 파일 시스템의 일부에 액세스하는 것을 포함하여 컨테이너의 기능을 제한합니다. 둘 중 하나에서 실행할 수 있습니다.`enforcement`또는`complain`모드를 선택합니다. AppArmor 프로파일을 작성하는 것은 어려울 수 있으므로베인. AppArmor 에 대한 자세한 내용은 공식AppArmor페이지를 참조하십시오.

> **Important**
>
> AppArmor 는 리눅스의 우분투와 데비안 배포판에서만 사용할 수 있습니다.

Recommendations

런타임 보안을 설정할 때는 다음 작업을 수행하는 것이 좋습니다.

런타임 방어를 위해 타사 솔루션 사용

런타임 방어를 위해 타사 솔루션을 사용합니다. Linux 보안의 작동 방식에 익숙하다면 seccomp 및 AppArmor 프로파일을 만들고 관리하십시오. 둘 다 오픈 소스 프로젝트입니다. 그렇지 않으면 대신 다른 타사 서비스를 사용하는 것이 좋습니다. 대부분 기계 학습을 사용하여 의심스러운 활동을 차단하거나 경고합니다. 사용 가능한 타사 솔루션 목록은AWS Marketplace용 컨테이너.

seccomp 정책을 사용하여 Linux 기능 추가 또는 제거

seccomp를 사용하여 Linux 기능을 더 잘 제어하고 syscall 검사 오류를 피할 수 있습니다. Seccomp은 특정 시스템 스캘을 실행하거나 특정 속성을 사용할 수 있는 권한을 취소하는 syscall 필터로 작동합니다.

AWS파트너

다음 중 하나를 사용할 수 있습니다.AWS파트너 제품으로 Amazon ECS 워크로드에 보안 및 기능을 추가할 수 있습니다. 자세한 내용은 단원을 참조하십시오.Amazon ECS 파트너.

Aqua Security

다음을 수행할 수 있습니다.Aqua Security를 사용하여 개발부터 프로덕션까지 클라우드 네이티브 애플리케이션을 보호할 수 있습니다. Aqua Cloud 네이티브 보안 플랫폼은 클라우드 네이티브 리소스 및 오케스트레이션 툴과 통합되어 투명하고 자동화된 보안을 제공합니다. 의심스러운 활동과 공격을 실시간으로 방지하고 정책을 시행하고 규정 준수를 간소화하는 데 도움이 됩니다.

Palo Alto Networks

Palo Alto Networks는 클라우드에서 그리고 개발 및 소프트웨어 라이프사이클 전반에 걸쳐 호스트, 컨테이너 및 서버리스 인프라에 대한 보안과 보호를 제공합니다.

트위스트 록은 팔로 알토 네트워크에 의해 공급되며 아마존 ECS FireLens 즈와 통합 될 수 있습니다. 이를 통해 높은 정확도의 보안 로그 및 인시던트에 액세스할 수 있으며, 이러한 로그는 여러AWS서비스. 여기에는 Amazon CloudWatch, Amazon Athena, Amazon Kinesis 등이 포함됩니다. Twistlock은 AWS컨테이너 서비스를 참조하십시오.

분류: Sysdig

다음을 수행할 수 있습니다.분류: Sysdig를 사용하여 프로덕션 시나리오에서 안전하고 규정을 준수하는 클라우드 네이티브 워크로드를 실행할 수 있습니다. Sysdig Secure DevOps 플랫폼에는 클라우드 네이티브 워크로드를 보호하는 보안 및 규정 준수 기능이 내장되어 있으며 엔터프라이즈급 확장성, 성능 및 사용자 지정도 제공합니다.

Amazon ECS 모범 사례 안내서의 문서 이력

다음 표에서는 Amazon ECS 모범 사례 안내서의 설명서 릴리스를 소개합니다.

업데이트 기록-변경	update-history-description	update-history-date
보안 모범 사례	Amazon ECS 워크로드의 보안 관리에 대한 모범 사례를 추가했습니다.	2021년 5월 26일
자동 확장 및 용량 관리 모범 사례	Amazon ECS 워크로드의 Auto Scaling 및 용량 관리에 대한 모범 사례를 추가했습니다.	2021년 5월 14일
영구 스토리지 모범 사례	Amazon ECS 워크로드의 영구 스토리지에 대한 모범 사례를 추가했습니다.	2021년 5월 7일
네트워킹 모범 사례	Amazon ECS 워크로드의 네트워킹 관리에 대한 모범 사례를 추가했습니다.	2021년 4월 6일
최초 릴리스	Amazon ECS 모범 사례 안내서의 최초 릴리스	2021년 4월 6일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.