Elastic Load Balancing

Application Load Balancer

Elastic Load Balancing: Application Load Balancer

Table of Contents

Application Load Balancer란 무엇입니까?

Elastic Load Balancing은 Application Load Balancer, Network Load Balancer, Classic Load Balancer의 세 가지 유형의 로드 밸런서를 지원합니다. 이 안내서에서는 Application Load Balancer에 대해 설명합니다. Network Load Balancer에 대한 자세한 내용은 Network Load Balancer 사용 설명서 단원을 참조하십시오. Classic Load Balancer에 대한 자세한 내용은 Classic Load Balancer 사용 설명서 단원을 참조하십시오.

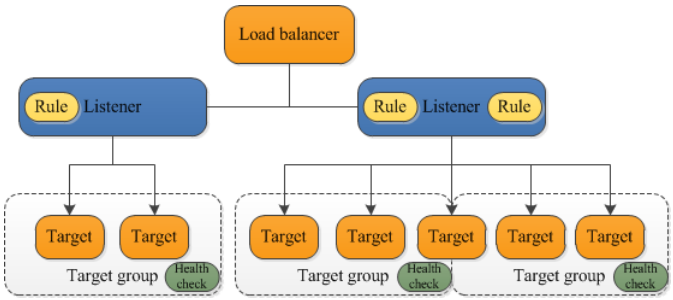
Application Load Balancer 구성 요소

로드 밸런서는 클라이언트에 대한 단일 접점 역할을 수행합니다. 로드 밸런서는 여러 가용 영역에서 EC2 인스턴스 같은 여러 대상에 수신 애플리케이션 트래픽을 분산합니다. 이렇게 하면 애플리케이션의 가용성이 향상됩니다. 로드 밸런서에 하나 이상의 리스너를 추가할 수 있습니다.

리스너는 사용자가 구성한 프로토콜 및 포트를 사용하여 클라이언트의 연결 요청을 확인하고, 사용자가 정의한 규칙에 따라 하나 이상의 대상 그룹에 요청을 전달합니다. 각 규칙은 대상 그룹, 조건 및 우선 순위를 지정합니다. 조건이 충족되면 대상 그룹으로 트래픽이 전달됩니다. 각 리스너에 대한 기본 규칙을 정의해야 하며 요청 콘텐츠를 기반으로 다른 대상 그룹을 지정하는 규칙을 추가할 수 있습니다(콘텐츠 기반 라우팅이라고도 함).

각 대상 그룹은 지정한 프로토콜과 포트 번호를 사용하여 EC2 인스턴스 같은 하나 이상의 등록된 대상으로 요청을 라우팅합니다. 여러 대상 그룹에 대상을 등록할 수 있습니다. 대상 그룹 기준으로 상태 확인을 구성할 수 있습니다. 로드 밸런서의 리스너 규칙에서 지정한 대상 그룹에 등록된 모든 대상에서 상태 검사가 수행됩니다.

다음 다이어그램은 기본 구성 요소를 보여 줍니다. 각 리스너에는 기본 규칙이 포함되어 있고 하나의 리스너에는 요청을 다른 대상 그룹으로 라우팅하는 다른 규칙이 포함되어 있습니다. 하나의 대상은 두 개의 대상 그룹에 등록됩니다.



자세한 내용은 다음 설명서를 참조하십시오.

- 로드밸런서 (p. 14)
- 리스너 (p. 24)
- 대상 그룹 (p. 50)

Application Load Balancer 개요

Application Load Balancer는 개방형 시스템 간 상호 연결(OSI) 모델의 일곱 번째 계층인 애플리케이션 계층에서 작동합니다. 로드 밸런서는 요청을 받으면 우선 순위에 따라 리스너 규칙을 평가하여 적용할 규칙을 결

정한 다음, 규칙 작업의 대상 그룹에서 대상을 선택합니다. 애플리케이션 트래픽의 콘텐츠를 기반으로 다른 대상 그룹에 요청을 라우팅하도록 리스너 규칙을 구성할 수 있습니다. 대상이 여러 개의 대상 그룹에 등록이 된 경우에도 각 대상 그룹에 대해 독립적으로 라우팅이 수행됩니다.

애플리케이션에 대한 요청의 전체적인 흐름을 방해하지 않고 필요에 따라 로드 밸런서에서 대상을 추가 및 제거할 수 있습니다. 애플리케이션에 대한 트래픽이 시간에 따라 변화하므로 Elastic Load Balancing가 로드 밸런서를 확장합니다. Elastic Load Balancing는 대다수의 워크로드에 자동으로 확장될 수 있습니다.

로드 밸런서가 정상적인 대상에만 요청을 보낼 수 있도록 등록된 대상의 상태를 모니터링하는 데 사용되는 상태 확인을 구성할 수 있습니다.

자세한 내용은 Elastic Load Balancing 사용 설명서의 Elastic Load Balancing 작동 방식을 참조하십시오.

Classic Load Balancer에서 마이그레이션하는 것의 장점

Classic Load Balancer 대신 Application Load Balancer를 사용하면 다음과 같은 장점이 있습니다.

- 경로 기반 라우팅을 지원합니다. 요청의 URL을 기반으로 요청을 전달하는 리스너에 대한 규칙을 구성할 수 있습니다. 이를 통해 애플리케이션을 규모가 더욱 작은 서비스로 구성하고, URL 콘텐츠를 기반으로 요청을 올바른 서비스로 라우팅할 수 있습니다.
- 호스트 기반 라우팅을 지원합니다. HTTP 헤더의 호스트 필드를 기반으로 요청을 전달하는 리스너에 대한 규칙을 구성할 수 있습니다. 따라서 단일 로드 밸런서를 사용하여 여러 개의 도메인에 요청을 라우팅할 수 있습니다.
- 표준 또는 사용자 지정 HTTP 헤더 및 메서드, 쿼리 파라미터, 소스 IP 주소 같은 요청의 필드를 기반으로 하는 라우팅을 지원합니다.
- 단일 EC2 인스턴스의 여러 애플리케이션으로 요청을 라우팅하는 것을 지원합니다. 여러 포트를 사용하여 각 인스턴스 또는 IP 주소를 동일한 대상 그룹에 등록할 수 있습니다.
- 한 URL에서 다른 URL로 요청을 리디렉션하는 작업을 지원합니다.
- 사용자 지정 HTTP 응답 회신을 지원합니다.
- 로드 밸런서의 VPC 외부 대상을 포함하여 IP 주소로 대상을 등록하는 것을 지원합니다.
- Lambda 함수를 대상으로 등록하는 작업을 지원합니다.
- 요청을 라우팅하기 전에 기업 또는 소셜 자격 증명을 통해 애플리케이션의 사용자를 인증할 수 있도록 로드 밸런서를 지원합니다.
- 컨테이너화된 애플리케이션을 지원합니다. Amazon Elastic Container Service(Amazon ECS)은 작업을 예약할 때 사용하지 않는 포트를 선택하고, 이 포트를 사용하여 대상 그룹에 작업을 등록할 수 있습니다. 이를 통해 클러스터를 효율적으로 사용할 수 있습니다.
- 대상 그룹 수준에서 상태 확인이 정의되고 많은 CloudWatch 지표가 보고되므로 각 서비스의 상태를 독립적으로 모니터링할 수 있게 지원합니다. 그룹에 대상 그룹을 연결하면 필요에 따라 동적으로 각 서비스를 확장할 수 있습니다.
- 액세스 로그는 추가 정보를 포함하며 압축된 형식으로 저장됩니다.
- 로드 밸런서 성능을 개선합니다.

각 로드 밸런서 유형에서 지원하는 기능에 대한 자세한 내용은 Elastic Load Balancing 제품 비교를 참조하십시오.

시작하는 방법

Application Load Balancer를 생성하려면 다음 자습서 중 하나를 시도해 보십시오.

- Elastic Load Balancing 사용 설명서의 Elastic Load Balancing 시작하기.
- 자습서: Application Load Balancer를 통한 경로 기반 라우팅 사용 (p. 7)
- 자습서: Application Load Balancer를 통해 마이크로서비스를 대상으로 사용 (p. 9)

관련 서비스

Elastic Load Balancing은 다음 서비스를 통해 애플리케이션의 가용성 및 확장성을 개선합니다.

- Amazon EC2 — 클라우드에서 애플리케이션을 실행하는 가상 서버입니다. 로드 밸런서를 구성하여 EC2 인스턴스에 트래픽을 라우팅할 수 있습니다.
- Amazon EC2 Auto Scaling — 인스턴스에 장애가 발생하더라도 원하는 수의 인스턴스를 실행하고 인스턴스의 수요가 변경되면 자동으로 인스턴스 수를 늘리거나 줄일 수 있게 해 줍니다. Elastic Load Balancing, 과 함께 Auto Scaling을 활성화하는 경우, Auto Scaling이 시작한 인스턴스는 자동으로 로드 밸런서에 등록되고 Auto Scaling이 종료하는 인스턴스는 자동으로 로드 밸런서에서 등록 취소됩니다.
- AWS Certificate Manager — HTTPS 리스너를 생성할 때 ACM에서 제공한 인증서를 지정할 수 있습니다. 로드 밸런서는 인증서를 사용하여 연결을 종료하고 클라이언트의 요청을 암호화 해제합니다. 자세한 내용은 SSL 인증서 (p. 33) 단원을 참조하십시오.
- Amazon CloudWatch — 로드 밸런서를 모니터링하고 필요에 따라 조치를 취할 수 있게 해 줍니다. 자세한 내용은 Application Load Balancer를 위한 CloudWatch 지표 (p. 70) 단원을 참조하십시오.
- Amazon ECS — EC2 인스턴스 클러스터에서 Docker 컨테이너를 실행, 중단 및 관리 가능합니다. 로드 밸런서를 구성하여 컨테이너에 트래픽을 라우팅할 수 있습니다. 자세한 내용은 Amazon Elastic Container Service Developer Guide에서 서비스 로드 밸런싱을 참조하십시오.
- Route 53 — 도메인 이름(예: `www.example.com`)을 컴퓨터를 사용하여 서로 연결해주는 숫자로 된 IP 주소(예: `192.0.2.1`)로 변환하여 방문자를 안정적이며 비용 효율적으로 웹 사이트로 라우팅하도록 합니다. AWS는 로드 밸런서와 같은 사용자의 AWS 리소스에 URL을 배정합니다. 그러나 기억하기 쉬운 URL이 필요한 경우도 있습니다. 예를 들어 도메인 이름을 로드 밸런서로 매핑할 수 있습니다.
- AWS WAF — Application Load Balancer와 함께 AWS WAF를 사용하여 웹 ACL(웹 액세스 제어 목록)의 규칙에 따라 요청을 허용하거나 차단할 수 있습니다. 자세한 내용은 Application Load Balancer 및 AWS WAF (p. 17) 단원을 참조하십시오.

로드 밸런서와 통합된 서비스에 대한 정보를 보려면 AWS Management 콘솔에서 로드 밸런서를 선택하고 Integrated services(통합 서비스) 탭을 선택합니다.

요금

로드 밸런서에서는 사용한 만큼만 지불하면 됩니다. 자세한 내용은 Elastic Load Balancing 요금을 참조하십시오.

Application Load Balancer 시작하기

이 자습서에서는 웹 기반 인터페이스인 AWS Management 콘솔을 통해 Application Load Balancer에 대한 실습 소개를 제공합니다. 첫 번째 를 생성하려면 다음 단계를 완료하십시오.

작업

또는 Network Load Balancer을(를) 생성하려면 Network Load Balancer 사용 설명서에서 Network Load Balancer 시작하기를 참조하십시오. Classic Load Balancer를 생성하려면 Classic Load Balancer 사용 설명서에서 Classic Load Balancer 생성을 참조하십시오.

시작하기 전에

- EC2 인스턴스에 대해 사용할 두 개의 가용 영역을 결정합니다. 각 가용 영역에 있는 하나 이상의 퍼블릭 서브넷으로 VPC(Virtual Private Cloud)를 구성합니다. 이 퍼블릭 서브넷은 로드 밸런서를 구성하는데 사용됩니다. 대신 이러한 가용 영역의 다른 서브넷에서 EC2 인스턴스를 시작할 수 있습니다.
- 각 가용 영역에서 하나 이상의 EC2 인스턴스를 시작합니다. 각 EC2 인스턴스에 Apache 또는 IIS(인터넷 정보 서비스)와 같은 웹 서버를 설치해야 합니다. 이들 인스턴스에 대한 보안 그룹이 포트 80에서 HTTP 액세스를 허용하는지 확인합니다.

1단계: 로드 밸런서 유형 선택

Elastic Load Balancing는 세 가지 로드 밸런서 유형을 지원합니다. 이 자습서에서는 를 생성합니다.

Application Load Balancer 생성

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 모음에서 로드 밸런서의 리전을 선택합니다. EC2 인스턴스에 사용한 리전과 동일한 리전을 선택해야 합니다.
3. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
4. 로드 밸런서 생성을 선택하십시오.
5. [Application Load Balancer]에서 [Create]를 선택합니다.

2단계 로드 밸런서 및 리스너 구성

[Configure Load Balancer] 페이지에서 다음 절차를 완료합니다.

로드 밸런서 및 리스너를 구성하려면

1. Name에 로드 밸런서 이름을 입력합니다.

   Application Load Balancer의 이름은 해당 리전의 Application Load Balancer 및 Network Load Balancer 세트 내에서 고유한 이름이어야 하고, 최대 32자여야 하며, 알파벳 문자 및 하이픈만 포함해야 하고, 하이픈으로 시작하거나 끝나지 않아야 하며 "internal-"로 시작하지 않아야 합니다.
2. [Scheme] 및 [IP address type]은 기본값으로 유지합니다.
3. 포트 80에서 HTTP 트래픽을 수락하는 리스너를 뜻하는 [Listeners]는 기본 값으로 유지합니다.
4. [Availability Zones]에서 EC2 인스턴스에 사용한 VPC를 선택합니다. EC2 인스턴스를 시작할 때 사용한 각 가용 영역에서 가용 영역을 선택한 후 해당 가용 영역에 대한 퍼블릭 서브넷을 선택합니다.
5. Next: Configure Security Settings를 선택합니다.
6. 이 자습서의 경우 HTTPS 리스너를 생성하지 않습니다. 다음: 보안 그룹 구성을 선택합니다.

3단계: 로드 밸런서에 대한 보안 그룹 구성

로드 밸런서에 대한 보안 그룹은 로드 밸런서가 리스너 포트 및 상태 확인 포트에서 등록된 대상과 통신할 수 있도록 허용해야 합니다. 콘솔에서는 올바른 프로토콜과 포트를 지정하는 규칙을 사용하여 로드 밸런서에 대한 보안 그룹을 자동으로 생성할 수 있습니다. 원하는 경우 자체 보안 그룹을 만들고 선택할 수 있습니다. 자세한 정보는 권장 규칙 (p. 21) 단원을 참조하십시오.

보안 그룹 구성 페이지에서 다음 절차를 완료하여 Elastic Load Balancing가 사용자를 대신하여 로드 밸런서에 대한 보안 그룹을 생성하게 하십시오.

로드 밸런서에 대한 보안 그룹을 구성하려면

1. Create a new security group을 선택합니다.
2. 보안 그룹의 이름과 설명을 입력하거나 기본 이름과 설명을 유지합니다. 이 새 보안 그룹에는 [Configure Load Balancer] 페이지에서 선택한 로드 밸런서 리스너 포트로 트래픽을 허용하는 규칙이 포함되어 있습니다.
3. [Next: Configure Routing]을 선택합니다.

4단계: 대상 그룹 구성

라우팅 요청에서 사용되는 대상 그룹을 만듭니다. 리스너의 기본 규칙은 이 대상 그룹에 등록된 대상에 대해 요청을 라우팅합니다. 로드 밸런서는 해당 대상 그룹에 대해 정의된 상태 확인 설정을 사용하여 이 대상 그룹의 대상 상태를 확인합니다. [Configure Routing] 페이지에서 다음 절차를 완료합니다.

대상 그룹을 구성하려면

1. Target group(대상 그룹)에는 기본 값인 New target group(새 대상 그룹)을 그대로 둡니다.
2. Name에 새 대상 그룹의 이름을 입력합니다.
3. 기본 대상 유형(Instance(인스턴스)), 프로토콜(HTTP) 및 유형(80)을 그대로 둡니다.
4. Health checks(상태 확인)에는 기본 설정을 그대로 둡니다.
5. Next: Register Targets(다음: 대상 등록)를 선택합니다.

5단계: 대상 그룹에 대상 등록

[Register Targets] 페이지에서 다음 절차를 완료합니다.

인스턴스를 대상 그룹에 등록하려면

1. [Instances]에서 인스턴스를 하나 이상 선택합니다.
2. 기본 포트(80)를 그대로 두고 Add to registered(등록된 항목에 추가)를 선택합니다.
3. 인스턴스 선택을 마치면 [Next: Review]를 선택합니다.

6단계: 로드 밸런서 생성 및 테스트

로드 밸런서를 생성하기 전에 선택한 설정을 검토합니다. 로드 밸런서를 생성한 후에는 EC2 인스턴스에 트래픽을 전송하고 있는지 확인할 수 있습니다.

로드 밸런서를 생성 및 확인하려면

1. [Review] 페이지에서 [Create]을 선택합니다.
2. 로드 밸런서가 생성되었다는 통보를 받은 후 [Close]를 선택합니다.
3. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
4. 새로 생성한 대상 그룹을 선택합니다.
5. [Targets] 탭에서 인스턴스가 준비되었는지 확인합니다. 인스턴스 상태가 `initial`인 경우 아직 인스턴스 등록이 진행 중이거나 정상으로 간주될 만한 최소 상태 확인 횟수를 통과하지 못했기 때문일 가능성이 높습니다. 하나 이상의 인스턴스 상태가`healthy`여야 로드 밸런서를 테스트할 수 있습니다.
6. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
7. 새로 생성한 로드 밸런서를 선택합니다.
8. [Description] 탭에서 로드 밸런서의 DNS 이름(예: my-load-balancer-1234567890.us-west-2.elb.amazonaws.com)을 복사합니다. DNS 이름을 인터넷에 연결된 웹 브라우저의 주소 필드에 붙여넣습니다. 모든 것이 잘 작동하는 경우 브라우저에 서버 기본 페이지가 표시됩니다.
9. (선택 사항) 추가 리스너를 정의하려면 규칙 추가 (p. 38) 단원을 참조하십시오.

7단계: 로드 밸런서 삭제(선택 사항)

로드 밸런서를 사용할 수 있는 순간부터 실행이 지속되는 매 시간 단위 또는 60분 미만의 시간 단위로 비용이 청구됩니다. 더 이상 로드 밸런서가 필요 없을 때는 이를 삭제할 수 있습니다. 로드 밸런서가 삭제되면 그 즉시 요금 발생이 중지됩니다. 로드 밸런서를 삭제해도 로드 밸런서에 등록된 대상에는 영향을 미치지 않습니다. 예를 들어 EC2 인스턴스는 계속 실행됩니다.

로드 밸런서를 삭제하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서에 대한 확인란을 선택한 후 [Actions], [Delete]를 선택합니다.
4. 확인 메시지가 나타나면 Yes, Delete를 선택합니다.

Application Load Balancer 자습서

다음은 Application Load Balancer를 사용하는 일반적인 작업의 수행 방법을 보여 주는 Elastic Load Balancing 자습서입니다.

- Elastic Load Balancing 시작하기(Elastic Load Balancing 사용 설명서)
- 자습서: Application Load Balancer를 통한 경로 기반 라우팅 사용 (p. 7)
- 자습서: Application Load Balancer를 통해 마이크로서비스를 대상으로 사용 (p. 9)
- 자습서: AWS CLI를 이용하여 Application Load Balancer 생성 (p. 11)

자습서: Application Load Balancer를 통한 경로 기반 라우팅 사용

규칙을 통해 URL 경로를 기반으로 요청을 전달하는 리스너를 생성할 수 있습니다. 이를 경로 기반 라우팅이라고 합니다. 마이크로서비스를 실행 중인 경우 경로 기반 라우팅을 사용하여 여러 백 엔드 서비스에 트래픽을 라우팅할 수 있습니다. 예를 들어 일반 요청을 하나의 대상 그룹으로 라우팅하고 이미지를 다른 대상 그룹에 렌더링하도록 요청할 수 있습니다.

시작하기 전

- Virtual Private Cloud(VPC)에서 EC2 인스턴스를 시작합니다. 이들 인스턴스에 대한 보안 그룹이 리스너 포트 및 상태 확인 포트에서 액세스를 허용하는지 확인합니다. 자세한 내용은 대상 보안 그룹 (p. 59) 단원을 참조하십시오.
- 등록하려는 EC2 인스턴스에 마이크로서비스가 배포되어 있는지 확인합니다.

로드 밸런서 생성

경로 기반 라우팅을 사용하는 로드 밸런서를 만들려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 모음에서 EC2 인스턴스에 대해 선택한 것과 동일한 리전을 선택합니다.
3. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
4. 다음과 같이 첫 번째 대상 집합에 대한 대상 그룹을 만듭니다.
   a. [Create target group]을 선택합니다.
   b. 대상 그룹에 대한 이름, 프로토콜, 포트 및 VPC를 지정한 다음, [Create]를 선택합니다.
   c. 새로운 대상 그룹을 선택합니다.
   d. [Targets] 탭에서 [Edit]를 선택합니다.
   e. [Instances]에서 인스턴스를 하나 이상 선택합니다. 인스턴스에 대한 포트를 지정하고 [Add to registered]를 선택한 다음 [Save]를 선택합니다.

      인스턴스가 등록되고 상태 확인을 통과할 때까지 인스턴스의 상태는 `initial`이고, 로드 밸런서에서 트래픽을 수신하도록 대상 그룹을 구성할 때까지는 `unused`입니다.
5. 다음과 같이 두 번째 대상 집합에 대한 대상 그룹을 만듭니다.

a. [Create target group]을 선택합니다.
b. 대상 그룹에 대한 이름, 프로토콜, 포트 및 VPC를 지정한 다음, [Create]를 선택합니다.
c. [Targets] 탭에서 [Edit]를 선택합니다.
d. [Instances]에서 인스턴스를 하나 이상 선택합니다. 인스턴스에 대한 포트를 지정하고 [Add to registered]를 선택한 다음 [Save]를 선택합니다.

   인스턴스가 등록되고 상태 확인을 통과할 때까지 인스턴스의 상태는 `initial`이고, 로드 밸런서에서 트래픽을 수신하도록 대상 그룹을 구성할 때까지는 `unused`입니다.

6. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
7. [Create Load Balancer]를 선택합니다.
8. [Select load balancer type]에서 [Application Load Balancer]를 선택합니다.
9. [Continue]를 선택합니다.
10. 다음과 같이 Configure Load Balancer 페이지를 완료합니다.

    a. Name에 로드 밸런서 이름을 입력합니다.

       Application Load Balancer의 이름은 해당 리전의 Application Load Balancer 및 Network Load Balancer 집합 내에서 고유한 이름이어야 하고, 최대 32자여야 하며, 알파벳 문자 및 하이픈만 포함해야 하고, 하이픈으로 시작하거나 끝나지 않아야 합니다.

    b. Scheme에서 인터넷 경계 로드 밸런서는 인터넷을 통해 클라이언트의 요청을 대상으로 라우팅합니다. 내부 로드 밸런서는 프라이빗 IP 주소를 사용하여 요청을 대상으로 라우팅합니다.
    c. Listeners에서 기본값은 포트 80에서 HTTP 트래픽을 수락하는 리스너입니다. 기본 리스너 설정을 그대로 두거나 리스너의 프로토콜 또는 포트를 수정하거나 Add를 선택해 다른 리스너를 추가할 수 있습니다.
    d. [Availability Zones]에서 EC2 인스턴스에 사용한 VPC를 선택합니다. 최소 두 개의 가용 영역을 선택합니다. 가용 영역에 대해 서브넷 한 개가 있는 경우 해당 서브넷이 선택됩니다. 가용 영역에 대해 서브넷이 두 개 이상 있는 경우 서브넷 중 하나를 선택합니다. 가용 영역당 서브넷을 한 개만 선택할 수 있습니다.
    e. Next: Configure Security Settings를 선택합니다.

11. (선택 사항) 이전 단계에서 안전한 리스너를 만들었다면 다음과 같이 [Configure Security Settings] 페이지를 완료합니다.

    a. AWS Certificate Manager를 사용해 인증서를 생성하거나 가져왔다면, Choose an existing certificate from AWS Certificate Manager(ACM)(AWS Certificate Manager(ACM)에서 기존 인증서 선택)를 선택한 후 인증서 이름에서 인증서를 선택하십시오.
    b. IAM을 사용하여 인증서를 업로드했다면 [Choose an existing certificate from AWS Identity and Access Management (IAM)]를 선택한 다음 [Certificate name]에서 인증서를 선택합니다.
    c. 업로드할 인증서가 있지만 ACM이 해당 리전에서 지원되지 않는 경우에는 [Upload a new SSL Certificate to AWS Identity and Access Management (IAM)]를 선택합니다. Certificate name에 인증서의 이름을 입력합니다. Private Key에 프라이빗 키 파일(PEM 인코딩)의 콘텐츠를 복사해 붙여넣습니다. Public Key Certificate에 퍼블릭 키 인증서 파일(PEM 인코딩)의 콘텐츠를 복사해 붙여넣습니다. 자체 서명 인증서를 사용하고 있지 않고 브라우저가 인증서를 묵시적으로 수락하는 것이 중요하지 않다면 Certificate Chain에 인증서 체인 파일(PEM 인코딩)의 콘텐츠를 복사해 붙여넣습니다.
    d. [Select policy]에는 기본 보안 정책을 유지합니다.

12. Next: Configure Security Groups를 선택합니다.
13. 다음과 같이 [Configure Security Groups] 페이지를 완료합니다.

    a. [Create a new security group]을 선택합니다.
    b. 보안 그룹의 이름과 설명을 입력하거나 기본 이름과 설명을 유지합니다. 이 새 보안 그룹에는 [Configure Load Balancer] 페이지에서 로드 밸런서에 대해 선택한 포트로 트래픽을 허용하는 규칙이 포함되어 있습니다.
    c. [Next: Configure Routing]을 선택합니다.

14. 다음과 같이 [Configure Routing] 페이지를 완료합니다.
    a. [Target group]에서 `Existing target group`을 선택합니다.
    b. [Name]에서 생성한 첫 번째 대상 그룹을 선택합니다.
    c. Next: Register Targets를 선택합니다.
15. [Register Targets] 페이지에서 대상 그룹에 등록한 인스턴스가 [Registered instances] 아래에 표시됩니다. 마법사를 완료할 때까지 대상 그룹에 등록된 대상을 변경할 수 없습니다. [Next: Review]를 선택합니다.
16. [Review] 페이지에서 [Create]을 선택합니다.
17. 로드 밸런서가 생성되었다는 통보를 받은 후 [Close]를 선택합니다.
18. 새로 생성한 로드 밸런서를 선택합니다.
19. 리스너 탭에서 규칙 보기/편집을 선택한 다음, 규칙 추가 아이콘(더하기 기호)을 선택합니다. 다음과 같이 규칙을 지정합니다.
    a. [Insert Rule]을 선택합니다.
    b. 조건 추가, 경로...를 선택한 다음 경로 기반 라우팅에 대해 사용할 패턴을 정확하게 입력합니다(예: `/img/*`). 조건을 저장하려면 확인 표시 아이콘을 선택합니다. 자세한 내용은 리스너 규칙 (p. 24) 단원을 참조하십시오.
    c. 작업 추가, 전달 대상...을 선택한 다음 생성한 두 번째 대상 그룹을 선택합니다. 작업을 저장하려면 확인 표시 아이콘을 선택합니다.
    d. 규칙을 저장하려면 저장을 선택합니다.

자습서: Application Load Balancer를 통해 마이크로서비스를 대상으로 사용

마이크로서비스 아키텍처를 사용하여 애플리케이션을 독립적으로 개발하고 배포할 수 있는 서비스로 구성할 수 있습니다. 이러한 서비스 중 하나 이상을 각 서비스가 다른 포트에서 연결을 수신하는 상태로 각 EC2 인스턴스에 설치할 수 있습니다. 단일 Application Load Balancer를 사용하여 요청을 애플리케이션의 모든 서비스로 라우팅할 수 있습니다. EC2 인스턴스를 대상 그룹에 등록하면 여러 번 등록할 수 있습니다. 각 서비스에 대해 해당 서비스의 포트를 사용하여 인스턴스를 등록합니다.

> **Important**
>
> Amazon Elastic Container Service(Amazon ECS)을 사용하여 서비스를 배포하면 동적인 포트 매핑을 사용하여 동일한 컨테이너 인스턴스의 단일 서비스에서 여러 작업을 지원할 수 있습니다. Amazon ECS는 각 컨테이너에 대해 인스턴스 ID와 포트를 사용하여 대상 그룹으로 컨테이너를 자동 등록 및 등록 취소함으로써 서비스 업데이트를 관리합니다. 자세한 내용은 Amazon Elastic Container Service Developer Guide에서 서비스 로드 밸런싱을 참조하십시오.

시작하기 전에

- EC2 인스턴스를 시작합니다. 인스턴스의 보안 그룹이 리스너 포트 및 상태 확인 포트에서 로드 밸런서 보안 그룹을 통한 액세스를 허용하는지 확인합니다. 자세한 내용은 대상 보안 그룹 (p. 59) 단원을 참조하십시오.
- 서비스를 EC2 인스턴스에 배포합니다(예: 컨테이너 사용).

로드 밸런서 생성

여러 서비스를 대상으로 사용하는 로드 밸런서를 만들려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.

2. 탐색 모음에서 EC2 인스턴스에 대해 선택한 것과 동일한 리전을 선택합니다.
3. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
4. [Create Load Balancer]를 선택합니다.
5. [Select load balancer type]에서 [Application Load Balancer]를 선택합니다.
6. [Continue]를 선택합니다.
7. 다음과 같이 Configure Load Balancer 페이지를 완료합니다.

   a. Name에 로드 밸런서 이름을 입력합니다.

      Application Load Balancer의 이름은 해당 리전의 Application Load Balancer 및 Network Load Balancer 집합 내에서 고유한 이름이어야 하고, 최대 32자여야 하며, 알파벳 문자 및 하이픈만 포함해야 하고, 하이픈으로 시작하거나 끝나지 않아야 합니다.

   b. Scheme에서 인터넷 경계 로드 밸런서는 인터넷을 통해 클라이언트의 요청을 대상으로 라우팅합니다. 내부 로드 밸런서는 프라이빗 IP 주소를 사용하여 요청을 대상으로 라우팅합니다.

   c. Listeners에서 기본값은 포트 80에서 HTTP 트래픽을 수락하는 리스너입니다. 기본 리스너 설정을 그대로 두거나 리스너의 프로토콜 또는 포트를 수정하거나 Add를 선택해 다른 리스너를 추가할 수 있습니다.

   d. [Availability Zones]에서 EC2 인스턴스에 사용한 VPC를 선택합니다. 최소 두 개의 가용 영역을 선택합니다. 가용 영역에 대해 서브넷 한 개가 있는 경우 해당 서브넷이 선택됩니다. 가용 영역에 대해 서브넷이 두 개 이상 있는 경우 서브넷 중 하나를 선택합니다. 가용 영역당 서브넷을 한 개만 선택할 수 있습니다.

   e. Next: Configure Security Settings를 선택합니다.

8. (선택 사항) 이전 단계에서 안전한 리스너를 만들었다면 다음과 같이 [Configure Security Settings] 페이지를 완료합니다.

   a. AWS Certificate Manager를 사용해 인증서를 생성하거나 가져왔다면, Choose an existing certificate from AWS Certificate Manager(ACM)(AWS Certificate Manager(ACM)에서 기존 인증서 선택)를 선택한 후 인증서 이름에서 인증서를 선택하십시오.

   b. IAM을 사용하여 인증서를 업로드한 경우 Choose an existing certificate from AWS Identity and Access Management (IAM)(AWS 자격 증명 및 액세스 관리(IAM)에서 기존 인증서 선택)을 선택한 후 인증서 이름에서 인증서를 선택하십시오.

   c. 업로드할 인증서가 있지만 ACM이 해당 리전에서 지원되지 않는 경우에는 [Upload a new SSL Certificate to AWS Identity and Access Management (IAM)]를 선택합니다. Certificate name에 인증서의 이름을 입력합니다. Private Key에 프라이빗 키 파일(PEM 인코딩)의 콘텐츠를 복사해 붙여넣습니다. Public Key Certificate에 퍼블릭 키 인증서 파일(PEM 인코딩)의 콘텐츠를 복사해 붙여넣습니다. 자체 서명 인증서를 사용하고 있지 않고 브라우저가 인증서를 묵시적으로 수락하는 것이 중요하지 않다면 Certificate Chain에 인증서 체인 파일(PEM 인코딩)의 콘텐츠를 복사해 붙여넣습니다.

   d. [Select policy]에는 기본 보안 정책을 유지합니다.

9. Next: Configure Security Groups를 선택합니다.
10. 다음과 같이 [Configure Security Groups] 페이지를 완료합니다.

    a. [Create a new security group]을 선택합니다.

    b. 보안 그룹의 이름과 설명을 입력하거나 기본 이름과 설명을 유지합니다. 이 새 보안 그룹에는 [Configure Load Balancer] 페이지에서 로드 밸런서에 대해 선택한 포트로 트래픽을 허용하는 규칙이 포함되어 있습니다.

    c. [Next: Configure Routing]을 선택합니다.

11. 다음과 같이 [Configure Routing] 페이지를 완료합니다.

    a. [Target group]에서는 기본 값인 `New target group`을 유지합니다.

    b. Name에 새 대상 그룹의 이름을 입력합니다.

    c. 필요에 따라 Protocol과 Port를 설정합니다.

    d. Health checks은 기본 상태 확인 설정을 그대로 둡니다.

e. Next: Register Targets를 선택합니다.

12. [Register Targets]에서 다음 작업을 수행합니다.
    a. [Instances]에서 EC2 인스턴스를 선택합니다.
    b. 서비스에서 사용하는 포트를 입력한 다음 [Add to registered]를 선택합니다.
    c. 등록할 각 서비스에 대해 반복합니다. 작업을 마쳤으면 [Next: Review]를 선택합니다.
13. [Review] 페이지에서 [Create]을 선택합니다.
14. 로드 밸런서가 생성되었다는 통보를 받은 후 [Close]를 선택합니다.

자습서: AWS CLI를 이용하여 Application Load Balancer 생성

이 자습서에서는 AWS CLI를 통해 Application Load Balancer에 대한 실습 소개를 제공합니다.

시작하기 전

- 다음 명령을 사용하여 Application Load Balancer를 지원하는 AWS CLI 버전을 실행하고 있는지 확인하십시오.

```
aws elbv2 help
```

  elbv2가 유효한 선택이 아니라는 오류 메시지가 표시되면 AWS CLI를 업데이트하십시오. 자세한 내용은 AWS Command Line Interface 사용 설명서의 AWS 명령줄 인터페이스 설치를 참조하십시오.
- Virtual Private Cloud(VPC)에서 EC2 인스턴스를 시작합니다. 이들 인스턴스에 대한 보안 그룹이 리스너 포트 및 상태 확인 포트에서 액세스를 허용하는지 확인합니다. 자세한 내용은 대상 보안 그룹 (p. 59) 단원을 참조하십시오.

로드 밸런서 생성

첫 번째 로드 밸런서를 생성하려면 다음 단계를 완료합니다.

로드 밸런서를 생성하려면

1. create-load-balancer 명령을 사용하여 로드 밸런서를 생성합니다. 동일한 가용 영역의 서브넷이 아닌 2개의 서브넷을 지정해야 합니다.

```
aws elbv2 create-load-balancer --name my-load-balancer  \
--subnets subnet-12345678 subnet-23456789 --security-groups sg-12345678
```

   출력에는 다음 형식과 함께 로드 밸런서의 Amazon 리소스 이름(ARN)이 포함됩니다.

```
arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/my-load-
balancer/1234567890123456
```

2. create-target-group 명령을 사용하여 대상 그룹을 만들고 EC2 인스턴스에 사용한 VPC와 동일한 VPC를 지정합니다.

```
aws elbv2 create-target-group --name my-targets --protocol HTTP --port 80 \
--vpc-id vpc-12345678
```

출력에는 다음 형식과 함께 대상 그룹의 ARN이 포함됩니다.

```
arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-
targets/1234567890123456
```

3. 다음과 같이 register-targets 명령을 사용하여 인스턴스를 대상 그룹에 등록합니다.

```
aws elbv2 register-targets --target-group-arn targetgroup-arn  \
--targets Id=i-12345678 Id=i-23456789
```

4. 다음과 같이 create-listener 명령을 사용하여 요청을 대상 그룹에 전달하는 기본 규칙이 있는 로드 밸런서에 대한 하나 이상의 리스너를 생성합니다.

```
aws elbv2 create-listener --load-balancer-arn loadbalancer-arn \
--protocol HTTP --port 80  \
--default-actions Type=forward,TargetGroupArn=targetgroup-arn
```

출력에는 다음 형식과 함께 리스너의 ARN이 포함됩니다.

```
arn:aws:elasticloadbalancing:us-east-2:123456789012:listener/app/my-load-
balancer/1234567890123456/1234567890123456
```

5. (선택 사항) 다음과 같이 이 https://docs.aws.amazon.com/cli/latest/reference/elbv2/describe-target-health.html 명령을 사용하여 대상 그룹에 등록된 대상의 상태를 확인할 수 있습니다.

```
aws elbv2 describe-target-health --target-group-arn targetgroup-arn
```

HTTPS 리스너 추가

HTTPS 리스너가 있는 로드 밸런서가 있는 경우 다음과 같이 HTTPS 리스너를 추가할 수 있습니다.

로드 밸런서에 HTTPS 리스너를 추가하려면

1. 다음 방법 중 하나를 사용하여 로드 밸런서와 함께 사용할 SSL 인증서를 만듭니다.
   - AWS Certificate Manager(ACM)를 사용하여 인증서를 만들거나 가져오십시오. 자세한 내용은 AWS Certificate Manager 사용 설명서에서 인증서 요청 또는 인증서 가져오기를 참조하십시오.
   - AWS Identity and Access Management(IAM)를 사용하여 인증서를 업로드하십시오. 자세한 내용은 IAM 사용 설명서에서 서버 인증서 작업을 참조하십시오.
2. create-listener 명령을 사용하여 요청을 대상 그룹에 전달하는 기본 규칙이 있는 하나 이상의 리스너를 생성합니다. HTTPS 리스너를 만들 때 SSL 인증서를 지정해야 합니다. `--ssl-policy` 옵션을 사용하여 기본값 이외의 SSL 정책을 지정할 수 있습니다.

```
aws elbv2 create-listener --load-balancer-arn loadbalancer-arn \
--protocol HTTPS --port 443  \
--certificates CertificateArn=certificate-arn \
--default-actions Type=forward,TargetGroupArn=targetgroup-arn
```

포트 재정의를 사용하여 대상 추가

단일 인스턴스에 여러 ECS 컨테이너가 있는 경우 각 컨테이너는 다른 포트에서 연결을 허용합니다. 매번 다른 포트를 사용하여 대상 그룹에 인스턴스를 여러 번 등록할 수 있습니다.

포트 재정의를 사용하여 대상을 추가하려면

1. 다음과 같이 create-target-group 명령을 사용하여 대상 그룹을 만듭니다.

```
aws elbv2 create-target-group --name my-targets --protocol HTTP --port 80 \
--vpc-id vpc-12345678
```

2. 다음과 같이 register-targets 명령을 사용하여 인스턴스를 대상 그룹에 등록합니다. 각 컨테이너마다 인스턴스 ID가 동일하지만 포트는 서로 다릅니다.

```
aws elbv2 register-targets --target-group-arn targetgroup-arn  \
--targets Id=i-12345678,Port=80 Id=i-12345678,Port=766
```

3. 다음과 같이 create-rule 명령을 사용하여 요청을 대상 그룹에 전달하는 규칙을 리스너에 추가합니다.

```
aws elbv2 create-rule --listener-arn listener-arn --priority 10 \
--actions Type=forward,TargetGroupArn=targetgroup-arn
```

경로 기반 라우팅 추가

하나의 대상 그룹에 요청을 전달하는 기본 규칙이 있는 리스너가 있는 경우, URL을 기반으로 다른 대상 그룹에 요청을 전달하는 규칙을 추가할 수 있습니다. 예를 들어 일반 요청을 하나의 대상 그룹으로 라우팅하고 이미지를 다른 대상 그룹에 표시하도록 요청할 수 있습니다.

경로 패턴이 있는 리스너에 규칙을 추가하려면

1. 다음과 같이 create-target-group 명령을 사용하여 대상 그룹을 만듭니다.

```
aws elbv2 create-target-group --name my-targets --protocol HTTP --port 80 \
--vpc-id vpc-12345678
```

2. 다음과 같이 register-targets 명령을 사용하여 인스턴스를 대상 그룹에 등록합니다.

```
aws elbv2 register-targets --target-group-arn targetgroup-arn  \
--targets Id=i-12345678 Id=i-23456789
```

3. URL에 지정된 패턴이 있는 경우 다음과 같이 create-rule 명령을 사용하여 요청을 대상 그룹에 전달하는 규칙을 리스너에 추가합니다.

```
aws elbv2 create-rule --listener-arn listener-arn --priority 10 \
--conditions Field=path-pattern,Values='/img/*' \
--actions Type=forward,TargetGroupArn=targetgroup-arn
```

로드 밸런서 삭제

더 이상 로드 밸런서 및 대상 그룹이 필요하지 않으면 다음과 같이 삭제할 수 있습니다.

```
aws elbv2 delete-load-balancer --load-balancer-arn loadbalancer-arn
aws elbv2 delete-target-group --target-group-arn targetgroup-arn
```

Application Load Balancers

로드 밸런서는 클라이언트에 대한 단일 점점 역할을 수행합니다. 클라이언트는 로드 밸런서에 요청을 전송하고 로드 밸런서는 두 개 이상의 가용 영역에 있는 EC2 인스턴스 같은 대상으로 로드 밸런서를 전송합니다. 로드 밸런서를 구성하려는 경우, 대상 그룹 (p. 50)을 생성한 다음 대상을 해당 대상 그룹에 등록합니다. 리스너 (p. 24)를 생성하여 클라이언트의 연결 요청을 확인하고, 클라이언트에서 하나 이상의 대상 그룹에 있는 대상으로 요청을 라우팅하는 리스너 규칙을 만듭니다.

자세한 내용은 Elastic Load Balancing 사용 설명서의 Elastic Load Balancing 작동 방식을 참조하십시오.

목차

로드 밸런서를 위한 서브넷

로드 밸런서 생성 시 2개 이상의 가용 영역에서 퍼블릭 서브넷 1개를 지정해야 합니다. 가용 영역당 1개의 퍼블릭 서브넷만 지정할 수 있습니다.

로드 밸런서가 적절하게 확장 가능하도록 로드 밸런서를 위한 각 서브넷에 최소 한 개의 `/27`비트 마스크(예: `10.0.0.0/27`)를 가진 CIDR 블록이 있고 사용 가능한 IP 주소가 8개 이상 있는지 확인합니다. 로드 밸런서는 이러한 IP 주소를 사용하여 대상에 대한 연결을 설정합니다.

로드 밸런서 보안 그룹

보안 그룹은 로드 밸런서와 송수신이 허용되는 트래픽을 제어하는 방화벽 역할을 합니다. 인바운드 및 아웃바운드 트래픽을 허용하는 포트 및 프로토콜을 선택할 수 있습니다.

로드 밸런서 보안 그룹과 관련된 보안 그룹에 대한 규칙은 리스너와 상태 확인 포트 모두에서 양방향으로 트래픽을 허용해야 합니다. 로드 밸런서에 리스너를 추가하거나 대상 그룹의 상태 확인 포트를 업데이트할 때마다 보안 그룹 규칙을 검토하여 양방향으로 새로운 포트에서 양방향 트래픽을 허용하는지 확인해야 합니다. 자세한 내용은 권장 규칙 (p. 21) 단원을 참조하십시오.

로드 밸런서 상태

로드 밸런서는 다음 중 하나의 상태일 수 있습니다.

`provisioning`

로드 밸런서를 설정하는 중입니다.

`active`

로드 밸런서가 완전히 설정되어 트래픽을 라우팅할 준비가 되었습니다.

`failed`

로드 밸런서를 설정할 수 없습니다.

로드 밸런서 속성

다음은 로드 밸런서의 속성입니다.

`access_logs.s3.enabled`

Amazon S3의 액세스 로그를 활성화할지 여부를 나타냅니다. 기본값은 `false`입니다.

`access_logs.s3.bucket`

액세스 로그에 대한 S3 버킷 이름입니다. 이 속성은 액세스 로그가 활성화된 경우에 필요합니다. 자세한 내용은 버킷 권한 (p. 88) 단원을 참조하십시오.

`access_logs.s3.prefix`

S3 버킷의 위치에 대한 접두사입니다.

`deletion_protection.enabled`

삭제 방지 기능의 활성화 여부를 나타냅니다. 기본값은 `false`입니다.

`idle_timeout.timeout_seconds`

유휴 제한 시간 값(초). 기본값은 60초입니다.

`routing.http2.enabled`

HTTP/2가 활성화되었는지를 나타냅니다. 기본값은 `true`입니다.

IP 주소 유형

인터넷 경계 로드 밸런서를 생성할 때 또는 활성화한 후에 IP 주소 유형을 설정할 수 있습니다. 내부 로드 밸런서는 IPv4 주소를 사용해야 합니다.

다음은 로드 밸런서 IP 주소 유형입니다.

`ipv4`

로드 밸런서는 IPv4 주소만 지원합니다(예: 192.0.2.1).

`dualstack`

로드 밸런서는 IPv4 및 IPv6 주소를 모두 지원합니다(예: 2001:0db8:85a3:0:0:8a2e:0370:7334).

IPv4 주소를 사용하여 로드 밸런서와 통신하는 클라이언트는 A 레코드를 확인하고 IPv6 주소를 사용하여 로드 밸런서와 통신하는 클라이언트는 AAAA 레코드를 확인합니다. 하지만 클라이언트가 로드 밸런서와 통신하는 방법에 관계없이 로드 밸런서는 IPv4 주소를 사용하여 대상과 통신합니다.

자세한 내용은 Application Load Balancer IP 주소 유형 (p. 22) 단원을 참조하십시오.

삭제 방지

로드 밸런서가 실수로 삭제되지 않도록 방지하려면, 삭제 방지 기능을 활성화할 수 있습니다. 기본 설정상 로드 밸런서에 대한 삭제 방지 기능은 비활성화되어 있습니다.

로드 밸런서용 삭제 방지 기능을 활성화하는 경우 로드 밸런서를 삭제하기 전에 이 기능을 먼저 비활성화해야 합니다.

콘솔을 사용하여 삭제 방지 기능을 활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. [Description] 탭에서 [Edit attributes]를 선택합니다.
5. [Edit load balancer attributes] 페이지에서 [Delete Protection]에 대해 [Enable]을 선택한 다음 [Save]를 선택합니다.
6. Save를 선택합니다.

콘솔을 사용하여 삭제 방지 기능을 비활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. [Description] 탭에서 [Edit attributes]를 선택합니다.
5. [Edit load balancer attributes] 페이지에서 [Delete Protection]에 대해 [Enable]을 취소한 다음 [Save]를 선택합니다.
6. Save를 선택합니다.

AWS CLI를 사용하여 삭제 방지 기능을 활성화 또는 비활성화하려면

`deletion_protection.enabled` 속성과 함께 modify-load-balancer-attributes 명령을 사용합니다.

유휴 연결 제한 시간

클라이언트가 로드 밸런서를 통해 생성하는 각 요청에 대해 로드 밸런서는 두 가지 연결을 유지합니다. 프런트 엔드 연결은 클라이언트와 로드 밸런서 사이이고 백 엔드 연결은 로드 밸런서와 대상 사이의 연결입니다. 로드 밸런서는 지정된 시간 동안 프런트 엔드 연결을 통해 전송되는 데이터가 없을 때 트리거되는 유휴 제한 시간을 관리합니다. 유휴 제한 시간이 경과할 때까지 데이터가 전송되거나 전송 또는 수신되지 않으면 로드 밸런서는 프런트 엔드 연결을 종료합니다.

기본적으로 Elastic Load Balancing은 유휴 제한 시간을 60초로 설정합니다. 따라서 요청이 진행되는 동안 대상에서 최소 60초마다 데이터를 전송하지 않으면 로드 밸런서가 프런트 엔드 연결을 종료할 수 있습니다. 파

일 업로드 같이 시간이 오래 걸리는 작업이 완료될 수 있도록 시간 여유를 두려면 유휴 제한 시간이 지나기 전에 최소 1바이트의 데이터를 전송하고 필요에 따라 유휴 제한 시간의 길이를 늘립니다.

백 엔드 연결의 경우 EC2 인스턴스에 대해 HTTP 연결 유지 옵션을 활성화하는 것이 좋습니다. HTTP 연결 유지는 EC2 인스턴스의 웹 서버 설정에서 활성화할 수 있습니다. HTTP 연결 유지를 활성화하면 연결 유지 초과 시간이 만료될 때까지 로드 밸런서가 백엔드 연결을 다시 사용할 수 있습니다. 또한 애플리케이션의 유휴 제한 시간을 로드 밸런서에 구성된 유휴 제한 시간보다 크게 설정하는 것이 좋습니다.

콘솔을 사용하여 유휴 제한 시간 값을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. [Description] 탭에서 [Edit attributes]를 선택합니다.
5. [Edit load balancer attributes] 페이지에서 [Idle timeout] 값(초)을 입력합니다. 값의 범위는 1~4000초입니다. 기본값은 60초입니다.
6. Save를 선택합니다.

AWS CLI를 사용하여 유휴 제한 시간 값을 업데이트하려면

`idle_timeout.timeout_seconds` 속성과 함께 modify-load-balancer-attributes 명령을 사용합니다.

Application Load Balancer 및 AWS WAF

AWS WAF와 함께 Application Load Balancer를 사용하여 웹 ACL(웹 액세스 제어 목록)의 규칙에 따라 요청을 허용하거나 차단할 수 있습니다. 자세한 내용은 AWS WAF 개발자 안내서에서 Web ACL 작업을 참조하십시오.

로드 밸런서가 AWS WAF와 통합되는지 여부를 확인하려면 AWS Management 콘솔에서 로드 밸런서를 선택하고 Integrated services(통합 서비스) 탭을 선택합니다.

Application Load Balancer 생성

로드 밸런서는 클라이언트로부터 요청을 가져와서 대상 그룹의 대상에 이를 분산합니다.

시작하기 전에 대상에서 사용하는 각 가용 영역에 하나 이상의 퍼블릭 서브넷이 있는 VPC(Virtual Private Cloud)가 있는지 확인합니다.

AWS CLI를 사용하여 로드 밸런서 생성하려면 자습서: AWS CLI를 이용하여 Application Load Balancer 생성 (p. 11) 단원을 참조하십시오.

AWS Management 콘솔을 사용하여 로드 밸런서를 생성하려면 다음 작업을 완료합니다.

작업

- 1단계: 로드 밸런서 및 리스너 구성 (p. 4)
- 2단계: HTTPS 리스너에 대한 보안 설정 구성 (p. 18)
- 3단계: 보안 그룹 구성 (p. 18)
- 4단계: 대상 그룹 구성 (p. 5)
- 5단계: 대상 그룹에 대한 대상 구성 (p. 19)
- 6단계: 로드 밸런서 생성 (p. 20)

1단계: 로드 밸런서 및 리스너 구성

먼저 이름, 네트워크, 1개 이상의 리스너 등 로드 밸런서의 몇 가지 기본 구성 정보를 제공합니다. 리스너는 연결 요청을 확인하는 프로세스입니다. 클라이언트와 로드 밸런서 간의 연결을 위한 프로토콜 및 포트로 구성됩니다. 지원되는 프로토콜 및 포트에 대한 자세한 내용은 리스너 구성 (p. 24) 섹션을 참조하십시오.

로드 밸런서 및 리스너를 구성하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서 생성을 선택하십시오.
4. [Application Load Balancer]에서 [Create]를 선택합니다.
5. Name에 로드 밸런서 이름을 입력합니다. 예, **`my-alb`**.
6. Scheme에서 인터넷 경계 로드 밸런서는 인터넷을 통해 클라이언트의 요청을 대상으로 라우팅합니다. 내부 로드 밸런서는 프라이빗 IP 주소를 사용하여 요청을 대상으로 라우팅합니다.
7. IP address type(IP 주소 유형)에서 서브넷이 IPv4 주소를 사용하는 경우 ipv4를 선택하고 서브넷이 IPv4 주소와 IPv6 주소를 모두 사용하는 경우 dualstack을 선택합니다.
8. Listeners에서 기본값은 포트 80에서 HTTP 트래픽을 수락하는 리스너입니다. 기본 리스너 설정을 그대로 두거나 프로토콜 또는 포트를 변경합니다. [Add]를 선택하여 다른 리스너를 추가합니다(예: HTTPS 리스너).
9. [Availability Zones]의 경우 VPC에서 둘 이상의 가용 영역을 선택합니다. 가용 영역에 대해 서브넷 한 개가 있는 경우 해당 서브넷이 선택됩니다. 가용 영역에 대해 서브넷이 두 개 이상 있는 경우 서브넷 중 하나를 선택합니다. 가용 영역당 서브넷을 한 개만 선택할 수 있습니다.
10. 다음: 보안 설정 구성을 선택합니다.

2단계: HTTPS 리스너에 대한 보안 설정 구성

이전 단계에서 HTTPS 리스너를 생성한 경우 필수 보안 설정을 구성합니다. 또는 마법사의 다음 페이지로 이동합니다.

로드 밸런서 리스너에서 HTTPS를 사용할 때 로드 밸런서에 SSL 인증서를 반드시 배포해야 합니다. 로드 밸런서는 이 인증서를 사용해 연결을 종료하고 대상으로 전송하기 전에 클라이언트의 요청을 해독합니다. 자세한 내용은 SSL 인증서 (p. 33) 단원을 참조하십시오. 로드 밸런서가 클라이언트와 SSL 연결을 협상하는 데 사용하는 보안 정책도 지정해야 합니다. 자세한 내용은 보안 정책 (p. 35) 섹션을 참조하십시오.

인증서 및 보안 정책을 구성하려면

1. [Select default certificate]에서 다음 중 하나를 수행합니다.
   - AWS Certificate Manager를 사용하여 인증서를 생성하거나 가져왔다면 Choose a certificate from ACM(ACM에서 인증서 선택)를 선택한 후 인증서 이름에서 인증서를 선택하십시오.
   - IAM을 사용하여 인증서를 업로드했다면 Choose a certificate from IAM(IAM에서 인증서 선택)을 선택한 후 인증서 이름에서 인증서를 선택하십시오.
2. [Security policy]에서 기본 보안 정책을 유지하는 것이 좋습니다.
3. Next: Configure Security Groups(다음: 보안 그룹 구성)를 선택하십시오.

3단계: 보안 그룹 구성

로드 밸런서에 대한 보안 그룹은 로드 밸런서가 리스너 포트 및 상태 확인 포트에서 등록된 대상과 통신할 수 있도록 허용해야 합니다. 콘솔은 이 통신을 허용하는 규칙을 통해 사용자 대신 로드 밸런서에 대한 보

안 그룹을 만들 수 있습니다. 원하는 경우 보안 그룹을 만들고 선택할 수 있습니다. 자세한 정보는 권장 규칙 (p. 21) 단원을 참조하십시오.

로드 밸런서에 대한 보안 그룹을 구성하려면

1. Create a new security group을 선택합니다.
2. 보안 그룹의 이름과 설명을 입력하거나 기본 이름과 설명을 유지합니다. 이 새 보안 그룹에는 [Configure Load Balancer] 페이지에서 로드 밸런서에 대해 선택한 포트로 트래픽을 허용하는 규칙이 포함되어 있습니다.
3. [Next: Configure Routing]을 선택합니다.

4단계: 대상 그룹 구성

대상 그룹에 대상을 등록합니다. 이 단계에서 구성하는 대상 그룹은 기본 리스너 규칙의 대상 그룹으로 사용되며, 이 규칙은 요청을 대상 그룹에 전달합니다. 자세한 정보는 Application Load Balancer 대상 그룹 (p. 50) 단원을 참조하십시오.

대상 그룹을 구성하려면

1. 대상 그룹에서는 기본값인 New target group(새 대상 그룹)을 유지합니다.
2. [Name]에 대상 그룹의 이름을 입력합니다.
3. Target type(대상 유형)에서 인스턴스 ID를 기준으로 대상을 등록하려면 Instance(인스턴스), IP 주소를 등록하려면 IP, Lambda 함수를 등록하려면 Lambda function(Lambda 함수)를 선택합니다.
4. (선택 사항) 대상 유형이 Instance(인스턴스) 또는 IP인 경우 필요에 따라 포트와 프로토콜을 수정합니다.
5. (선택 사항) 대상 유형이 Lambda function(Lambda 함수)인 경우 필요에 따라 상태 확인을 활성화합니다.
6. Health checks은 기본 상태 확인 설정을 그대로 둡니다.
7. Next: Register Targets(다음: 대상 등록)를 선택합니다.

5단계: 대상 그룹에 대한 대상 구성

Application Load Balancer를 사용하면 대상 그룹의 대상 유형에 따라 대상을 대상 그룹에 등록하는 방법이 결정됩니다.

인스턴스 ID로 대상을 등록하려면

1. [Instances]에서 인스턴스를 하나 이상 선택합니다.
2. 인스턴스 리스너 포트를 입력한 다음 [Add to registered]를 선택합니다.
3. 인스턴스 등록을 마치면 [Next: Review]를 선택합니다.

IP 주소를 등록하려면

1. 등록할 각 IP 주소에 대해 다음을 수행합니다.
   a. IP 주소가 대상 그룹 VPC의 서브넷에서 온 경우 [Network]에서 VPC를 선택합니다. 그렇지 않은 경우 [Other private IP address]를 선택합니다.
   b. [IP]에는 IP 주소를 입력합니다.
   c. [Port]에 포트를 입력합니다.
   d. [Add to list]를 선택합니다.

2. 목록에 IP 주소를 추가했으면 [Next: Review]를 선택합니다.

Lambda 함수를 등록하려면

1. Lambda function(Lambda 함수)에 대해 다음 중 하나를 수행합니다.
   - Lambda 함수 선택
   - 새 Lambda 함수를 생성하고 선택
   - 대상 그룹을 생성한 후 Lambda 함수를 등록
2. [Next: Review]를 선택합니다.

6단계: 로드 밸런서 생성

로드 밸런서를 생성한 후, 대상이 초기 상태 확인을 통과했는지 확인한 다음 로드 밸런서가 대상으로 트래픽을 전송하고 있는지 테스트할 수 있습니다. 로드 밸런서를 완료하면 이를 삭제할 수 있습니다. 자세한 내용은 Application Load Balancer 삭제 (p. 23) 섹션을 참조하십시오.

로드 밸런서를 생성하려면

1. [Review] 페이지에서 [Create]을 선택합니다.
2. 로드 밸런서가 생성된 후 [Close]를 선택합니다.
3. (선택 사항) 경로 패턴이나 호스트 이름을 기반으로 요청을 전달하는 추가 리스너 규칙을 정의하려면 규칙 추가 (p. 38) 단원을 참조하십시오.

Application Load Balancer 가용 영역

로드 밸런서의 가용 영역을 언제든지 활성화 또는 비활성화할 수 있습니다. 가용 영역을 활성화하고 나면 로드 밸런서가 해당 가용 영역의 등록 대상으로 요청을 라우팅하기 시작합니다. 활성화된 각 가용 영역에 등록된 대상이 하나 이상 있는지 확인하는 경우에 로드 밸런서가 가장 효과적입니다.

가용 영역을 비활성화하고 나면 해당 가용 영역의 대상은 로드 밸런서에 등록된 상태로 유지되지만 로드 밸런서는 요청을 대상으로 라우팅하지 않습니다.

콘솔을 사용하여 가용 영역을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. [Description] 탭에서 [Basic Configuration] 아래 [Edit Availability Zones]를 선택합니다.
5. 가용 영역을 활성화하려면 해당 가용 영역 확인란을 선택합니다. 가용 영역에 대해 서브넷 한 개가 있는 경우 해당 서브넷이 선택됩니다. 가용 영역에 대해 서브넷이 두 개 이상 있는 경우 서브넷 중 하나를 선택합니다. 가용 영역당 서브넷을 한 개만 선택할 수 있습니다.
6. 활성화된 가용 영역의 서브넷을 변경하려면 [Change subnet]을 선택하고 다른 서브넷 중 하나를 선택합니다.
7. 가용 영역을 제거하려면 해당 가용 영역 확인란을 선택 취소합니다.
8. Save를 선택합니다.

AWS CLI를 사용하여 가용 영역을 업데이트하려면

set-subnets 명령을 사용합니다.

Application Load Balancer 보안 그룹

로드 밸런서가 리스너 포트 및 상태 확인 포트에서 등록된 대상과 통신을 할 수 있는지 확인해야 합니다. 로드 밸런서에 리스너를 추가하거나 로드 밸런서가 요청을 라우팅하기 위해 사용하는 대상 그룹의 상태 확인 포트를 업데이트할 때마다 로드 밸런서와 연결된 보안 그룹이 새로운 포트에서 양방향 트래픽을 허용하는지 확인해야 합니다. 그렇지 않은 경우 현재 연결된 보안 그룹의 규칙을 편집하거나 여러 보안 그룹을 로드 밸런서와 연결할 수 있습니다.

권장 규칙

권장 규칙은 로드 밸런서의 유형(인터넷 경계 또는 내부)에 따라 다릅니다.

인터넷 경계 로드 밸런서

Inbound		
Source	Port Range	Comment
0.0.0.0/0	*###*	로드 밸런서 리스너 포트에서 모든 인바운드 트래픽을 허용
Outbound		
Destination	Port Range	Comment
#### ## ##	*#### ###*	인스턴스 리스너 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다
#### ## ##	*## ##*	상태 확인 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다

내부 로드 밸런서

Inbound		
Source	Port Range	Comment
VPC CIDR	*###*	로드 밸런서 리스너 포트에서 VPC CIDR에서 오는 인바운드 트래픽을 허용
Outbound		
Destination	Port Range	Comment
#### ## ##	*#### ###*	인스턴스 리스너 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다
#### ## ##	*## ##*	상태 확인 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다

인바운드 ICMP 트래픽이 경로 MTU 검색을 지원하도록 허용하는 것이 좋습니다. 자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서에서 경로 MTU 검색을 참조하십시오.

연결된 보안 그룹 업데이트

로드밸런서와 연결된 보안 그룹을 언제든지 업데이트할 수 있습니다.

콘솔을 사용하여 보안 그룹을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. [Description] 탭에서 [Security] 아래 [Edit security groups]를 선택합니다.
5. 로드 밸런서에 보안 그룹을 연결하려면 보안 그룹을 선택합니다. 로드 밸런서에서 보안 그룹을 제거하려면 보안 그룹을 선택 취소합니다.
6. Save를 선택합니다.

AWS CLI를 사용하여 보안 그룹을 업데이트하려면

set-security-groups 명령을 사용합니다.

Application Load Balancer IP 주소 유형

Application Load Balancer를 구성하여 IPv4 트래픽만 라우팅하거나 IPv4 및 IPv6 트래픽을 모두 라우팅할 수 있습니다. 자세한 내용은 IP 주소 유형 (p. 15) 섹션을 참조하십시오.

IPv6 요구 사항

- 인터넷 경계 로드 밸런서.
- Virtual Private Cloud(VPC)에는 연결된 IPv6 CIDR 블록이 있는 서브넷이 있습니다. 자세한 내용은 Amazon EC2 사용 설명서에서 IPv6 주소를 참조하십시오.

콘솔을 사용하여 IP 주소 유형을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. [Actions], [Edit IP address type]을 선택합니다.
5. IP address type의 경우 ipv4를 선택하여 IPv4 주소만 지원하거나 dualstack을 선택하여 IPv4 주소와 IPv6 주소를 모두 지원합니다.
6. Save를 선택합니다.

AWS CLI를 사용하여 IP 주소 유형을 업데이트하려면

set-ip-address-type 명령을 사용합니다.

Application Load Balancer 태그

태그는 용도, 소유자, 환경 등 다양한 방식으로 로드 밸런서를 분류할 수 있도록 해줍니다.

각 로드 밸런서에 여러 태그를 추가할 수 있습니다. 태그 키는 각 로드 밸런서에 대해 고유해야 합니다. 로드 밸런서에 이미 연결된 키를 통해 태그를 추가하면 해당 태그의 값이 업데이트됩니다.

태그 사용을 마치면 로드 밸런서에서 이를 제거할 수 있습니다.

제한 사항

- 리소스당 최대 태그 수 — 50개

- 최대 키 길이 — 유니코드 문자 127자
- 최대 값 길이 — 유니코드 문자 255자
- 태그 키와 값은 대/소문자를 구분합니다. 허용되는 문자는 UTF-8로 표현할 수 있는 문자, 공백 및 숫자와 특수 문자: + - = . _ : / @. 선행 또는 후행 공백을 사용하면 안 됩니다.
- 태그 이름이나 값에서 `aws:` 접두사는 사용하지 마십시오. 이 단어는 AWS용으로 예약되어 있습니다. 이 접두사가 지정된 태그 이름이나 값은 편집하거나 삭제할 수 없습니다. 이 접두사가 지정된 태그는 리소스당 태그 수 제한에 포함되지 않습니다.

콘솔을 사용하여 로드 밸런서 태그를 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. [Tags] 탭에서 [Add/Edit Tags]를 선택하고 다음 중 하나 이상의 작업을 수행합니다.
   a. 태그를 업데이트하려면 [Key] 및 [Value] 값을 수정합니다.
   b. 새로운 태그를 추가하려면 [Create Tag]를 선택한 다음 [Key] 및 [Value] 값을 입력합니다.
   c. 태그를 삭제하려면 해당 태그 옆의 삭제 아이콘(X)을 선택합니다.
5. 태그 업데이트를 마쳤으면 [Save]를 선택합니다.

AWS CLI를 사용하여 로드 밸런서 태그를 업데이트하려면

add-tags 및 remove-tags 명령을 사용합니다.

Application Load Balancer 삭제

로드 밸런서를 사용할 수 있는 순간부터 실행이 지속되는 매 시간 단위 또는 60분 미만의 시간 단위로 비용이 청구됩니다. 더 이상 로드 밸런서가 필요 없을 때는 이를 삭제할 수 있습니다. 로드 밸런서가 삭제되면 그 즉시 요금 발생이 중지됩니다.

삭제 방지 기능이 활성화되어 있으면 로드 밸런서를 삭제할 수 없습니다. 자세한 내용은 삭제 방지 (p. 16) 단원을 참조하십시오.

로드 밸런서를 삭제해도 등록된 대상에는 영향을 미치지 않습니다. 예를 들어 EC2 인스턴스는 계속 실행되고 대상 그룹에 계속 등록됩니다. 대상 그룹을 삭제하려면 대상 그룹 삭제 (p. 68) 섹션을 참조하십시오.

콘솔을 사용하여 로드 밸런서를 삭제하려면

1. 로드 밸런서를 가리키는 도메인을 위한 CNAME 레코드가 있는 경우에는 새로운 위치를 가리키도록 하고 로드 밸런서를 삭제하기 전에 DNS 변경이 적용될 때까지 기다립니다.
2. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
3. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
4. 로드 밸런서를 선택한 다음 [Actions], [Delete]를 차례로 선택합니다.
5. 확인 메시지가 나타나면 예, 삭제를 선택하십시오.

AWS CLI를 사용하여 로드 밸런서를 삭제하려면

delete-load-balancer 명령을 사용합니다.

애플리케이션 로드 밸런서를 위한 리스너

Application Load Balancer를 사용하기 전에 먼저 하나 이상의 리스너를 추가해야 합니다. 리스너는 구성한 프로토콜 및 포트를 사용하여 연결 요청을 확인하는 프로세스입니다. 리스너에 대해 정의한 규칙에 따라 로드 밸런서가 하나 이상의 대상 그룹에서 대상으로 요청을 라우팅하는 방법이 결정됩니다.

목차

리스너 구성

리스너는 다음과 같은 프로토콜 및 포트를 지원합니다.

- 프로토콜: HTTP, HTTPS
- 포트: 1-65535

애플리케이션이 비즈니스 로직에 집중할 수 있도록 HTTPS 리스너를 사용하여 암호화 및 암호 해독 작업을 로드 밸런서로 오프로드할 수 있습니다. 리스너 프로토콜이 HTTPS인 경우에는 리스너에 한 개 이상의 SSL 서버 인증서를 반드시 배포해야 합니다. 자세한 정보는 Application Load Balancer용 HTTPS 리스너 생성 (p. 33) 단원을 참조하십시오.

Application Load Balancer는 WebSockets에 대한 기본 지원을 제공합니다. HTTP 및 HTTPS 리스너 모두에서 WebSockets를 사용할 수 있습니다.

Application Load Balancer는 HTTPS 리스너에서 HTTP/2에 대한 기본 지원을 제공합니다. 하나의 HTTP/2 연결을 이용해 최대 128개의 요청을 동시에 전송할 수 있습니다. 로드 밸런서는 이러한 요청을 개별 HTTP/1.1 요청으로 변환하고 대상 그룹의 정상 상태 대상 간에 배포합니다. HTTP/2는 프런트 엔드 연결을 보다 효율적으로 사용하기 때문에 클라이언트와 로드 밸런서 간의 연결을 줄일 수 있습니다. HTTP/2의 서버 푸시 기능을 사용할 수 없습니다.

자세한 내용은 Elastic Load Balancing 사용 설명서의 라우팅 요청을 참조하십시오.

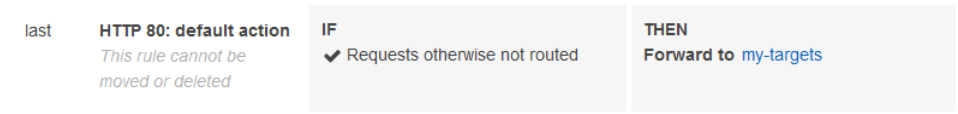
리스너 규칙

각 리스너는 기본 규칙을 가지고 있으며, 선택에 따라 추가 규칙을 정의할 수 있습니다. 각 규칙은 우선 순위, 하나 이상의 작업, 하나 이상의 조건으로 구성됩니다. 언제든 규칙을 추가하거나 편집할 수 있습니다. 자세한 내용은 규칙 편집 (p. 39) 단원을 참조하십시오.

기본 규칙

리스너를 생성할 때 기본 규칙에 대한 작업을 정의합니다. 기본 규칙은 조건을 가질 수 없습니다. 리스너의 규칙에 대한 조건이 충족되지 않으면 기본 규칙에 대해 작업이 수행됩니다.

다음은 콘솔에 나타난 기본 규칙을 보여줍니다.



규칙 우선 순위

각 규칙마다 우선 순위가 있습니다. 규칙은 가장 낮은 값에서 가장 높은 값에 이르기까지 우선 순위에 따라 평가됩니다. 기본 규칙은 마지막에 평가됩니다. 기본이 아닌 규칙의 우선 순위는 언제든지 변경이 가능합니다. 기본 규칙의 우선 순위는 변경할 수 없습니다. 자세한 내용은 규칙 재정렬 (p. 40) 단원을 참조하십시오.

규칙 작업

각 규칙 작업에는 작업을 수행하는 데 필요한 유형, 순서 및 정보가 있습니다. 자세한 내용은 규칙 작업 유형 (p. 25) 단원을 참조하십시오.

규칙 조건

각 규칙 조건에는 유형과 구성 정보가 있습니다. 규칙에 대한 조건이 충족되면 작업이 수행됩니다. 자세한 내용은 규칙 조건 형식 (p. 28) 단원을 참조하십시오.

규칙 작업 유형

규칙에 대해 지원되는 작업 유형은 다음과 같습니다.

`authenticate-cognito`

[HTTPS 리스너] Amazon Cognito를 사용하여 사용자를 인증합니다. 자세한 내용은 Application Load Balancer를 사용하여 사용자 인증 (p. 43) 단원을 참조하십시오.

`authenticate-oidc`

[HTTPS 리스너] OpenID Connect(OIDC)와 호환되는 자격 증명 공급자를 사용하여 사용자를 인증합니다.

`fixed-response`

사용자 지정 HTTP 응답을 반환합니다. 자세한 내용은 고정 응답 작업 (p. 26) 단원을 참조하십시오.

`forward`

요청을 지정된 대상 그룹으로 전달합니다.

`redirect`

한 URL의 요청을 다른 URL로 리디렉션합니다. 자세한 내용은 리디렉션 작업 (p. 26) 단원을 참조하십시오.

가장 낮은 순서 값이 있는 작업이 첫 번째로 수행됩니다. 각 규칙에는 `forward`, `redirect` 또는 `fixed-response` 작업 중 하나가 꼭 포함되어 있어야 하며, 이 작업이 수행할 마지막 작업이어야 합니다.

고정 응답 작업

`fixed-response` 작업을 사용하여 클라이언트 요청을 삭제하고 사용자 지정 HTTP 응답을 반환할 수 있습니다. 이 작업을 사용하여 2XX, 4XX, 5XX 응답 코드와 선택적 메시지를 반환할 수 있습니다.

`fixed-response` 작업이 수행되면 해당 작업과 리디렉션 대상의 URL이 액세스 로그에 기록됩니다. 자세한 내용은 액세스 로그 항목 (p. 82) 섹션을 참조하십시오. 성공한 `fixed-response` 작업의 개수는 `HTTP_Fixed_Response_Count` 지표에 보고됩니다. 자세한 내용은 Application Load Balancer 지표 (p. 71)을(를) 참조하십시오.

Example AWS CLI의 고정 응답 작업 예

규칙을 만들거나 수정할 때 작업을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하십시오. 다음 작업은 지정된 상태 코드와 메시지 본문이 있는 고정 응답을 보냅니다.

```
[
  {
      "Type": "fixed-response",
      "FixedResponseConfig": {
          "StatusCode": "200",
          "ContentType": "text/plain",
          "MessageBody": "Hello world"
      }
  }
]
```

전달 작업

`forward` 작업을 사용하여 요청을 지정된 대상 그룹으로 라우팅할 수 있습니다.

Example AWS CLI의 Forward 작업 예

규칙을 만들거나 수정할 때 작업을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하십시오. 다음 작업은 요청을 지정된 대상 그룹으로 전달합니다.

```
[
  {
      "Type": "forward",
      "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a06"
  }
]
```

리디렉션 작업

`redirect` 작업을 사용하여 한 URL의 클라이언트 요청을 다른 URL로 리디렉션할 수 있습니다. 요구 사항에 따라 리디렉션을 임시(HTTP 302) 또는 영구(HTTP 301)로 구성할 수 있습니다.

URI는 다음과 같은 구성 요소로 이루어집니다.

```
protocol://hostname:port/path?query
```

리디렉션 루프를 피하기 위해 프로토콜, 호스트 이름 포트, 경로 중 최소 하나를 수정해야 합니다. 수정하지 않은 구성 요소는 원래 값을 유지합니다.

protocol

프로토콜(HTTP 또는 HTTPS)입니다. HTTP를 HTTP로, HTTP를 HTTPS로, HTTPS를 HTTPS로 리디렉션할 수 있습니다. HTTPS를 HTTP로 리디렉션할 수는 없습니다.

hostname

호스트 이름입니다. 호스트 이름은 대/소문자를 구분하지 않고, 길이가 최대 128자일 수 있으며, 영숫자, 와일드카드(* 및 ?), 하이픈(-)으로 구성됩니다.

port

포트입니다(1~65535).

경로

"/"로 시작하는 절대 경로입니다. 경로는 대/소문자를 구분하고, 길이가 최대 128자일 수 있으며, 영숫자, 와일드카드(* 및 ?), &(& 사용), 특수 문자 _-.$/~'"@:+로 구성됩니다.

query

쿼리 파라미터입니다

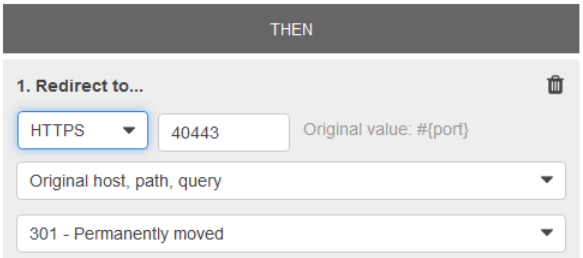
다음 예약 키워드를 사용하여 원래 URL의 URI 구성 요소를 대상 URL에 다시 사용할 수 있습니다.

- `#{protocol}` - 프로토콜을 포함합니다. protocol 및 query 구성 요소에 사용합니다.
- `#{host}` - 도메인을 포함합니다. hostname, path, query 구성 요소에 사용합니다.
- `#{port}` - 포트를 포함합니다. port, path, query 구성 요소에 사용합니다.
- `#{path}` - 경로를 포함합니다. path 및 query 구성 요소에 사용합니다.
- `#{query}` - 쿼리 파라미터를 포함합니다. query 구성 요소에 사용합니다.

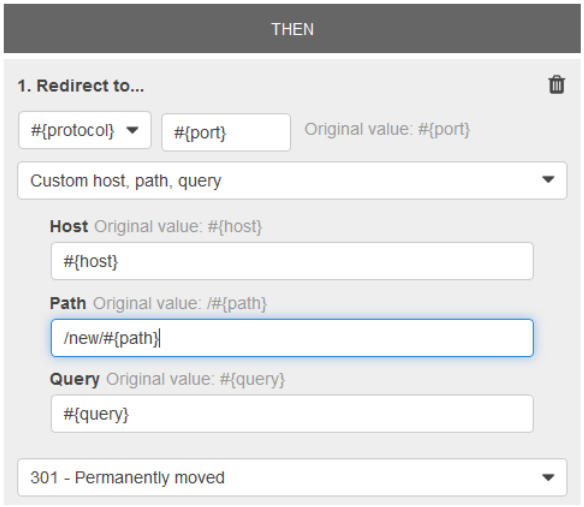
`redirect`작업이 수행되면 액세스 로그에 작업이 기록됩니다. 자세한 내용은 액세스 로그 항목 (p. 82) 섹션을 참조하십시오. 성공한 `redirect` 작업의 개수는 `HTTP_Redirect_Count` 지표에 보고됩니다. 자세한 내용은 Application Load Balancer 지표 (p. 71)을(를) 참조하십시오.

Example 콘솔을 사용한 리디렉션 작업 예

다음 규칙은 HTTPS 프로토콜과 지정된 포트(40443)를 사용하는 URL로의 영구 리디렉션을 설정하지만, 원래 호스트 이름, 경로, 쿼리 파라미터를 포함합니다. 이 화면은 "https://#{host}:40443/#{path}?#{query}"와 동일합니다.



다음 규칙은 원래 프로토콜, 포트, 호스트 이름, 쿼리 파라미터를 포함하는 URL로의 영구 리디렉션을 설정하고, `#{path}` 키워드를 사용하여 수정된 경로를 생성합니다. 이 화면은 "#{protocol}://#{host}:#{port}/new/#{path}?#{query}"와 동일합니다.



Example AWS CLI의 리디렉션 작업 예

규칙을 만들거나 수정할 때 작업을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하십시오. 다음 작업은 포트 443에서 HTTP 요청을 HTTPS 요청으로 리디렉션합니다. 호스트 이름, 경로, 쿼리 문자열은 HTTP 요청과 동일합니다.

```
[
  {
      "Type": "redirect",
      "RedirectConfig": {
          "Protocol": "HTTPS",
          "Port": "443",
          "Host": "#{host}",
          "Path": "/#{path}",
          "Query": "#{query}",
          "StatusCode": "HTTP_301"
      }
  }
]
```

규칙 조건 형식

규칙에 대해 지원되는 조건 형식은 다음과 같습니다.

`host-header`

각 요청의 호스트 이름을 기반으로 라우팅합니다. 자세한 내용은 호스트 조건 (p. 30) 단원을 참조하십시오.

`http-header`

각 요청의 HTTP 헤더를 기반으로 라우팅합니다. 자세한 내용은 HTTP 헤더 조건 (p. 29) 단원을 참조하십시오.

`http-request-method`

각 요청의 HTTP 요청 메서드를 기반으로 라우팅합니다. 자세한 내용은 HTTP 요청 메서드 조건 (p. 29) 단원을 참조하십시오.

`path-pattern`

요청 URL의 경로 패턴을 기반으로 라우팅합니다. 자세한 내용은 경로 조건 (p. 30) 단원을 참조하십시오.

`query-string`

쿼리 문자열의 키/값 페어 또는 값을 기반으로 라우팅합니다. 자세한 내용은 쿼리 문자열 조건 (p. 31) 단원을 참조하십시오.

`source-ip`

각 요청의 소스 IP 주소를 기반으로 라우팅합니다. 자세한 내용은 소스 IP 주소 조건 (p. 32) 단원을 참조하십시오.

각 규칙에는 `host-header`, `http-request-method`, `path-pattern`, `source-ip` 조건 중 0개 또는 1개가 포함될 수 있으며, `http-header` 및 `query-string` 조건 중 0개 이상이 포함될 수 있습니다.

조건당 최대 3개의 일치 평가를 지정할 수 있습니다. 예를 들어 각 `http-header` 조건에 대해 요청의 HTTP 헤더 값과 비교할 최대 3개의 문자열을 지정할 수 있습니다. 문자열 중 하나가 HTTP 헤더 값과 일치하면 조건이 충족됩니다. 모든 문자열이 일치하도록 요구하려면 일치 평가마다 조건 하나를 만듭니다.

규칙당 최대 5개의 일치 평가를 지정할 수 있습니다. 예를 들어 조건 5개 각각에 일치 평가가 하나씩 있는 규칙을 만들 수 있습니다.

`http-header`, `host-header`, `path-pattern`, `query-string` 조건의 일치 평가에 와일드카드 문자를 포함시킬 수 있습니다. 규칙당 와일드카드 문자는 5개로 제한됩니다.

HTTP 헤더 조건

HTTP 헤더 조건을 사용하여 요청의 HTTP 헤더를 기반으로 요청을 라우팅하는 규칙을 구성할 수 있습니다. 표준 또는 사용자 지정 HTTP 헤더 필드의 이름을 지정할 수 있습니다. 헤더 이름과 일치 평가는 대/소문자를 구분하지 않습니다. 비교 문자열에서는 *(0개 이상의 문자 일치) 및 ?(정확히 1자 일치) 와일드카드 문자가 지원됩니다. 와일드카드 문자는 헤더 이름에서는 지원되지 않습니다.

Example AWS CLI의 HTTP 헤더 조건 예

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하십시오. 다음 조건은 지정된 문자열 중 하나와 일치하는 User-Agent 헤더가 있는 요청에 의해 충족됩니다.

```
[
  {
      "Field": "http-header",
      "HttpHeaderConfig": {
          "HttpHeaderName": "User-Agent",
          "Values": ["*Chrome*", "*Safari*"]
      }
  }
]
```

HTTP 요청 메서드 조건

HTTP 요청 메서드 조건을 사용하여 요청의 HTTP 요청 메서드를 기반으로 요청을 라우팅하는 규칙을 구성할 수 있습니다. 표준 또는 사용자 지정 HTTP 메서드를 지정할 수 있습니다. 일치 평가는 대/소문자를 구분합니다. 와일드카드 문자는 지원되지 않으므로 메서드 이름이 정확히 일치해야 합니다.

GET 및 HEAD 요청을 동일한 방식으로 라우팅하는 것이 좋습니다. HEAD 요청에 대한 응답이 캐싱될 수 있기 때문입니다.

Example AWS CLI의 HTTP 메서드 조건 예

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하십시오. 다음 조건은 지정된 메서드를 사용하는 요청에 의해 충족됩니다.

```
[
  {
      "Field": "http-request-method",
      "HttpRequestMethodConfig": {
          "Values": ["CUSTOM-METHOD"]
      }
  }
]
```

호스트 조건

호스트 조건을 사용하여 호스트 헤더의 호스트 이름을 기반으로 요청을 라우팅하는 규칙을 정의할 수 있습니다(호스트 기반 라우팅이라고도 함). 따라서 단일 로드 밸런서를 사용하여 여러 개의 도메인을 지원할 수 있습니다.

호스트 이름은 대/소문자를 구별하지 않고 최대 255자이며 다음과 같은 문자를 포함할 수 있습니다:

- A–Z, a–z, 0–9
- - .
- * (0개 이상의 문자에 해당)
- ?(정확히 1자 일치)

최소 "." 한 글자 포함해야 합니다. 마지막 "." 문자 다음에는 알파벳만 포함할 수 있습니다.

호스트 이름 예제

- **example.com**
- **test.example.com**
- ***.example.com**

규칙 ***.example.com**은 **test.example.com**과 일치하나 **example.com**과 일치하지 않습니다.

Example AWS CLI의 호스트 헤더 조건 예

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하십시오. 다음 조건은 지정된 문자열과 일치하는 호스트 헤더가 있는 요청에 의해 충족됩니다.

```
[
  {
      "Field": "host-header",
      "HostHeaderConfig": {
          "Values": ["*.example.com"]
      }
  }
]
```

경로 조건

경로 조건을 사용하여 요청의 URL을 기반으로 요청을 라우팅하는 규칙을 정의할 수 있습니다(경로 기반 라우팅이라고도 함).

경로 패턴은 쿼리 파라미터가 아닌 URL의 경로에만 적용됩니다.

경로 이름은 대/소문자를 구별하지 않고 최대 128자이며 다음과 같은 문자를 포함할 수 있습니다.

- A–Z, a–z, 0–9
- _ - . $ / ~ " ' @ : +
- &(& 사용)
- * (0개 이상의 문자에 해당)
- ?(정확히 한 글자에 해당)

경로 패턴 예제

- `/img/*`
- `/js/*`

경로 패턴은 요청을 라우팅하는 데 사용되지만 요청을 변경하지 않습니다. 예를 들어 `/img/*`라는 경로 패턴을 가진 규칙은 `/img/picture.jpg`에 대한 요청을 `/img/picture.jpg`를 위한 요청으로서 지정된 대상 그룹에 전달합니다.

Example AWS CLI의 경로 패턴 조건 예

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하십시오. 다음 조건은 지정된 문자열이 포함된 URL이 있는 요청에 의해 충족됩니다.

```
[
  {
      "Field": "path-pattern",
      "PathPatternConfig": {
          "Values": ["/img/*"]
      }
  }
]
```

쿼리 문자열 조건

쿼리 문자열 조건을 사용하여 쿼리 문자열의 키/값 페어 또는 값을 기반으로 요청을 라우팅하는 규칙을 구성할 수 있습니다. 일치 평가는 대/소문자를 구분하지 않습니다. *(0개 이상의 문자 일치) 및 ?(정확히 1자 일치) 와일드카드 문자가 지원됩니다.

Example AWS CLI의 쿼리 문자열 조건 예

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하십시오. 다음 조건은 키/값 페어 "version=v1" 또는 “example”로 설정된 키가 포함된 쿼리 문자열이 있는 요청에 의해 충족됩니다.

```
[
  {
      "Field": "query-string",
      "QueryStringConfig": {
          "Values": [
            {
                "Key": "version",
                "Value": "v1"
            },
            {
                "Value": "example"
            }
```

```
            ]
        }
    }
]
```

소스 IP 주소 조건

소스 IP 주소 조건을 사용하여 요청의 소스 IP 주소를 기반으로 요청을 라우팅하는 규칙을 구성할 수 있습니다. IP 주소는 CIDR 형식으로 지정해야 합니다. IPv4 및 IPv6 주소를 모두 사용할 수 있습니다. 와일드카드 문자는 지원되지 않습니다.

클라이언트가 프록시 뒤에 있는 경우, 이는 클라이언트의 IP 주소가 아니라 프록시의 IP 주소입니다.

이 조건은 X-Forwarded-For 헤더의 주소에 의해 충족되지 않습니다. X-Forwarded-For 헤더에서 주소를 검색하려면 `http-header` 조건을 사용합니다.

Example AWS CLI의 소스 IP 조건 예

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하십시오. 다음 조건은 지정된 CIDR 블록 중 하나에 소스 IP 주소가 있는 요청에 의해 충족됩니다.

```
[
  {
      "Field": "source-ip",
      "SourceIpConfig": {
          "Values": ["192.0.2.0/24", "198.51.100.10/32"]
      }
  }
]
```

Application Load Balancer용 HTTP 리스너 생성

리스너는 연결 요청을 확인하는 프로세스입니다. 로드 밸런서를 생성할 때 리스너를 정의하면 언제라도 로드 밸런서에 리스너를 추가할 수 있습니다.

이 페이지의 정보는 로드 밸런서용 HTTP 리스너를 생성하는 데 도움이 됩니다. 로드 밸런서에 HTTPS 리스너를 추가하려면 Application Load Balancer용 HTTPS 리스너 생성 (p. 33) 단원을 참조하십시오.

사전 조건

- 기본 리스너 규칙에 전달 작업을 추가하려면 사용 가능한 대상 그룹을 지정해야 합니다. 자세한 내용은 대상 그룹 생성 (p. 54) 단원을 참조하십시오.

HTTP 리스너 추가

리스너에서 클라이언트에서 로드 밸런서로의 연결을 위한 프로토콜 및 포트 번호와 기본 리스너 규칙에 대한 대상 그룹을 구성합니다. 자세한 내용은 리스너 구성 (p. 24) 단원을 참조하십시오.

콘솔을 사용하여 HTTPS 리스너를 추가하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱 아래에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택한 후 [Listeners], [Add listener]를 선택합니다.
4. 프로토콜 : 포트에서 HTTP를 선택하고 기본 포트를 유지하거나 다른 포트를 입력합니다.

5. 기본 작업에서 다음 중 하나를 수행합니다.
   - 작업 추가, 전달 대상을 선택하고 대상 그룹을 선택합니다.
   - 작업 추가, 다음으로 리디렉션을 선택하고 리디렉션 URL을 입력하십시오. 자세한 내용은 리디렉션 작업 (p. 26)를 참조하십시오.
   - 작업 추가, 고정 응답 반환을 선택하고 응답 코드와 응답 본문(선택 사항)을 입력하십시오. 자세한 내용은 고정 응답 작업 (p. 26)를 참조하십시오.

   작업을 저장하려면 확인 표시 아이콘을 선택합니다.
6. Save를 선택합니다.
7. (선택 사항) 경로 패턴이나 호스트 이름을 기반으로 요청을 전달하는 추가 리스너 규칙을 정의하려면 규칙 추가 (p. 38) 단원을 참조하십시오.

AWS CLI를 사용하여 HTTP 리스너를 추가하려면

리스너 및 기본 규칙을 생성하려면 create-listener 명령을, 추가 리스너 규칙을 정의하려면 create-rule 명령을 사용하십시오.

Application Load Balancer용 HTTPS 리스너 생성

리스너는 연결 요청을 확인하는 프로세스입니다. 로드 밸런서를 생성할 때 리스너를 정의하면 언제라도 로드 밸런서에 리스너를 추가할 수 있습니다.

암호화된 연결(SSL 오프로드라고도 함)을 사용하는 HTTPS 리스너를 생성할 수 있습니다. 이 기능을 사용하면 로드 밸런서와 SSL 또는 TLS 세션을 시작하는 클라이언트 간에 트래픽 암호화가 가능합니다.

이 페이지의 정보는 로드 밸런서용 HTTPS 리스너를 생성하는 데 도움이 됩니다. 로드 밸런서에 HTTP 리스너를 추가하려면 Application Load Balancer용 HTTP 리스너 생성 (p. 32) 단원을 참조하십시오.

목차

SSL 인증서

HTTPS 리스너를 사용하려면 로드 밸런서에 한 개 이상의 SSL/TLS 서버 인증서를 반드시 배포해야 합니다. 로드 밸런서는 이 인증서를 사용해 연결을 종료한 다음, 대상으로 전송하기 전에 클라이언트의 요청을 해독합니다.

로드 밸런서에는 X.509 인증서(SSL/TLS 서버 인증서)가 필요합니다. 인증서는 인증 기관(CA)에서 발행한 디지털 형태의 ID 증명서입니다. 인증서에는 식별 정보, 유효 기간, 퍼블릭 키, 일련번호, 발행자의 디지털 서명이 들어 있습니다.

로드 밸런서와 함께 사용할 인증서를 생성할 때 도메인 이름을 지정해야 합니다.

AWS Certificate Manager (ACM)을(를) 사용하여 로드 밸런서에 대한 인증서를 생성하는 것이 좋습니다. ACM은(는) Elastic Load Balancing과(와) 통합하므로 로드 밸런서에 인증서를 배포할 수 있습니다. 자세한 내용은 AWS Certificate Manager 사용 설명서 섹션을 참조하십시오.

Important

ACM은 4096 키 길이의 RSA 및 EC 인증서를 지원합니다. 하지만, ACM.과의 통합을 통해 로드 밸런서에 이러한 인증서를 설치할 수는 없습니다. 로드 밸런서와 함께 사용하기 위해서는 이러한 인증서를 IAM에 업로드해야 합니다.

또는 SSL/TLS 도구를 사용해 인증서 서명 요청(CSR)을 생성하고 CA가 서명한 CSR을 가져와서 인증서를 만든 다음, ACM로 인증서를 가져오거나 AWS Identity and Access Management(IAM)으로 인증서를 업로드할 수 있습니다. 인증서를 ACM(으)로 가져오는 작업에 대한 자세한 내용은 AWS Certificate Manager 사용 설명서에서 인증서 가져오기를 참조하십시오. IAM에 인증서를 업로드하는 방법에 대한 자세한 내용은 IAM 사용 설명서에서 서버 인증서 작업을 참조하십시오.

기본 인증서

HTTPS 리스너를 생성할 때 인증서 하나를 꼭 지정해야 합니다. 이 인증서를 기본 인증서라고 합니다.

인증서 목록 (p. 34)에서 추가 인증서를 지정하면 클라이언트가 SNI(서버 이름 표시) 프로토콜을 사용하지 않고 호스트 이름을 지정하여 연결하거나 인증서 목록에 일치하는 인증서가 없는 경우에만 기본 인증서가 사용됩니다.

추가 인증서를 지정하지 않지만 단일 로드 밸런서를 통해 보안 애플리케이션을 여러 개 호스팅해야 하는 경우, 와일드카드 인증서를 사용하거나 인증서에 각 추가 도메인의 주체 대체 이름(SAN)을 추가할 수 있습니다.

인증서 목록

HTTPS 리스너를 생성한 후 리스너에는 기본 인증서와 빈 인증서 목록이 있습니다. 필요에 따라 리스너의 인증서 목록에 인증서를 추가할 수 있습니다. 인증서 목록을 사용하면 로드 밸런서가 동일한 포트의 여러 도메인을 지원하고 각 도메인에 대해 다른 인증서를 제공할 수 있습니다.

로드 밸런서는 SNI를 지원하는 스마트 인증서 선택 알고리즘을 사용합니다. 클라이언트가 제공한 호스트 이름이 인증서 목록의 단일 인증서와 일치하면 로드 밸런서는 이 인증서를 선택합니다. 클라이언트가 제공한 호스트 이름이 인증서 목록의 여러 인증서와 일치하면 로드 밸런서는 클라이언트가 지원할 수 있는 최선의 인증서를 선택합니다. 인증서 선택은 다음 조건에 따라 다음 순서대로 이루어집니다.

- 퍼블릭 키 알고리즘(RSA보다 ECDSA 선호)
- 해싱 알고리즘(MD5보다 SHA 선호)
- 키 길이(가장 큰 길이 선호)
- 유효 기간

로드 밸런서 액세스 로그 항목은 클라이언트가 지정한 호스트 이름과 클라이언트에 제공된 인증서를 나타냅니다. 자세한 내용은 액세스 로그 항목 (p. 82) 단원을 참조하십시오.

인증서 갱신

각 인증서에는 유효 기간이 있습니다. 유효 기간이 끝나기 전에 로드 밸런서의 각 인증서를 갱신 또는 교체해야 합니다. 여기에는 기본 인증서와 인증서 목록의 인증서가 포함됩니다. 인증서를 갱신 또는 교체해도 로드 밸런서 노드에 수신되어 상태가 양호한 대상으로 라우팅이 보류 중인 진행 중 요청에는 영향을 주지 않습니다. 인증서를 갱신하면 새 요청에서 갱신된 인증서를 사용합니다. 인증서를 교체하면 새 요청에서 새 인증서를 사용합니다.

인증서 갱신 및 교체를 다음과 같이 관리할 수 있습니다.

- AWS Certificate Manager가 제공하고 로드 밸런서에 배포된 인증서는 자동으로 갱신이 가능합니다. ACM는 인증서가 만료되기 전에 인증서 갱신을 시도합니다. 자세한 내용은 AWS Certificate Manager 사용 설명서의 관리형 갱신을 참조하십시오.
- ACM에 인증서를 가져온 경우에는 인증서의 만료일을 반드시 모니터링해서 만료되기 전에 인증서를 갱신해야 합니다. 자세한 내용은 AWS Certificate Manager 사용 설명서에서 인증서 가져오기를 참조하십시오.

- IAM으로 인증서를 가져온 경우, 새 인증서를 만들어 ACM 또는 IAM으로 가져온 후 로드 밸런서에 새 인증서를 추가하고, 만료된 인증서를 로드 밸런서에서 제거해야 합니다.

보안 정책

Elastic Load Balancing는 보안 정책이라고 하는 Secure Socket Layer(SSL) 협상 구성을 사용해 클라이언트와 로드 밸런서 간의 연결을 협상하십시오. 보안 정책은 프로토콜과 암호의 조합입니다. 프로토콜은 클라이언트와 서버 간에 보안 연결을 설정하여 클라이언트와 로드 밸런서 간에 전달되는 모든 데이터를 안전하게 보호합니다. 암호는 코딩된 메시지를 생성하기 위해 암호화 키를 사용하는 암호화 알고리즘입니다. 프로토콜은 여러 개의 암호를 사용해 인터넷 상의 데이터를 암호화합니다. 연결 협상이 이루어지는 동안 클라이언트와 로드 밸런서는 각각이 지원하는 암호 및 프로토콜 목록을 선호도 순으로 표시합니다. 기본적으로 서버의 목록에서 클라이언트의 암호 중 하나와 일치하는 첫 번째 암호가 보안 연결을 위해 선택됩니다.

Application Load Balancer는 클라이언트 또는 대상 연결에 대한 SSL 재협상을 지원하지 않습니다.

프런트 엔드 연결에서 사용되는 보안 정책을 선택할 수 있습니다. 백엔드 연결에는 `ELBSecurityPolicy-2016-08` 보안 정책이 항상 사용됩니다. Application Load Balancer는 사용자 지정 보안 정책을 지원하지 않습니다.

Elastic Load Balancing는 Application Load Balancer에 대해 다음의 보안 정책을 제공합니다.

- `ELBSecurityPolicy-2016-08`
- `ELBSecurityPolicy-FS-2018-06`
- `ELBSecurityPolicy-TLS-1-2-2017-01`
- `ELBSecurityPolicy-TLS-1-2-Ext-2018-06`
- `ELBSecurityPolicy-TLS-1-1-2017-01`
- `ELBSecurityPolicy-2015-05`
- `ELBSecurityPolicy-TLS-1-0-2015-04`

일반 용도에 적합한 `ELBSecurityPolicy-2016-08` 정책을 사용하는 것이 좋습니다. FS(Forward Secrecy)가 필요한 경우 `ELBSecurityPolicy-FS-2018-06` 정책을 사용할 수 있습니다. `ELBSecurityPolicy-TLS` 정책 중 하나를 사용하여 특정한 TLS 프로토콜 버전을 비활성화해야 하는 규정 준수 및 보안 표준을 충족하거나 암호 사용 중지가 필요한 기존 클라이언트를 지원할 수 있습니다. 인터넷 클라이언트만 TLS 버전 1.0이 필요한 인터넷 클라이언트 비율은 적습니다. 로드 밸런서에 대한 요청에서 TLS 프로토콜 버전을 확인하려면 로드 밸런서에서 액세스 로깅을 활성화하고 액세스 로그를 검사하십시오. 자세한 내용은 액세스 로그 (p. 81)를 참조하십시오.

다음 표는 Application Load Balancer에 대해 정의된 보안 정책을 설명합니다.

보안 정책	2016-08 *	FS-2018-0	TLS-1-2	TLS-1-2-Ext	TLS-1-1	TLS-1-0 †
TLS 프로토콜						
Protocol-TLSv1	♦	♦				♦
Protocol-TLSv1.1	♦	♦			♦	♦
Protocol-TLSv1.2	♦	♦	♦	♦	♦	♦
TLS 암호						
ECDHE-ECDSA-AES128-GCM-SHA256	♦	♦	♦	♦	♦	♦

보안 정책	2016-08 *	FS-2018-0	TLS-1-2	TLS-1-2-Ext	TLS-1-1	TLS-1-0 †
ECDHE-RSA-AES128-GCM-SHA256	♦	♦	♦	♦	♦	♦
ECDHE-ECDSA-AES128-SHA256	♦	♦	♦	♦	♦	♦
ECDHE-RSA-AES128-SHA256	♦	♦	♦	♦	♦	♦
ECDHE-ECDSA-AES128-SHA	♦	♦		♦	♦	♦
ECDHE-RSA-AES128-SHA	♦	♦		♦	♦	♦
ECDHE-ECDSA-AES256-GCM-SHA384	♦	♦	♦	♦	♦	♦
ECDHE-RSA-AES256-GCM-SHA384	♦	♦	♦	♦	♦	♦
ECDHE-ECDSA-AES256-SHA384	♦	♦	♦	♦	♦	♦
ECDHE-RSA-AES256-SHA384	♦	♦	♦	♦	♦	♦
ECDHE-RSA-AES256-SHA	♦	♦		♦	♦	♦
ECDHE-ECDSA-AES256-SHA	♦	♦		♦	♦	♦
AES128-GCM-SHA256	♦		♦	♦	♦	♦
AES128-SHA256	♦		♦	♦	♦	♦
AES128-SHA	♦			♦	♦	♦
AES256-GCM-SHA384	♦		♦	♦	♦	♦
AES256-SHA256	♦		♦	♦	♦	♦
AES256-SHA	♦			♦	♦	♦
DES-CBC3-SHA						♦

* Application Load Balancer에 대한 `ELBSecurityPolicy-2016-08` 및 `ELBSecurityPolicy-2015-05` 보안 정책은 동일합니다.

† 보안이 약한 DES-CBC3-SHA 암호를 필요로 하는 기존 클라이언트를 지원하지 않는 한 이 보안 정책을 사용해서는 안 됩니다.

AWS CLI을(를) 사용하여 Application Load Balancer에 대한 보안 정책 구성을 보려면 describe-ssl-policies 명령을 사용하십시오.

HTTPS 리스너 추가

리스너에서 클라이언트에서 로드 밸런서로의 연결을 위한 프로토콜 및 포트 번호와 기본 리스너 규칙에 대한 대상 그룹을 구성합니다. 자세한 내용은 리스너 구성 (p. 24) 단원을 참조하십시오.

사전 조건

- 기본 리스너 규칙에 전달 작업을 추가하려면 사용 가능한 대상 그룹을 지정해야 합니다. 자세한 내용은 대상 그룹 생성 (p. 54) 단원을 참조하십시오.

- HTTPS 리스너를 생성하려면 인증서와 보안 정책을 지정해야 합니다. 로드 밸런서는 이 인증서를 사용해 연결을 종료하고 대상으로 전송하기 전에 클라이언트의 요청을 해독합니다. 로드 밸런서는 클라이언트와 SSL 연결을 협상할 때 보안 정책을 사용합니다.

콘솔을 사용하여 HTTPS 리스너를 추가하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱 아래에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택한 후 [Listeners], [Add listener]를 선택합니다.
4. 프로토콜 : 포트에서 HTTPS를 선택하고 기본 포트를 유지하거나 다른 포트를 입력합니다.
5. (선택 사항) 사용자를 인증하려면 Default actions(기본 작업)에서 Add action(작업 추가), Authenticate(인증)를 선택하고 요청된 정보를 제공합니다. 작업을 저장하려면 확인 표시 아이콘을 선택합니다. 자세한 내용은 Application Load Balancer를 사용하여 사용자 인증 (p. 43)를 참조하십시오.
6. 기본 작업에서 다음 중 하나를 수행합니다.
   - 작업 추가, 전달 대상을 선택하고 대상 그룹을 선택합니다.
   - 작업 추가, 다음으로 리디렉션을 선택하고 리디렉션 URL을 입력하십시오. 자세한 내용은 리디렉션 작업 (p. 26)를 참조하십시오.
   - 작업 추가, 고정 응답 반환을 선택하고 응답 코드와 응답 본문(선택 사항)을 입력하십시오. 자세한 내용은 고정 응답 작업 (p. 26)를 참조하십시오.

   작업을 저장하려면 확인 표시 아이콘을 선택합니다.
7. [Security policy]에서 기본 보안 정책을 유지하는 것이 좋습니다.
8. Default SSL certificate(기본 SSL 인증서)에 대해 다음 중 하나를 수행합니다.
   - AWS Certificate Manager을 사용하여 인증서를 생성하거나 가져온 경우 ACM에서 시작을 선택하고 인증서를 선택하십시오.
   - IAM을 사용하여 인증서를 업로드한 경우 IAM에서 시작을 선택하고 인증서를 선택하십시오.
9. Save를 선택합니다.
10. (선택 사항) 경로 패턴이나 호스트 이름을 기반으로 요청을 전달하는 추가 리스너 규칙을 정의하려면 규칙 추가 (p. 38) 단원을 참조하십시오.
11. (선택 사항) SNI 프로토콜에 사용할 인증서 목록을 추가하려면 인증서 목록에 인증서 추가 (p. 42) 단원을 참조하십시오.

AWS CLI를 사용하여 HTTPS 리스너를 추가하려면

리스너 및 기본 규칙을 생성하려면 create-listener 명령을, 추가 리스너 규칙을 정의하려면 create-rule 명령을 사용하십시오.

HTTPS 리스너 업데이트

HTTPS 리스너를 생성한 후 기본 인증서를 교체하거나, 인증서 목록을 업데이트하거나, 보안 정책을 교체할 수 있습니다. 자세한 내용은 Application Load Balancer용 HTTPS 리스너 업데이트 (p. 41) 단원을 참조하십시오.

애플리케이션 로드 밸런서를 위한 리스너 규칙

리스너에 대해 정의한 규칙에 따라 로드 밸런서가 하나 이상의 대상 그룹에서 대상으로 요청을 라우팅하는 방법이 결정됩니다.

각 규칙은 우선 순위, 하나 이상의 작업, 하나 이상의 조건으로 구성됩니다. 자세한 내용은 리스너 규칙 (p. 24) 단원을 참조하십시오.

Note

콘솔에는 규칙 우선 순위가 아니라 각 규칙에 대한 상대적 순차 번호가 표시됩니다. AWS CLI 또는 Elastic Load Balancing API를 사용하여 설명함으로써 규칙의 우선 순위를 알아낼 수 있습니다.

요구 사항

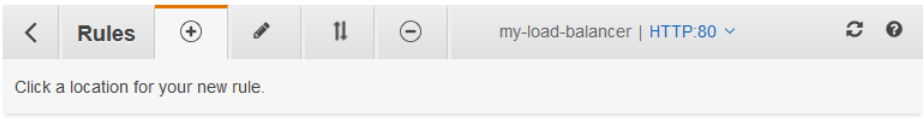
- 각 규칙에는 `forward`, `redirect` 또는 `fixed-response` 작업 중 하나가 꼭 포함되어 있어야 하며, 이 작업이 수행할 마지막 작업이어야 합니다.
- 각 규칙에는 `host-header`, `http-request-method`, `path-pattern`, `source-ip` 조건 중 0개 또는 1개가 포함될 수 있으며, `http-header` 및 `query-string` 조건 중 0개 이상이 포함될 수 있습니다.
- 조건당 최대 3개의 비교 문자열과 규칙당 최대 5개의 비교 문자열을 지정할 수 있습니다.
- `forward` 작업은 요청을 대상 그룹으로 라우팅합니다. `forward` 작업을 추가하기 전에 대상 그룹을 만들고 대상 그룹에 대상을 추가합니다. 자세한 내용은 대상 그룹 생성 (p. 54) 단원을 참조하십시오.

규칙 추가

리스너를 생성할 때 기본 규칙을 정의하면 언제라도 기본이 아닌 추가 규칙을 정의할 수 있습니다.

콘솔을 사용하여 규칙을 추가하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택한 다음 [Listeners]를 선택합니다.
4. 리스너를 업데이트하려면 [View/edit rules]를 선택합니다.
5. 메뉴 모음에서 [Add rules] 아이콘(+ 기호)을 선택하여 우선 순위에 따라 규칙을 삽입할 수 있는 위치에 Insert Rule 아이콘을 추가합니다.



6. 이전 단계에서 추가한 Insert Rule(규칙 삽입) 아이콘 중 하나를 선택합니다.
7. 다음과 같이 조건을 하나 이상 추가합니다.

   a. 호스트 헤더 조건을 추가하려면 Add condition(조건 추가), Host header(호스트 헤더)를 선택하고 호스트 이름을 입력합니다(예: **`*.example.com`**). 조건을 저장하려면 확인 표시 아이콘을 선택합니다.

      각 문자열의 최대 크기는 128자입니다. 이 비교는 대/소문자를 구분하지 않습니다. * 및 ? 와일드카드 문자가 지원됩니다.

   b. 경로 조건을 추가하려면 Add condition(조건 추가), Path(경로)를 선택하고 경로 패턴을 입력합니다(예: `/img/*`). 조건을 저장하려면 확인 표시 아이콘을 선택합니다.

      각 문자열의 최대 크기는 128자입니다. 이 비교는 대/소문자를 구분합니다. * 및 ? 와일드카드 문자가 지원됩니다.

   c. HTTP 헤더 조건을 추가하려면 Add condition(조건 추가), Http header(Http 헤더)를 선택합니다. 헤더의 이름을 입력하고 비교 문자열을 하나 이상 추가합니다. 조건을 저장하려면 확인 표시 아이콘을 선택합니다.

      각 헤더 이름의 최대 크기는 40자이고, 헤더 이름은 대/소문자를 구분하지 않으며, 와일드카드는 지원되지 않습니다. 각 비교 문자열의 최대 크기는 128자이고, * 및 ? 와일드카드 문자가 지원됩니다. 이 비교는 대/소문자를 구분하지 않습니다.

d. HTTP 요청 메서드 조건을 추가하려면 Add condition(조건 추가), Http request method(Http 요청 메서드)를 선택하고 메서드 이름을 하나 이상 추가합니다. 조건을 저장하려면 확인 표시 아이콘을 선택합니다.

   각 이름의 최대 크기는 40자입니다. 허용되는 문자는 A-Z, 하이픈(-), 밑줄(_)입니다. 이 비교는 대/소문자를 구분합니다. 와일드카드는 지원되지 않습니다.

e. 쿼리 문자열 조건을 추가하려면 Add condition(조건 추가), Query string(쿼리 문자열)을 선택하고 키/값 페어를 하나 이상 추가합니다. 각 키/값 페어에 대해 키를 생략하고 값만 지정할 수 있습니다. 조건을 저장하려면 확인 표시 아이콘을 선택합니다.

   각 문자열의 최대 크기는 128자입니다. 이 비교는 대/소문자를 구분하지 않습니다. * 및 ? 와일드카드 문자가 지원됩니다.

f. 소스 IP 조건을 추가하려면 Add condition(조건 추가), Source IP(소스 IP)를 선택하고 CIDR 블록을 하나 이상 추가합니다. 조건을 저장하려면 확인 표시 아이콘을 선택합니다.

   IPv4 및 IPv6 주소를 모두 사용할 수 있습니다. 와일드카드는 지원되지 않습니다.

8. (선택 사항, HTTPS 리스너) 사용자를 인증하려면 작업 추가, 인증을 선택하고 요청된 정보를 제공합니다. 작업을 저장하려면 확인 표시 아이콘을 선택합니다. 자세한 내용은 Application Load Balancer를 사용하여 사용자 인증 (p. 43) 단원을 참조하십시오.
9. 다음 작업 중 하나를 추가합니다.
   - 전달 작업을 추가하려면 Add action(작업 추가), Forward to(전달)를 선택하고 대상 그룹을 선택합니다. 작업을 저장하려면 확인 표시 아이콘을 선택합니다.
   - 리디렉션 작업을 추가하려면 작업 추가, 다음으로 리디렉션을 선택하고 리디렉션 URL을 입력합니다. 작업을 저장하려면 확인 표시 아이콘을 선택합니다. 자세한 내용은 리디렉션 작업 (p. 26) 단원을 참조하십시오.
   - 고정 응답 작업을 추가하려면 작업 추가, 고정 응답 반환을 선택하고 응답 코드와 응답 본문(선택 사항)을 입력합니다. 작업을 저장하려면 확인 표시 아이콘을 선택합니다. 자세한 내용은 고정 응답 작업 (p. 26) 단원을 참조하십시오.

10. Save를 선택합니다.
11. (선택 사항) 규칙 순서를 변경하려면 화살표를 사용한 후 Save(저장)를 선택합니다. 기본 규칙은 항상 last(마지막) 우선 순위입니다.
12. 이 화면에서 나가려면 메뉴 모음에서 [Back to the load balancer] 아이콘(뒤로 버튼)을 선택합니다.

AWS CLI을 사용하여 규칙을 추가하려면

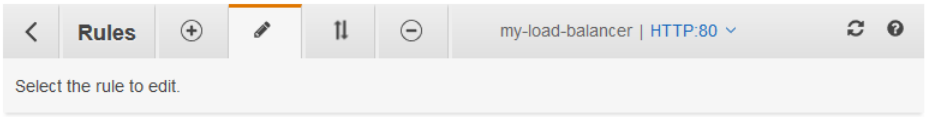
규칙을 생성하려면 create-rule 명령을 사용하십시오. 규칙에 대한 정보를 보려면 describe-rules 명령을 사용하십시오.

규칙 편집

규칙에 대한 작업 및 조건은 언제든지 편집이 가능합니다.

콘솔을 사용하여 규칙을 편집하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택한 다음 [Listeners]를 선택합니다.
4. 리스너를 업데이트하려면 [View/edit rules]를 선택합니다.
5. 메뉴 모음에서 [Edit rules] 아이콘(연필)을 선택합니다.



6. 규칙을 편집하려면 [Edit rules] 아이콘(연필)을 선택합니다.
7. (선택 사항) 필요에 따라 조건 및 작업을 수정합니다. 예를 들어 조건 또는 작업을 편집(연필 아이콘)하거나, 조건을 추가하거나, HTTPS 리스너의 규칙에 인증 작업을 추가하거나, 조건 또는 작업을 삭제(휴지통 아이콘)할 수 있습니다. 기본 규칙에는 조건을 추가할 수 없습니다.

8. [Update]를 선택합니다.
9. 이 화면에서 나가려면 메뉴 모음에서 [Back to the load balancer] 아이콘(뒤로 버튼)을 선택합니다.

AWS CLI을 사용하여 규칙을 편집하려면

modify-rule 명령을 사용하십시오.

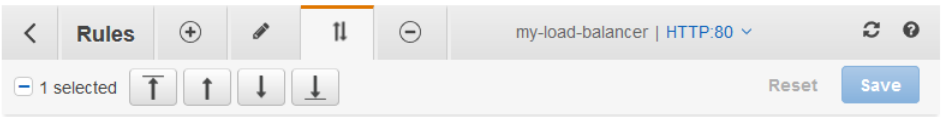
규칙 재정렬

규칙은 가장 낮은 값에서 가장 높은 값에 이르기까지 우선 순위에 따라 평가됩니다. 기본 규칙은 마지막에 평가됩니다. 기본이 아닌 규칙의 우선 순위는 언제든지 변경이 가능합니다. 기본 규칙의 우선 순위는 변경할 수 없습니다.

> **Note**
>
> 콘솔에는 규칙 우선 순위가 아니라 각 규칙에 대한 상대적 순차 번호가 표시됩니다. 콘솔을 사용해 규칙을 재정렬할 때 기존의 규칙 우선 순위에 따라 새로운 규칙 우선 순위가 매겨집니다. 규칙의 우선 순위를 특정 값으로 설정하려면 AWS CLI 또는 Elastic Load Balancing API를 사용하십시오.

콘솔을 사용하여 규칙을 재정렬하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택한 다음 [Listeners]를 선택합니다.
4. 리스너를 업데이트하려면 [View/edit rules]를 선택합니다.
5. 메뉴 모음에서 [Reorder rules] 아이콘(화살표)을 선택합니다.



6. 규칙 옆에 있는 확인란을 선택한 다음, 화살표를 이용해 규칙에 새로운 우선 순위를 부여합니다. 기본 규칙은 항상 마지막 우선 순위입니다.
7. 규칙 재정렬을 마쳤으면 [Save]를 선택합니다.
8. 이 화면에서 나가려면 메뉴 모음에서 [Back to the load balancer] 아이콘(뒤로 버튼)을 선택합니다.

AWS CLI를 사용하여 규칙 우선 순위를 업데이트하려면

set-rule-priorities 명령을 사용하십시오.

규칙 삭제

리스너에 대한 기본이 아닌 규칙은 언제든지 삭제할 수 있습니다. 리스너에 대한 기본 규칙은 삭제할 수 없습니다. 리스너를 삭제하면 모든 리스너 규칙이 삭제됩니다.

콘솔을 사용하여 규칙을 삭제하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택한 다음 [Listeners]를 선택합니다.
4. 리스너를 업데이트하려면 [View/edit rules]를 선택합니다.
5. 메뉴 모음에서 Delete rules(규칙 삭제) 아이콘(마이너스 부호)을 선택합니다.
6. 규칙의 확인란을 선택하고 Delete(삭제)를 선택합니다. 리스너에 대한 기본 규칙은 삭제할 수 없습니다.
7. 이 화면에서 나가려면 메뉴 모음에서 [Back to the load balancer] 아이콘(뒤로 버튼)을 선택합니다.

AWS CLI를 사용하여 규칙을 삭제하려면

delete-rule 명령을 사용하십시오.

Application Load Balancer용 HTTPS 리스너 업데이트

HTTPS 리스너를 생성한 후 기본 인증서를 교체하거나, 인증서 목록을 업데이트하거나, 보안 정책을 교체할 수 있습니다.

제한

ACM은 4096 키 길이의 RSA 및 EC 인증서를 지원합니다. 하지만, ACM.과의 통합을 통해 로드 밸런서에 이러한 인증서를 설치할 수는 없습니다. 로드 밸런서와 함께 사용하기 위해서는 이러한 인증서를 IAM에 업로드해야 합니다.

작업

- 기본 인증서 교체 (p. 41)
- 인증서 목록에 인증서 추가 (p. 42)
- 인증서 목록에서 인증서 제거 (p. 43)
- 보안 정책 업데이트 (p. 43)

기본 인증서 교체

다음 절차에 따라 리스너의 기본 인증서를 교체할 수 있습니다. 자세한 내용은 SSL 인증서 (p. 33) 단원을 참조하십시오.

콘솔을 사용하여 기본 인증서를 변경하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택한 다음 [Listeners]를 선택합니다.
4. 리스너의 확인란을 선택하고 Edit(편집)를 선택합니다.
5. Default SSL certificate(기본 SSL 인증서)에 대해 다음 중 하나를 수행합니다.
   - AWS Certificate Manager을 사용하여 인증서를 생성하거나 가져온 경우 ACM에서 시작을 선택하고 인증서를 선택하십시오.
   - IAM을 사용하여 인증서를 업로드한 경우 IAM에서 시작을 선택하고 인증서를 선택하십시오.
6. Save를 선택합니다.

AWS CLI를 사용하여 기본 인증서를 변경하려면

modify-listener 명령을 사용하십시오.

인증서 목록에 인증서 추가

다음 절차에 따라 리스너 인증서 목록에 인증서를 추가할 수 있습니다. HTTPS 리스너를 처음 생성할 때 인증서 목록은 비어 있습니다. 인증서를 하나 이상 추가할 수 있습니다. 필요에 따라 기본 인증서를 추가하여 이 인증서가 기본 인증서로 교체되더라도 SNI 프로토콜에 이 인증서가 사용되도록 할 수 있습니다. 자세한 내용은 SSL 인증서 (p. 33) 단원을 참조하십시오.

콘솔을 사용하여 인증서 목록에 인증서를 추가하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱 아래에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택한 다음 [Listeners]를 선택합니다.
4. 업데이트할 HTTPS 리스너에 대해 View/edit certificates(인증서 보기/편집)를 선택합니다. 그러면 기본 인증서가 표시되고 리스너에 추가한 다른 인증서가 그 다음에 표시됩니다.
5. 메뉴 모음에서Add certificates(인증서 추가) 아이콘(플러스 기호)을 선택하십시오. 그러면 기본 인증서가 나타나고 ACM 및 IAM에서 관리하는 기타 인증서가 그 뒤에 표시됩니다. 리스너에 인증서를 이미 추가한 경우 해당 확인란이 선택되고 비활성화됩니다.
6. ACM 또는 IAM에서 이미 관리하는 인증서를 추가하려면 인증서의 확인란을 선택하고 추가를 선택하십시오.
7. ACM 또는 IAM에서 관리하지 않는 인증서가 있으면 다음과 같이 ACM로 가져온 후 리스너에 추가하십시오.
   a. [Import certificate]를 선택합니다.
   b. [Certificate private key]에 인증서의 암호화되지 않은 PEM 인코딩 형식 프라이빗 키를 붙여 넣습니다.
   c. [Certificate body]에 PEM 인코딩 형식의 인증서를 붙여 넣습니다.
   d. (선택 사항) [Certificate chain]에 PEM 인코딩된 인증서 체인을 붙여넣습니다.
   e. [Import]를 선택합니다. 새로 가져온 인증서가 사용 가능 인증서 목록에 나타나고 선택됩니다.
   f. [추가]를 선택합니다.
8. 이 화면에서 나가려면 메뉴 모음에서 [Back to the load balancer] 아이콘(뒤로 버튼)을 선택합니다.

AWS CLI를 사용하여 인증서 목록에 인증서를 추가하려면

add-listener-certificates 명령을 사용합니다.

인증서 목록에서 인증서 제거

다음 절차에 따라 HTTPS 리스너의 인증서 목록에서 인증서를 제거할 수 있습니다. HTTPS 리스너의 기본 인증서를 제거하려면 기본 인증서 교체 (p. 41) 단원을 참조하십시오.

콘솔을 사용하여 인증서 목록에서 인증서를 제거하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱 아래에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택한 다음 [Listeners]를 선택합니다.
4. 업데이트할 리스너에 대해 View/edit certificates(인증서 보기/편집)를 선택합니다. 그러면 기본 인증서가 표시되고 리스너에 추가한 다른 인증서가 그 다음에 표시됩니다.
5. 메뉴 모음에서 [Remove certificates] 아이콘(마이너스 기호)을 선택합니다.
6. 인증서의 확인란을 선택하고 Remove(제거)를 선택합니다.
7. 이 화면에서 나가려면 메뉴 모음에서 [Back to the load balancer] 아이콘(뒤로 버튼)을 선택합니다.

AWS CLI를 사용하여 인증서 목록에서 인증서를 제거하려면

remove-listener-certificates 명령을 사용합니다.

보안 정책 업데이트

HTTPS 리스너를 생성할 때 요구를 충족하는 보안 정책을 선택할 수 있습니다. 새로운 보안 정책이 추가되면 새로운 보안 정책을 사용하도록 HTTPS 리스너를 업데이트할 수 있습니다. Application Load Balancer는 사용자 지정 보안 정책을 지원하지 않습니다. 자세한 내용은 보안 정책 (p. 35) 섹션을 참조하십시오.

콘솔을 사용하여 보안 정책을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택한 다음 [Listeners]를 선택합니다.
4. HTTPS 리스너의 확인란을 선택하고 Edit(편집)를 선택합니다.
5. Security policy(보안 정책)에서 보안 정책을 선택합니다.
6. [Update]를 선택합니다.

AWS CLI를 사용하여 보안 정책을 업데이트하려면

modify-listener 명령을 사용하십시오.

Application Load Balancer를 사용하여 사용자 인증

애플리케이션에 액세스하는 사용자를 안전하게 인증하도록 Application Load Balancer를 구성할 수 있습니다. 이렇게 하면 애플리케이션이 비즈니스 로직에 집중할 수 있도록 사용자 인증 작업을 로드 밸런서로 오프로드할 수 있습니다.

지원되는 사용 사례는 다음과 같습니다.

- OpenID Connect(OIDC) 호환 자격 증명 공급자(IdP)를 통해 사용자를 인증합니다.
- Amazon Cognito에서 지원되는 사용자 풀을 통해 Amazon, Facebook 또는 Google과 같은 잘 알려진 소셜 IdP를 통해 사용자를 인증합니다.

- Amazon Cognito에서 지원되는 사용자 풀을 통해 SAML, LDAP 또는 Microsoft AD를 사용하는 기업 자격 증명을 통해 사용자를 인증합니다.

OIDC 호환 IdP 사용 준비

Application Load Balancer에서 OIDC 호환 IdP를 사용하는 경우 다음을 수행합니다.

- IdP에서 새 OIDC 앱을 생성합니다. 클라이언트 ID와 클라이언트 암호를 구성해야 합니다.
- IdP가 게시하는 권한 부여, 토큰 및 사용자 정보와 같은 엔드포인트를 가져옵니다. 잘 알려진 구성에서 이 정보를 찾을 수 있습니다.
- IdP 앱에서 리디렉션 URL 중 하나를 화이트리스트에 추가합니다. 그러면 해당 URL을 사용자가 사용하게 됩니다. 여기서 DNS는 로드 밸런서의 도메인 이름이고 CNAME은 애플리케이션의 DNS 별칭입니다.
  - https://*DNS*/oauth2/idpresponse
  - https://*CNAME*/oauth2/idpresponse

Amazon Cognito 사용 준비

Application Load Balancer에서 Amazon Cognito 사용자 풀을 사용하는 경우 다음을 수행하십시오.

- 사용자 풀을 생성합니다. 자세한 내용은 Amazon Cognito 개발자 안내서에서 Amazon Cognito 사용자 풀을 참조하십시오.
- 사용자 풀 클라이언트를 생성합니다. 클라이언트가 클라이언트 암호를 생성하고, 코드 부여 흐름을 사용하며, 로드 밸런서가 사용하는 것과 동일한 OAuth 범위를 지원하도록 클라이언트를 구성해야 합니다. 자세한 내용은 Amazon Cognito 개발자 안내서에서 사용자 풀 앱 클라이언트 구성을 참조하십시오.
- 사용자 풀 도메인을 생성합니다. 자세한 내용은 Amazon Cognito 개발자 안내서에서 사용자 풀의 도메인 이름 추가를 참조하십시오.
- 요청된 범위가 ID 토큰을 반환하는지 확인하십시오. 예를 들어, 기본값 범위 `openid`는 ID 토큰을 반환하지만 `aws.cognito.signin.user.admin` 범위는 그렇지 않습니다.
- 소셜 또는 기업 IdP와 연동하려면 연동 섹션에서 IdP를 활성화합니다. 자세한 내용은 Amazon Cognito 개발자 안내서에서 사용자 풀에 소셜 로그인 추가 또는 사용자 풀에 SAML IdP 로그인 추가를 참조하십시오.
- Amazon Cognito의 콜백 URL 필드에서 다음 리디렉션 URL을 화이트리스트에 추가합니다. 여기서 DNS는 로드 밸런서의 도메인 이름이고 CNAME은 애플리케이션의 DNS 별칭입니다(사용하는 경우).
  - https://*DNS*/oauth2/idpresponse
  - https://*CNAME*/oauth2/idpresponse
- IdP 앱의 콜백 URL에 있는 사용자 풀 도메인을 화이트리스트에 추가합니다. IdP의 형식을 사용합니다. 예:
  - https://*domain-prefix*.auth.*region*.amazoncognito.com/saml2/idpresponse
  - https://*user-pool-domain*/oauth2/idpresponse

IAM 사용자가 Amazon Cognito를 사용하여 사용자를 인증하도록 로드 밸런서를 구성할 수 있도록 하려면 `cognito-idp:DescribeUserPoolClient` 작업을 호출할 수 있는 사용자 권한을 부여해야 합니다.

Amazon CloudFront 사용 준비

Application Load Balancer 앞에서 CloudFront 배포를 사용하는 경우 다음 설정을 활성화하십시오.

- 전달 요청 헤더(모두) - CloudFront가 인증된 요청에 대한 응답을 캐시하지 않도록 합니다. 이렇게 하면 인증 세션이 만료된 후 응답이 캐시에서 제공되지 않습니다. 또는 캐싱이 활성화되어 있는 동안 이 위험을 줄이기 위해 CloudFront 배포의 소유자는 인증 쿠키가 만료되기 전에 TTL(Time to Live) 값이 만료되도록 설정할 수 있습니다.

- 쿼리 문자열 전달 및 캐싱(모두) - 로드 밸런서가 IdP로 사용자를 인증하는 데 필요한 쿼리 문자열 파라미터에 액세스할 수 있도록 합니다.
- 쿠키 전달(모두) - CloudFront가 모든 인증 쿠키를 로드 밸런서로 전달하도록 합니다.

사용자 인증 구성

하나 이상의 리스너 규칙에 대한 인증 작업을 생성하여 사용자 인증을 구성합니다. `authenticate-cognito` 및 `authenticate-oidc` 작업 유형은 HTTPS 리스너에서만 지원됩니다. 해당 필드에 대한 설명은 Elastic Load Balancing API 참조 버전 2015-12-01에서 AuthenticateCognitoActionConfig 및 AuthenticateOidcActionConfig를 참조하십시오.

기본적으로 `SessionTimeout` 필드는 7일로 설정됩니다. 더 짧은 세션이 필요한 경우 세션 제한 시간을 1초까지 짧게 구성할 수 있습니다. 자세한 내용은 인증 로그아웃 및 세션 제한 시간 (p. 48) 섹션을 참조하십시오.

애플리케이션에 적절하게 `OnUnauthenticatedRequest` 필드를 구성합니다. 예:

- 사용자가 소셜 또는 기업 자격 증명을 사용하여 로그인해야 하는 애플리케이션—이 작업은 기본값 옵션에서 지원됩니다 `authenticate`. 사용자가 로그인하지 않은 경우 로드 밸런서는 요청을 IdP 권한 부여 엔드포인트로 리디렉션하고 IdP는 사용자에게 사용자 인터페이스를 사용하여 로그인하라고 알립니다.
- 로그인한 사용자에게 개인 설정된 보기를 제공하거나 로그인하지 않은 사용자에게 일반 보기를 제공하는 애플리케이션—이 유형의 애플리케이션을 지원하려면 `allow` 옵션을 사용하십시오. 사용자가 로그인한 경우 로드 밸런서는 사용자 클레임을 제공하며 애플리케이션은 개인 설정된 보기를 제공할 수 있습니다. 사용자가 로그인하지 않은 경우 로드 밸런서는 사용자 클레임 없이 요청을 전달하며 애플리케이션은 일반 보기를 제공할 수 있습니다.
- 몇 초마다 로드되는 JavaScript가 있는 단일 페이지 애플리케이션—기본적으로 인증 세션 쿠키가 만료된 후 AJAX 호출이 IdP로 리디렉션되고 차단됩니다. `deny` 옵션을 사용하는 경우 로드 밸런서는 이러한 AJAX 호출에 HTTP 401 권한 없음 오류를 반환합니다.

로드 밸런서는 IdP 토큰 엔드포인트(`TokenEndpoint`) 및 IdP 사용자 정보 엔드포인트(`UserInfoEndpoint`)와 통신할 수 있어야 합니다. 로드 밸런서의 보안 그룹과 VPC의 네트워크 ACL이 이러한 엔드포인트에 대한 발신 액세스를 허용하는지 확인합니다. VPC에서 인터넷에 액세스할 수 있는지 확인합니다. 내부 로드 밸런서가 있는 경우, NAT 게이트웨이를 사용하여 로드 밸런서가 이러한 엔드포인트를 액세스할 수 있도록 설정합니다.

다음 create-rule 명령을 사용하여 사용자 인증을 구성합니다.

```
aws elbv2 create-rule --listener-arn listener-arn --priority 10 \
--conditions Field=path-pattern,Values="/login" --actions file://actions.json
```

다음은 `authenticate-oidc` 작업 및 `forward` 작업을 지정하는 `actions.json` 파일의 예입니다.

```
[{
    "Type": "authenticate-oidc",
    "AuthenticateOidcConfig": {
        "Issuer": "https://idp-issuer.com",
        "AuthorizationEndpoint": "https://authorization-endpoint.com",
        "TokenEndpoint": "https://token-endpoint.com",
        "UserInfoEndpoint": "https://user-info-endpoint.com",
        "ClientId": "abcdefghijklmnopqrstuvwxyz123456789",
        "ClientSecret": "12345678901234567890123456789O",
        "SessionCookieName": "my-cookie",
        "SessionTimeout": 3600,
        "Scope": "email",
        "AuthenticationRequestExtraParams": {
            "display": "page",
```

```
            "prompt": "login"
        },
        "OnUnauthenticatedRequest": "deny"
    },
    "Order": 1
},
{
    "Type": "forward",
    "TargetGroupArn": "arn:aws:elasticloadbalancing:region-code:account-
id:targetgroup/target-group-name/target-group-id",
    "Order": 2
}]
```

다음은 `authenticate-cognito` 작업 및 `forward` 작업을 지정하는 `actions.json` 파일의 예입니다.

```
[{
    "Type": "authenticate-cognito",
    "AuthenticateCognitoConfig": {
        "UserPoolArn": "arn:aws:cognito-idp:region-code:account-id:userpool/user-pool-id",
        "UserPoolClientId": "abcdefghijklmnopqrstuvwxyz123456789",
        "UserPoolDomain": "userPoolDomain1",
        "SessionCookieName": "my-cookie",
        "SessionTimeout": 3600,
        "Scope": "email",
        "AuthenticationRequestExtraParams": {
            "display": "page",
            "prompt": "login"
        },
        "OnUnauthenticatedRequest": "deny"
    },
    "Order": 1
},
{
    "Type": "forward",
    "TargetGroupArn": "arn:aws:elasticloadbalancing:region-code:account-
id:targetgroup/target-group-name/target-group-id",
    "Order": 2
}]
```

자세한 내용은 리스너 규칙 (p. 24) 단원을 참조하십시오.

인증 흐름

Elastic Load Balancing은 다음 단계가 포함된 OIDC 권한 부여 코드 흐름을 사용합니다.

1. 인증 작업이 포함된 규칙에 대한 조건이 충족되면 로드 밸런서는 요청 헤더에서 인증 세션 쿠키를 확인합니다. 쿠키가 없는 경우 로드 밸런서는 IdP가 사용자를 인증할 수 있도록 사용자를 IdP 권한 부여 엔드포인트로 리디렉션합니다.
2. 사용자가 인증된 후 IdP는 권한 부여 코드를 사용하여 사용자를 로드 밸런서로 다시 리디렉션합니다. 로드 밸런서는 코드를 IdP 토큰 엔드포인트에 제공하여 ID 토큰과 액세스 토큰을 가져옵니다.
3. 로드 밸런서는 ID 토큰을 확인한 후 액세스 토큰을 IdP 사용자 정보 엔드포인트와 교환하여 사용자 클레임을 가져옵니다.
4. 로드 밸런서는 클라이언트의 사용자 에이전트가 요청을 수행할 때 쿠키를 로드 밸런서에 전송할 수 있도록 인증 세션 쿠키를 생성하여 클라이언트에 전송합니다. 대부분의 브라우저는 쿠키 크기를 4K로 제한하기 때문에 로드 밸런서는 크기가 4K를 초과하는 쿠키를 여러 쿠키로 샤드합니다. IdP에서 수신한 사용자 클레임과 액세스 토큰의 총 크기가 11K 바이트를 초과하면 로드 밸런서는 클라이언트에게 HTTP 500 오류를 반환하고 `ELBAuthUserClaimsSizeExceeded` 지표를 증가시킵니다.
5. 로드 밸런서는 HTTP 헤더에서 사용자 클레임을 대상에 전송합니다. 자세한 내용은 사용자 클레임 인코딩 및 서명 확인 (p. 47) 단원을 참조하십시오.

6. IdP가 ID 토큰에서 유효한 새로 고침 토큰을 제공하는 경우 로드 밸런서는 새로 고침 토큰을 저장하고 세션 제한 시간이 초과되거나 IdP 새로 고침이 실패할 때까지 액세스 토큰이 만료될 때마다 이 토큰을 사용하여 사용자 클레임을 새로 고칩니다. 사용자가 로그아웃하면 새로 고침이 실패하고 로드 밸런서는 사용자를 IdP 권한 부여 엔드포인트로 리디렉션합니다. 이렇게 하면 사용자가 로그아웃한 후 로드 밸런서가 세션을 중단할 수 있습니다. 자세한 내용은 인증 로그아웃 및 세션 제한 시간 (p. 48) 단원을 참조하십시오.

사용자 클레임 인코딩 및 서명 확인

로드 밸런서는 사용자를 성공적으로 인증한 후 IdP에서 수신된 사용자 클레임을 대상에 전송합니다. 로드 밸런서는 애플리케이션이 서명을 확인하고 클레임이 로드 밸런서에서 전송되었음을 확인할 수 있도록 사용자 클레임에 서명합니다.

로드 밸런서는 다음 HTTP 헤더를 추가합니다.

`x-amzn-oidc-accesstoken`

토큰 엔드포인트의 액세스 토큰, 일반 텍스트.

`x-amzn-oidc-identity`

사용자 정보 엔드포인트의 제목 필드(sub), 일반 텍스트.

`x-amzn-oidc-data`

사용자 클레임, JSON 웹 토큰(JWT) 형식.

전체 사용자 클레임이 필요한 애플리케이션은 표준 JWT 라이브러리를 사용하여 JWT 토큰을 확인할 수 있습니다. 이러한 토큰은 JWT 형식을 따르지만 ID 토큰이 아닙니다. JWT 형식에는 base64 URL 방식으로 인코딩된 헤더, 페이로드 및 서명이 포함되고 끝에 패딩 문자가 포함됩니다. JWT 서명은 ECDSA + P-256 + SHA256입니다.

JWT 헤더는 다음 필드가 있는 JSON 객체입니다.

```
{
   "alg": "algorithm",
   "kid": "12345678-1234-1234-1234-123456789012",
   "signer": "arn:aws:elasticloadbalancing:region-code:account-id:loadbalancer/app/load-
balancer-name/load-balancer-id",
   "iss": "url",
   "client": "client-id",
   "exp": "expiration"
}
```

JWT 페이로드는 IdP 사용자 정보 엔드포인트에서 수신된 사용자 클레임이 포함되는 JSON 객체입니다.

```
{
   "sub": "1234567890",
   "name": "name",
   "email": "alias@example.com",
   ...
}
```

로드 밸런서는 사용자 클레임을 암호화하지 않기 때문에 HTTPS를 사용하도록 대상 그룹을 구성하는 것이 좋습니다. HTTP를 사용하도록 대상 그룹을 구성하는 경우 반드시 보안 그룹을 사용하여 트래픽을 로드 밸런서로 리디렉션해야 합니다. 또한 클레임을 기반으로 권한 부여를 수행하기 전에 서명을 확인하는 것이 좋습니다. 퍼블릭 키를 가져오려면 JWT 헤더에서 키 ID를 가져오고 이 정보를 사용하여 다음 리전 엔드포인트에서 퍼블릭 키를 조회합니다.

```
https://public-keys.auth.elb.region.amazonaws.com/key-id
```

AWS GovCloud(미국 서부)의 경우 엔드포인트는 다음과 같습니다.

```
https://s3-us-gov-west-1.amazonaws.com/aws-elb-public-keys-prod-us-gov-west-1/key-id
```

AWS GovCloud(미국 동부)의 경우 엔드포인트는 다음과 같습니다.

```
https://s3-us-gov-east-1.amazonaws.com/aws-elb-public-keys-prod-us-gov-east-1/key-id
```

다음 예에서는 Python 3.x로 퍼블릭 키를 가져오는 방법을 보여줍니다.

```
import jwt
import requests
import base64
import json

# Step 1: Get the key id from JWT headers (the kid field)
encoded_jwt = headers.dict['x-amzn-oidc-data']
jwt_headers = encoded_jwt.split('.')[0]
decoded_jwt_headers = base64.b64decode(jwt_headers)
decoded_jwt_headers = decoded_jwt_headers.decode("utf-8")
decoded_json = json.loads(decoded_jwt_headers)
kid = decoded_json['kid']

# Step 2: Get the public key from regional endpoint
url = 'https://public-keys.auth.elb.' + region + '.amazonaws.com/' + kid
req = requests.get(url)
pub_key = req.text

# Step 3: Get the payload
payload = jwt.decode(encoded_jwt, pub_key, algorithms=['ES256'])
```

다음 예에서는 Python 2.7로 퍼블릭 키를 가져오는 방법을 보여줍니다.

```
import jwt
import requests
import base64
import json

# Step 1: Get the key id from JWT headers (the kid field)
encoded_jwt = headers.dict['x-amzn-oidc-data']
jwt_headers = encoded_jwt.split('.')[0]
decoded_jwt_headers = base64.b64decode(jwt_headers)
decoded_json = json.loads(decoded_jwt_headers)
kid = decoded_json['kid']

# Step 2: Get the public key from regional endpoint
url = 'https://public-keys.auth.elb.' + region + '.amazonaws.com/' + kid
req = requests.get(url)
pub_key = req.text

# Step 3: Get the payload
payload = jwt.decode(encoded_jwt, pub_key, algorithms=['ES256'])
```

인증 로그아웃 및 세션 제한 시간

애플리케이션은 인증된 사용자를 로그아웃해야 하는 경우 인증 세션 쿠키의 만료 시간을 -1로 설정하고 클라이언트를 IdP 로그아웃 엔드포인트(IdP가 지원하는 경우)로 리디렉션해야 합니다. 사용자가 삭제된 쿠키를

재사용할 수 없도록 하려면 액세스 토큰의 만료 시간을 적절히 짧게 구성하는 것이 좋습니다. 클라이언트가 NULL이 아닌 새로 고침 토큰과 함께 만료된 액세스 토큰이 있는 권한 부여 세션 쿠키를 로드 밸런서에 제공하는 경우 로드 밸런서는 IdP에 연락하여 사용자가 여전히 로그인하고 있는지 확인합니다.

새로 고침 토큰과 세션 제한 시간은 다음과 같이 연계됩니다.

- 세션 제한 시간이 액세스 토큰 만료 시간보다 짧은 경우 로드 밸런서는 세션 제한 시간을 준수하고 인증 세션 제한 시간이 초과된 후 사용자가 다시 로그인하도록 합니다.
- 세션 제한 시간이 액세스 토큰 만료 시간보다 길고 IdP가 새로 고침 토큰을 지원하지 않는 경우 로드 밸런서는 제한 시간이 초과될 때까지 인증 세션을 유지한 다음 사용자가 다시 로그인하도록 합니다.
- 세션 제한 시간이 액세스 토큰 만료 시간보다 길고 IdP가 새로 고침 토큰을 지원하는 경우 로드 밸런서는 액세스 토큰이 만료될 때마다 사용자 세션을 새로 고칩니다. 로드 밸런서는 인증 세션 제한 시간이 초과되거나 새로 고침 흐름이 실패한 후에만 사용자가 다시 로그인하도록 합니다.

애플리케이션 로드 밸런서를 위한 리스너 삭제

언제든 리스너를 삭제할 수 있습니다. 로드 밸런서를 삭제하면 모든 해당 리스너도 삭제됩니다.

콘솔을 이용하여 리스너를 삭제하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Load Balancers]를 선택합니다.
3. 로드 밸런서를 선택한 다음 [Listeners]를 선택합니다.
4. HTTPS 리스너의 확인란을 선택하고 Delete(삭제)를 선택합니다.
5. 확인 메시지가 나타나면 예, 삭제를 선택하십시오.

AWS CLI를 사용하여 리스너를 삭제하려면

delete-listener 명령을 사용하십시오.

Application Load Balancer 대상 그룹

각 대상 그룹은 하나 이상의 등록된 대상에 요청을 라우팅하는 데 사용됩니다. 각 리스너 규칙을 생성할 때 대상 그룹 및 조건을 지정합니다. 규칙 조건이 충족되면 해당하는 대상 그룹으로 트래픽이 전달됩니다. 서로 다른 유형의 요청에 대해 서로 다른 대상 그룹을 생성할 수 있습니다. 예를 들어, 일반 요청인 경우 하나의 대상 그룹을 생성하고 애플리케이션에 대한 마이크로 서비스의 요청인 경우 다른 대상 그룹을 생성합니다. 자세한 내용은 Application Load Balancer 구성 요소 (p. 1) 섹션을 참조하십시오.

대상 그룹 기준으로 로드 밸런서에 대한 상태 확인 설정을 정의합니다. 대상 그룹을 만들거나 나중에 변경할 때 재정의하지 않는 이상 각 대상 그룹은 기본 상태 확인 설정을 사용합니다. 리스너에 대한 규칙에 대상 그룹을 지정한 후, 로드 밸런서는 해당 로드 밸런서에 대해 활성화된 가용 영역의 대상 그룹에 등록된 모든 대상의 상태를 지속적으로 모니터링합니다. 로드 밸런서는 정상 상태로 등록된 대상으로 요청을 라우팅합니다.

목차

라우팅 구성

기본적으로 로드 밸런서는 대상 그룹을 생성할 때 지정한 프로토콜과 포트 번호를 사용하여 대상으로 요청을 라우팅합니다. 또는 대상 그룹에 등록할 때 대상으로 트래픽을 라우팅하는 데 사용되는 포트를 재정의할 수 있습니다.

대상 그룹은 다음과 같은 프로토콜 및 포트를 지원합니다.

- 프로토콜: HTTP, HTTPS
- 포트: 1-65535

대상 그룹이 HTTPS 프로토콜로 구성되거나 HTTPS 상태 확인을 사용하는 경우, 대상에 대한 SSL 연결은 `ELBSecurityPolicy2016-08` 정책의 보안 설정을 사용합니다.

대상 유형

대상 그룹을 생성할 때 대상 유형을 지정합니다. 이 값에 따라 이 대상 그룹에 대상을 등록할 때 지정하는 대상의 유형이 결정됩니다. 대상 그룹을 생성한 후에는 대상 유형을 변경할 수 없습니다.

가능한 대상 유형은 다음과 같습니다.

`instance`

대상이 인스턴스 ID에 의해 지정됩니다.

`ip`

대상이 IP 주소입니다.

`lambda`

대상이 Lambda 함수입니다.

대상 유형이 `ip`인 경우, 다음 CIDR 블록 중 하나에서 IP 주소를 지정할 수 있습니다.

- 대상 그룹에 대한 VPC의 서브넷
- 10.0.0.0/8(RFC 1918)
- 100.64.0.0/10(RFC 6598)
- 172.16.0.0/12(RFC 1918)
- 192.168.0.0/16(RFC 1918)

지원되는 이러한 CIDR 블록을 사용하여 ClassicLink 인스턴스, 피어링된 VPC의 인스턴스, IP 주소 및 포트로 주소를 지정할 수 있는 AWS 리소스(예: 데이터베이스), AWS Direct Connect 또는 VPN 연결을 통해 AWS에 연결되는 온프레미스 리소스를 대상 그룹에 등록할 수 있습니다.

Important

공개적으로 라우팅 가능한 IP 주소는 지정할 수 없습니다.

인스턴스 ID를 사용하여 대상을 지정하면 해당 인스턴스의 기본 네트워크 인터페이스에 지정된 기본 프라이빗 IP 주소를 사용하여 트래픽이 인스턴스로 라우팅됩니다. IP 주소를 사용하여 대상을 지정하면 하나 이상의 네트워크 인터페이스에서 프라이빗 IP 주소를 사용하여 트래픽을 인스턴스로 라우팅할 수 있습니다. 그러면 한 인스턴스의 여러 애플리케이션이 동일한 포트를 사용할 수 있습니다. 각 네트워크 인터페이스에는 자체 보안 그룹이 있을 수 있습니다.

대상 그룹의 대상 유형이 `lambda`인 경우 단일 Lambda 함수를 등록할 수 있습니다. 로드 밸런서가 Lambda 함수에 대한 요청을 수신하면 Lambda 함수를 호출합니다. 자세한 내용은 대상으로서 Lambda 함수 (p. 62) 단원을 참조하십시오.

등록된 대상

로드 밸런서는 클라이언트에 대해 단일 접점의 역할을 하며 정상적으로 등록된 대상 간에 수신 트래픽을 자동으로 분산합니다. 하나 이상의 대상 그룹에 각 대상을 등록할 수 있습니다. 서로 다른 포트를 사용하여 각 EC2 인스턴스 또는 IP 주소를 동일한 대상 그룹에 여러 번 등록할 수 있으며, 이를 통해 로드 밸런서는 요청을 마이크로서비스로 라우팅할 수 있습니다.

애플리케이션에 대한 요구가 증가하면 이를 처리하기 위해 하나 이상의 대상 그룹에 추가 대상을 등록할 수 있습니다. 로드 밸런서는 등록 프로세스가 완료되고 대상이 초기 상태 확인을 통과하자마자 새로 등록된 대상에 대한 라우팅 요청을 시작합니다.

애플리케이션에 대한 요구가 감소하거나 대상을 서비스해야 하는 경우에는 대상 그룹에서 대상 등록을 취소할 수 있습니다. 대상을 등록 취소하면 대상 그룹에서 제거되지만 대상에 영향을 미치지는 않습니다. 등록이 취소되는 즉시 로드 밸런서는 대상으로의 요청 라우팅을 중지합니다. 진행 중인 요청이 완료될 때까지 해당 대상은 `draining` 상태를 유지합니다. 요청 수신을 다시 시작할 준비가 되면 대상 그룹에 대상을 다시 등록할 수 있습니다.

인스턴스 ID로 대상을 등록하는 경우 Auto Scaling 그룹에 로드 밸런서를 사용할 수 있습니다. Auto Scaling 그룹에 대상 그룹을 연결하면 Auto Scaling는 대상을 시작할 때 대상 그룹에 해당 대상을 등록합니다. 자세한 내용은 Amazon EC2 Auto Scaling 사용 설명서에서 Auto Scaling 그룹에 로드 밸런서 연결을 참조하십시오.

제한

- 동일한 VPC에서 다른 Application Load Balancer의 IP 주소는 등록할 수 없습니다. 다른 Application Load Balancer가 피어링된 VPC에 있는 경우, 그 IP 주소는 등록할 수 있습니다.

대상 그룹 속성

대상 그룹 유형이 `instance` 또는 `ip`인 경우 다음 대상 그룹 속성이 지원됩니다.

`deregistration_delay.timeout_seconds`

Elastic Load Balancing에서 대상을 등록 취소하기 전에 대기하는 시간. 범위는 0–3600초입니다. 기본값은 300초입니다.

`slow_start.duration_seconds`

로드 밸런서가 대상 그룹에 대해 선형으로 증가하는 트래픽 공유를 새로 등록된 대상에 보내는 시간(초 단위). 범위는 330–900초입니다(15분). 기본값은 0초입니다(비활성화).

`stickiness.enabled`

고정 세션을 활성화할지 여부를 나타냅니다.

`stickiness.lb_cookie.duration_seconds`

쿠키 만료 기간(초). 이 기간이 지난 후 쿠키는 무효로 간주됩니다. 최소값은 1초이고 최대값은 7일(604800초)입니다. 기본값은 1일(86400초)입니다.

`stickiness.type`

고정의 유형. 가능한 값은 `lb_cookie`입니다.

대상 그룹 유형이 `lambda`인 경우 다음 대상 그룹 속성이 지원됩니다.

`lambda.multi_value_headers.enabled`

로드 밸런서와 Lambda 함수 간에 교환되는 요청 및 응답 헤더에 값 또는 문자열의 배열이 포함됩니다. 가능한 값은 `true` 또는 `false`입니다. 기본값은 `false`입니다. 자세한 내용은 다중 값 헤더 (p. 65) 단원을 참조하십시오.

등록 취소 지연

Elastic Load Balancing은 등록 취소 중인 대상으로 요청을 전송하는 것을 중지합니다. 기본적으로 은 등록 취소 프로세스를 완료하기 전에 300초 동안 대기하는데, 이는 대상에 대해 진행 중인 요청을 완료하는 데 도움이 될 수 있습니다. Elastic Load Balancing가 대기하는 시간을 변경하려면 등록 취소 지연 값을 업데이트하십시오.

등록을 취소하는 대상의 초기 상태는 `draining`입니다. 등록 취소 지연이 경과한 후 등록 취소 프로세스가 완료되며 대상 상태는 `unused`입니다. 대상이 그룹의 일부인 경우 종료 및 대체될 수 있습니다.

등록을 취소하는 대상에 진행 중인 요청이 없고 활성 연결이 없는 경우 Elastic Load Balancing은 등록 취소 지연 시간이 경과할 때까지 대기하지 않고 등록 취소 프로세스를 즉시 완료합니다. 하지만 대상 등록 취소가 완료되더라도 대상 상태는 등록 취소 지연이 경과할 때까지 `draining`으로 표시됩니다.

등록 취소 지연이 경과되기 전에 등록을 취소하는 대상이 연결을 종료하면 클라이언트는 500 레벨 오류 응답을 수신합니다.

콘솔을 사용하여 등록 취소 지연 값을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택합니다. 현재 값이 Description(설명) 탭에 Deregistration delay(등록 취소 지연)로 표시됩니다.
4. [Description] 탭에서 [Edit attributes]를 선택합니다.
5. [Edit attributes] 페이지에서 필요에 따라 [Deregistration delay]의 값을 변경한 다음 [Save]를 선택합니다.

AWS CLI를 사용하여 등록 취소 지연 값을 업데이트하려면

`deregistration_delay.timeout_seconds` 속성과 함께 modify-target-group-attributes 명령을 사용합니다.

느린 시작 모드

기본적으로, 대상은 대상 그룹으로 등록되자 마자 전체 요청 공유를 받기 시작하고 초기 상태 확인을 전달합니다. 느린 시작 모드를 사용하면 로드 밸런서가 대상으로 전체 요청 공유를 보내기 전에 대상에 워밍업 시간이 제공됩니다. 대상 그룹을 위해 느린 시작을 활성화한 후에는, 대상이 대상 그룹으로 등록될 때 느린 시작 모드를 시작하고 구성된 느린 시작 기간이 경과하면 느린 시작 모드를 종료합니다. 느린 시작 모드에서는 로드 밸런서가 대상으로 보낼 수 있는 요청의 수를 선형으로 증가시킵니다. 대상이 느린 시작 모드를 종료한 후에는 로드 밸런서가 대상으로 전체 요청 공유를 보낼 수 있습니다.

고려 사항

- 대상 그룹을 위해 느린 시작을 활성화하면, 대상 그룹으로 이미 등록된 대상은 느린 시작 모드를 시작하지 않습니다.
- 비어있는 대상 그룹을 위해 느린 시작을 활성화한 다음 단일 등록 작업을 사용하여 하나 이상의 대상을 등록하면, 이러한 대상들은 느린 시작 모드를 시작하지 않습니다. 느린 시작 모드 상태가 아닌 등록된 대상이 최소한 하나 이상 있는 경우에만 새로 등록된 대상이 느린 시작 모드를 시작합니다.
- 느린 시작 모드에서 대상을 등록 취소하는 경우 대상이 느린 시작 모드를 종료합니다. 동일한 대상을 다시 등록하는 경우 대상이 다시 느린 시작 모드를 시작합니다.
- 느린 시작 모드의 대상이 비정상적인 상태가 되었다가 기간이 경과하기 전에 다시 정상적인 상태가 되는 경우, 대상이 느린 시작 모드로 유지되고 남은 기간이 경과하면 느린 시작 모드를 종료합니다. 느린 시작 모드가 아닌 대상이 비정상에서 정상으로 바뀌는 경우에는 느린 시작 모드를 시작하지 않습니다.

콘솔을 사용하여 느린 시작 지속시간 값을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택합니다. 현재 값이 Description(설명) 탭에 Slow start duration(느린 시작 지속시간)으로 표시됩니다.
4. [Description] 탭에서 [Edit attributes]를 선택합니다.
5. Edit attributes(속성 편집) 페이지에서 필요에 따라 Slow start duration(느린 시작 지속시간)의 값을 변경한 다음 저장을 선택합니다. 느린 시작 모드를 비활성화하려면 지속시간을 0으로 설정합니다.

AWS CLI를 사용하여 느린 시작 지속시간 값을 업데이트하려면

`slow_start.duration_seconds` 속성과 함께 modify-target-group-attributes 명령을 사용합니다.

고정 세션

고정 세션은 대상 그룹의 동일한 대상으로 요청을 라우팅하는 메커니즘입니다. 이는 클라이언트에게 지속적인 경험을 제공하기 위해 상태 정보를 유지하는 서버에 유용합니다. 고정 세션을 사용하려면 클라이언트는 쿠키를 지원해야 합니다.

처음 클라이언트의 요청을 받으면 로드 밸런서는 요청을 대상에 전달하고 클라이언트에 대한 응답에 포함할 쿠키를 생성합니다. 해당 클라이언트의 다음 요청에는 쿠키가 포함됩니다. 고정 세션이 대상 그룹에 대해 활성화되어 있고 요청이 동일한 대상 그룹으로 전송되는 경우, 로드 밸런서는 쿠키를 감지하고 해당 요청을 동일한 대상으로 라우팅합니다.

Application Load Balancer는 로드 밸런서가 생성한 쿠키만 지원합니다. 쿠키의 이름은 AWSALB입니다. 이러한 쿠키의 콘텐츠는 교체 키를 사용하여 암호화됩니다. 로드 밸런서가 생성한 쿠키는 해독하거나 변경할 수 없습니다.

WebSocket 연결은 본질적으로 고정됩니다. 클라이언트가 WebSocket에 대한 연결 업그레이드를 요청하는 경우, HTTP 101 상태 코드를 반환하여 연결 업그레이드를 승인하는 대상은 WebSocket 연결에 사용되는 대상입니다. WebSocket 업그레이드가 완료되면 쿠키 기반 고정성이 사용되지 않습니다.

대상 그룹 레벨에서 고정 세션을 활성화합니다. 또한 로드 밸런서가 생성한 쿠키의 고정 시간(초)을 설정할 수도 있습니다. 지속 시간은 요청이 있을 때마다 설정됩니다. 따라서 각 지속 시간이 만료되기 전에 클라이언트에서 요청을 보내는 경우, 고정 세션은 지속됩니다.

Application Load Balancer는 Max-Age 헤더 대신에 쿠키 헤더의 Expires 속성을 사용합니다.

콘솔을 사용하여 고정 세션을 활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택합니다.
4. [Description] 탭에서 [Edit attributes]를 선택합니다.
5. [Edit attributes] 페이지에서 다음을 수행합니다.
   a. [Enable load balancer generated cookie stickiness]를 선택합니다.
   b. [Stickiness duration]에서 1초에서 7일 사이의 값을 지정합니다.
   c. Save를 선택합니다.

AWS CLI를 사용하여 고정 세션을 활성화하려면

`stickiness.enabled` 및 `stickiness.lb_cookie.duration_seconds` 속성과 함께 modify-target-group-attributes 명령을 사용합니다.

대상 그룹 생성

대상 그룹에 대상을 등록합니다. 기본적으로 로드 밸런서는 대상 그룹에 대해 지정한 프로토콜과 포트 번호를 사용하여 등록된 대상으로 요청을 전송합니다. 또는 대상 그룹에 각 대상을 등록할 때 이 포트를 재정의할 수 있습니다.

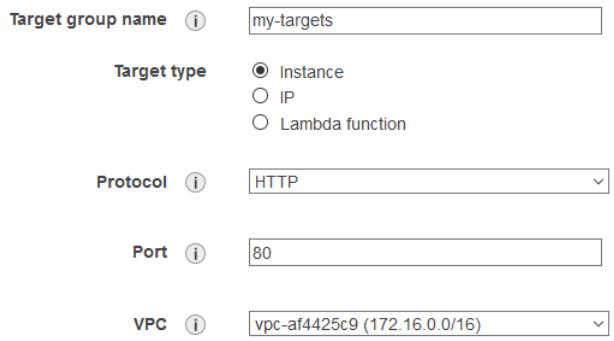
대상 그룹을 만든 후에는 태그를 추가할 수 있습니다.

대상 그룹의 대상으로 트래픽을 라우팅하려면 리스너 또는 리스너에 대한 규칙을 생성할 때 작업에 대상 그룹을 지정합니다. 자세한 내용은 리스너 규칙 (p. 24) 단원을 참조하십시오.

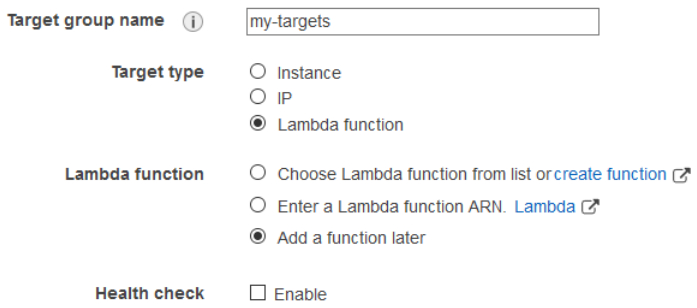
언제든지 대상 그룹에서 대상을 추가하거나 삭제할 수 있습니다. 자세한 내용은 대상 그룹에 대상 등록 (p. 59) 섹션을 참조하십시오. 대상 그룹에 대한 상태 확인 설정을 변경할 수도 있습니다. 자세한 내용은 대상 그룹의 상태 설정 변경 (p. 58) 섹션을 참조하십시오.

콘솔을 사용하여 대상 그룹을 생성하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. [Create target group]을 선택합니다.
4. [Target group name]에 대상 그룹의 이름을 입력합니다.
5. Target type(대상 유형)에서 인스턴스 ID를 기준으로 대상을 등록하려면 Instance(인스턴스), IP 주소를 등록하려면 IP, Lambda 함수를 등록하려면 Lambda function(Lambda 함수)를 선택합니다.
6. 대상 유형이 Instance(인스턴스) 또는 IP인 경우 다음을 수행합니다.
   a. (선택 사항) [Protocol]과 [Port]에서 필요에 따라 기본값을 변경합니다.
   b. [VPC]에서 Virtual Private Cloud(VPC)를 선택합니다.



7. 대상 유형이 Lambda function(Lambda 함수)인 경우 다음을 수행합니다.
   a. Lambda function(Lambda 함수)에 대해 다음 중 하나를 수행합니다.
      - Lambda 함수 선택
      - 새 Lambda 함수를 생성하고 선택
      - 대상 그룹을 생성한 후 Lambda 함수를 등록
   b. (선택 사항) 상태 확인을 활성화하려면 Health check(상태 확인), Enable(활성화)를 선택합니다.



8. (선택 사항) [Health check settings] 및 [Advanced health check settings]에서 필요에 따라 기본 설정을 변경합니다.
9. Create를 선택합니다.
10. (선택 사항) 다음과 같이 하나 이상의 태그를 추가합니다.
    a. 새로 생성한 대상 그룹을 선택합니다.
    b. [Tags] 탭에서 [Add/Edit Tags]를 선택합니다.

c. [Add/Edit Tags] 페이지에서 추가한 각 태그에 대해 [Create Tag]를 클릭한 다음 태그 키와 태그 값을 지정합니다. 태그 추가를 마쳤으면 [Save]를 선택합니다.

11. (선택 사항) 대상을 대상 그룹에 추가하려면 대상 그룹에 대상 등록 (p. 59) 섹션을 참조하십시오.

AWS CLI를 사용하여 대상 그룹을 생성하려면

create-target-group 명령을 사용하여 대상 그룹을 생성하고, add-tags 명령으로 대상 그룹에 태그를 지정하고, register-targets 명령으로 대상을 추가합니다.

대상 그룹에 대한 상태 확인

Application Load Balancer는 등록된 대상으로 요청을 주기적으로 전송하여 상태를 확인합니다. 이러한 테스트를 바로 상태 확인이라고 합니다.

각각의 교차 영역 노드는 로드 밸런서의 활성화된 가용 영역에서 정상 상태 대상에만 요청을 전송합니다. 각각의 로드 밸런서 노드는 대상이 등록된 대상 그룹에 대한 상태 확인 설정을 사용하여 각 대상의 상태를 확인합니다. 대상이 등록된 후에는 상태 확인을 통과해야만 정상 상태로 간주됩니다. 각각의 상태 확인이 완료되고 나면 로드 밸런서 노드는 상태 확인을 위해 설정된 연결을 종료합니다.

대상 그룹에 비정상인 등록된 대상만 포함되는 경우 로드 밸런서 노드는 비정상인 대상 간에 요청을 라우팅합니다.

WebSockets에서는 상태 확인이 지원되지 않습니다.

상태 확인 설정

다음 설정을 사용하여 대상 그룹에서 대상에 대한 상태 확인을 구성합니다. 로드 밸런서는 지정된 포트, 프로토콜 및 ping 경로를 사용하여 HealthCheckIntervalSeconds 초마다 모든 등록 대상에 상태 확인 요청을 전송합니다. 각 상태 확인 요청은 독립적이며 전체 간격 동안 지속됩니다. 대상이 응답하는 데 걸리는 시간은 다음 상태 확인 요청의 간격에 영향을 미치지 않습니다. 상태 확인이 UnhealthyThresholdCount 연속 실패를 초과하면 로드 밸런서는 대상을 서비스에서 제외합니다. 상태 확인이 HealthyThresholdCount 연속 성공을 초과하면 로드 밸런서는 대상을 다시 서비스합니다.

설정	설명
HealthCheckProtocol	대상에 대한 상태 확인을 수행할 때 로드 밸런서가 사용하는 프로토콜입니다. HTTP, HTTPS 등의 프로토콜이 여기에 해당됩니다. HTTP 프로토콜이 기본 설정값입니다.
HealthCheckPort	대상에 대한 상태 확인을 수행할 때 로드 밸런서가 사용하는 포트입니다. 각 대상이 로드 밸런서에서 트래픽을 수신하는 포트를 사용하도록 기본 설정되어 있습니다.
HealthCheckPath	상태 확인을 위한 대상에서 목적지가 되는 ping 경로입니다. 유효한 URI를 지정합니다(/path?query). 기본값은 /입니다.
HealthCheckTimeoutSeconds	상태 확인 실패를 의미하는 대상으로부터 응답이 없는 기간(초 단위)입니다. 범위는 2–120초입니다. 대상 유형이 `instance` 또는 `ip`이면 기본값은 5초이며, 대상 유형이 `lambda`이면 기본값은 30초입니다.
HealthCheckIntervalSeconds	개별 인스턴스의 상태 확인 간의 대략적인 간격(초 단위)입니다. 범위는 5–300초입니다. 대상 유형이

설정	설명
	`instance` 또는 `ip`이면 기본값은 30초이며, 대상 유형이 `lambda`이면 기본값은 35초입니다.
HealthyThresholdCount	비정상 상태의 대상을 정상으로 간주하기까지 필요한 연속적인 상태 확인 성공 횟수입니다. 범위는 2–10입니다. 기본값은 5입니다.
UnhealthyThresholdCount	대상을 비정상 상태로 간주하기까지 필요한 연속적인 상태 확인 실패 횟수입니다. 범위는 2–10입니다. 기본값은 2입니다.
Matcher	대상으로부터 응답 성공을 확인할 때 사용하는 HTTP 코드입니다. 200~499까지 값 또는 값 범위를 지정할 수 있습니다. 기본값은 200입니다.

대상 상태

로드 밸런서가 대상으로 상태 확인 요청을 전송할 수 있으려면 먼저 대상 그룹에 이를 등록하고 리스너 규칙에서 대상 그룹을 지정한 다음, 로드 밸런서에서 대상의 가용 영역을 활성화해야 합니다. 대상이 로드 밸런서에서 요청을 수신할 수 있으려면 먼저 초기 상태 확인을 통과해야 합니다. 대상이 초기 상태 확인을 통과한 후에는 상태가 `Healthy`가 됩니다.

다음 표에는 등록 대상의 상태로 가능한 값이 나와 있습니다.

값	설명
`initial`	로드 밸런서에서는 대상 등록이나 대상에 대해 초기 상태 확인이 진행 중에 있습니다.
`healthy`	대상이 정상 상태입니다.
`unhealthy`	대상이 상태 확인에 응답하지 않았거나 상태 확인에 실패했습니다.
`unused`	대상이 대상 그룹에 등록되어 있지 않거나, 대상 그룹이 로드 밸런서를 위한 리스너 규칙에서 사용되지 않거나, 대상이 로드 밸런서에서 활성화되지 않은 가용 영역에 있습니다.
`draining`	대상이 등록 취소되고 있으며 연결 드레이닝이 진행 중입니다.

상태 확인 사유 코드

대상의 상태가 `Healthy` 이외의 값인 경우에는 API가 문제에 대한 사유 코드와 설명을 반환하고 콘솔이 도구 설명에 동일한 설명을 표시합니다. `Elb`로 시작되는 사유 코드는 로드 밸런서 측에서 호출되고, `Target`으로 시작되는 사유 코드는 대상 측에서 호출됩니다.

사유 코드	설명
`Elb.InitialHealthChecking`	초기 상태 확인이 진행 중
`Elb.InternalError`	내부 오류로 인한 상태 확인 실패
`Elb.RegistrationInProgress`	대상 등록이 진행 중

사유 코드	설명
`Target.DeregistrationInProgress`	대상 등록 취소가 진행 중
`Target.FailedHealthChecks`	상태 확인 실패
`Target.InvalidState`	대상이 중지 상태에 있음 대상이 종료 상태에 있음 대상이 종료 또는 중지 상태에 있음 대상이 잘못된 상태에 있음
`Target.IpUnusable`	IP 주소는 로드 밸런서에서 사용 중이므로 대상으로 사용할 수 없습니다.
`Target.NotInUse`	대상 그룹이 로드 밸런서에서 트래픽을 수신하도록 구성되지 않음 대상이 로드 밸런서에서 활성화되지 않은 가용 영역에 있음
`Target.NotRegistered`	대상이 대상 그룹에 등록되지 않음
`Target.ResponseCodeMismatch`	[code] 등의 코드에서 상태 확인 실패
`Target.Timeout`	요청 시간 초과

대상의 상태 확인

대상 그룹에 등록된 대상의 상태를 확인할 수 있습니다.

콘솔을 사용하여 대상의 상태를 확인하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹을 선택합니다.
4. 대상 탭에서 상태 열은 각 대상의 상태를 나타냅니다.
5. 상태가 `Healthy` 이외의 값인 경우에는 도구 설명에서 자세한 내용을 확인합니다.

AWS CLI를 사용하여 대상의 상태를 확인하려면

describe-target-health 명령을 사용하십시오. 이 명령의 출력 화면에는 대상 상태 설명이 포함됩니다. 상태가 `Healthy` 이외의 값인 경우에는 출력 화면에도 사유 코드가 포함됩니다.

대상 그룹의 상태 설정 변경

대상 그룹에 대한 상태 확인 설정을 언제든지 변경할 수도 있습니다.

콘솔을 사용하여 대상 그룹의 상태 확인 설정을 변경하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹을 선택합니다.
4. 상태 확인 탭에서 편집을 선택합니다.

5. 대상 그룹 편집 페이지에서 필요에 따라 설정을 변경한 다음 저장을 선택합니다.

AWS CLI를 사용하여 대상 그룹의 상태 확인 설정을 변경하려면

modify-target-group 명령을 사용하십시오.

대상 그룹에 대상 등록

대상 그룹에 대상을 등록합니다. 대상 그룹을 생성할 때 대상 유형을 지정하며, 이 값에 따라 대상을 등록하는 방법이 결정됩니다. 예를 들어 인스턴스 ID, IP 주소 또는 Lambda 함수를 등록할 수 있습니다. 자세한 내용은 Application Load Balancer 대상 그룹 (p. 50) 단원을 참조하십시오.

최근 등록된 대상에 대한 요구가 증가하면 이를 처리하기 위해 하나 이상의 대상 그룹에 추가 대상을 등록할 수 있습니다. 대상이 요청을 처리할 준비가 되면 대상 그룹에 등록합니다. 로드 밸런서는 등록 프로세스가 완료되고 대상이 초기 상태 확인을 통과하자마자 해당 대상에 대한 라우팅 요청을 시작합니다.

등록된 대상에 대한 요구가 감소하거나 대상을 서비스해야 하는 경우에는 대상 그룹에서 등록을 취소할 수 있습니다. 등록이 취소되는 즉시 로드 밸런서는 대상으로의 요청 라우팅을 중지합니다. 대상이 요청 수신을 시작할 준비가 되면 대상 그룹에 다시 등록할 수 있습니다.

대상이 등록 취소되면 로드 밸런서는 진행 중인 요청이 완료될 때까지 대기합니다. 이를 연결 드레이닝이라고 합니다. 연결 드레이닝이 진행 중인 동안 대상의 상태는 `draining`입니다.

IP 주소로 등록된 대상을 등록 취소하면 등록 취소 지연이 완료될 때까지 기다려야 동일한 IP 주소를 다시 등록할 수 있습니다.

인스턴스 ID로 대상을 등록하는 경우 Auto Scaling 그룹에 로드 밸런서를 사용할 수 있습니다. Auto Scaling 그룹에 대상 그룹을 연결하고 해당 그룹이 확장되면, Auto Scaling 그룹에서 시작한 인스턴스가 대상 그룹에 자동으로 등록됩니다. 그룹에서 대상 그룹을 분리하면 인스턴스가 대상 그룹에서 자동으로 등록 취소됩니다. 자세한 내용은 Amazon EC2 Auto Scaling 사용 설명서의 Auto Scaling 그룹에 로드 밸런서 연결을 참조하십시오.

대상 보안 그룹

EC2 인스턴스를 대상으로 등록할 때 인스턴스에 대한 보안 그룹은 로드 밸런서가 리스너 포트 및 상태 확인 포트에서 인스턴스와 통신할 수 있도록 허용해야 합니다.

권장 규칙

Inbound		
Source	Port Range	Comment
## ### ## ##	*#### ###*	인스턴스 리스너 포트의 로드 밸런서에서 트래픽을 허용합니다
## ### ## ##	*## ##*	상태 확인 포트의 로드 밸런서에서 트래픽을 허용합니다

인바운드 ICMP 트래픽이 경로 MTU 검색을 지원하도록 허용하는 것이 좋습니다. 자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서에서 경로 MTU 검색을 참조하십시오.

대상 등록 또는 등록 취소

대상 그룹의 대상 유형에 따라 해당 대상 그룹에 대상을 등록하는 방법이 결정됩니다. 자세한 내용은 대상 유형 (p. 50) 단원을 참조하십시오.

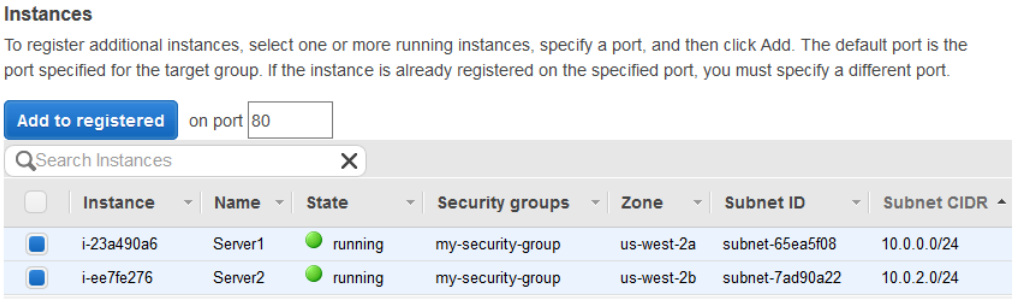
목차

인스턴스 ID로 대상 등록 또는 등록 취소

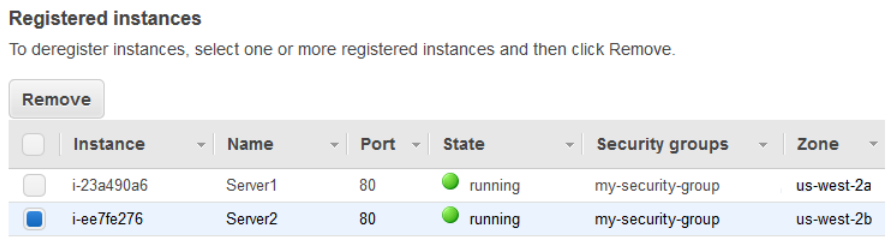
인스턴스는 대상 그룹에 대해 지정한 VPC(Virtual Private Cloud)에 있어야 합니다. 또한 인스턴스를 등록할 때 인스턴스가 `running` 상태여야 합니다.

인스턴스 ID로 대상을 등록하거나 등록 취소하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택합니다.
4. [Targets] 탭에서 [Edit]를 선택합니다.
5. 인스턴스를 등록하려면 [Instances]에서 인스턴스를 선택하고 필요에 따라 기본 인스턴스를 변경한 다음 [Add to registered]를 선택합니다.



6. 인스턴스를 등록 취소하려면 [Registered instances]에서 인스턴스를 선택한 다음 [Remove]를 선택합니다.



7. Save를 선택합니다.

IP 주소로 대상 등록 또는 등록 취소

사용자가 등록하는 IP 주소는 다음 CIDR 블록 중 하나를 출처로 한 주소여야 합니다.

- 대상 그룹에 대한 VPC의 서브넷
- 10.0.0.0/8(RFC 1918)
- 100.64.0.0/10(RFC 6598)
- 172.16.0.0/12(RFC 1918)

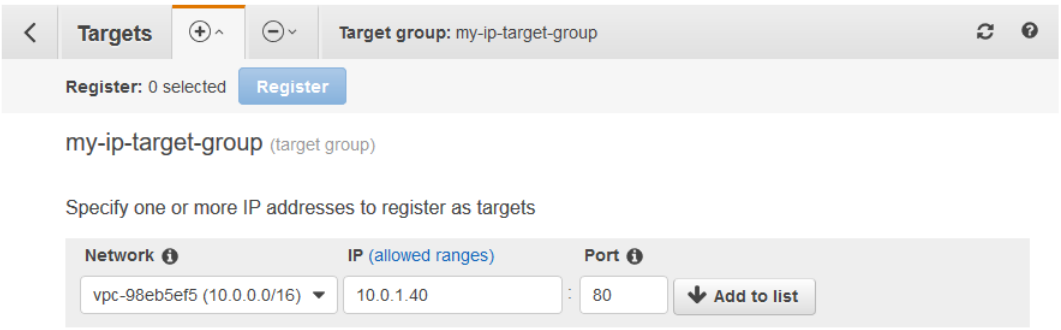
- 192.168.0.0/16(RFC 1918)

제한

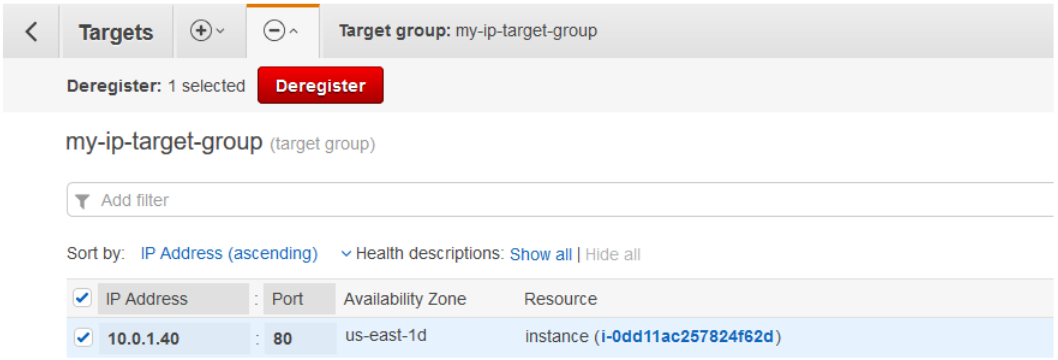
- 동일한 VPC에서 다른 Application Load Balancer의 IP 주소는 등록할 수 없습니다. 다른 Application Load Balancer가 피어링된 VPC에 있는 경우, 그 IP 주소는 등록할 수 있습니다.

IP 주소로 대상을 등록하거나 등록 취소하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택합니다.
4. [Targets] 탭에서 [Edit]를 선택합니다.
5. IP 주소를 등록하려면 메뉴 모음의 [Register targets] 아이콘(더하기 기호)을 선택합니다. 각 IP 주소에 대해 네트워크를 선택하고, IP 주소 및 포트를 입력한 다음 Add to list(목록에 추가)를 선택합니다. 주소 지정을 마치면 [Register]를 선택합니다.



6. IP 주소를 등록 취소하려면 메뉴 모음의 [Deregister targets] 아이콘(마이너스 기호)을 선택합니다. 등록 취소된 IP 주소가 많은 경우 필터를 추가하거나 정렬 순서를 변경하는 것이 유용할 수 있습니다. IP 주소를 선택한 다음 [Deregister]를 선택합니다.



7. 이 화면에서 나가려면 메뉴 모음에서 [Back to target group] 아이콘(뒤로 버튼)을 선택합니다.

Lambda 함수 등록 또는 등록 취소

단일 Lambda 함수를 각 대상 그룹에 등록할 수 있습니다. Elastic Load Balancing는 Lambda 함수를 호출할 권한이 있어야 합니다. 트래픽을 Lambda 함수에 더 이상 전송할 필요가 없는 경우 해당 함수의 등록을 취소할 수 있습니다. Lambda 함수의 등록을 취소한 후에는 처리 중인 요청이 HTTP 5XX 오류와 함께 실패합니

다. Lambda 함수를 바꾸려면 그 대신 새 대상 그룹을 생성하는 것이 더 좋습니다. 자세한 내용은 대상으로서 Lambda 함수 (p. 62) 단원을 참조하십시오.

Lambda 함수를 등록하거나 등록 취소하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택한 다음 Targets(대상) 탭을 선택합니다.
4. 등록된 Lambda 함수가 없는 경우 Register(등록)를 선택합니다. Lambda 함수를 선택한 다음 Register(등록)를 선택합니다.
5. Lambda 함수의 등록을 취소하려면 Deregister(등록 취소)를 선택합니다. 확인 메시지가 나타나면 [Deregister]를 선택합니다.

AWS CLI를 사용하여 대상 등록 또는 등록 취소

register-targets 명령을 사용하여 대상을 추가하고 deregister-targets 명령을 사용하여 대상을 제거합니다.

대상으로서 Lambda 함수

Lambda 함수를 대상으로 등록하고 Lambda 함수에 대한 대상 그룹에 요청을 전달하도록 리스너 규칙을 구성할 수 있습니다. 로드 밸런서가 Lambda 함수를 대상으로 사용하는 대상 그룹에 요청을 전달하면 Lambda 함수를 호출하고 요청 콘텐츠를 Lambda 함수에 JSON 형식으로 전달합니다.

제한

- Lambda 함수와 대상 그룹은 동일한 계정에 있어야 합니다.
- Lambda 함수에 전송할 수 있는 요청 본문의 최대 크기는 1MB입니다. 관련 크기 제한은 HTTP 헤더 제한을 참조하십시오.
- Lambda 함수가 전송할 수 있는 응답 JSON의 최대 크기는 1MB입니다.
- WebSocket은 지원되지 않습니다. 업그레이드 요청은 HTTP 400 코드와 함께 거부됩니다.

목차

Lambda 함수 준비

다음 권장 사항은 Application Load Balancer와 함께 Lambda 함수를 사용하는 경우에 적용됩니다.

Lambda 함수를 호출할 권한

AWS Management 콘솔을 사용하여 대상 그룹을 생성하고 Lambda 함수를 등록하면 콘솔은 사용자를 대신하여 필수 권한을 Lambda 함수 정책에 추가합니다. 그렇지 않으면 AWS CLI를 사용하여 대상 그룹을 생성하고 함수를 등록한 후 add-permission 명령을 사용하여 Lambda 함수를 호출할 Elastic Load Balancing 권한을

부여해야 합니다. `--source-arn` 파라미터를 포함시켜서 함수 호출을 지정된 대상 그룹으로 제한하는 것이 좋습니다.

```
aws lambda add-permission \
--function-name lambda-function-arn-with-alias-name \
--statement-id elb1 \
--principal elasticloadbalancing.amazonaws.com \
--action lambda:InvokeFunction \
--source-arn target-group-arn
```

Lambda 함수 버전 관리

대상 그룹당 하나의 Lambda 함수를 등록할 수 있습니다. Lambda 함수를 변경할 수 있는지 확인하고 로드 밸런서가 항상 현재 버전의 Lambda 함수를 호출하도록 하려면 Lambda 함수를 로드 밸런서에 등록할 때 함수 별칭을 생성하고 별칭을 함수 ARN에 포함시킵니다. 자세한 내용은 AWS Lambda Developer Guide의 AWS Lambda 함수 버전 관리 및 별칭 및 별칭을 사용한 트래픽 이동을 참조하십시오.

함수 제한 시간.

로드 밸런서는 Lambda 함수가 응답하거나 시간 초과될 때까지 대기합니다. 예상 실행 시간을 기반으로 Lambda 함수의 제한 시간을 구성하는 것이 좋습니다. 기본 제한 시간 값과 이 값을 변경하는 방법에 대한 자세한 내용은 기본 AWS Lambda 함수 구성을 참조하십시오. 구성할 수 있는 최대 제한 시간 값에 대한 자세한 내용은 AWS Lambda 제한을 참조하십시오.

Lambda 함수에 대한 대상 그룹 생성

라우팅 요청에서 사용되는 대상 그룹을 만듭니다. 요청 콘텐츠가 해당 콘텐츠를 이 대상 그룹에 전달하는 작업이 포함된 리스너 규칙과 일치하는 경우 로드 밸런서는 등록된 Lambda 함수를 호출합니다.

대상 그룹을 생성하고 Lambda 함수를 등록하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. [Create target group]을 선택합니다.
4. [Target group name]에 대상 그룹의 이름을 입력합니다.
5. Target type(대상 유형)에 Lambda function(Lambda 함수)을 선택합니다.
6. Lambda function(Lambda 함수)에 대해 다음 중 하나를 수행합니다.
   - Lambda 함수 선택
   - 새 Lambda 함수를 생성하고 선택
   - 대상 그룹을 생성한 후 Lambda 함수를 등록
7. (선택 사항) 상태 확인을 활성화하려면 Health check(상태 확인), Enable(활성화)를 선택합니다.
8. Create를 선택합니다.

AWS CLI를 사용하여 대상 그룹을 생성하고 Lambda 함수의 등록을 취소하려면

create-target-group 및 register-targets 명령을 사용합니다.

로드 밸런서에서 이벤트 수신

로드 밸런서는 HTTP 및 HTTPS를 통한 요청에 대한 Lambda 호출을 지원합니다. 로드 밸런서는 JSON 형식으로 이벤트를 전송합니다. 로드 밸런서는 `X-Amzn-Trace-Id`, `X-Forwarded-For`, `X-Forwarded-Port` 및 `X-Forwarded-Proto` 헤더를 모든 요청에 추가합니다.

콘텐츠 유형이 text/*, application/json, application/javascript 및 application/xml 중 하나인 경우 로드 밸런서는 본문을 Lambda 함수에 그대로 전송하고 `isBase64Encoded`를 `false`로 설정합니다. 기타 모든 유형의 경우 로드 밸런서는 본문을 Base64로 인코딩하고 `isBase64Encoded`를 `true`로 설정합니다.

다음은 이벤트 예제입니다.

```
{
    "requestContext": {
        "elb": {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/my-target-
group/6d0ecf831eec9f09"
        }
    },
    "httpMethod": "GET",
    "path": "/",
    "queryStringParameters": {parameters},
    "headers": {
        "accept": "text/html,application/xhtml+xml",
        "accept-language": "en-US,en;q=0.8",
        "content-type": "text/plain",
        "cookie": "cookies",
        "host": "lambda-846800462-us-east-2.elb.amazonaws.com",
        "user-agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6)",
        "x-amzn-trace-id": "Root=1-5bdb40ca-556d8b0c50dc66f0511bf520",
        "x-forwarded-for": "72.21.198.66",
        "x-forwarded-port": "443",
        "x-forwarded-proto": "https"
    },
    "isBase64Encoded": false,
    "body": "request_body"
}
```

로드 밸런서에 응답

Lambda 함수의 응답에는 Base64 인코딩 상태, 상태 코드, 상태 설명 및 헤더가 포함됩니다. 본문을 생략할 수 있습니다. `statusDescription` 헤더에는 단일 공백으로 분리된 상태 코드와 이유 문구가 포함되어야 합니다.

응답의 본문에 바이너리 콘텐츠를 포함시키려면 콘텐츠를 Base64로 인코딩하고 `isBase64Encoded`를 `true`를 설정해야 합니다. 로드 밸런서는 콘텐츠를 디코딩하여 바이너리 콘텐츠를 수신하고 이 콘텐츠를 HTTP 응답의 본문으로 클라이언트에 전송합니다.

로드 밸런서는 `Connection` 또는 `Transfer-Encoding`과 같은 홉별 헤더를 따르지 않습니다. 응답을 클라이언트에 전송하기 전에 로드 밸런서가 컴퓨팅하기 때문에 `Content-Length` 헤더를 생략할 수 있습니다.

다음은 Lambda 함수의 응답 예제입니다.

```
{
    "isBase64Encoded": false,
    "statusCode": 200,
    "statusDescription": "200 OK",
    "headers": {
        "Set-cookie": "cookies",
        "Content-Type": "application/json"
    },
    "body": "Hello from Lambda (optional)"
}
```

Application Load Balancer와 작동하는 Lambda 함수 템플릿에 대해서는 github의 application-load-balancer-serverless-app을 참조하십시오. 또는, Lambda 콘솔을 열고 함수를 생성한 다음, AWS Serverless Application Repository에서 다음 중 하나를 선택합니다.

- ALB-Lambda-Target-HelloWorld

- ALB-Lambda-Target-UploadFiletoS3
- ALB-Lambda-Target-BinaryResponse
- ALB-Lambda-Target-WhatisMyIP

다중 값 헤더

클라이언트의 요청 또는 Lambda 함수의 응답에 다중 값이 있는 헤더가 포함되거나 동일한 헤더가 여러 번 포함되는 경우 다중 값 헤더 구문에 대한 지원을 활성화할 수 있습니다. 다중 값 헤더를 활성화한 후에는 로드 밸런서와 Lambda 함수 간에 교환되는 요청 및 응답 헤더에 배열이 사용됩니다. 그렇지 않으면 로드 밸런서는 수신한 마지막 값을 사용합니다.

목차

다중 값 헤더가 있는 요청

헤더 및 쿼리 문자열 파라미터에 사용되는 필드 이름은 대상 그룹의 다중 값 헤더를 활성화하는지 여부에 따라 달라집니다.

다음 요청 예제에는 동일한 키의 쿼리 파라미터가 두 개 있습니다.

```
http://www.example.com?&myKey=val1&myKey=val2
```

기본 형식을 사용할 경우 로드 밸런서는 클라이언트에서 전송된 마지막 값을 사용하고 `queryStringParameters`를 사용하여 쿼리 문자열 파라미터가 포함된 이벤트를 전송합니다. 예:

```
"queryStringParameters": { "myKey": "val2"},
```

다중 값 헤더를 사용할 경우 로드 밸런서는 클라이언트에서 전송된 두 개의 키 값을 모두 사용하고 `multiValueQueryStringParameters`를 사용하여 쿼리 문자열 파라미터가 포함된 이벤트를 전송합니다. 예:

```
"multiValueQueryStringParameters": { "myKey": ["val1", "val2"] },
```

마찬가지로, 클라이언트가 헤더에 쿠키 두 개가 있는 요청을 전송한다고 가정합니다.

```
"cookie": "name1=value1",
"cookie": "name2=value2",
```

기본 형식을 사용할 경우 로드 밸런서는 클라이언트에서 전송된 마지막 쿠키를 사용하고 `headers`를 사용하여 헤더가 포함된 이벤트를 전송합니다. 예:

```
"headers": {
    "cookie": "name2=value2",
    ...
},
```

다중 값 헤더를 사용할 경우 로드 밸런서는 클라이언트에서 전송된 두 개의 쿠키를 모두 사용하고 `multiValueHeaders`를 사용하여 헤더가 포함된 이벤트를 전송합니다. 예:

```
"multiValueHeaders": {
    "cookie": ["name1=value1", "name2=value2"],
    ...
},
```

다중 값 헤더가 있는 응답

헤더에 사용되는 필드 이름은 대상 그룹의 다중 값 헤더를 활성화하는지 여부에 따라 달라집니다. 다중 값 헤더 및 `headers`를 활성화한 경우 `multiValueHeaders`를 사용해야 합니다.

기본 형식을 사용할 경우 단일 쿠키를 지정할 수 있습니다.

```
{
  "headers": {
      "Set-cookie": "cookie-name=cookie-value;Domain=myweb.com;Secure;HttpOnly",
      "Content-Type": "application/json"
  },
}
```

다중 값 헤더를 활성화할 경우 다음과 같이 여러 쿠키를 지정할 수 있습니다.

```
{
  "multiValueHeaders": {
      "Set-cookie": ["cookie-name=cookie-value;Domain=myweb.com;Secure;HttpOnly","cookie-
name=cookie-value;Expires=May 8, 2019"],
      "Content-Type": ["application/json"]
  },
}
```

다중 값 헤더 활성화

대상 유형이 `lambda`인 대상 그룹의 다중 값 헤더를 활성화하거나 비활성화할 수 있습니다.

콘솔을 사용하여 다중 값 헤더를 활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택합니다.
4. [Description] 탭에서 [Edit attributes]를 선택합니다.
5. Multi value headers(다중 값 헤더)에서 Enable(활성화)을 선택합니다.
6. Save를 선택합니다.

AWS CLI를 사용하여 다중 값 헤더를 활성화하려면

`lambda.multi_value_headers.enabled` 속성과 함께 modify-target-group-attributes 명령을 사용합니다.

상태 확인 활성화

기본적으로 상태 확인은 `lambda` 유형의 대상 그룹에 대해 비활성화됩니다. Amazon Route 53를 사용하여 DNS 장애 조치를 구현하기 위해 상태 확인을 활성화할 수 있습니다. Lambda 함수는 상태 확인 요청에 응답하기 전에 다운스트림 서비스의 상태를 확인할 수 있습니다. Lambda 함수의 응답이 상태 확인 실패를 나타내는 경우 상태 확인 실패가 Route 53에 전달됩니다. 백업 애플리케이션 스택으로 장애 조치하도록 Route 53를 구성할 수 있습니다.

Lambda 함수 호출에 대한 요금과 마찬가지로 상태 확인에 대한 요금이 부과됩니다.

다음은 Lambda 함수에 전송되는 상태 확인 이벤트의 형식입니다. 이벤트가 상태 확인 이벤트인지 여부를 확인하려면 사용자 에이전트 필드의 값을 확인합니다. 상태 확인의 사용자 에이전트는 `ELB-HealthChecker/2.0`입니다.

```
{
    "requestContext": {
        "elb": {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/my-target-
group/6d0ecf831eec9f09"
        }
    },
    "httpMethod": "GET",
    "path": "/",
    "queryStringParameters": {},
    "headers": {
        "user-agent": "ELB-HealthChecker/2.0"
    },
    "body": "",
    "isBase64Encoded": false
}
```

대상 그룹에 대한 상태 확인을 활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택합니다.
4. Health checks(상태 확인) 탭에서 Edit health check(상태 확인 편집)를 선택합니다.
5. Health Check(상태 확인)에서 Enable(활성화)을 선택합니다.
6. Save를 선택합니다.

AWS CLI를 사용하여 대상 그룹에 대한 상태 확인을 활성화하려면

`--health-check-enabled` 옵션과 함께 modify-target-group 명령을 사용합니다.

Lambda 함수 등록 취소

트래픽을 Lambda 함수에 더 이상 전송할 필요가 없는 경우 해당 함수의 등록을 취소할 수 있습니다. Lambda 함수의 등록을 취소한 후에는 처리 중인 요청이 HTTP 5XX 오류와 함께 실패합니다.

Lambda 함수를 바꾸려면 새 대상 그룹을 생성하고, 새 함수를 새 대상 그룹에 등록한 다음, 새 대상 그룹을 기존 대상 그룹 대신 사용하도록 리스너 규칙을 업데이트하는 것이 좋습니다.

Lambda 함수의 등록을 취소하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택합니다.
4. Targets(대상) 탭에서 Deregister(등록 취소)를 선택합니다.
5. [Deregister]를 선택합니다.

AWS CLI를 사용하여 Lambda 함수의 등록을 취소하려면

deregister-targets 명령을 사용합니다.

대상 그룹에 대한 태그

태그를 사용하면 용도, 소유자 또는 환경 등에 따라 대상 그룹을 다양한 방식으로 분류할 수 있습니다.

각 대상 그룹에 여러 태그를 추가할 수 있습니다. 태그 키는 대상 그룹별로 고유해야 합니다. 대상 그룹에 이미 연결된 키를 통해 태그를 추가하면 해당 태그의 값이 업데이트됩니다.

사용이 끝난 태그는 삭제할 수 있습니다.

제한 사항

- 리소스당 최대 태그 수 — 50개
- 최대 키 길이 — 유니코드 문자 127자
- 최대 값 길이 — 유니코드 문자 255자
- 태그 키와 값은 대/소문자를 구분합니다. 허용되는 문자는 UTF-8로 표현할 수 있는 문자, 공백 및 숫자와 특수 문자: + - = . _ : / @. 선행 또는 후행 공백을 사용하면 안 됩니다.
- 태그 이름이나 값에서 `aws:` 접두사는 사용하지 마십시오. 이 단어는 AWS용으로 예약되어 있습니다. 이 접두사가 지정된 태그 이름이나 값은 편집하거나 삭제할 수 없습니다. 이 접두사가 지정된 태그는 리소스당 태그 수 제한에 포함되지 않습니다.

콘솔을 사용하여 대상 그룹 태그를 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택합니다.
4. Tags(태그) 탭에서 Add/Edit Tags(태그 추가/편집)를 선택하고 다음 중 하나 이상의 작업을 수행합니다.
   a. 태그를 업데이트하려면 [Key] 및 [Value] 값을 수정합니다.
   b. 새로운 태그를 추가하려면 Create Tag(태그 생성)를 선택한 다음 Key(키) 값 및 Value(값)를 입력합니다.
   c. 태그를 삭제하려면 해당 태그 옆의 삭제 아이콘(X)을 선택합니다.
5. 태그 업데이트를 마쳤으면 [Save]를 선택합니다.

AWS CLI를 사용하여 대상 그룹 태그를 업데이트하려면

add-tags 및 remove-tags 명령을 사용합니다.

대상 그룹 삭제

작업에서 참조하지 않는 경우 대상 그룹을 삭제할 수 있습니다. 대상 그룹을 삭제해도 대상 그룹에 등록된 대상에는 영향을 미치지 않습니다. 등록된 EC2 인스턴스가 더 이상 필요하지 않은 경우 중지 또는 종료할 수 있습니다.

콘솔을 사용하여 대상 그룹을 삭제하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 [LOAD BALANCING] 아래에서 [Target Groups]를 선택합니다.
3. 대상 그룹을 선택하고 [Actions], [Delete]를 차례로 선택합니다.

4. 확인 메시지가 표시되면 [Yes]를 선택합니다.

AWS CLI를 사용하여 대상 그룹을 삭제하려면

delete-target-group 명령을 사용합니다.

애플리케이션 로드 밸런서 모니터링

다음 기능을 사용하여 로드 밸런서를 모니터링하고 트래픽 패턴을 분석하며 로드 밸런서 및 대상의 문제를 해결할 수 있습니다.

CloudWatch 지표

Amazon CloudWatch를 사용하면 로드 밸런서 및 대상을 위한 데이터 요소에 대한 통계를 지표라고 하는 정렬된 시계열 집합으로 검색할 수 있습니다. 이러한 지표를 사용하여 시스템이 예상대로 수행되고 있는지 확인할 수 있습니다. 자세한 내용은 Application Load Balancer를 위한 CloudWatch 지표 (p. 70)를 참조하십시오.

액세스 로그

액세스 로그를 사용하여 로드 밸런서에 보낸 요청에 대한 자세한 정보를 캡처하고 Amazon S3에 로그 파일로 저장할 수 있습니다. 또한 이러한 액세스 로그를 사용하여 트래픽 패턴을 분석하고 대상의 문제를 해결할 수 있습니다. 자세한 내용은 애플리케이션 로드 밸런서를 위한 액세스 로그 (p. 81) 섹션을 참조하십시오.

요청 추적

요청 추적을 사용하여 HTTP 요청을 추적할 수 있습니다. 로드 밸런서는 수신한 각 요청에 트레이스 식별자가 있는 헤더를 추가합니다. 자세한 정보는 Application Load Balancer를 위한 요청 추적 (p. 92) 단원을 참조하십시오.

CloudTrail 로그

AWS CloudTrail을 사용하여 Elastic Load Balancing API에 보낸 요청에 대한 자세한 정보를 캡처하고 Amazon S3에 로그 파일로 저장할 수 있습니다. 이러한 CloudTrail 로그를 사용하여 어떤 요청이 이루어졌는지, 어떤 소스 IP 주소에서 요청을 했는지, 누가 언제 요청했는지 등을 확인할 수 있습니다. 자세한 내용은 AWS CloudTrail을 사용하여 Application Load Balancer에 대한 API 호출 로깅 (p. 93)을(를) 참조하십시오.

Application Load Balancer를 위한 CloudWatch 지표

Elastic Load Balancing는 해당 로드 밸런서 및 대상에 대한 데이터 포인트를 Amazon CloudWatch에 게시합니다. CloudWatch를 사용하면 그러한 데이터 포인트에 대한 통계를 지표라고 하는 정렬된 시계열 데이터 집합으로 검색할 수 있습니다. 지표를 모니터링할 변수로 생각하면 데이터 요소는 시간에 따른 변수의 값을 나타냅니다. 예를 들어 지정된 기간 동안 로드 밸런서에 대한 정상 상태 대상의 총 수를 모니터링할 수 있습니다. 각 데이터 요소에는 연결된 타임스탬프와 측정 단위(선택 사항)가 있습니다.

지표를 사용하여 시스템이 예상대로 수행되고 있는지 확인할 수 있습니다. 예를 들어 CloudWatch 경보를 생성하여 지정된 지표를 모니터링할 수 있으며, 지표가 허용 범위를 벗어난다고 간주되는 경우 작업(예: 이메일 주소로 알림 전송)를 시작할 수 있습니다.

Elastic Load Balancing는 요청이 로드 밸런서를 통과하는 경우에만 CloudWatch에 지표를 보고합니다. 로드 밸런서를 통과하는 요청이 있는 경우, Elastic Load Balancing는 60초마다 지표를 측정하여 전송합니다. 로드 밸런서를 통과하고 있는 요청이 없는 경우나 지표에 대한 데이터가 없는 경우에는 지표가 보고되지 않습니다.

자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하십시오.

목차

- 로드 밸런서에 대한 CloudWatch 지표 보기 (p. 79)

Application Load Balancer 지표

AWS/ApplicationELB 네임스페이스에는 다음 로드 밸런서 지표가 포함되어 있습니다.

지표	설명
ActiveConnectionCount	클라이언트에서 로드 밸런서로, 그리고 로드 밸런서에서 대상으로 동시에 연결되는 활성 TCP 연결 총 수. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer
ClientTLSNegotiationErrorCount	로드 밸런서와 세션을 구성하지 않은 클라이언트에서 시작된 TLS 연결 수. 가능한 원인으로는 암호 또는 프로토콜 불일치가 있습니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer
ConsumedLCUs	로드 밸런서에서 사용하는 로드 밸런서 용량 단위(LCU) 수. 시간 단위로 사용한 LCU 수만큼 요금을 지불하면 됩니다. 자세한 내용은 Elastic Load Balancing 요금을 참조하십시오. 보고 기준: 항상 보고 통계: 모두 Dimensions • LoadBalancer
HTTP_Fixed_Response_Count	성공한 고정 응답 작업의 수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. Dimensions • LoadBalancer
HTTP_Redirect_Count	성공한 리디렉션 작업의 수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다.

지표	설명
	Dimensions • LoadBalancer
HTTP_Redirect_Url_Limit_Exceeded_Count	유닌 인천 해당의 URL이 8K보다 크기 때문에 완료할 수 없는 리디렉션 작업의 수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. Dimensions • LoadBalancer
HTTPCode_ELB_3XX_Count	로드 밸런서에서 생성되는 HTTP 3XX 리디렉션 코드의 수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. Dimensions • LoadBalancer
HTTPCode_ELB_4XX_Count	로드 밸런서에서 생성된 HTTP 4XX 클라이언트 오류 코드 수. 클라이언트 오류는 요청 형식이 잘못되었거나 불완전할 때 생성됩니다. 이러한 요청은 대상에서 수신되지 않습니다. 단, 대상에서 생성된 응답 코드 수는 여기에 포함되지 않습니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Minimum, Maximum 및 Average는 모두 1을 반환합니다. Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer
HTTPCode_ELB_5XX_Count	로드 밸런서에서 생성된 HTTP 5XX 서버 오류 코드 수. 단, 대상에서 생성된 응답 코드 수는 여기에 포함되지 않습니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Minimum, Maximum 및 Average는 모두 1을 반환합니다. Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer
HTTPCode_ELB_500_Count	로드 밸런서에서 생성된 HTTP 500 오류 코드 수. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다.

지표	설명
HTTPCode_ELB_502_Count	로드 밸런서에서 생성된 HTTP 502 오류 코드 수. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다.
HTTPCode_ELB_503_Count	로드 밸런서에서 생성된 HTTP 503 오류 코드 수. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다.
HTTPCode_ELB_504_Count	로드 밸런서에서 생성된 HTTP 504 오류 코드 수. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다.
IPv6ProcessedBytes	로드 밸런서에서 IPv6를 통해 처리된 총 바이트 수. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer
IPv6RequestCount	로드 밸런서가 수신한 IPv6 요청 수. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Minimum, Maximum 및 Average는 모두 1을 반환합니다. Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer • TargetGroup, LoadBalancer • TargetGroup, AvailabilityZone, LoadBalancer
NewConnectionCount	클라이언트에서 로드 밸런서로, 그리고 로드 밸런서에서 대상으로 새롭게 구성된 TCP 연결 총 수 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer

지표	설명
ProcessedBytes	로드 밸런서에서 IPv4 및 IPv6를 통해 처리된 총 바이트 수. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer
RejectedConnectionCount	로드 밸런서가 최대 연결 수에 도달하여 거부된 연결 수 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer
RequestCount	IPv4 및 IPv6를 통해 처리된 요청 수입니다. 여기에는 로드 밸런서의 대상이 생성한 응답이 포함된 요청만 포함됩니다. 보고 기준: 항상 보고 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer • TargetGroup, LoadBalancer • TargetGroup, AvailabilityZone, LoadBalancer
RuleEvaluations	1시간 평균 요청 빈도를 기준으로 로드 밸런서에서 처리된 규칙 수 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer

AWS/ApplicationELB 네임스페이스에는 다음 대상 지표가 포함되어 있습니다.

지표	설명
HealthyHostCount	정상 상태로 간주되는 대상 수 Reporting criteria(보고 기준): 상태 확인을 활성화한 경우 보고됨 통계: 가장 유용한 통계는 Average, Minimum 및 Maximum입니다.

지표	설명
	Dimensions • TargetGroup, LoadBalancer • TargetGroup, AvailabilityZone, LoadBalancer
HTTPCode_Target_2XX_Count, HTTPCode_Target_3XX_Count, HTTPCode_Target_4XX_Count, HTTPCode_Target_5XX_Count	대상에서 생성된 HTTP 응답 코드 수. 단, 로드 밸런서에서 생성된 응답 코드수는 여기에 포함되지 않습니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Minimum, Maximum 및 Average는 모두 1을 반환합니다. Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer • TargetGroup, LoadBalancer • TargetGroup, AvailabilityZone, LoadBalancer
NonStickyRequestCount	로드 밸런서가 기존 고정 세션을 사용할 수 없기 때문에 새 대상을 선택한 요청 수입니다. 예를 들어, 요청이 새 클라이언트의 첫 번째 요청이었고 고정 쿠키가 제공되지 않았거나, 고정 쿠키가 제공되었지만 이 대상 그룹에 등록된 대상을 지정하지 않았거나, 고정 그룹이 잘못된 형식이거나 만료되었거나, 내부 오류로 인해 로드 밸런서가 고정 쿠키를 읽을 수 없었습니다. Reporting criteria(보고 기준): 대상 그룹에서 고정이 활성화됩니다. 통계: 유일하게 의미 있는 통계는 Sum입니다.
RequestCountPerTarget	대상 그룹에서 대상별로 수신한 평균 요청 수. 대상 그룹은 TargetGroup 차원을 사용하여 지정해야 합니다. 대상이 Lambda 함수인 경우 이 지표는 적용되지 않습니다. 보고 기준: 항상 보고 통계: 유일하게 유효한 통계는 Sum입니다. 단, 이 통계는 합산이 아닌 평균을 의미합니다. Dimensions • TargetGroup • TargetGroup, LoadBalancer

지표	설명
TargetConnectionErrorCount	로드 밸런서와 대상 사이에 성공적으로 구성되지 않은 연결 수 대상이 Lambda 함수인 경우 이 지표는 적용되지 않습니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer • TargetGroup, LoadBalancer • TargetGroup, AvailabilityZone, LoadBalancer
TargetResponseTime	로드 밸런서에서 요청 신호를 전송한 후 대상에서 응답 신호가 수신될 때까지 결과된 시간(초). 이 지표는 액세스 로그에서 target_processing_time 필드와 동일합니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Average 및 pNN.NN입니다(백분위수). Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer • TargetGroup, LoadBalancer • TargetGroup, AvailabilityZone, LoadBalancer
TargetTLSNegotiationErrorCount	대상과 세션을 구성하지 않은 로드 밸런서에서 시작된 TLS 연결 수. 가능한 원인으로는 암호 또는 프로토콜 불일치가 있습니다. 대상이 Lambda 함수인 경우 이 지표는 적용되지 않습니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone, LoadBalancer • TargetGroup, LoadBalancer • TargetGroup, AvailabilityZone, LoadBalancer
UnHealthyHostCount	비정상 상태로 간주되는 대상 수 Reporting criteria(보고 기준): 상태 확인을 활성화한 경우 보고됨 통계: 가장 유용한 통계는 Average, Minimum 및 Maximum입니다. Dimensions • TargetGroup, LoadBalancer • TargetGroup, AvailabilityZone, LoadBalancer

AWS/ApplicationELB 네임스페이스에는 대상으로 등록된 Lambda 함수에 대해 다음과 같은 지표가 포함됩니다.

지표	설명
LambdaInternalError	로드 밸런서 또는 AWS Lambda에 내부적인 문제 때문에 실패한 Lambda 함수에 대한 요청 수입니다. 오류 이유 코드를 가져오려면 액세스 로그의 오류 이유 필드를 확인하십시오. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. 차원: TargetGroup
LambdaTargetProcessedBytes	Lambda 함수의 요청과 응답에 대해 로드 밸런서에서 처리된 총 바이트 수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. 차원: LoadBalancer
LambdaUserError	Lambda 함수 문제 때문에 실패한 Lambda 함수에 대한 요청 수입니다. 예를 들어, 로드 밸런서가 함수를 호출할 권한이 없거나, 형식이 잘못되거나 필수 필드가 누락된 함수에서 로드 밸런서가 JSON을 수신했거나, 요청 본문 또는 응답의 크기가 1MB의 최대 크기를 초과했습니다. 오류 이유 코드를 가져오려면 액세스 로그의 오류 이유 필드를 확인하십시오. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. 차원: TargetGroup

AWS/ApplicationELB 네임스페이스에는 사용자 인증에 대한 다음 지표가 포함되어 있습니다.

지표	설명
ELBAuthError	인증 작업이 잘못 구성되었거나, 로드 밸런서가 IdP와 연결을 설정할 수 없었거나, 로드 밸런서가 내부 오류로 인해 인증 흐름을 완료할 수 없었기 때문에 완료되지 않은 사용자 인증 수. 오류 이유 코드를 가져오려면 액세스 로그의 오류 이유 필드를 확인하십시오. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. 차원: LoadBalancer
ELBAuthFailure	IdP가 사용자에 대한 액세스를 거부했거나 인증 코드가 두 번 이상 사용되었기 때문에 완료되지 않은 사용자 인증 수. 오류 이유 코드를 가져오려면 액세스 로그의 오류 이유 필드를 확인하십시오. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다.

지표	설명
	차원: LoadBalancer
ELBAuthLatency	IdP에 ID 토큰 및 사용자 정보를 쿼리하는 데 경과한 시간(단위: 밀리초)입니다. 이러한 작업이 하나 이상 실패할 경우 이 지표는 실패까지의 시간입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 모든 통계가 의미 있습니다. 차원: LoadBalancer
ELBAuthRefreshTokenSuccess	로드 밸런서가 IdP에서 제공된 새로 고침 토큰을 사용하여 사용자 클레임을 성공적으로 새로 고친 횟수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. 차원: LoadBalancer
ELBAuthSuccess	성공한 인증 작업의 수. 이 지표는 로드 밸런서가 IdP로부터 사용자 클레임을 검색한 후 인증 워크플로 종료 시 증가합니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. 차원: LoadBalancer
ELBAuthUserClaimsSizeExceeded	구성된 IdP가 11K 바이트 크기를 초과하는 사용자 클레임을 반환한 횟수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. 차원: LoadBalancer

Application Load Balancer 지표 차원

Application Load Balancer 지표를 필터링하려면 다음 차원을 사용하십시오.

차원	설명
AvailabilityZone	가용 영역을 기준으로 지표 데이터를 필터링합니다.
LoadBalancer	로드 밸런서를 기준으로 지표 데이터를 필터링합니다. 로드 밸런서는 다음과 같이 지정합니다. app/load-balancer-name/1234567890123456(로드 밸런서 ARN의 마지막 구간)
TargetGroup	대상 그룹을 기준으로 지표 데이터를 필터링합니다. 대상 그룹은 다음과 같이 지정합니다. targetgroup/target-group-name/1234567890123456(대상 그룹 ARN의 마지막 구간).

Application Load Balancer 지표에 대한 통계

CloudWatch는 Elastic Load Balancing가 게시한 지표 데이터 포인트에 따라 통계를 제공합니다. 통계는 지정한 기간에 걸친 지표 데이터 집계입니다. 통계를 요청하면 지표 이름 및 차원으로 반환된 데이터 스트림이 식별됩니다. 차원이란 지표를 고유하게 식별하는 데 도움이 되는 이름-값 쌍을 말합니다. 예를 들어 특정 가용 영역에서 시작된 로드 밸런서를 지원하는 정상 상태의 모든 EC2 인스턴스에 대한 통계를 요청할 수 있습니다.

`Minimum` 및 `Maximum` 통계는 개별 로드 밸런서 노드가 보고한 최소 및 최대 값을 반영합니다. 예를 들어 로드 밸런서 노드가 2개라고 가정해 보겠습니다. 하나의 노드에는 `Minimum`이 2, `Maximum`이 10, `Average`가 6인 `HealthyHostCount`가 있으며 다른 노드에는 `Minimum`이 1, `Maximum`이 5, `Average`가 3인 `HealthyHostCount`가 있습니다. 따라서 로드 밸런서의 `Minimum`은 1, `Maximum`은 10, `Average`는 4입니다.

`Sum` 통계는 모든 로드 밸런서 노드의 집계 값입니다. 지표에는 기간별 보고서가 여러 개 있기 때문에 `Sum`은 모든 로드 밸런서 노드에서 집계된 지표에만 적용할 수 있습니다.

`SampleCount` 통계는 측정된 샘플의 수입니다. 지표는 샘플링 간격 및 이벤트를 토대로 수집이 되기 때문에 일반적으로 이 통계는 유용하지 않습니다. 예를 들어 `HealthyHostCount`에 대해 `SampleCount`는 각 로드 밸런서 노드가 보고하는 샘플 수를 기반으로 하며 정상 호스트 수는 아닙니다.

백분위 수는 데이터 세트에서 값의 상대적 위치를 나타냅니다. 소수점 두 자리까지 사용하여 백분위 수를 지정할 수 있습니다(예: p95.45). 예를 들어 95 백분위는 데이터의 95%가 이 값보다 아래에 있고 5%가 위에 있다는 것을 의미합니다. 백분위 수는 종종 이상치를 격리하는 데 사용됩니다. 예를 들어 애플리케이션이 캐시에서 오는 요청의 대다수를 1-2 ms에 처리하지만, 캐시가 비어 있는 경우에는 처리에 100 - 200 ms가 걸린다고 가정해 봅시다. 최대값은 가장 느린 경우(200 ms 정도)를 반영합니다. 평균은 데이터의 분산을 나타내지 않습니다. 백분위 수는 애플리케이션 성능을 훨씬 의미 있는 방식으로 볼 수 있습니다. Auto Scaling 트리거 또는 CloudWatch 경보로 99 백분위를 사용하면 처리에 2 ms가 넘게 걸리는 요청이 전체의 1%를 넘지 않게 할 수 있습니다.

로드 밸런서에 대한 CloudWatch 지표 보기

Amazon EC2 콘솔을 사용해 로드 밸런서에 대한 CloudWatch 지표를 볼 수 있습니다. 이 측정치들은 모니터링 그래프로 표시됩니다. 로드 밸런서가 활성 상태로 요청을 수신 중에 있으면 모니터링 그래프에 데이터 요소가 표시됩니다.

또는 CloudWatch 콘솔을 사용해 로드 밸런서를 위한 지표를 볼 수 있습니다.

Amazon EC2 콘솔을 사용한 메트릭 확인

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 대상 그룹을 기준으로 필터링한 지표를 보려면 다음 작업을 수행합니다.
   a. 탐색 창에서 [Target Groups]를 선택합니다.
   b. 대상 그룹을 선택한 다음 [Monitoring] 탭을 선택합니다.
   c. (선택 사항) 시간을 기준으로 결과를 필터링하려면 [Showing data for]에서 시간 범위를 선택합니다.
   d. 단일 지표를 크게 보려면 그래프를 선택합니다.
3. 로드 밸런서를 기준으로 필터링한 지표를 보려면 다음 작업을 수행합니다.
   a. 탐색 창에서 [Load Balancers]를 클릭합니다.
   b. 로드 밸런서를 선택한 다음 [Monitoring] 탭을 선택합니다.
   c. (선택 사항) 시간을 기준으로 결과를 필터링하려면 [Showing data for]에서 시간 범위를 선택합니다.
   d. 단일 지표를 크게 보려면 그래프를 선택합니다.

CloudWatch 콘솔을 사용한 메트릭 확인

1. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 [Metrics]를 선택합니다.
3. [ApplicationELB] 네임스페이스를 선택합니다.
4. (선택 사항) 모든 차원의 지표를 보려면 검색 필드에 이름을 입력합니다.
5. (선택 사항) 차원을 기준으로 필터링하려면 다음 중 하나를 선택하십시오.

   - 로드 밸런서에 보고된 지표만 표시하려면 [Per AppELB Metrics]를 선택합니다. 단일 로드 밸런서에 대한 지표를 보려면 검색 필드에 해당되는 이름을 입력합니다.
   - 대상 그룹에 보고된 지표만 표시하려면 [Per AppELB, per TG Metrics]를 선택합니다. 단일 대상 그룹에 대한 지표를 보려면 검색 필드에 해당되는 이름을 입력합니다.
   - 가용 영역이 로드 밸런서에 대해 보고한 지표만 표시하려면 [Per AppELB, per AZ Metrics]를 선택합니다. 단일 로드 밸런서에 대한 지표를 보려면 검색 필드에 해당되는 이름을 입력합니다. 단일 로드 밸런서에 대한 지표를 보려면 검색 필드에 해당되는 이름을 입력합니다.
   - 가용 영역 및 대상 그룹이 로드 밸런서에 대해 보고한 지표만 표시하려면 [Per AppELB, per AZ, per TG Metrics]를 선택합니다. 단일 로드 밸런서에 대한 지표를 보려면 검색 필드에 해당되는 이름을 입력합니다. 단일 대상 그룹에 대한 지표를 보려면 검색 필드에 해당되는 이름을 입력합니다. 단일 로드 밸런서에 대한 지표를 보려면 검색 필드에 해당되는 이름을 입력합니다.

AWS CLI을(를) 사용하여 지표를 보려면

사용 가능한 지표의 목록을 표시하려면 아래 list-metrics 명령을 사용하십시오.

```
aws cloudwatch list-metrics --namespace AWS/ApplicationELB
```

AWS CLI를 사용하여 지표에 대한 통계를 구하려면

지정된 지표 및 차원에 대한 통계를 구하려면 아래 get-metric-statistics 명령을 사용하십시오. CloudWatch는 각각의 고유한 차원의 조합을 별도의 지표로 처리합니다. 특별 게시가 되지 않은 차원의 조합을 사용해 통계를 검색할 수는 없습니다. 지표 생성 시 사용한 것과 동일하게 차원을 지정해야 합니다.

```
aws cloudwatch get-metric-statistics --namespace AWS/ApplicationELB \
--metric-name UnHealthyHostCount --statistics Average  --period 3600 \
--dimensions Name=LoadBalancer,Value=app/my-load-balancer/50dc6c495c0c9188 \
Name=TargetGroup,Value=targetgroup/my-targets/73e2d6bc24d8a067 \
--start-time 2016-04-18T00:00:00Z --end-time 2016-04-21T00:00:00Z
```

다음은 예제 출력입니다.

```
{
    "Datapoints": [
        {
            "Timestamp": "2016-04-18T22:00:00Z",
            "Average": 0.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2016-04-18T04:00:00Z",
            "Average": 0.0,
            "Unit": "Count"
        },
        ...
    ],
    "Label": "UnHealthyHostCount"
}
```

애플리케이션 로드 밸런서를 위한 액세스 로그

Elastic Load Balancing은 로드 밸런서에 전송된 요청에 대한 자세한 정보를 캡처하는 액세스 로그를 제공합니다. 각 로그에는 요청을 받은 시간, 클라이언트의 IP 주소, 지연 시간, 요청 경로 및 서버 응답과 같은 정보가 포함되어 있습니다. 이러한 액세스 로그를 사용하여 트래픽 패턴을 분석하고 문제를 해결할 수 있습니다.

액세스 로그는 Elastic Load Balancing의 옵션 기능으로, 기본적으로 비활성화됩니다. 로드 밸런서에 대해 액세스 로그를 활성화하면 Elastic Load Balancing가 로그를 캡처하여 압축 파일을 지정한 Amazon S3 버킷에 저장합니다. 언제든지 액세스 로그를 비활성화할 수 있습니다.

S3 버킷에 대한 Amazon S3 관리형 암호화 키(SSE-S3)를 사용하여 서버 측 암호화를 활성화하는 경우, 각 액세스 로그 파일은 S3 버킷에 저장되기 전에 자동으로 암호화되고 액세스 시 암호화가 해제됩니다. 암호화된 로그 파일이나 암호화되지 않은 로그 파일을 액세스하는 방식과 다르지 않으므로 별도의 조치가 필요 없습니다. 각 로그 파일은 고유 키로 암호화되며, 주기적으로 바뀌는 마스터 키를 사용하여 키 자체가 암호화됩니다. 자세한 내용은 Amazon Simple Storage Service 개발자 가이드에서 Amazon S3 관리형 암호화 키(SSE-S3)로 서버 측 암호화를 사용하여 데이터 보호를 참조하십시오.

액세스 로그에 대한 추가 요금은 없습니다;. Amazon S3에 대한 스토리지 비용은 청구되지만 Elastic Load Balancing가 Amazon S3에 로그 파일을 전송하기 위해 사용하는 대역폭에 대해서는 비용이 청구되지 않습니다. 스토리지 비용에 대한 자세한 내용은 Amazon S3 요금을 참조하십시오.

목차

액세스 로그 파일

Elastic Load Balancing는 5분마다 각 로드 밸런서 노드에 대한 로그 파일을 게시합니다. 로그 전달은 결과의 일관성이 있습니다. 로드 밸런서는 같은 기간 동안 여러 개의 로그를 전달할 수 있습니다. 이러한 상황은 보통 사이트에 트래픽이 많은 경우에 발생합니다.

액세스 로그의 파일 이름은 다음 형식을 사용합니다.

```
bucket[/prefix]/AWSLogs/aws-account-id/elasticloadbalancing/region/yyyy/mm/dd/aws-account-id_elasticloadbalancing_region_load-balancer-id_end-time_ip-address_random-string.log.gz
```

버킷

S3 버킷의 이름

prefix

버킷의 접두사(논리적 계층 구조)입니다. 접두사를 지정하지 않는 경우 로그는 버킷의 루트 수준에 저장됩니다.

aws-account-id

소유자의 AWS 계정 ID입니다.

region

로드 밸런서 및 S3 버킷을 위한 리전입니다.

yyyy/mm/dd

로그가 전달된 날짜입니다.

load-balancer-id

로드 밸런서의 리소스 ID입니다. 리소스 ID에 포함되어 있는 슬래시(/)가 마침표(.)로 대체됩니다.

end-time

로깅 간격이 끝나는 날짜와 시간입니다. 예를 들어 종료 시간이 20140215T2340Z이면 23:35와 23:40 사이의 요청에 대한 항목이 포함됩니다.

ip-address

요청을 처리한 로드 밸런서 노드의 IP 주소입니다. 내부 로드 밸런서의 경우 프라이빗 IP 주소가 됩니다.

random-string

시스템에서 생성된 임의 문자열입니다.

다음은 로그 파일 이름의 예제입니다.

```
s3://my-bucket/prefix/AWSLogs/123456789012/elasticloadbalancing/us-
east-2/2016/05/01/123456789012_elasticloadbalancing_us-east-2_my-
loadbalancer_20140215T2340Z_172.160.001.192_20sg8hgm.log.gz
```

원하는 기간만큼 버킷에 로그 파일을 저장할 수 있습니다. 그러나 Amazon S3 수명 주기 규칙을 정의하여 자동으로 로그 파일을 보관하거나 삭제할 수도 있습니다. 자세한 내용은 Amazon Simple Storage Service 개발자 가이드에서 객체 수명 주기 관리를 참조하십시오.

액세스 로그 항목

Elastic Load Balancing는 대상으로 전달되지 않는 요청을 포함해 로드 밸런서로 전송된 모든 요청을 기록합니다. 예를 들어 클라이언트가 잘못된 요청을 보내거나 요청에 응답할 정상 인스턴스가 없는 경우에도 요청은 계속 기록됩니다. 는 상태 확인 요청을 기록하지 않습니다.

각 로그 항목에는 로드 밸런서에 대한 단일 요청(WebSockets의 경우 연결)의 세부 정보가 포함되어 있습니다. WebSockets의 경우, 연결이 종료된 이후에만 항목이 기록됩니다. 업그레이드 연결을 설정할 수 없는 경우에는 HTTP 또는 HTTPS 요청과 항목이 동일합니다.

Important

Elastic Load Balancing은 최대한 요청을 기록합니다. 모든 요청을 완벽하게 기록하기 위한 용도가 아니라 요청 특성을 이해하는 데 액세스 로그를 사용하는 것이 좋습니다.

구문

다음 표에서는 액세스 로그 항목의 필드를 순서대로 설명합니다. 모든 필드는 공백으로 구분됩니다. 새 필드가 도입되면 로그 항목 끝에 추가됩니다. 예상하지 못했던 방식으로 로그 항목이 끝나면 모든 필드를 무시해야 합니다.

필드	설명
type	요청 또는 연결의 유형입니다. 사용 가능한 값은 다음과 같습니다(기타 값은 모두 무시). • `http` — HTTP • SSL/TLS를 통한 `https` — HTTP • SSL/TLS를 통한 `h2` — HTTP/2

필드	설명
	• ws — 웹 소켓 • wss — SSL/TLS 상에서 웹 소켓
타임스탬프	로드 밸런서가 클라이언트에 응답을 생성한 시간(ISO 8601 형식)입니다. WebSockets의 경우, 연결이 종료된 시점이 됩니다.
elb	로드 밸런서의 리소스 ID입니다. 액세스 로그 항목을 분석하고 있다면 리소스 ID에 슬래시(/)가 포함될 수 있다는 것을 기억하십시오.
client:port	요청을 하는 클라이언트의 IP 주소 및 포트입니다.
target:port	이 요청을 처리한 대상의 IP 주소 및 포트입니다. 클라이언트가 전체 요청을 전송하지 않은 경우에는 로드 밸런서가 대상으로 요청을 디스패치 할 수 없고 이 값은 -로 설정됩니다. 대상이 Lambda 함수인 경우 이 값은 -로 설정됩니다. AWS WAF에서 요청을 차단한 경우, 이 값은 -로 설정되고 elb_status_code 값은 403으로 설정됩니다.
request_processing_time	로드 밸런서가 요청을 수신한 시간부터 대상으로 이를 전송한 시간까지의 총 경과 시간(초, 밀리초 단위)입니다. 로드 밸런서가 대상으로 요청을 디스패치할 수 없는 경우 이 값은 -1로 설정됩니다. 대상이 유휴 제한 시간 전에 연결을 종료하거나 클라이언트가 잘못된 요청을 보내는 경우에 이런 상황이 발생할 수 있습니다. 등록된 대상 유휴 시간 초과 횟수 이전에 응답하지 않는 경우에도 이 값이 -1로 설정될 수 있습니다.
target_processing_time	로드 밸런서가 대상에 요청을 보낸 시간부터 대상이 응답 헤더를 보내기 시작할 때까지의 총 경과 시간(초, 밀리초 단위)입니다. 로드 밸런서가 대상으로 요청을 디스패치할 수 없는 경우 이 값은 -1로 설정됩니다. 대상이 유휴 제한 시간 전에 연결을 종료하거나 클라이언트가 잘못된 요청을 보내는 경우에 이런 상황이 발생할 수 있습니다. 등록된 대상 유휴 시간 초과 횟수 이전에 응답하지 않는 경우에도 이 값이 -1로 설정될 수 있습니다.
response_processing_time	로드 밸런서가 대상에서 응답 헤더를 수신한 시간부터 클라이언트에 응답을 보내기 시작할 때까지의 총 경과 시간(초, 밀리초 단위)입니다. 여기에는 로드 밸런서의 대기 시간과 로드 밸런서에서 클라이언트까지의 연결 확보 시간이 모두 포함됩니다. 로드 밸런서가 대상으로 요청을 전송할 수 없는 경우 이 값은 -1로 설정됩니다. 대상이 유휴 제한 시간 전에 연결을 종료하거나 클라이언트가 잘못된 요청을 보내는 경우에 이런 상황이 발생할 수 있습니다.
elb_status_code	로드 밸런서의 응답 상태 코드입니다.
target_status_code	대상의 응답 상태 코드입니다. 대상으로 연결이 설정되고 대상이 응답을 전송한 경우에만 이 값이 기록됩니다. 그렇지 않으면 -에 설정됩니다.
received_bytes	클라이언트(요청자)로부터 수신된 요청의 크기(바이트)입니다. HTTP 요청의 경우, 헤더가 포함이 됩니다. WebSockets의 경우에는 연결 시 클라이언트에서 수신된 총 바이트 수입니다.

필드	설명
sent_bytes	클라이언트(요청자)에게 보낸 응답의 크기(바이트)입니다. HTTP 요청의 경우, 헤더가 포함이 됩니다. WebSockets의 경우에는 연결 시 클라이언트에 전송된 총 바이트 수입니다.
"요청"	큰 따옴표로 묶여 있고 HTTP 메서드 + protocol://host:port/uri + HTTP 버전 형식을 사용해 기록된 요청 줄입니다. 로드 밸런서는 요청 URI를 기록할 때 클라이언트가 보낸 URL을 원본 그대로 보관합니다. 또한 액세스 로그 파일에 대한 콘텐츠 유형을 설정하지 않습니다. 이 필드를 처리하는 경우 해당 클라이언트가 URL을 보낸 방법을 고려하십시오.
"user_agent"	요청을 보낸 클라이언트를 식별하는 사용자 에이전트 문자열입니다(큰 따옴표로 묶임). 이 문자열은 하나 이상의 제품 식별자, 제품[/버전]으로 이루어져 있습니다. 문자열이 8 KB보다 길면 잘리게 됩니다.
ssl_cipher	[HTTPS 리스너] SSL 암호입니다. 리스너가 HTTPS 리스너가 아닌 경우 이 값은 -로 설정됩니다.
ssl_protocol	[HTTPS 리스너] SSL 프로토콜입니다. 리스너가 HTTPS 리스너가 아닌 경우 이 값은 -로 설정됩니다.
target_group_arn	대상 그룹의 ARN(Amazon 리소스 이름)입니다.
"trace_id"	X-Amzn-Trace-Id 헤더의 콘텐츠가 기록됩니다(큰 따옴표로 묶임).
"domain_name"	[HTTPS 리스너] TLS 핸드셰이크 중에 클라이언트가 제공한 SNI 도메인(큰 따옴표로 묶임). 클라이언트가 SNI를 지원하지 않거나, 도메인이 인증서와 일치하지 않으면 이 값이 -로 설정되고 클라이언트에 기본 인증서가 제공됩니다.
"chosen_cert_arn"	[HTTPS 리스너] 클라이언트에 제공된 인증서의 ARN(큰 따옴표로 묶임). 세션이 재사용되는 경우 이 값은 `session-reused`로 설정됩니다.
matched_rule_priority	요청과 일치하는 규칙의 우선 순위 값입니다. 규칙이 일치하면 1~50,000 사이의 값입니다. 규칙이 일치하지 않고 기본 작업이 수행된 경우 이 값은 0에 설정됩니다. 규칙 평가 중 오류가 발생하면 -1에 설정됩니다. 기타 오류의 경우 -에 설정됩니다.
request_creation_time	로드 밸런서가 클라이언트에서 요청을 받은 시간입니다(ISO 8601 형식).
"실행된 작업"	요청을 처리할 때 수행된 작업(큰 따옴표로 묶임). 이 값은 `waf`, `waf-failed`, `authenticate`, `redirect`, `fixed-response`, `forward`와 같은 가능한 값을 포함할 수 있는 쉼표로 구분된 목록입니다. 잘못된 요청 등에 대해 수행된 작업이 없는 경우 이 값은 -로 설정됩니다.
"redirect_url"	HTTP 응답의 위치 헤더에 대한 리디렉션 대상 URL로, 큰따옴표로 묶여 있습니다. 수행된 리디렉션 작업이 없는 경우 이 값은 -로 설정됩니다.
"error_reason"	큰 따옴표로 묶인 오류 이유 코드입니다. 요청이 실패하면 이 값은 오류 이유 코드 (p. 84)에 설명된 오류 코드 중 하나입니다. 수행한 작업에 인증 작업이 포함되지 않거나 대상이 Lambda 함수가 아닌 경우, 이 값은 -로 설정됩니다.

오류 이유 코드

로드 밸런서가 인증 작업을 완료할 수 없는 경우, 로드 밸런서는 액세스 로그의 error_reason 필드에 다음 이유 코드 중 하나를 저장합니다. 또한 로드 밸런서는 해당 CloudWatch 지표를 증가시킵니다. 자세한 내용은 Application Load Balancer를 사용하여 사용자 인증 (p. 43) 단원을 참조하십시오.

Code	설명	지표
AuthInvalidCookie	인증 쿠키가 유효하지 않습니다.	ELBAuthFailure
AuthInvalidGrantError	토큰 엔드포인트의 권한 부여 코드가 유효하지 않습니다.	ELBAuthFailure
AuthInvalidIdToken	ID 토큰이 유효하지 않습니다.	ELBAuthFailure
AuthInvalidStateParam	상태 파라미터가 유효하지 않습니다.	ELBAuthFailure
AuthInvalidTokenResponse	토큰 엔드포인트의 응답이 유효하지 않습니다.	ELBAuthFailure
AuthInvalidUserinfoResponse	사용자 정보 엔드포인트의 응답이 유효하지 않습니다.	ELBAuthFailure
AuthMissingCodeParam	권한 부여 엔드포인트의 인증 응답에 'code'라는 쿼리 파라미터가 없습니다.	ELBAuthFailure
AuthMissingHostHeader	권한 부여 엔드포인트의 인증 응답에 호스트 헤더 필드가 없습니다.	ELBAuthError
AuthMissingStateParam	권한 부여 엔드포인트의 인증 응답에 'state'라는 쿼리 파라미터가 없습니다.	ELBAuthFailure
AuthTokenEpRequestFailed	토큰 엔드포인트에서 오류 응답(2XX 아님)이 있습니다.	ELBAuthError
AuthTokenEpRequestTimeout	로드 밸런서가 토큰 엔드포인트와 통신할 수 없습니다.	ELBAuthError
AuthUnhandledException	로드 밸런서에 처리할 수 없는 예외가 발생했습니다.	ELBAuthError
AuthUserinfoEpRequestFailed	IdP 사용자 정보 엔드포인트에서 오류 응답(2XX 아님)이 있습니다.	ELBAuthError
AuthUserinfoEpRequestTimeout	로드 밸런서가 IdP 사용자 정보 엔드포인트와 통신할 수 없습니다.	ELBAuthError
AuthUserinfoResponseSizeExceeded	IdP에서 반환한 클레임의 크기가 11K 바이트를 초과했습니다.	ELBAuthUserClaimsSizeExceeded

Lambda 함수에 대한 요청이 실패하면 로드 밸런서가 액세스 로그의 오류 이유 필드에 다음 이유 코드 중 하나를 저장합니다. 또한 로드 밸런서는 해당 CloudWatch 지표를 증가시킵니다. 자세한 내용은 Lambda 호출 작업을 참조하십시오.

코드	설명	지표
LambdaAccessDenied	로드 밸런서는 Lambda 함수를 호출할 권한이 없습니다.	LambdaUserError
LambdaConnectionTimeout	Lambda에 연결하려는 시도가 시간 초과되었습니다.	LambdaInternalError
LambdaEC2AccessDeniedException	Amazon EC2가 함수 초기화 중 Lambda에 대한 액세스를 거부했습니다.	LambdaUserError

코드	설명	지표
LambdaEC2ThrottledException	Amazon EC2가 함수 초기화 중 Lambda를 제한했습니다.	LambdaUserError
LambdaEC2UnexpectedException	Amazon EC2에서 함수 초기화 중 예기치 않은 예외가 발생했습니다.	LambdaUserError
LambdaENILimitReachedException	Lambda는 네트워크 인터페이스에 대한 제한을 초과했기 때문에 Lambda 함수의 구성에 지정된 VPC에서 네트워크 인터페이스를 생성할 수 없습니다.	LambdaUserError
LambdaInvalidResponse	Lambda 함수의 응답이 잘못된 형식이거나 필수 필드가 누락되었습니다.	LambdaUserError
LambdaInvalidRuntimeException	지정된 버전의 Lambda 런타임이 지원되지 않습니다.	LambdaUserError
LambdaInvalidSecurityGroupException	Lambda 함수의 구성에 지정된 보안 그룹 ID가 유효하지 않습니다.	LambdaUserError
LambdaInvalidSubnetException	Lambda 함수의 구성에 지정된 서브넷 ID가 유효하지 않습니다.	LambdaUserError
LambdaInvalidZipFileException	Lambda에서 지정된 함수 zip 파일의 압축을 풀 수 없습니다.	LambdaUserError
LambdaKMSAccessDeniedException	KMS 키에 대한 액세스가 거부되었기 때문에 Lambda에서 환경 변수의 암호화를 해제할 수 없습니다. Lambda 함수의 KMS 권한을 확인하십시오.	LambdaUserError
LambdaKMSDisabledException	지정된 KMS 키가 비활성화되었기 때문에 Lambda에서 환경 변수의 암호화를 해제할 수 없습니다. Lambda 함수의 KMS 키 설정을 확인하십시오.	LambdaUserError
LambdaKMSInvalidStateException	KMS 키의 상태가 유효하지 않기 때문에 Lambda에서 환경 변수의 암호화를 해제할 수 없습니다. Lambda 함수의 KMS 키 설정을 확인하십시오.	LambdaUserError
LambdaKMSNotFoundException	KMS 키를 찾을 수 없기 때문에 Lambda에서 환경 변수의 암호화를 해제할 수 없습니다. Lambda 함수의 KMS 키 설정을 확인하십시오.	LambdaUserError
LambdaRequestTooLarge	요청 본문의 크기가 1MB를 초과했습니다.	LambdaUserError
LambdaResourceNotFound	Lambda 함수를 찾을 수 없습니다.	LambdaUserError
LambdaResponseTooLarge	응답의 크기가 1MB를 초과했습니다.	LambdaUserError
LambdaServiceException	Lambda에 내부 오류가 발생했습니다.	LambdaInternalError
LambdaSubnetIPAddressLimitReachedException	Lambda에서 서브넷에 사용 가능한 IP 주소가 없으므로 Lambda에서 Lambda 함수에 대한 VPC 액세스를 설정할 수 없습니다.	LambdaUserError
LambdaThrottling	요청이 너무 많기 때문에 Lambda 함수가 제한되었습니다.	LambdaUserError
LambdaUnhandled	Lambda 함수에 처리할 수 없는 예외가 발생했습니다.	LambdaUserError

예제

다음은 로그 항목의 예제입니다. 보다 읽기 쉽도록 텍스트가 여러 줄에 나타납니다.

HTTP 항목 예제

다음은 HTTP 리스너(포트 80에서 포트 80)를 위한 로그 항목 예제입니다.

```
http 2018-07-02T22:23:00.186641Z app/my-loadbalancer/50dc6c495c0c9188
192.168.131.39:2817 10.0.0.1:80 0.000 0.001 0.000 200 200 34 366
"GET http://www.example.com:80/ HTTP/1.1" "curl/7.46.0" - -
arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
"Root=1-58337262-36d228ad5d99923122bbe354" "-" "-"
0 2018-07-02T22:22:48.364000Z "forward" "-" "-"
```

HTTPS 항목 예제

다음은 HTTP 리스너(포트 443에서 포트 80)를 위한 로그 항목 예제입니다.

```
https 2018-07-02T22:23:00.186641Z app/my-loadbalancer/50dc6c495c0c9188
192.168.131.39:2817 10.0.0.1:80 0.086 0.048 0.037 200 200 0 57
"GET https://www.example.com:443/ HTTP/1.1" "curl/7.46.0" ECDHE-RSA-AES128-GCM-SHA256
 TLSv1.2
arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
"Root=1-58337281-1d84f3d73c47ec4e58577259" "www.example.com" "arn:aws:acm:us-
east-2:123456789012:certificate/12345678-1234-1234-1234-123456789012"
1 2018-07-02T22:22:48.364000Z "authenticate,forward" "-" "-"
```

HTTP/2 항목 예제

다음은 HTTP/2 스트림을 위한 로그 항목 예제입니다.

```
h2 2018-07-02T22:23:00.186641Z app/my-loadbalancer/50dc6c495c0c9188
10.0.1.252:48160 10.0.0.66:9000 0.000 0.002 0.000 200 200 5 257
"GET https://10.0.2.105:773/ HTTP/2.0" "curl/7.46.0" ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2
arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
"Root=1-58337327-72bd00b0343d75b906739c42" "-" "-"
1 2018-07-02T22:22:48.364000Z "redirect" "https://example.com:80/" "-"
```

WebSockets 항목 예제

다음은 WebSockets 연결을 위한 로그 항목 예제입니다.

```
ws 2018-07-02T22:23:00.186641Z app/my-loadbalancer/50dc6c495c0c9188
10.0.0.140:40914 10.0.1.192:8010 0.001 0.003 0.000 101 101 218 587
"GET http://10.0.0.30:80/ HTTP/1.1" "-" - -
arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
"Root=1-58337364-23a8c76965a2ef7629b185e3" "-" "-"
1 2018-07-02T22:22:48.364000Z "forward" "-" "-"
```

보안 WebSockets 항목 예제

다음은 보안 WebSockets 연결을 위한 로그 항목 예제입니다.

```
wss 2018-07-02T22:23:00.186641Z app/my-loadbalancer/50dc6c495c0c9188
10.0.0.140:44244 10.0.0.171:8010 0.000 0.001 0.000 101 101 218 786
"GET https://10.0.0.30:443/ HTTP/1.1" "-" ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2
arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
"Root=1-58337364-23a8c76965a2ef7629b185e3" "-" "-"
```

```
1 2018-07-02T22:22:48.364000Z "forward" "-" "-"
```

Lambda 함수에 대한 예제 항목

다음은 성공한 Lambda 함수 요청에 대한 예제 로그 항목입니다.

```
http 2018-11-30T22:23:00.186641Z app/my-loadbalancer/50dc6c495c0c9188
192.168.131.39:2817 - 0.000 0.001 0.000 200 200 34 366
"GET http://www.example.com:80/ HTTP/1.1" "curl/7.46.0" - -
arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
"Root=1-58337364-23a8c76965a2ef7629b185e3" "-" "-"
0 2018-11-30T22:22:48.364000Z "forward" "-" "-"
```

다음은 실패한 Lambda 함수 요청에 대한 예제 로그 항목입니다.

```
http 2018-11-30T22:23:00.186641Z app/my-loadbalancer/50dc6c495c0c9188
192.168.131.39:2817 - 0.000 0.001 0.000 502 - 34 366
"GET http://www.example.com:80/ HTTP/1.1" "curl/7.46.0" - -
arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
"Root=1-58337364-23a8c76965a2ef7629b185e3" "-" "-"
0 2018-11-30T22:22:48.364000Z "forward" "-" "LambdaInvalidResponse"
```

버킷 권한

액세스 로그를 활성화할 때는 반드시 액세스 로그에 대한 S3 버킷을 지정해야 합니다. 버킷은 다음 요구 사항을 충족해야 합니다.

요구 사항

- 버킷은 로드 밸런서와 같은 리전에 있어야 합니다.
- 버킷에 대한 액세스 로그 쓰기 권한을 Elastic Load Balancing에 부여하는 버킷 정책을 이 버킷이 보유해야 합니다. 버킷 정책은 버킷에 대한 액세스 권한을 정의하기 위해 액세스 정책 언어로 작성된 JSON 문의 집합입니다. 각 문에는 단일 권한에 대한 정보와 일련의 요소들이 포함되어 있습니다.

다음 옵션 중 하나를 사용하여 S3 버킷을 액세스 로그에 대해 준비하십시오.

옵션

- 버킷을 생성해야 하고 콘솔을 사용하여 로깅 액세스를 활성화할 계획인 경우 액세스 로그 활성화 (p. 90)로 건너뛰고 옵션을 선택하여 콘솔을 통해 버킷 및 버킷 정책을 생성할 수 있습니다.
- 액세스 로그를 위한 버킷을 생성하고 AWS CLI 또는 API를 사용하는 경우에는 다음 절차를 통해 버킷을 생성하고 필요한 버킷 정책을 수동으로 추가하십시오.
- 액세스 로그를 위한 버킷을 이미 보유한 경우에는 다음 절차의 1단계에 따라 Amazon S3 콘솔을 연 후 4단계로 건너뛰어 버킷 정책을 추가하거나 업데이트하십시오.

필요한 권한을 가진 Amazon S3 버킷을 생성하려면

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. [기존 버킷을 사용하도록 건너뛰기] 버킷 만들기를 선택하십시오.
3. [기존 버킷을 사용하도록 건너뛰기] 버킷 만들기 대화 상자에서 다음을 수행하십시오.

    a. [Bucket Name]에서 버킷 이름을 입력합니다(예: `my-loadbalancer-logs`). 선택한 이름은 Amazon S3에 있는 어떤 기존 버킷 이름과도 중복되지 않아야 합니다. 일부 리전에서는 버킷 이름에 대한 추가 제한이 있을 수 있습니다. 자세한 내용은 Amazon Simple Storage Service 개발자 가이드의 Bucket Restrictions and Limitations를 참조하십시오.

b. [Region]에서 로드 밸런서를 생성한 리전을 선택합니다.

c. Create를 선택합니다.

4. 버킷을 선택한 다음 [Permissions]를 선택합니다.
5. [Bucket Policy]를 선택합니다. 버킷에 이미 정책이 연결되어 있는 경우 필수 구문을 기존 정책에 추가 할 수 있습니다.
6. [Policy generator]를 선택합니다. [AWS Policy Generator] 페이지에서 다음을 수행합니다.

   a. Select Type of Policy에 대해 S3 Bucket Policy를 선택합니다.

   b. [Effect]에서 [Allow]를 선택합니다.

   c. Principal에서 다음 AWS 계정 ID 중 하나를 지정하여 Elastic Load Balancing에 S3 버킷에 대한 액세스 권한을 부여하십시오. 로드 밸런서 및 버킷을 위한 리전에 해당되는 계정 ID를 사용합니다.

리전	리전 이름	탄력적 로드 밸런싱 계정 ID입니다.
us-east-1	미국 동부(버지니아 북부)	127311923021
us-east-2	미국 동부(오하이오)	033677994240
us-west-1	미국 서부(캘리포니아 북부 지역)	027434742980
us-west-2	미국 서부(오레곤)	797873946194
ca-central-1	캐나다(중부)	985666609251
eu-central-1	EU(프랑크푸르트)	054676820928
eu-west-1	EU(아일랜드)	156460612806
eu-west-2	EU(런던)	652711504416
eu-west-3	EU(파리)	009996457667
eu-north-1	EU(스톡홀름)	897822967062
ap-east-1	아시아 태평양(홍콩)	754344448648
ap-northeast-1	아시아 태평양(도쿄)	582318560864
ap-northeast-2	아시아 태평양(서울)	600734575887
ap-northeast-3	아시아 태평양(오사카-로컬)	383597477331
ap-southeast-1	아시아 태평양(싱가포르)	114774131450
ap-southeast-2	아시아 태평양(시드니)	783225319266
ap-south-1	아시아 태평양(뭄바이)	718504428378
sa-east-1	남아메리카(상파울루)	507241528517
us-gov-west-1*	AWS GovCloud (US-West)	048591011584
us-gov-east-1*	AWS GovCloud(US-East)	190560391635

리전	리전 이름	탄력적 로드 밸런싱 계정 ID입니다.
cn-north-1*	중국(베이징)	638102146993
cn-northwest-1*	중국(닝샤)	037604701340

* 이 리전에는 별도의 계정이 필요합니다. 자세한 내용은 AWS GovCloud(미국 서부) 및 중국(베이징)을(를) 참조하십시오.

d. 작업에서 PutObject를 선택하여 Elastic Load Balancing가 S3 버킷에 객체를 저장할 수 있습니다.

e. Amazon 리소스 이름(ARN)에서 S3 버킷의 ARN을 다음 형식으로 입력합니다. [*aws-account-id*]에서 로드 밸런서를 소유하고 있는 AWS 계정의 ID(예: *123456789012*)를 지정합니다. 계정 ID에 와일드카드를 지정하면 다른 계정이 사용자 버킷에 액세스 로그를 쓸 수 있으니 지정하지 마십시오. 여러 계정의 로드 밸런서에서 단일 버킷을 사용하여 액세스 로그를 저장하려면 각 ARN에서 해당하는 AWS 계정 ID를 사용하여 버킷 정책에서 계정당 하나의 ARN을 지정하십시오.

```
arn:aws:s3:::bucket/prefix/AWSLogs/aws-account-id/*
```

us-gov-west-1 리전을 사용 중인 경우 ARN에 arn:aws 대신 arn:aws-us-gov를 지정합니다.

f. [Add Statement], [Generate Policy]를 선택합니다. 정책 문서는 다음과 같이 표시됩니다.

```
{
  "Id": "Policy1429136655940",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1429136633762",
      "Action": [
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::my-loadbalancer-logs/my-app/AWSLogs/123456789012/
*",
      "Principal": {
        "AWS": [
          "797873946194"
        ]
      }
    }
  ]
}
```

g. 버킷 정책을 새로 생성하는 경우에는 전체 정책 문서를 복사한 다음 [Close]를 선택합니다.

기존 버킷 정책을 편집하는 경우에는 정책 문서(Statement요소의 [and] 사이에 있는 텍스트)에서 새 문을 복사하고 [Close]를 선택합니다.

7. Amazon S3 콘솔로 이동한 후 해당 정책을 텍스트 영역에 적절히 붙여 넣으십시오.
8. Save를 선택합니다.

액세스 로그 활성화

로드 밸런서에 대한 액세스 로그를 활성화할 때는 로드 밸런서가 로그를 저장할 S3 버킷의 이름을 지정해야 합니다. 이 버킷은 로드 밸런서와 같은 리전에 위치해야 하고, Elastic Load Balancing에 버킷에 액세스 로그 쓰기 권한을 부여하는 버킷 정책을 가지고 있어야 합니다. 로드 밸런서를 소유한 계정과 다른 계정으로 버킷을 소유할 수 있습니다.

콘솔을 이용하여 액세스 로그를 활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 [Load Balancers]를 클릭합니다.
3. 로드 밸런서를 선택합니다.
4. [Description] 탭에서 [Edit attributes]를 선택합니다.
5. [Edit load balancer attributes] 페이지에서 다음 작업을 수행합니다.
   a. [Enable access logs]를 선택합니다.
   b. S3 location]에 접두사를 포함하는 S3 버킷의 이름을 입력합니다(예: `my-loadbalancer-logs/my-app`). 기존 버킷의 이름이나 새 버킷의 이름을 지정할 수 있습니다. 기존 버킷을 지정하는 경우, 해당 버킷을 소유해야 하고 필요한 버킷 정책을 구성해야 합니다.
   c. (선택 사항) 버킷이 존재하지 않는 경우에는 [Create this location for me]를 선택합니다. Amazon S3의 모든 기존 버킷 이름에 대해 고유한 이름을 지정해야 하고 DNS 이름 지정 규칙을 따릅니다. 자세한 내용은 Amazon Simple Storage Service 개발자 가이드의 버킷 이름 지정 규칙을 참조하십시오.
   d. Save를 선택합니다.

AWS CLI를 이용하여 액세스 로그를 활성화하려면

modify-load-balancer-attributes 명령을 사용합니다.

S3 버킷에서 Elastic Load Balancing가 테스트 파일을 생성했는지 확인하려면

로드 밸런서에 대해 액세스 로그가 활성화되면 Elastic Load Balancing는 버킷 정책이 필요한 권한을 지정하도록 S3 버킷을 확인하고 테스트 파일을 생성합니다. 콘솔을 사용하여 테스트 파일이 생성되었는지 확인할 수 있습니다. 테스트 파일은 실제 액세스 로그 파일이 아니기 때문에 예제 레코드가 포함되어 있지 않습니다.

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. [All Buckets]에서 S3 버킷을 선택합니다.
3. 테스트 로그 파일을 탐색합니다. 경로는 다음과 같습니다.

```
my-bucket/prefix/AWSLogs/123456789012/ELBAccessLogTestFile
```

액세스 로그에 대해 S3 버킷을 관리하려면

액세스 로그를 활성화한 경우, 해당 액세스 로그를 포함하는 버킷을 삭제하기 전에 액세스 로그를 비활성화해야 합니다. 이렇게 하지 않으면 이름이 동일한 새로운 버킷이 있고, 필요한 버킷 정책이 다른 AWS 계정에 생성된 경우, Elastic Load Balancing가 내 로드 밸런서의 액세스 로그를 이 새로운 버킷에 쓸 수 있습니다.

액세스 로그 비활성화

언제든지 로드 밸런서에 대한 액세스 로그를 비활성화할 수 있습니다. 액세스 로그를 비활성화하면 액세스 로그는 사용자가 삭제할 때까지 S3 버킷에 남아 있습니다. 자세한 내용은 Amazon Simple Storage Service 콘솔 사용 설명서에서 버킷을 사용한 작업을 참조하십시오.

콘솔을 이용하여 액세스 로그를 비활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 [Load Balancers]를 클릭합니다.
3. 로드 밸런서를 선택합니다.
4. [Description] 탭에서 [Edit attributes]를 선택합니다.
5. [Edit load balancer attributes] 페이지에서 [Enable access logs]를 선택 취소합니다.
6. Save를 선택합니다.

AWS CLI를 이용하여 액세스 로그를 비활성화하려면

modify-load-balancer-attributes 명령을 사용합니다.

액세스 로그 파일 처리

액세스 로그 파일은 압축이 됩니다. Amazon S3; 콘솔을 사용해 파일을 열면 압축이 해제되고 정보가 표시됩니다. 파일을 다운로드하는 경우에는 압축을 해제해야 정보를 볼 수 있습니다.

웹 사이트에서 요청이 많은 경우에는 로드 밸런서가 수 기가바이트의 데이터로 로그 파일을 생성할 수 있습니다. 라인별 처리로는 이렇게 대량의 데이터를 처리할 수 없습니다. 따라서 병렬 처리 솔루션을 제공하는 분석 도구를 사용해야 할 수 있습니다. 예를 들어, 다음과 같은 분석 도구를 사용하여 액세스 로그를 분석 및 처리할 수 있습니다.

- Amazon Athena은 Amazon S3에서 표준 SQL을 사용하여 데이터를 쉽게 분석할 수 있는 대화형 쿼리 서비스입니다. 자세한 내용은 Amazon Athena 사용 설명서에서 Application Load Balancer 로그 쿼리를 참조하십시오.
- Loggly
- Splunk
- Sumo Logic

Application Load Balancer를 위한 요청 추적

요청 추적을 사용하여 클라이언트에서 대상 또는 기타 서비스로 가는 HTTP 요청을 추적할 수 있습니다. 클라이언트에서 요청을 받으면 로드 밸런서는 대상에 요청을 전달하기 전에 X-Amzn-Trace-Id 헤더를 추가 또는 업데이트합니다. 로드 밸런서와 대상 간의 서비스 또는 애플리케이션도 이 헤더를 추가 또는 업데이트할 수 있습니다.

액세스 로그를 활성화하면 X-Amzn-Trace-Id 헤더의 콘텐츠가 기록됩니다. 자세한 내용은 애플리케이션 로드 밸런서를 위한 액세스 로그 (p. 81) 섹션을 참조하십시오.

구문

X-Amzn-Trace-Id 헤더에는 다음 형식을 가진 필드가 포함되어 있습니다.

```
Field=version-time-id
```

필드

필드의 이름입니다. 지원되는 값은 `Root` 및 `Self`입니다.

애플리케이션은 자체 용도로 임의 필드를 추가할 수 있습니다. 로드 밸런서는 이들 필드를 보관은 하지만 사용하지는 않습니다.

version

버전 번호입니다.

시간

epoch 시간(초)입니다.

id

트레이스 식별자입니다.

예제

X-Amzn-Trace-Id 헤더가 들어오는 요청에 존재하지 않는 경우에는 로드 밸런서가 `Root` 필드를 가진 헤더를 생성하고 요청을 전달합니다. 예:

```
X-Amzn-Trace-Id: Root=1-67891233-abcdef012345678912345678
```

X-Amzn-Trace-Id 헤더가 존재하고 `Root` 필드를 가지고 있는 경우에는 로드 밸런서가 `Self` 필드를 삽입하고 요청을 전달합니다. 예:

```
X-Amzn-Trace-Id: Self=1-67891234-12456789abcdef012345678;Root=1-67891233-
abcdef012345678912345678
```

애플리케이션이 `Root` 필드와 사용자 지정 필드를 가진 헤더를 추가하는 경우에는 로드 밸런서가 두 필드를 모두 보관하고 `Self` 필드를 삽입한 다음 요청을 전달합니다.

```
X-Amzn-Trace-Id: Self=1-67891234-12456789abcdef012345678;Root=1-67891233-
abcdef012345678912345678;CalledFrom=app
```

X-Amzn-Trace-Id 헤더가 존재하고 `Self` 필드를 가지고 있는 경우에는 로드 밸런서가 `Self` 필드의 값을 업데이트합니다.

제한

- 로드 밸런서는 응답을 수신할 때가 아니라 요청을 수신할 때 헤더를 업데이트합니다.
- HTTP 헤더가 7 KB보다 크면 로드 밸런서는 `Root` 필드를 가진 X-Amzn-Trace-Id 헤더를 재작성합니다.
- WebSockets에서는 업그레이드 요청이 성공할 때까지만 추적이 가능합니다.

AWS CloudTrail을 사용하여 Application Load Balancer에 대한 API 호출 로깅

Elastic Load Balancing은 AWS CloudTrail에서 사용자, 역할 또는 AWS 서비스가 수행한 작업에 대한 레코드를 제공하는 서비스인 Elastic Load Balancing과 통합됩니다. CloudTrail은 Elastic Load Balancing에 대한 모든 API 호출을 이벤트로 캡처합니다. 캡처되는 호출에는 AWS Management 콘솔의 호출과 Elastic Load Balancing API 작업에 대한 호출이 포함됩니다. 추적을 생성하면 Elastic Load Balancing에 대한 이벤트를 비롯하여 CloudTrail 이벤트를 Amazon S3 버킷으로 지속적으로 배포할 수 있습니다. 추적을 구성하지 않은 경우 Event history(이벤트 기록)에서 CloudTrail 콘솔의 최신 이벤트를 볼 수도 있습니다. CloudTrail에서 수집하는 정보를 사용하여 Elastic Load Balancing에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

CloudTrail에 대한 자세한 내용은 AWS CloudTrail User Guide 단원을 참조하십시오.

클라이언트가 로드 밸런서에 요청을 할 때와 같이 로드 밸런서를 위한 기타 작업을 모니터링하려면 액세스 로그를 사용하십시오. 자세한 정보는 애플리케이션 로드 밸런서를 위한 액세스 로그 (p. 81) 단원을 참조하십시오.

CloudTrail의 Elastic Load Balancing 정보

CloudTrail은 계정 생성 시 AWS 계정에서 활성화됩니다. Elastic Load Balancing에서 활동이 수행되면 해당 활동은 이벤트 기록에서 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. AWS 계정에서 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 내용은 CloudTrail 이벤트 기록에서 이벤트 보기를 참조하십시오.

Elastic Load Balancing 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려는 경우 추적을 생성합니다. 추적은 CloudTrail이 Amazon S3 버킷으로 로그 파일을 전송할 수 있도록 합니다. 콘솔에서 추적을

생성하면 기본적으로 모든 AWS 리전에 추적이 적용됩니다. 추적은 AWS 파티션에 있는 모든 리전의 이벤트를 로깅하고 지정된 Amazon S3 버킷으로 로그 파일을 전송합니다. 또는 CloudTrail 로그에서 수집된 이벤트 데이터를 추가 분석 및 처리하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 정보는 다음을 참조하십시오.

- 추적 생성 개요
- CloudTrail 지원 서비스 및 통합
- CloudTrail에 대한 Amazon SNS 알림 구성
- 여러 리전에서 CloudTrail 로그 파일 받기 및 여러 계정에서 CloudTrail 로그 파일 받기

Application Load Balancer에 대한 모든 Elastic Load Balancing 작업을 CloudTrail에서 로깅하여 Elastic Load Balancing API 참조 버전 2015-12-01에 기록됩니다. 예를 들어, `CreateLoadBalancer` 및 `DeleteLoadBalancer` 작업을 호출하면 CloudTrail 로그 파일에 항목이 생성됩니다.

모든 이벤트 및 로그 항목에는 요청을 생성한 사용자에 대한 정보가 들어 있습니다. 자격 증명 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 AWS Identity and Access Management(IAM) 사용자 자격 증명으로 했는지 여부
- 역할 또는 연합된 사용자에 대한 임시 보안 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 다른 AWS 서비스에서 요청했는지 여부.

자세한 내용은 CloudTrail userIdentity 요소를 참조하십시오.

Elastic Load Balancing 로그 파일 항목 이해

추적은 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 제공할 수 있도록 해 주는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함됩니다. 이벤트는 어떤 소스로부터의 단일 요청을 나타내며 요청된 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 포함되어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 호출의 주문 스택 추적이 아니므로 특정 순서로 표시되지 않습니다.

로그 파일에는 Elastic Load Balancing API 호출 외에 AWS 계정의 모든 AWS API 호출 이벤트가 포함되어 있습니다. `elasticloadbalancing.amazonaws.com` 값이 있는 `eventSource` 요소를 확인하여 Elastic Load Balancing API에 대한 호출의 위치를 찾을 수 있습니다. `CreateLoadBalancer` 같은 특정 작업에 대한 레코드를 보려면 작업 이름이 있는 `eventName` 요소를 확인합니다.

다음은 AWS CLI를 사용하여 Application Load Balancer를 생성한 후 삭제한 사용자의 Elastic Load Balancing에 대한 CloudTrail 로그 레코드의 예입니다. `userAgent` 요소를 사용해 CLI를 식별할 수 있습니다. `eventName` 요소를 사용해 요청된 API 호출을 식별할 수 있습니다. 그리고 사용자(`Alice`)에 대한 정보는 `userIdentity` 요소를 보면 알 수 있습니다.

Example 예: CreateLoadBalancer


```
{
    "eventVersion": "1.03",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Alice"
    },
    "eventTime": "2016-04-01T15:31:48Z",
    "eventSource": "elasticloadbalancing.amazonaws.com",
    "eventName": "CreateLoadBalancer",
    "awsRegion": "us-west-2",
```

```
    "sourceIPAddress": "198.51.100.1",
    "userAgent": "aws-cli/1.10.10 Python/2.7.9 Windows/7 botocore/1.4.1",
    "requestParameters": {
        "subnets": ["subnet-8360a9e7","subnet-b7d581c0"],
        "securityGroups": ["sg-5943793c"],
        "name": "my-load-balancer",
        "scheme": "internet-facing"
    },
    "responseElements": {
        "loadBalancers":[{
            "type": "application",
            "loadBalancerName": "my-load-balancer",
            "vpcId": "vpc-3ac0fb5f",
            "securityGroups": ["sg-5943793c"],
            "state": {"code":"provisioning"},
            "availabilityZones": [
               {"subnetId":"subnet-8360a9e7","zoneName":"us-west-2a"},
               {"subnetId":"subnet-b7d581c0","zoneName":"us-west-2b"}
            ],
            "dNSName": "my-load-balancer-1836718677.us-west-2.elb.amazonaws.com",
            "canonicalHostedZoneId": "Z2P70J7HTTTPLU",
            "createdTime": "Apr 11, 2016 5:23:50 PM",
            "loadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/ffcddace1759e1d0",
            "scheme": "internet-facing"
        }]
    },
    "requestID": "b9960276-b9b2-11e3-8a13-f1ef1EXAMPLE",
    "eventID": "6f4ab5bd-2daa-4d00-be14-d92efEXAMPLE",
    "eventType": "AwsApiCall",
    "apiVersion": "2015-12-01",
    "recipientAccountId": "123456789012"
}
```

Example 예: DeleteLoadBalancer

```
{
    "eventVersion": "1.03",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Alice"
    },
    "eventTime": "2016-04-01T15:31:48Z",
    "eventSource": "elasticloadbalancing.amazonaws.com",
    "eventName": "DeleteLoadBalancer",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "198.51.100.1",
    "userAgent": "aws-cli/1.10.10 Python/2.7.9 Windows/7 botocore/1.4.1",
    "requestParameters": {
        "loadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/ffcddace1759e1d0"
    },
    "responseElements": null,
    "requestID": "349598b3-000e-11e6-a82b-298133eEXAMPLE",
    "eventID": "75e81c95-4012-421f-a0cf-babdaEXAMPLE",
    "eventType": "AwsApiCall",
    "apiVersion": "2015-12-01",
    "recipientAccountId": "123456789012"
}
```

애플리케이션 로드 밸런서 문제 해결

다음 정보는 Application Load Balancer와 관련된 문제를 해결하는 데 도움이 될 수 있습니다.

문제

등록된 대상은 서비스되지 않고 있습니다.

대상이 `InService` 상태로 들어가는 데 예상보다 시간이 오래 걸릴 경우 상태 확인에 실패할 수 있습니다. 한번이라도 상태 확인을 통과할 때까지 대상이 서비스되지 않습니다. 자세한 내용은 대상 그룹에 대한 상태 확인 (p. 56) 섹션을 참조하십시오.

인스턴스가 상태 확인에 실패하고 있는지 확인한 다음, 다음을 점검합니다.

보안 그룹이 트래픽을 허용하지 않음

인스턴스에 연결된 보안 그룹은 반드시 상태 확인 포트와 상태 확인 프로토콜을 사용하여 로드 밸런서에서의 트래픽을 허용해야 합니다. 로드 밸런서 보안 그룹에서의 모든 트래픽을 허용할 수 있도록 인스턴스 보안 그룹에 규칙을 추가할 수 있습니다. 또한 로드 밸런서를 위한 보안 그룹은 반드시 인스턴스로의 트래픽을 허용해야 합니다.

ACL(액세스 제어 목록)이 트래픽을 허용하지 않음

인스턴스를 위해 서브넷에 연결된 네트워크 ACL은 상태 확인 포트에서 인바운드 트래픽을, 휘발성 포트(1024-65535)에서 아웃바운드 트래픽을 허용해야 합니다. 로드 밸런서 노드를 위해 서브넷에 연결된 네트워크 ACL은 휘발성 포트에서 인바운드 트래픽을, 상태 확인 및 휘발성 포트에서 아웃바운드 트래픽을 허용해야 합니다.

핑 경로가 존재하지 않습니다.

상태 확인을 위한 대상 페이지를 생성하고 이것의 경로를 핑 경로로 지정합니다.

유휴 연결 제한 시간

먼저, 대상의 프라이빗 IP 주소와 상태 확인 프로토콜을 사용하여 네트워크 내에서 직접 대상을 연결할 수 있는지 확인합니다. 연결이 불가능하면 인스턴스가 과도하게 사용되고 있는지 확인하고, 이 인스턴스가 처리할 요청이 너무 많은 경우 대상 그룹에 더 많은 대상을 추가합니다. 연결이 가능한 경우, 상태 확인 시간 제한이 시작되기 전에 대상 페이지가 응답을 하지 않을 수 있습니다. 상태 확인을 위해 더 간단한 대상 페이지를 선택하거나 상태 확인 설정을 조정합니다.

대상이 성공적 응답 코드를 반환하지 않음

성공 코드는 200으로 기본 설정되어 있지만, 상태 확인을 구성할 때 선택에 따라 성공 코드를 추가적으로 지정할 수 있습니다. 성공 코드가 로드 밸런서가 기대하고 있는 것인지, 그리고 성공 시 이들 코드를 반환하도록 애플리케이션이 구성되어 있는지 확인합니다.

클라이언트가 인터넷 경계 로드 밸런서에 연결이 불가능

로드 밸런서가 요청에 응답하지 않는 경우에는 다음을 점검하십시오.

인터넷 경계 로드 밸런서가 프라이빗 서브넷에 연결

로드 밸런서를 위한 퍼블릭 서브넷을 지정했는지 확인합니다. 퍼브릭 서브넷은 가상 프라이빗 클라우드(VPC)를 위한 인터넷 게이트웨이로 연결되는 경로를 가지고 있습니다.

보안 그룹이나 네트워크 ACL이 트래픽을 허용하지 않음

로드 밸런서를 위한 보안 그룹과 로드 밸런서 서브넷을 위한 모든 네트워크 ACL은 클라이언트에서의 인바운드 트래픽과 리스너 포트의 클라이언트로의 아웃바운드 트래픽을 허용해야 합니다.

로드 밸런서가 비정상 상태 대상에 요청을 전송

대상 그룹에 정상인 대상이 최소 하나 이상 있는 경우 로드 밸런서는 정상인 대상으로만 요청을 라우팅합니다. 대상 그룹에 비정상인 대상만 포함되는 경우 로드 밸런서는 비정상인 대상으로 요청을 라우팅합니다.

로드 밸런서가 HTTP 오류 코드를 생성

로드 밸런서는 다음과 같은 HTTP 오류 코드를 생성합니다. 로드 밸런서는 클라이언트에 HTTP 코드를 전송하고 액세스 로그에 대한 요청을 저장하며, `HTTPCode_ELB_4XX_Count` 또는 `HTTPCode_ELB_5XX_Count` 지표를 증분합니다.

오류

HTTP 400: 잘못된 요청

가능한 원인:

- 클라이언트가 HTTP 사양을 충족하지 않는 잘못된 형식의 요청을 전송했습니다.

- 클라이언트가 사용한 HTTP CONNECT 방법은 Application Load Balancer에서 지원하지 않습니다.
- 요청 헤더가 요청 줄당 16K, 단일 헤더당 16K 또는 전체 헤더에서 64K를 초과했습니다.

HTTP 401: 권한 없음

사용자를 인증하도록 리스너 규칙을 구성했습니다. 인증되지 않은 사용자를 거부하도록 `OnUnauthenticatedRequest`를 구성했거나 IdP가 액세스를 거부했습니다.

HTTP 403: 금지됨

Application Load Balancer에 대한 요청을 모니터링하기 위해 AWS WAF 웹 액세스 제어 목록(웹 ACL)을 구성하여 요청이 차단되었습니다.

HTTP 408: Request Timeout

클라이언트가 유휴 제한 시간 만료 전에 데이터를 전송하지 않았습니다. TCP 연결 유지를 전송해도 이 시간 제한을 막지 못합니다. 각 유휴 제한 시간이 지나기 전에 최소 1바이트의 데이터를 전송하십시오. 필요한 만큼 유휴 제한 시간의 길이를 늘립니다.

HTTP 413: 페이로드가 너무 큼

대상이 Lambda 함수이고 요청 본문이 1MB를 초과합니다.

HTTP 414: URI가 너무 김

요청 URL 또는 쿼리 문자열 파라미터가 너무 큽니다.

HTTP 460

로드 밸런서가 클라이언트에서 요청을 수신했지만, 유휴 제한 시간이 종료되기 전에 클라이언트가 로드 밸런서와의 연결을 종료했습니다.

클라이언트 제한 시간이 로드 밸런서의 유휴 제한 시간보다 큰지 확인합니다. 클라이언트 제한 시간이 끝나기 전에 대상이 클라이언트에 응답을 제공하는지 확인하거나, 클라이언트가 제한 시간을 지원할 경우 로드 밸런서의 유휴 제한 시간에 맞게 클라이언트 제한 시간을 늘립니다.

HTTP 463

로드 밸런서가 IP 주소가 30개가 넘는 X-Forwarded-For 요청 헤더를 받았습니다.

HTTP 500: 내부 서버 오류

가능한 원인:

- AWS WAF 웹 액세스 제어 목록(웹 ACL)을 구성했으며 웹 ACL 규칙을 실행하는 데 오류가 발생했습니다.
- 사용자를 인증하도록 리스너 규칙을 구성했지만, 다음 중 하나가 true입니다.
  - 로드 밸런서가 IdP 토큰 엔드포인트 또는 IdP 사용자 정보 엔드포인트와 통신할 수 없습니다. 로드 밸런서의 보안 그룹과 VPC의 네트워크 ACL이 이러한 엔드포인트에 대한 발신 액세스를 허용하는지 확인합니다. VPC에서 인터넷에 액세스할 수 있는지 확인합니다. 내부 로드 밸런서가 있는 경우, NAT 게이트웨이를 사용하여 인터넷 액세스를 활성화하십시오.
  - IdP에서 반환된 클레임 크기가 로드 밸런서에서 지원되는 최대 크기를 초과했습니다.

- 클라이언트가 호스트 헤더 없이 HTTP/1.0 요청을 제출했으며, 로드 밸런서가 리디렉션 URL을 생성하지 못했습니다.
- 클라이언트가 HTTP 프로토콜 없이 요청을 제출했으며, 로드 밸런서가 리디렉션 URL을 생성하지 못했습니다.
- 요청된 범위가 ID 토큰을 반환하지 않습니다.

HTTP 501: 구현되지 않음

로드 밸런서가 미지원 값이 포함된 Transfer-Encoding 헤더를 받았습니다. Transfer-Encoding에서 지원하는 값은 `chunked` 및 `identity`입니다. 대신 Content-Encoding 헤더를 사용할 수 있습니다.

HTTP 502: 잘못된 게이트웨이

가능한 원인:

- 연결 설정을 시도하는 동안 로드 밸런서가 대상에서 TCP RST를 수신했습니다.
- 연결을 설정하려고 했을 때 로드 밸런서가 "ICMP 대상에 연결할 수 없음(호스트에 연결할 수 없음"과 같이 대상으로부터 예기치 않은 응답을 받았습니다. 로드 밸런서 서브넷부터 대상 포트의 대상에 이르기까지 트래픽 허용 여부를 점검하십시오.
- 로드 밸런서가 대상에 대해 대기 중인 요청을 가지고 있는 상태에서 대상이 TCP RST 또는 TCP FIN과의 연결을 종료했습니다. 대상의 연결 유지 기간이 로드 밸런서의 유휴 제한 시간 값보다 짧은지 확인합니다.
- 대상 응답이 잘못된 형식이거나 유효하지 않은 HTTP를 포함하고 있습니다.
- 로드 밸런서가 대상에 연결할 때 SSL 핸드셰이크 오류 또는 SSL 핸드셰이크 제한 시간(10초)가 발생했습니다.
- 등록 취소된 대상에 의해 처리 중인 요청에 대해 경과된 등록 취소 지연 시간 오래 걸리는 작업이 완료될 수 있도록 지연 기간을 늘립니다.
- 대상이 Lambda 함수이고 응답 본문이 1MB를 초과합니다.
- 대상이 구성된 제한 시간에 도달하기 전에 응답하지 않은 Lambda 함수입니다.

HTTP 503: 서비스 사용 불가

로드 밸런서의 대상 그룹에 등록된 대상이 없습니다.

HTTP 504: 게이트웨이 제한 시간

가능한 원인:

- 연결 제한 시간이 만료(10초)되기 전에 로드 밸런서가 대상에 대한 연결을 설정하지 못했습니다.
- 로드 밸런서가 대상에 대한 연결을 설정했지만, 유휴 제한 시간이 끝나기 전에 대상이 응답을 하지 않았습니다.
- 서브넷의 네트워크 ACL이 휘발성 포트(1024-65535)에서 대상에서 로드 밸런서 노드로의 트래픽을 허용하지 않았습니다.
- 대상이 개체 몸체보다 큰 콘텐츠 길이 헤더를 반환합니다. 로드 밸런서가 놓친 바이트를 기다리는 도중 시간이 초과되었습니다.
- 대상이 구성된 최대 제한 시간에 도달하기 전에 응답하지 않은 Lambda 함수입니다.

HTTP 561: 권한 없음

사용자를 인증하도록 리스너 규칙을 구성했지만, 사용자를 인증할 때 IdP가 오류 코드를 반환했습니다.

대상이 HTTP 오류 코드를 생성

로드 밸런서가 HTTP 오류를 포함하여 대상에서 클라이언트로 유효한 HTTP 응답을 전달합니다. 대상이 생성한 HTTP 오류 코드는 `HTTPCode_Target_4XX_Count` and `HTTPCode_Target_5XX_Count` 지표에 기록이 됩니다.

Application Load Balancer에 대한 제한

Application Load Balancer에 대한 현재 제한을 보려면 Amazon EC2 콘솔에서 제한 페이지를 확인하거나 describe-account-limits(AWS CLI) 명령을 사용합니다. 제한 증가를 요청하려면 Elastic Load Balancing 제한 양식을 사용하십시오.

AWS 계정에는 Application Load Balancer와 관련된 다음과 같은 제한이 있습니다.

리전 제한

- 리전당 로드 밸런서: 20
- 리전당 대상 그룹: 3000

로드 밸런서 제한

- 로드 밸런서당 리스너: 50
- 로드 밸런서당 대상: 1000
- 로드 밸런서당 가용 영역당 서브넷: 1
- 로드 밸런서당 보안 그룹: 5
- 로드 밸런서당 규칙(기본 규칙은 계산하지 않음): 100
- 로드 밸런서당 인증서(기본 인증서는 포함되지 않음): 25
- 로드 밸런서당 대상을 등록할 수 있는 횟수: 100

대상 그룹 제한

- 대상 그룹당 로드 밸런서: 1
- 대상 그룹당 대상(인스턴스 또는 IP 주소): 1000
- 대상 그룹당 대상(Lambda 함수): 1

규칙 제한

- 규칙당 일치 평가: 5
- 규칙당 와일드카드: 5
- 규칙당 작업: 2(하나는 옵션 인증 작업, 하나는 필수 작업)

Application Load Balancer 문서 기록

다음 표에서는 Application Load Balancer 릴리스를 설명합니다.

기능	설명	날짜
고급 라우팅 요청	이 릴리스에서는 표준 및 사용자 지정 HTTP 헤더 및 메서드, 쿼리 파라미터, 소스 IP 주소를 기반으로 하는 리스너 규칙의 조건을 추가하여 호스트 헤더 및 경로 기반 라우팅에 대한 기존 지원을 확장합니다. 자세한 내용은 규칙 조건 형식 (p. 28) 단원을 참조하십시오.	2019년 3월 27일
대상으로서 Lambda 함수	이 릴리스에는 Lambda 함수를 대상으로 등록하기 위한 지원이 추가됩니다. 자세한 내용은 대상으로서 Lambda 함수 (p. 62) 단원을 참조하십시오.	2018년 11월 29일
고정 응답 작업	이 릴리스에는 로드 밸런서가 사용자 지정 HTTP 응답을 반환하기 위한 지원이 추가되었습니다. 자세한 내용은 고정 응답 작업 (p. 26) 단원을 참조하십시오.	2018년 7월 25일
리디렉션 작업	이 릴리스에는 로드 밸런서가 요청을 다른 URL로 리디렉션하기 위한 지원이 추가되었습니다. 자세한 정보는 리디렉션 작업 (p. 26) 단원을 참조하십시오.	2018년 7월 25일
FS 및 TLS 1.2 보안 정책	이 릴리스에는 FS(Forward Secrecy) 및 TLS 1.2 보안 정책이 추가되었습니다. 자세한 정보는 보안 정책 (p. 35) 단원을 참조하십시오.	2018년 6월 6일
인증 지원	이 릴리스에서는 요청을 라우팅하기 전에 기업 또는 소셜 자격 증명을 사용하여 애플리케이션의 사용자를 인증할 수 있도록 로드 밸런서에 대한 지원이 추가되었습니다. 자세한 내용은 Application Load Balancer를 사용하여 사용자 인증 (p. 43) 단원을 참조하십시오.	2018년 5월 30일
느린 시작 모드	이 릴리스는 느린 시작 모드(로드 밸런서가 워밍업 동안 새로 등록된 대상으로 보내는 요청 공유를 점차 증가시킴)를 위한 지원을 추	2018년 3월 24일

기능	설명	날짜
	가합니다. 자세한 내용은 느린 시작 모드 (p. 53) 단원을 참조하십시오.	
리소스 수준 권한	이 릴리스에서는 리소스 레벨 권한 및 태깅 조건 키에 대한 지원이 추가되었습니다. 자세한 내용은 Elastic Load Balancing 사용 설명서에서 인증 및 액세스 제어를 참조하십시오.	2018년 5월 10일
SNI 지원	이번 릴리스에는 SNI(Server Name Indication)에 대한 지원이 추가되었습니다. 자세한 내용은 SSL 인증서 (p. 33) 단원을 참조하십시오.	2017년 10월 10일
IP 주소를 대상으로 사용	이 릴리스에서는 IP 주소를 대상으로 등록에 대한 지원이 추가되었습니다. 자세한 내용은 대상 유형 (p. 50) 단원을 참조하십시오.	2017년 8월 31일
호스트 기반 라우팅	이 릴리스에서는 호스트 헤더의 호스트 이름을 기반으로 하는 라우팅 요청에 대한 지원이 추가되었습니다. 자세한 정보는 호스트 조건 (p. 30) 단원을 참조하십시오.	2017년 5월 4일
TLS 1.1 및 TLS 1.2 보안 정책	이 릴리스에는 TLS 1.1 및 TLS 1.2. 보안 정책이 추가되었습니다. 자세한 정보는 보안 정책 (p. 35) 단원을 참조하십시오.	2017년 2월 6일
IPv6 지원	이 릴리스에는 IPv6 주소에 대한 지원이 추가되었습니다. 자세한 내용은 IP 주소 유형 (p. 15) 섹션을 참조하십시오.	2017년 1월 25일
요청 추적	이 릴리스에는 요청 추적에 대한 지원이 추가되었습니다. 자세한 내용은 Application Load Balancer를 위한 요청 추적 (p. 92) 단원을 참조하십시오.	2016년 11월 22일
TargetResponseTime 지표 백분위수 지원	이 릴리스에는 Amazon CloudWatch가 지원하는 새로운 백분위수 통계 자료가 추가되었습니다. 자세한 내용은 Application Load Balancer 지표에 대한 통계 (p. 79) 섹션을 참조하십시오.	2016년 11월 17일
새로운 로드 밸런서 유형	이 Elastic Load Balancing 릴리스에는 Application Load Balancer가 도입되었습니다.	2016년 8월 11일