Application Load Balancers

Elastic Load Balancing

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Elastic Load Balancing: Application Load Balancers

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon의 상표 및 트레이드 드레스는 Amazon 외 제품 또는 서비스와 함께, Amazon 브랜드 이미지를 떨어뜨리거나 고객에게 혼동을 일으킬 수 있는 방식으로 사용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon 계열사, 관련 업체 또는 Amazon의 지원 업체 여부에 상관없이 해당 소유자의 자산입니다.

Table of Contents

Application Load Balancer란 무엇입니까? 1
Application Load Balancer 구성 요소 1
Application Load Balancer 개요 2
Classic Load Balancer에서 마이그레이션할 때의 이점 3
관련 서비스 4
요금 5
시작 6
시작하기 전 준비 사항 6
1단계: 대상 그룹 구성 6
2단계: 로드 밸런서 유형 선택 7
3단계: 로드 밸런서 및 리스너 구성 7
4단계: 로드 밸런서 테스트 8
5단계: (선택 사항) 로드 밸런서 삭제 9
사용 시작하기 AWS CLI 10
시작하기 전 준비 사항 10
로드 밸런서 생성 10
HTTPS 리스너 추가 12
경로 기반 라우팅 추가 13
로드 밸런서 삭제 13
Application Load Balancers 14
로드 밸런서를 위한 서브넷 15
가용 영역 서브넷 15
로컬 영역 서브넷 16
Outpost 서브넷 16
로드 밸런서 보안 그룹 17
로드 밸런서 상태 18
로드 밸런서 속성 18
IP 주소 유형 20
IPAM IP 주소 풀 21
로드 밸런서 연결 22
교차 영역 로드 밸런싱 22
DNS 이름 23
로드 밸런서 생성 24
1단계: 대상 그룹 구성 6

2단계: 대상 등록 25
3단계: 로드 밸런서 및 리스너 구성 26
4단계: 로드 밸런서 테스트 8
가용 영역 업데이트 30
보안 그룹 업데이트 31
권장 규칙 31
연결된 보안 그룹 업데이트 33
IP 주소 유형 업데이트 34
IPAM IP 주소 풀 업데이트 35
로드 밸런서 통합 35
Amazon Application Recovery Controller(ARC) 36
Amazon CloudFront + AWS WAF 38
AWS Global Accelerator 38
AWS Config 39
AWS WAF 39
로드 밸런서 속성 편집 40
연결 유휴 제한 시간 40
HTTP 클라이언트 연결 유지 기간 41
삭제 방지 42
Desync Mitigation Mode 43
Host header preservation 45
로드 밸런서에 태그 지정 47
로드 밸런서 삭제 48
리소스 맵 보기 49
리소스 맵 구성 요소 49
LCU 예약 50
예약 요청 51
예약 업데이트 또는 종료 52
예약 모니터링 52
리스너 및 규칙 54
리스너 구성 54
리스너 속성 55
리스너 규칙 57
기본 규칙 57
규칙 우선 순위 58
규칙 작업 58

규칙 조건 58
규칙 작업 유형 58
고정 응답 작업 59
전달 작업 59
리디렉션 작업 62
규칙 조건 형식 65
HTTP 헤더 조건 66
HTTP 요청 메서드 조건 67
호스트 조건 68
경로 조건 69
쿼리 문자열 조건 70
소스 IP 주소 조건 71
X-Forwarded 헤더 71
X-Forwarded-For 72
X-Forwarded-Proto 75
X-Forwarded-Port 76
HTTP 리스너 생성 76
사전 조건 76
HTTP 리스너 추가 76
SSL 인증서 77
기본 인증서 78
인증서 목록 78
인증서 갱신 79
보안 정책 79
TLS 보안 정책 81
FIPS 보안 정책 105
FS 지원 정책 120
HTTPS 리스너 생성 126
사전 조건 126
HTTPS 리스너 추가 126
리스너 규칙 업데이트 128
요구 사항 128
규칙 추가 129
규칙 편집 131
규칙 재정렬 132
규칙 삭제 133

HTTPS 리스너 업데이트 133
기본 인증서 교체 134
인증서 목록에 인증서 추가 134
인증서 목록에서 인증서 제거 135
보안 정책 업데이트 136
HTTP 헤더 수정 137
상호 TLS 인증 137
시작하기 전 준비 사항 138
HTTP 헤더 141
CA 제목 이름 알리기 142
연결 로그 143
상호 TLS 구성 143
트러스트 스토어 공유 148
사용자 인증 구성 153
OIDC 호환 IdP 사용 준비 154
Amazon Cognito 사용 준비 154
Amazon CloudFront 사용 준비 156
사용자 인증 구성 157
인증 흐름 159
사용자 클레임 인코딩 및 서명 확인 161
제한 시간 165
인증 로그아웃 166
리스너에 태그 지정 166
리스너 태그를 업데이트합니다 167
규칙 태그를 업데이트합니다 168
리스너 삭제 168
헤더 수정 169
mTLS/TLS 헤더 이름 바꾸기 169
응답 헤더 추가 171
헤더 비활성화 173
제한 사항 173
헤더 수정 활성화 173
대상 그룹 177
라우팅 구성 178
대상 유형 178
IP 주소 유형 180

프로토콜 버전 180
등록된 대상 182
대상 그룹 속성 182
라우팅 알고리즘 184
대상 그룹의 라우팅 알고리즘 수정 185
대상 그룹 상태 186
비정상 상태 작업 186
요구 사항 및 고려 사항 187
모니터링 188
예제 188
로드 밸런서에 대한 Route 53 DNS 장애 조치 사용 189
대상 그룹 생성 190
상태 설정 업데이트 192
상태 확인 구성 193
상태 확인 설정 194
대상 상태 196
상태 확인 사유 코드 197
대상 상태 확인 198
상태 확인 설정 업데이트 199
대상 그룹 속성 편집 199
등록 취소 지연 199
느린 시작 모드 200
교차 영역 로드 밸런싱 201
자동 대상 가중치(ATW) 204
고정 세션 207
대상 등록 214
대상 보안 그룹 214
공유 서브넷 215
대상 등록 또는 등록 취소 215
Lambda 함수를 대상으로 사용 218
Lambda 함수 준비 219
Lambda 함수에 대한 대상 그룹 생성 217
로드 밸런서에서 이벤트 수신 220
로드 밸런서에 응답 221
다중 값 헤더 222
상태 확인 활성화 225

Lambda 함수 등록 취소 ... 226
대상 그룹에 태그 지정 ... 227
대상 그룹 삭제 ... 228
로드 밸런서 모니터링 ... 229
CloudWatch 지표 ... 230
Application Load Balancer 지표 ... 230
Application Load Balancer의 지표 차원 ... 250
Application Load Balancer 지표에 대한 통계 ... 251
로드 밸런서에 대한 CloudWatch 지표 보기 ... 252
액세스 로그 ... 254
액세스 로그 파일 ... 255
액세스 로그 항목 ... 256
로그 항목 예제 ... 269
액세스 로그 파일 처리 ... 271
액세스 로그 활성화 ... 272
액세스 로그 비활성화 ... 282
연결 로그 ... 283
연결 로그 파일 ... 283
연결 로그 항목 ... 285
로그 항목 예제 ... 288
연결 로그 파일 처리 ... 289
연결 로그 활성화 ... 289
연결 로깅 비활성화 ... 299
요청 추적 ... 299
구문 ... 299
제한 사항 ... 301
로드 밸런서 문제 해결 ... 302
등록된 대상은 서비스되지 않고 있습니다. ... 302
클라이언트가 인터넷 경계 로드 밸런서에 연결할 수 없음 ... 304
사용자 지정 도메인으로 전송된 요청은 로드 밸런서에 수신되지 않음 ... 304
로드 밸런서로 전송된 HTTPS 요청은 “NET: :ERR_CERT_COMMON_NAME_INVALID”를 반환합니다. ... 305
로드 밸런서가 높은 처리 시간을 표시합니다 ... 305
로드 밸런서가 응답 코드 000을 보냅니다. ... 305
로드 밸런서가 HTTP 오류 코드를 생성 ... 305
HTTP 400: 잘못된 요청 ... 306

HTTP 401: 권한 없음 306
HTTP 403: 금지됨 307
HTTP 405: 허용되지 않은 메서드 307
HTTP 408: 요청 제한 시간 307
HTTP 413: 페이로드가 너무 큼 307
HTTP 414: URI가 너무 깊 307
HTTP 460 307
HTTP 463 308
HTTP 464 308
HTTP 500: 내부 서버 오류 308
HTTP 501: 구현되지 않음 309
HTTP 502: 잘못된 게이트웨이 309
HTTP 503: 서비스 사용 불가 309
HTTP 504: 게이트웨이 제한 시간 309
HTTP 505: 버전이 지원되지 않습니다. 310
HTTP 507: 스토리지 부족 310
HTTP 561: 권한 없음 310
대상이 HTTP 오류 코드를 생성 310
AWS Certificate Manager 인증서를 사용할 수 없음 310
여러 줄의 헤더는 지원되지 않습니다. 311
리소스 맵을 사용하여 비정상 대상 문제 해결 311
할당량 313
로드 밸런서 313
대상 그룹 314
규칙 314
트러스트 스토어 315
인증서 315
HTTP 헤더 315
Load Balancer서 용량 단위 316
문서 기록 317
..... cccxxiii

Application Load Balancer란 무엇입니까?

Elastic Load Balancing은 둘 이상의 가용 영역에서 EC2 인스턴스, 컨테이너, IP 주소 등 여러 대상에 걸쳐 수신되는 트래픽을 자동으로 분산합니다. 등록된 대상의 상태를 모니터링하면서 상태가 양호한 대상으로만 트래픽을 라우팅합니다. Elastic Load Balancing은 수신 트래픽이 시간이 지남에 따라 변경됨에 따라 로드 밸런서를 확장합니다. 대다수의 워크로드에 맞게 자동으로 조정할 수 있습니다.

Elastic Load Balancing은 다음 로드 밸런서를 지원합니다. Application Load Balancers, Network Load Balancers, Gateway Load Balancers 및 Classic Load Balancer 각자 필요에 따라 가장 적합한 로드 밸런서 유형을 선택할 수 있습니다. 이 안내서에서는 Application Load Balancer에 대해 설명합니다. 다른 로드 밸런서에 대한 자세한 내용은 Network Load Balancer 사용 설명서, Gateway Load Balancer 사용 설명서, Classic Load Balancer 사용 설명서를 참조하세요.

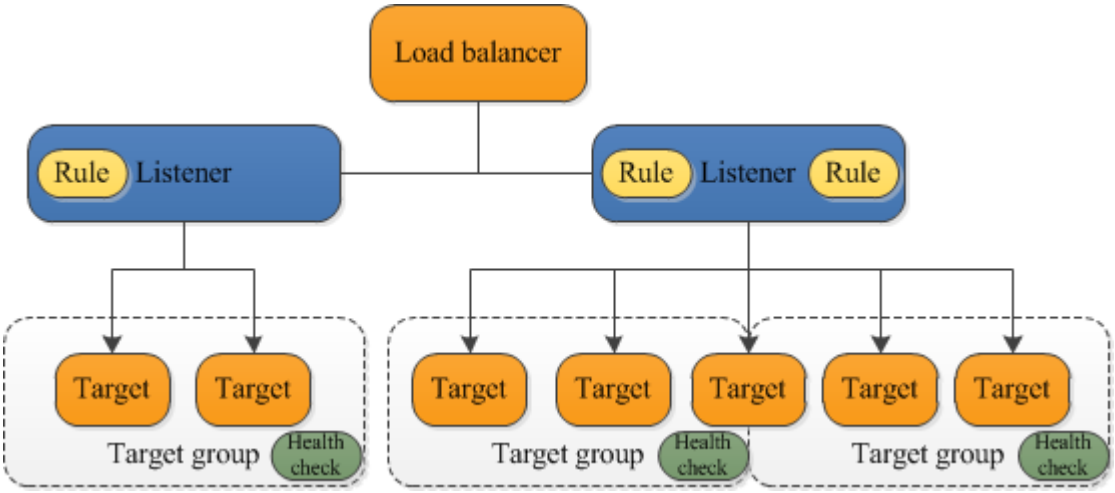
Application Load Balancer 구성 요소

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 구성한 프로토콜 및 포트를 사용하여 클라이언트의 연결 요청을 확인합니다. 리스너에 대해 정의한 규칙에 따라 로드 밸런서가 등록된 대상으로 요청을 라우팅하는 방법이 결정됩니다. 각 규칙은 우선 순위, 하나 이상의 작업, 하나 이상의 조건으로 구성됩니다. 규칙에 대한 조건이 충족되면 작업이 수행됩니다. 각 리스너에 대한 기본 규칙을 정의해야 하며, 필요에 따라 추가 규칙을 정의할 수 있습니다.

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 참조하세요.

- 로드 밸런서
- 리스너
- 대상 그룹

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 결정한 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. 대상 그룹 레벨에서 사용되는 라우팅 알고리즘을 구성할 수 있습니다. 기본 라우팅 알고리즘은 라운드 로빈입니다. 그 대신 최소 미해결 요청 라우팅 알고리즘을 지정할 수 있습니다.

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 Balancing은 로드 밸런서를 확장합니다. Elastic Load Balancing은 대다수의 워크로드에 맞게 자동으로 조정할 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 참조하세요.

Classic Load Balancer에서 마이그레이션할 때의 이점

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

- 경로 조건에 대한 지원. 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 조건에 대한 지원. 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.
- HTTP 헤더 조건 및 메서드, 쿼리 파라미터, 소스 IP 주소 등 요청의 필드를 기반으로 하는 라우팅을 지원합니다.
- 단일 EC2 인스턴스의 여러 애플리케이션으로 요청을 라우팅하는 것을 지원합니다. 인스턴스 또는 IP 주소를 각각 다른 포트에 있는 여러 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)는 태스크를 예약할 때 사용되지 않는 포트를 선택하고 이 포트를 사용하여 대상 그룹에 태스크를 등록할 수 있습니다. 이를 통해 클러스터를 효율적으로 사용할 수 있습니다.
- 상태 확인은 대상 그룹 수준에서 정의되고 많은 CloudWatch 지표가 대상 그룹 수준에서 보고되므로 각 서비스의 상태를 독립적으로 모니터링할 수 있습니다. Auto Scaling 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 기능을 참조하세요.

관련 서비스

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

- Amazon EC2 — 클라우드에서 애플리케이션을 실행할 수 있는 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 Balancing과 함께 Auto Scaling을 사용하는 경우, Auto Scaling이 시작한 인스턴스는 자동으로 대상 그룹에 등록되고 Auto Scaling이 종료하는 인스턴스는 자동으로 대상 그룹에서 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 Application Load Balancer용 SSL 인증서 단원을 참조하세요.
- Amazon CloudWatch — 로드 밸런서를 모니터링하고 필요에 따라 조치를 취할 수 있게 해줍니다. 자세한 내용은 Application Load Balancer의 CloudWatch 지표 단원을 참조하세요.
- Amazon ECS — EC2 인스턴스 클러스터에서 Docker 컨테이너를 실행, 중단 및 관리할 수 있게 해줍니다. 로드 밸런서를 구성하여 컨테이너에 트래픽을 라우팅할 수 있습니다. 자세한 내용은 Amazon Elastic Container Service 개발자 안내서에서 서비스 로드 밸런싱 세션을 참조하세요.
- AWS Global Accelerator - 애플리케이션의 가용성과 성능을 향상시킵니다. 액셀러레이터를 사용하여 하나 이상의 AWS 리전에서 여러 로드 밸런서에 트래픽을 분산합니다. 자세한 내용은 AWS Global Accelerator 개발자 안내서를 참조하세요.
- Route 53 - 컴퓨터 간 연결을 위해 사용되는 숫자 형식의 IP 주소(예: `192.0.2.1`)로 도메인 이름(예: `www.example.com`)을 변환하여 안정적이며 경제적인 방식으로 방문자를 웹 사이트로 연결합니다. AWS 에서는 리소스에 URL을 지정합니다(예: 로드 밸런서). 그러나 기억하기 쉬운 URL이 필요한 경우도 있습니다. 예를 들어 도메인 이름을 로드 밸런서로 매핑할 수 있습니다. 자세한 내용은 Amazon Route 53 개발자 안내서의 ELB 로드 밸런서로 트래픽 라우팅을 참조하세요.
- AWS WAF - Application Load Balancer와 AWS WAF 함께를 사용하여 웹 액세스 제어 목록(웹 ACL)의 규칙에 따라 요청을 허용하거나 차단할 수 있습니다. 자세한 내용은 AWS WAF 단원을 참조하십시오.

로드 밸런서와 통합된 서비스에 대한 정보를 보려면에서 로드 밸런서를 AWS Management Console 선택하고 통합 서비스 탭을 선택합니다.

요금

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

Application Load Balancer 시작하기

이 자습서에서는 웹 기반 인터페이스인를 통해 Application Load Balancer를 실습 AWS Management Console으로 소개합니다. 첫 번째 Application Load Balancer를 생성하려면 다음 단계를 완료합니다.

내용

- 시작하기 전 준비 사항
- 1단계: 대상 그룹 구성
- 2단계: 로드 밸런서 유형 선택
- 3단계: 로드 밸런서 및 리스너 구성
- 4단계: 로드 밸런서 테스트
- 5단계: (선택 사항) 로드 밸런서 삭제

일반적인 로드 밸런서 구성에 대한 데모는 Elastic Load Balancing 데모를 참조하세요.

시작하기 전 준비 사항

- 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: 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.

콘솔을 사용하여 대상 그룹을 구성하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.

3. 대상 그룹 생성을 선택합니다.
4. 기본 구성(Basic configuration) 아래에서 대상 유형(Target type)을 인스턴스로 유지합니다.
5. 대상 그룹 이름(Target group name)에 새로운 대상 그룹의 이름을 입력합니다.
6. 기본 프로토콜(HTTP) 및 포트(80)를 유지합니다.
7. 사용자의 인스턴스를 포함하는 VPC를 선택합니다. 프로토콜 버전을 HTTP1로 유지합니다.
8. Health checks(상태 확인)에는 기본 설정을 그대로 둡니다.
9. Next(다음)를 선택합니다.
10. 대상 등록(Register Targets) 페이지에서 다음 단계를 완료합니다. 로드 밸런서를 생성하기 위한 선택적 단계입니다. 그러나 로드 밸런서를 테스트하고 대상으로 트래픽을 라우팅하고 있는지 확인하려면 대상을 등록해야 합니다.

    a. 사용 가능한 인스턴스(Available instance)에서 인스턴스를 하나 이상 선택합니다.

    b. 기본 포트 80을 유지하고 아래에서 보류 중인 것으로 포함(Include as pending below)을 선택합니다.
11. 대상 그룹 생성을 선택합니다.

2단계: 로드 밸런서 유형 선택

A: Elastic Load Balancing은 여러 유형의 로드 밸런서를 지원합니다. 이 자습서에서는 Application Load Balancer를 생성합니다.

콘솔을 사용하여 Application Load Balancer를 생성하려면

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 Balancing에서 로드 밸런서를 선택합니다.
4. 로드 밸런서 생성(Create Load Balancer)을 선택합니다.
5. Application Load Balancer에서 생성을 선택합니다.

3단계: 로드 밸런서 및 리스너 구성

Application Load Balancer를 생성하려면 먼저 이름, 구성표 및 IP 주소 유형과 같은 로드 밸런서에 대한 기본 구성 정보를 제공해야 합니다. 그런 다음 네트워크와 하나 이상의 리스너에 대한 정보를 제공

합니다. 리스너는 연결 요청을 확인하는 프로세스입니다. 클라이언트와 로드 밸런서 간의 연결을 위한 프로토콜 및 포트로 구성됩니다. 지원되는 프로토콜 및 포트에 대한 자세한 내용은 리스너 구성 단원을 참조하십시오.

로드 밸런서 및 리스너를 구성하려면

1. 로드 밸런서 이름(Load Balancer name)에 로드 밸런서의 이름을 입력합니다. 예: `my-alb`.
2. [Scheme] 및 [IP address type]은 기본값으로 유지합니다.
3. 네트워크 매핑(Network mapping)에서 EC2 인스턴스에 사용한 VPC를 선택합니다. 2개 이상의 가용 영역과 영역당 1개의 서브넷을 선택합니다. EC2 인스턴스를 시작할 때 사용한 각 가용 영역에서 가용 영역을 선택한 후 해당 가용 영역에 대한 하나의 퍼블릭 서브넷을 선택합니다.
4. 보안 그룹의 경우 이전 단계에서 선택한 VPC의 기본 보안 그룹을 선택합니다. 그 대신, 다른 보안 그룹을 선택할 수 있습니다. 보안 그룹은 로드 밸런서가 리스너 포트 및 상태 확인 포트에서 등록된 대상과 통신하는 것을 허용하는 규칙이 포함되어야 합니다. 자세한 내용은 보안 그룹 규칙을 참조하십시오.
5. 리스너 및 라우팅(Listeners and routing)에 대해 기본 프로토콜과 포트를 유지하고 목록에서 대상 그룹을 선택합니다. 포트 80에서 HTTP 트래픽을 수락하고 기본으로 선택한 대상 그룹에 트래픽을 전달하는 리스너를 구성합니다. 이 자습서의 경우 HTTPS 리스너를 생성하지 않습니다.
6. 기본 작업(Default action)에 대해, 1단계: 대상 그룹 구성에서 생성하고 등록한 대상 그룹을 선택합니다.
7. (선택 사항) 태그를 추가하여 로드 밸런서를 분류합니다. 태그 키는 각 로드 밸런서에 대해 고유해야 합니다. 허용되는 문자는 문자, 공백, 숫자(UTF-8 형식) 및 특수 문자 + - = . _ : / @입니다. 선행 또는 후행 공백을 사용하면 안 됩니다. 태그 값은 대소문자를 구분합니다.
8. 구성을 검토하고 로드 밸런서 생성(Create load balancer)을 선택합니다. 생성 중에 로드 밸런서에 몇 가지 기본 특성이 적용됩니다. 로드 밸런서를 생성한 후 이를 보고 편집할 수 있습니다. 자세한 내용은 로드 밸런서 속성 섹션을 참조하세요.

4단계: 로드 밸런서 테스트

로드 밸런서를 생성한 후에는 EC2 인스턴스에 트래픽을 전송하고 있는지 확인할 수 있습니다.

로드 밸런서를 테스트하려면

1. 로드 밸런서가 생성되었다는 통보를 받은 후 [Close]를 선택합니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 새로 생성한 대상 그룹을 선택합니다.

4. [Targets]를 선택하고 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.
5. 탐색 창의 로드 밸런싱에서 로드 밸런서를 선택합니다.
6. 새로 생성한 로드 밸런서를 선택합니다.
7. [설명(Description)]을 선택하고, 로드 밸런서의 DNS 이름(예: my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com)을 복사합니다. DNS 이름을 인터넷에 연결된 웹 브라우저의 주소 필드에 붙여 넣습니다. 모든 것이 잘 작동하는 경우 브라우저에 서버 기본 페이지가 표시됩니다.
8. (선택 사항) 추가 리스너를 정의하려면 규칙 추가 단원을 참조하세요.

5단계: (선택 사항) 로드 밸런서 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. 로드 밸런서를 삭제해도 로드 밸런서에 등록된 대상에는 영향을 미치지 않습니다. 예를 들어 EC2 인스턴스는 이 가이드에서 생성된 로드 밸런서를 삭제한 후에도 계속 실행됩니다.

콘솔을 사용하여 로드 밸런서를 삭제하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱에서 로드 밸런서를 선택합니다.
3. 로드 밸런서에 대한 확인란을 선택한 후 작업(Actions), 삭제(Delete)를 선택합니다.
4. 확인 메시지가 나타나면 Yes, Delete(예, 삭제합니다)를 선택합니다.

를 사용하여 Application Load Balancer 시작하기 AWS CLI

이 자습서에서는를 통해 Application Load Balancer를 실습으로 소개합니다 AWS CLI.

내용

- 시작하기 전 준비 사항
- 로드 밸런서 생성
- HTTPS 리스너 추가
- 경로 기반 라우팅 추가
- 로드 밸런서 삭제

시작하기 전 준비 사항

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

  ```
  aws elbv2 help
  ```

  elbv2가 유효한 선택이 아니라는 오류 메시지가 표시되면 AWS CLI를 업데이트하십시오. 자세한 내용은 AWS Command Line Interface 사용 설명서의 의 최신 버전 설치를 AWS CLI 참조하세요.
- 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 섹션을 참조하세요.
- IPv4 또는 듀얼스택 로드 밸런서 중 무엇을 생성할지 결정합니다. 클라이언트가 IPv4 주소만을 사용하여 로드 밸런서와 통신하도록 하려는 경우 IPv4를 사용합니다. 클라이언트가 IPv4 및 IPv6 주소를 사용하여 로드 밸런서와 통신하도록 하려는 경우 듀얼스택을 사용합니다. 또한 듀얼스택을 사용하면 IPv6 애플리케이션 또는 듀얼스택 서브넷 등의 IPv6를 사용하는 백엔드 대상과 통신할 수 있습니다.
- 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.

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-0e3f5cac72EXAMPLE subnet-081ec835f3EXAMPLE --security-groups
 sg-07e8ffd50fEXAMPLE
```

   create-load-balancer 명령을 사용하여 **dualstack** 로드 밸런서를 생성합니다.

```
aws elbv2 create-load-balancer --name my-load-balancer  \
--subnets subnet-0e3f5cac72EXAMPLE subnet-081ec835f3EXAMPLE --security-groups
 sg-07e8ffd50fEXAMPLE --ip-address-type dualstack
```

   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 EC2 인스턴스에 사용한 VPC와 동일한 VPC를 지정해서 대상 그룹을 생성합니다.

   듀얼스택 로드 밸런서와 연결할 IPv4 및 IPv6 대상 그룹을 생성할 수 있습니다. 대상 그룹의 IP 주소 유형에 따라 로드 밸런서가 백엔드 대상과 통신하고 상태를 확인하는 데 사용할 IP 버전이 결정됩니다.

```
aws elbv2 create-target-group --name my-targets --protocol HTTP --port 80 \
--vpc-id vpc-0598c7d356EXAMPLE --ip-address-type [ipv4 or ipv6]
```

   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-0abcdef1234567890 Id=i-1234567890abcdef0
```

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) 다음 describe-target-health 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)을 사용하여 인증서를 생성하거나 가져옵니다. 자세한 내용은 AWS Certificate Manager 사용 설명서의 퍼블릭 인증서 요청 또는 인증서 가져오기를 참조하세요.
   - AWS Identity and Access Management (IAM)을 사용하여 인증서를 업로드합니다. 자세한 내용은 IAM 사용 설명서에서 서버 인증서 작업을 참조하세요.
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
```

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-0598c7d356EXAMPLE
```

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

```
aws elbv2 register-targets --target-group-arn targetgroup-arn  \
--targets Id=i-0abcdef1234567890 Id=i-1234567890abcdef0
```

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 접점 역할을 수행합니다. 클라이언트는 로드 밸런서에 요청을 전송하고 로드 밸런서는 EC2 인스턴스 같은 대상으로 로드 밸런서를 전송합니다. 로드 밸런서를 구성하려는 경우, 대상 그룹을 생성한 다음 대상을 해당 대상 그룹에 등록합니다. 리스너를 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 참조하세요.

목차

- 로드 밸런서를 위한 서브넷
- 로드 밸런서 보안 그룹
- 로드 밸런서 상태
- 로드 밸런서 속성
- IP 주소 유형
- IPAM IP 주소 풀
- 로드 밸런서 연결
- 교차 영역 로드 밸런싱
- DNS 이름
- Application Load Balancer 생성
- Application Load Balancer의 가용 영역 업데이트
- Application Load Balancer 보안 그룹
- Application Load Balancer의 IP 주소 유형 업데이트
- Application Load Balancer의 IPAM IP 주소 풀 업데이트
- Application Load Balancer 통합
- Application Load Balancer의 속성 편집
- Application Load Balancer에 태그 지정
- Application Load Balancer 삭제
- Application Load Balancer 리소스 맵 보기
- Application Load Balancer 용량 예약

로드 밸런서를 위한 서브넷

Application Load Balancer를 생성할 때는 대상이 포함된 영역을 활성화해야 합니다. 영역을 활성화하려면 영역에 서브넷을 지정합니다. Elastic Load Balancing은 지정한 각 영역에 로드 밸런서 노드를 생성합니다.

고려 사항

- 활성화된 각 영역에 등록된 대상이 하나 이상 있는지 확인하는 경우에 로드 밸런서가 가장 효과적입니다.
- 영역에 대상을 등록하지만 영역을 활성화하지 않는 경우 이러한 등록된 대상은 로드 밸런서로부터 트래픽을 수신하지 않습니다.
- 로드 밸런서에 여러 영역을 활성화하는 경우 영역은 동일한 유형이어야 합니다. 예를 들어 가용 영역과 로컬 영역을 모두 활성화할 수는 없습니다.
- 자신이 사용자와 공유한 서브넷은 지정할 수 있습니다.

Application Load Balancer는 다음 서브넷 유형을 지원합니다.

서브넷 유형

- 가용 영역 서브넷
- 로컬 영역 서브넷
- Outpost 서브넷

가용 영역 서브넷

두 개 이상의 가용 영역 서브넷을 선택해야 합니다. 다음과 같은 제한 사항이 있습니다.

- 각 서브넷이 서로 다른 가용 영역에 속해야 합니다.
- 로드 밸런서가 적절하게 확장 가능하도록 로드 밸런서의 각 가용 영역 서브넷에 /27 비트 마스크(예: 10.0.0.0/27)가 하나 이상인 CIDR 블록이 있고 서브넷당 사용 가능한 IP 주소가 8개 이상 있는지 확인합니다. 필요한 경우 로드 밸런서를 확장하려면 이러한 이 IP 주소 8개가 필요합니다. 로드 밸런서는 이러한 IP 주소를 사용하여 대상에 대한 연결을 설정합니다. 이 기능이 없으면 Application Load Balancer 노드가 어려울 수도 있으며 이로 인해 노드 교체가 실패 상태로 전환될 수도 있습니다.

참고: 확장을 시도하는 동안 Application Load Balancer 서브넷의 사용 가능한 IP 주소가 부족하면 애플리케이션 로드 밸런서는 충분한 용량으로 실행됩니다. 이 기간에 기존 노드는 계속해서 트래픽을 처리하지만 확장 시도가 중단되면 연결 설정 시도 시 5xx 오류가 발생하거나 시간 초과가 발생할 수도 있습니다.

로컬 영역 서브넷

하나 이상의 로컬 영역 서브넷을 지정할 수 있습니다. 다음과 같은 제한 사항이 있습니다.

- 로드 밸런 AWS WAF 서에서는를 사용할 수 없습니다.
- Lambda 함수를 대상으로 사용할 수 없습니다.
- 스티키 세션 또는 애플리케이션 고정성은 사용할 수 없습니다.

Outpost 서브넷

단일 Outpost 서브넷을 지정할 수 있습니다. 다음과 같은 제한 사항이 있습니다.

- 온프레미스 데이터 센터에 Outpost가 설치 및 구성되어 있어야 합니다. Outpost와 AWS 리전 간에 안정적인 네트워크 연결이 있어야 합니다. 자세한 내용은 AWS Outposts 사용 설명서를 참조하십시오.
- 로드 밸런서는 로드 밸런서 노드용 Outpost에 두 개의 `large` 인스턴스가 필요합니다. 지원되는 인스턴스 유형은 다음 표에 나와 있습니다. 로드 밸런서는 필요에 따라 확장되어 노드 크기를 한 번에 한 개씩(`large`에서 `xlarge`, 그 후 `xlarge`에서 `2xlarge`, 그 후 `2xlarge`에서 `4xlarge`) 조정합니다. 노드를 가장 큰 인스턴스 크기로 확장한 후 추가 용량이 필요한 경우, 로드 밸런서가 `4xlarge` 인스턴스를 로드 밸런서 노드로 추가합니다. 로드 밸런서를 확장하기 위한 인스턴스 용량이 충분하지 않거나 사용 가능한 IP 주소가 없는 경우 로드 밸런서는 이벤트를 AWS Health Dashboard에 보고하며 로드 밸런서 상태는 `active_impaired`입니다.
- 인스턴스 ID 또는 IP 주소로 대상을 등록할 수 있습니다. Outpost에 대해 AWS 리전에 대상을 등록하면 해당 대상이 사용되지 않습니다.
- 대상으로서 Lambda 함수, AWS WAF 통합, Sticky Session, 인증 지원, AWS Global Accelerator와의 통합 기능은 사용할 수 없습니다.

Application Load Balancer는 Outpost에서 c5/c5d, m5/m5d 또는 r5/r5d 인스턴스에 배포할 수 있습니다. 다음 표는 로드 밸런서가 Outpost에서 사용할 수 있는 인스턴스 유형별 크기 및 EBS 볼륨을 보여줍니다.

인스턴스 유형 및 크기	EBS 볼륨(GB)
c5/c5d	
large	50
xlarge	50
2xlarge	50
4xlarge	100
m5/m5d	
large	50
xlarge	50
2xlarge	100개
4xlarge	100
r5/r5d	
large	50
xlarge	100개
2xlarge	100개
4xlarge	100

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 밸런서와 관련된 보안 그룹에 대한 규칙은 리스너와 상태 확인 포트 모두에서 양방향으로 트래픽을 허용해야 합니다. 로드 밸런서에 리스너를 추가하거나 대상 그룹의 상태 확인 포트를 업데이트할 때

마다 보안 그룹 규칙을 검토하여 양방향으로 새로운 포트에서 양방향 트래픽을 허용하는지 확인해야 합니다. 자세한 내용은 권장 규칙 단원을 참조하세요.

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.

`active_impaired`

로드 밸런서가 트래픽을 라우팅하지만 확장에 필요한 리소스가 없습니다.

`failed`

로드 밸런서를 설정할 수 없습니다.

로드 밸런서 속성

해당 속성을 편집하여 Application Load Balancer를 구성할 수 있습니다. 자세한 내용은 로드 밸런서 속성 편집 단원을 참조하십시오.

다음은 로드 밸런서의 속성입니다.

`access_logs.s3.enabled`

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

`access_logs.s3.bucket`

액세스 로그에 대한 Amazon S3 버킷 이름입니다. 이 속성은 액세스 로그가 활성화된 경우에 필요합니다. 자세한 내용은 액세스 로그 활성화 단원을 참조하십시오.

`access_logs.s3.prefix`

Amazon S3 버킷의 위치에 대한 접두사입니다.

`client_keep_alive.seconds`

클라이언트 연결 유지 기간 값(초)입니다. 기본값은 3600초입니다.

`deletion_protection.enabled`

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

`idle_timeout.timeout_seconds`

유휴 제한 시간 값(초). 기본값은 60초입니다.

`ipv6.deny_all_igw_traffic`

인터넷 게이트웨이를 통해 내부 로드 밸런서에 대한 의도하지 않은 액세스가 발생하지 못하도록 로드 밸런서에 대한 인터넷 게이트웨이(IGW) 액세스를 차단합니다. 인터넷 연결 로드 밸런서에 대해서는 `false`로, 내부 로드 밸런서에 대해서는 `true`로 설정됩니다. 이 속성은 비IGW 인터넷 액세스(예: 피어링, Transit Gateway, AWS Direct Connect또는를 통한 액세스)를 방지하지 않습니다 AWS VPN.

`routing.http.desync_mitigation_mode`

애플리케이션에 보안 위험을 초래할 수 있는 요청을 로드 밸런서에서 처리하는 방법을 결정합니다. 가능한 값은 `monitor`, `defensive` 및 `strictest` 입니다. 기본값은 `defensive`입니다.

`routing.http.drop_invalid_header_fields.enabled`

헤더 필드가 있는 HTTP 헤더를 로드 밸런서(`true`)를 통해 제거할지 또는 대상(`false`)으로 라우팅할지 여부를 나타냅니다. 기본값은 `false`입니다. Elastic Load Balancing에서는 유효한 HTTP 헤더 이름이 HTTP 필드 이름 레지스트리에 설명된 바와 같이 정규 표현식 `[-A-Za-z0-9]+`를 준수해야 합니다. 각 이름은 영숫자 또는 하이픈으로 구성되어야 합니다. 이 패턴을 준수하지 않는 HTTP 헤더를 요청에서 제거하려는 경우 `true`를 선택합니다.

`routing.http.preserve_host_header.enabled`

Application Load Balancer가 HTTP 요청에 `Host` 헤더를 보존하고 변경 없이 대상에 전송해야 하는지 여부를 나타냅니다. 가능한 값은 `true`와 `false`입니다. 기본값은 `false`입니다.

`routing.http.x_amzn_tls_version_and_cipher_suite.enabled`

협상된 TLS 버전과 암호 그룹에 대한 정보를 포함하는 두 헤더(`x-amzn-tls-version` 및 `x-amzn-tls-cipher-suite`)가 대상에 전송되기 전에 클라이언트 요청에 추가되는지 여부를 나타냅니다. `x-amzn-tls-version` 헤더에는 클라이언트와 협상된 TLS 프로토콜 버전에 대한 정보가 있으며, `x-amzn-tls-cipher-suite` 헤더에는 클라이언트와 협상한 암호 그룹에 대한 정보가 있습니다. 두 헤더 모두 OpenSSL 형식입니다. 이 속성에 사용 가능한 값은 `true` 및 `false`입니다. 기본값은 `false`입니다.

`routing.http.xff_client_port.enabled`

X-Forwarded-For 헤더가 클라이언트의 로드 밸런서 연결에 사용한 소스 포트를 보존해야 하는지 여부를 나타냅니다. 가능한 값은 true과 false입니다. 기본값은 false입니다.

`routing.http.xff_header_processing.mode`

이를 사용하여, Application Load Balancer가 대상에 요청을 보내기 전에 HTTP 요청의 X-Forwarded-For 헤더를 수정, 보존 또는 제거할 수 있습니다. 가능한 값은 append, preserve 및 remove 입니다. 기본값은 append입니다.

- 값이 append인 경우, Application Load Balancer가 대상에 요청을 보내기 전에 HTTP 요청의 X-Forwarded-For 헤더에 클라이언트 IP 주소(마지막 홉)를 추가합니다.
- 값이 preserve인 경우, Application Load Balancer가 대상에 요청을 보내기 전에 HTTP 요청의 X-Forwarded-For 헤더를 보존합니다.
- 값이 remove인 경우, Application Load Balancer가 대상에 요청을 보내기 전에 HTTP 요청의 X-Forwarded-For 헤더를 제거합니다.

`routing.http2.enabled`

HTTP/2가 활성화되었는지를 나타냅니다. 기본값은 true입니다.

`waf.fail_open.enabled`

요청을 전달할 수 없는 경우 a AWS WAF지원 로드 밸런서가 요청을 대상으로 라우팅하도록 허용할지 여부를 나타냅니다 AWS WAF. 가능한 값은 true와 false입니다. 기본값은 false입니다.

> **Note**
>
> routing.http.drop_invalid_header_fields.enabled 속성은 HTTP Desync 보호 기능을 제공하기 위해 도입되었습니다. routing.http.desync_mitigation_mode 속성은 애플리케이션에 대해 HTTP Desync로부터 보다 포괄적인 보호를 제공하기 위해 추가되었습니다. 두 속성을 모두 사용할 필요는 없으며 애플리케이션 요구 사항에 따라 둘 중 하나를 선택할 수 있습니다.

IP 주소 유형

클라이언트가 인터넷 연결 내부 로드 밸런서에 액세스하기 위해 사용할 수 있는 IP 주소 유형을 설정할 수 있습니다.

Application Load Balancer는 다음 IP 주소 유형을 지원합니다.

`ipv4`

클라이언트는 IPv4 주소(예: 192.0.2.1)를 사용하여 로드 밸런서에 연결해야 합니다.

`dualstack`

클라이언트는 IPv4 주소(예: 192.0.2.1) 및 IPv6 주소(예: 2001:0db8:85a3:0:0:8a2e:0370:7334)를 사용하여 로드 밸런서에 연결할 수 있습니다.

`dualstack-without-public-ipv4`

클라이언트는 IPv6 주소(예: 2001:0db8:85a3:0:0:8a2e:0370:7334)를 사용하여 로드 밸런서에 연결해야 합니다.

고려 사항

- 로드 밸런서는 대상 그룹의 IP 주소 유형에 따라 대상과 통신합니다.
- 로드 밸런서에 대해 듀얼스택 모드를 활성화하면 Elastic Load Balancing에서 해당 로드 밸런서의 AAA DNS 레코드를 제공합니다. IPv4 주소를 사용하여 로드 밸런서와 통신하는 클라이언트는 A DNS 레코드를 확인합니다. IPv6 주소를 사용하여 로드 밸런서와 통신하는 클라이언트는 AAA DNS 레코드를 확인합니다.
- 의도하지 않은 인터넷 액세스를 방지하기 위해 인터넷 게이트웨이를 통한 내부 듀얼스택 로드 밸런서로의 액세스가 차단됩니다. 그러나 이는 비 IGW 인터넷 액세스(예: 피어링, Transit Gateway AWS Direct Connect또는를 통한 액세스)를 방지하지 않습니다 AWS VPN.
- Application Load Balancer 인증은 ID 제공업체(idP) 또는 Amazon Cognito 엔드포인트에 연결할 때 IPv4만 지원합니다. 퍼블릭 IPv4 주소가 없으면 로드 밸런서가 인증 프로세스를 완료할 수 없어 HTTP 500 오류가 발생합니다.

자세한 내용은 Application Load Balancer의 IP 주소 유형 업데이트 단원을 참조하십시오.

IPAM IP 주소 풀

IPAM IP 주소 풀은 Amazon VPC IP 주소 관리자(IPAM) 내의 연속 IP 주소 범위(또는 CIDRs) 모음입니다. Application Load Balancer에서 IPAM IP 주소 풀을 사용하면 라우팅 및 보안 요구 사항에 따라 IPv4 주소를 구성할 수 있습니다. IPAM IP 주소 풀은 먼저 IPAM 내에서 생성해야 Application Load Balancer에서 사용할 수 있습니다. 자세한 내용은 IPAM으로 IP 주소 가져오기를 참조하세요.

고려 사항

- IPAM IP 주소 풀은 내부 로드 밸런서 또는 퍼블릭 IPv4 IP 주소 유형이 없는 듀얼 스택과 호환되지 않습니다.
- 로드 밸런서에서 현재 사용 중인 IPAM IP 주소 풀의 IP 주소는 삭제할 수 없습니다.
- 다른 IPAM IP 주소 풀로 전환하는 동안 로드 밸런서 HTTP 클라이언트 연결 유지 기간에 따라 기존 연결이 종료됩니다.
- IPAM IP 주소 풀은 여러 계정에서 공유할 수 있습니다. 자세한 내용은 IPAM에 대한 통합 옵션 구성을 참조하세요.

IPAM IP 주소 풀을 사용하면 퍼블릭 IPv4 주소 범위의 일부 또는 전체를 로 가져 AWS 와 Application Load Balancer와 함께 사용할 수 있습니다. IP 주소 할당을 더 잘 제어하면 보안 정책 및 제어를 보다 효과적으로 관리하고 적용하는 동시에 비용을 절감할 수 있습니다. Application Load Balancer에서 IPAM IP 주소 풀을 사용하는 것과 관련된 추가 요금은 없지만 사용되는 계층에 따라 IPAM과 관련된 요금이 있을 수 있습니다. 자세한 내용은 Amazon VPC 요금을 참조하세요.

IPAM IP 주소 풀은 EC2 인스턴스 및 Application Load Balancer를 시작할 때 항상 우선 순위가 지정되며 IP 주소가 더 이상 사용되지 않으면 즉시 다시 사용할 수 있게 됩니다. IPAM IP 주소 풀에 더 이상 할당 가능한 IP 주소가 없는 경우 AWS 관리형 IP 주소가 할당됩니다. AWS 관리형 IP 주소에는 추가 비용이 발생합니다. IP 주소를 추가하려면 기존 IPAM IP 주소 풀에 새 IP 주소 범위를 추가할 수 있습니다.

로드 밸런서 연결

로드 밸런서는 요청을 처리할 때 클라이언트와의 연결과 대상과의 연결이라는 두 가지 연결을 유지합니다. 로드 밸런서와 클라이언트 사이의 연결은 프런트엔드 연결이라고도 합니다. 로드 밸런서와 대상 사이의 연결은 백엔드 연결이라고도 합니다.

교차 영역 로드 밸런싱

Application Load Balancer를 사용하면 기본적으로 교차 영역 로드 밸런싱이 켜져 있으며 로드 밸런서 수준에서 변경할 수 없습니다. 자세한 내용은 Elastic Load Balancing 사용 설명서의 교차 영역 로드 밸런싱 섹션을 참조하세요.

교차 영역 로드 밸런싱은 대상 그룹 수준에서 해제할 수 있습니다. 자세한 내용은 the section called “교차 영역 로드 밸런싱 해제” 단원을 참조하십시오.

DNS 이름

각 Application Load Balancer는 다음 구문을 사용하여 기본 도메인 이름 시스템(DNS)을 수신합니다. *name*-*id*.elb.*region*.amazonaws.com. 예: my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com.

기억하기 쉬운 DNS 이름을 사용하는 것을 선호하는 경우, 사용자 지정 도메인 이름을 생성하고 이를 Application Load Balancer의 DNS 이름과 연결할 수 있습니다. 클라이언트가 이러한 사용자 지정 도메인 이름을 사용해 요청을 하면 DNS 서버는 이를 Application Load Balancer의 DNS 이름으로 해석합니다.

먼저 인증된 도메인 등록 대행자를 이용해 도메인 이름을 등록합니다. 다음으로 도메인 등록 대행자와 같은 DNS 서비스를 사용하여 요청을 Application Load Balancer로 라우팅하는 DNS 레코드를 생성합니다. 자세한 내용은 DNS 서비스에 대한 설명서를 참조하세요. 예를 들어, DNS 서비스로 Amazon Route 53을 사용하는 경우 Application Load Balancer를 지정하는 별칭 레코드를 생성합니다. 자세한 내용은 Amazon Route 53 개발자 안내서의 ELB 로드 밸런서로 트래픽 라우팅을 참조하세요.

Application Load Balancer는 활성화된 각 가용 영역에 대하여 하나의 IP 주소를 가집니다. 이는 Application Load Balancer 노드의 IP 주소입니다. Application Load Balancer의 DNS 이름은 이러한 주소로 확인됩니다. 예를 들어, Application Load Balancer의 사용자 지정 도메인 이름이 `example.applicationloadbalancer.com`이라고 가정해 보겠습니다. 다음 dig 또는 nslookup 명령을 사용하여 Application Load Balancer 노드의 IP 주소를 확인합니다.

Linux 또는 Mac

```
$ dig +short example.applicationloadbalancer.com
```

Windows

```
C:\> nslookup example.applicationloadbalancer.com
```

Application Load Balancer는 노드를 위한 DNS 레코드를 가집니다. DNS 이름을 다음 구문과 함께 사용하여 Application Load Balancer 노드의 IP 주소를 확인할 수 있습니다. *az*.*name*-*id*.elb.*region*.amazonaws.com.

Linux 또는 Mac

```
$ dig +short us-east-2b.my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com
```

Windows

```
C:\> nslookup us-east-2b.my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com
```

Application Load Balancer 생성

로드 밸런서는 클라이언트로부터 요청을 가져와서 대상 그룹의 대상에 이를 분산합니다.

시작하기 전에 대상에서 사용하는 각 영역에 하나 이상의 퍼블릭 서브넷이 있는 Virtual Private Cloud(VPC)가 있는지 확인합니다. 자세한 내용은 the section called “로드 밸런서를 위한 서브넷” 단원을 참조하십시오.

를 사용하여 로드 밸런서를 생성하려면 섹션을 AWS CLI참조하세요를 사용하여 Application Load Balancer 시작하기 AWS CLI.

를 사용하여 로드 밸런서를 생성하려면 다음 작업을 AWS Management Console완료합니다.

업무

- 1단계: 대상 그룹 구성
- 2단계: 대상 등록
- 3단계: 로드 밸런서 및 리스너 구성
- 4단계: 로드 밸런서 테스트

1단계: 대상 그룹 구성

대상 그룹을 구성하면 EC2 인스턴스와 같은 대상을 등록할 수 있습니다. 이 단계에서 구성하는 대상 그룹은 로드 밸런서를 구성할 때 리스너 규칙의 대상 그룹으로 사용됩니다. 자세한 내용은 Application Load Balancer 대상 그룹 단원을 참조하십시오.

콘솔을 사용하여 대상 그룹을 구성하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 대상 그룹을 선택합니다.
3. 대상 그룹 생성을 선택합니다.
4. [기본 구성(Basic configuration)] 섹션에서 다음 파라미터를 설정합니다.

    a. 대상 유형 선택에서 인스턴스를 선택하여 인스턴스 ID로 대상을 지정하거나 IP 주소를 선택하여 IP 주소로만 대상을 지정합니다. 대상 유형이 Lambda 함수인 경우 상태 확인(Health checks) 섹션에서 활성화(Enable)를 선택하여 상태 확인을 활성화할 수 있습니다.

b. 대상 그룹 이름에 대상 그룹의 이름을 입력합니다.

c. 필요에 따라 포트와 프로토콜을 수정합니다.

d. 대상 유형이 인스턴스 또는 IP 주소인 경우 IPv4 또는 IPv6로 IP 주소 유형을 선택하고, 그 외의 경우에는 다음 단계로 건너뜁니다.

   선택한 IP 주소 유형을 가진 대상만 이 대상 그룹에 포함될 수 있습니다. 대상 그룹을 생성한 후에는 IP 주소 유형을 변경할 수 없습니다.

e. VPC의 경우 대상 그룹에 포함할 대상이 있는 Virtual Private Cloud(VPC)를 선택합니다.

f. 프로토콜 버전(Protocol version)에서 요청 프로토콜이 HTTP/1.1 또는 HTTP/2인 경우 HTTP1을 선택하고, 요청 프로토콜이 HTTP/2 또는 gRPC인 경우 HTTP2를 선택하고, 요청 프로토콜이 gRPC인 경우 gRPC를 선택합니다.

5. 상태 확인 섹션에서 필요에 따라 기본 설정을 수정합니다. 고급 상태 확인 설정(Advanced health check settings)의 경우 상태 확인 포트, 개수, 시간 초과, 간격을 선택하고 성공 코드를 지정합니다. 상태 확인이 비정상 임계값 수를 연속으로 초과하는 경우 로드 밸런서는 대상을 서비스 중단 상태로 만듭니다. 상태 확인이 정상 임계값 수를 연속으로 초과하는 경우 로드 밸런서는 대상을 다시 서비스 상태로 전환합니다. 자세한 내용은 Application Load Balancer 대상 그룹의 상태 확인 단원을 참조하세요.

6. (선택 사항) 다음과 같이 하나 이상의 태그를 추가합니다.

   a. 태그 섹션을 확장합니다.

   b. [Add tag]를 선택합니다.

   c. 태그 키 및 태그 값을입력합니다. 허용되는 문자는 문자, 공백, 숫자(UTF-8 형식) 및 특수 문자 + - = . _ : / @입니다. 선행 또는 후행 공백을 사용하면 안 됩니다. 태그 값은 대소문자를 구분합니다.

7. [Next]를 선택합니다.

2단계: 대상 등록

EC2 인스턴스, IP 주소 또는 Lambda 함수를 대상 그룹의 대상으로 등록할 수 있습니다. 로드 밸런서를 생성하기 위한 선택적 단계입니다. 그러나 대상을 등록해야 로드 밸런서가 트래픽을 해당 대상으로 라우팅합니다.

1. 대상 등록(Register targets) 페이지에서 다음과 같이 하나 이상의 대상을 추가합니다.

   - 대상 유형이 인스턴스인 경우 하나 이상의 인스턴스를 선택하고 하나 이상의 포트를 입력한 다음 아래에 보류 중인 것으로 포함을 선택합니다.

- 대상 유형이 IP 주소인 경우 다음을 수행합니다.

  a. 네트워크 VPC를 목록에서 선택하거나 기타 프라이빗 IP 주소를 선택합니다.

  b. IP 주소를 수동으로 입력하거나 인스턴스 세부 정보를 사용하여 IP 주소를 찾습니다. 한 번에 최대 5개의 IP 주소를 입력할 수 있습니다.

  c. 지정된 IP 주소로 트래픽을 라우팅할 포트를 입력합니다.

  d. 아래에서 보류 중인 것으로 포함을 선택합니다.

- 대상 유형이 Lambda인 경우 Lambda 함수를 선택하거나 Lambda 함수 ARN을 입력한 다음 아래에 보류 중인 것으로 포함(Include as pending below)을 선택합니다.

2. [Create target group]을 선택합니다.

3단계: 로드 밸런서 및 리스너 구성

Application Load Balancer를 생성하려면 먼저 이름, 구성표 및 IP 주소 유형과 같은 로드 밸런서에 대한 기본 구성 정보를 제공해야 합니다. 그런 다음 네트워크와 하나 이상의 리스너에 대한 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 단원을 참조하십시오.

콘솔을 사용하여 로드 밸런서 및 리스너를 구성하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 [Load Balancers]를 클릭합니다.
3. 로드 밸런서 생성을 선택합니다.
4. Application Load Balancer 아래에서 생성(Create)을 선택합니다.
5. 기본 구성

   a. 로드 밸런서 이름(Load Balancer name)에 로드 밸런서의 이름을 입력합니다. 예를 들어 **my-alb**입니다. Application Load Balancer의 이름은 해당 리전의 Application Load Balancer 및 네트워크 로드 밸런서 세트 내에서 고유해야 합니다. 이름은 최대 32자여야 하며 영숫자 및 하이픈만 포함할 수 있습니다. 하이픈 또는 `internal-`(으)로 시작하거나 끝나서는 안 됩니다. Application Load Balancer의 이름은 생성한 후에는 변경할 수 없습니다.

   b. 구성표(Scheme)에서 internet-facing 또는 internal을 선택합니다. internet-facing 로드 밸런서는 인터넷을 통해 클라이언트의 요청을 대상으로 라우팅합니다. 내부 로드 밸런서는 프라이빗 IP 주소를 사용하여 요청을 대상으로 라우팅합니다.

c. IP 주소 유형에서 IPv4, 듀얼 스택 또는 퍼블릭 IPv4가 없는 듀얼 스택을 선택합니다. 클라이언트가 IPv4 주소를 사용하여 로드 밸런서와 통신하는 경우 IPv4를 선택합니다. 클라이언트가 IPv4 및 IPv6 주소를 모두 사용해 로드 밸런서와 통신하는 경우 듀얼 스택(Dualstack)을 선택합니다. 클라이언트가 IPv6 주소만 사용해 로드 밸런서와 통신하는 경우 퍼블릭 IPv4가 없는 듀얼 스택을 선택합니다.

6. 네트워크 매핑

   a. VPC에서는 EC2 인스턴스에 사용한 것과 동일한 VPC를 선택합니다. 구성표(Scheme)에 대해 Internet-facing을 선택한 경우 인터넷 게이트웨이가 있는 VPC만 선택할 수 있습니다.

   b. IPAM IP 주소 풀의 경우 퍼블릭 IPv4 주소에 IPAM 풀 사용을 선택할 수 있습니다. 자세한 내용은 IPAM IP 주소 풀을 참조하세요.

   c. 가용 영역 및 서브넷의 경우 다음과 같이 서브넷을 선택하여 로드 밸런서의 영역을 활성화합니다.

      - 2개 이상의 가용 영역의 서브넷
      - 1개 이상의 로컬 영역의 서브넷
      - Outpost 서브넷 1개

      자세한 내용은 the section called “로드 밸런서를 위한 서브넷” 단원을 참조하십시오.

      내부 로드 밸런서의 경우 서브넷 CIDR에서 IPv4 및 IPv6 주소를 할당합니다.

      로드 밸런서에 대해 듀얼 스택 모드를 사용하도록 설정한 경우 IPv4 및 IPv6 CIDR 블록 모두가 있는 서브넷을 선택합니다.

7. 보안 그룹(Security groups)의 경우 기존 보안 그룹을 선택하거나 새 보안 그룹을 생성할 수 있습니다.

   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 보안 그룹을 만들 수 있습니다. 보안 그룹을 생성하여 대신 선택할 수도 있습니다. 자세한 내용은 권장 규칙 섹션을 참조하세요.

   (선택 사항) 로드 밸런서에 대한 새 보안 그룹을 생성하려면 새 보안 그룹 생성(Create a new security group)을 선택합니다.

8. 리스너 및 라우팅(Listeners and routing)에서 기본 리스너는 포트 80에서 HTTP 트래픽을 수락합니다. 기본 프로토콜과 포트를 유지하거나 다른 프로토콜과 포트를 선택할 수 있습니다. 기본 작업

(Default action)에서 앞서 생성한 대상 그룹을 선택합니다. 리스너 추가(Add listener)를 선택하여 다른 리스너(예: HTTPS 리스너)를 추가할 수도 있습니다.

9. (선택 사항) HTTPS 리스너를 사용하는 경우

   보안 정책의 경우 항상 최신 사전 정의 보안 정책을 사용하는 것이 좋습니다.

   a. 기본 SSL/TLS 인증서에서 다음 옵션을 사용할 수 있습니다.

      - 를 사용하여 인증서를 생성하거나 가져온 경우 ACM에서를 AWS Certificate Manager선택한 다음 인증서 선택에서 인증서를 선택합니다.
      - IAM을 사용하여 인증서를 가져온 경우 IAM에서를 선택하고 인증서 선택에서 인증서를 선택합니다.
      - 가져올 인증서가 있지만 리전에서 ACM을 이용할 수 없는 경우 가져오기를 선택하고 IAM으로를 선택합니다. 인증서 이름 필드에 인증서의 이름을 입력합니다. 인증서 프라이빗 키에서 프라이빗 키 파일(PEM 인코딩)의 콘텐츠를 복사해 붙여넣습니다. 인증서 본문에서 퍼블릭 키 인증서 파일(PEM 인코딩)의 콘텐츠를 복사해 붙여 넣습니다. 자체 서명 인증서를 사용하고 있지 않고 브라우저가 인증서를 묵시적으로 수락하는 것이 중요하지 않다면 인증서 체인(Certificate Chain)에 인증서 체인 파일(PEM 인코딩)의 콘텐츠를 복사해 붙여 넣습니다.

   b. (선택 사항) 상호 인증을 활성화하려면 클라이언트 인증서 처리에서 상호 인증(mTLS)을 활성화합니다.

      활성화되면 기본 상호 TLS 모드는 패스스루입니다.

      트러스트 스토어로 확인을 선택한 경우:

      - 기본적으로 만료된 클라이언트 인증서와의 연결은 거부됩니다. 이 동작을 변경하려면 고급 mTLS 설정 을 확장한 다음 클라이언트 인증서 만료에서 만료된 클라이언트 인증서 허용을 선택합니다.
      - 트러스트 스토어에서 기존 트러스트 스토어를 선택하거나 새 트러스트 스토어를 선택합니다.
        - 새 트러스트 스토어를 선택한 경우 트러스트 스토어 이름, S3 URI 인증 기관 위치, 선택적으로 S3 URI 인증서 해지 목록 위치를 제공합니다.
      - (선택 사항) TrustStore CA 제목 이름 알림을 활성화할지 선택합니다.

10. (선택 사항) 생성 중에 서비스 통합을 통한 최적화에서 다른 서비스를 로드 밸런서와 통합할 수 있습니다.

- 기존 또는 자동으로 생성된 웹 ACL과 함께 로드 밸런서에 대한 AWS WAF 보안 보호를 포함하도록 선택할 수 있습니다. 생성 후 AWS WAF 콘솔 에서 웹 ACL을 관리할 수 있습니다. 자세한 내용은 AWS WAF 개발자 안내서의 웹 ACL을 AWS 리소스와 연결 또는 연결 해제를 참조하세요.
- AWS Global Accelerator에서 액셀러레이터를 생성하고 로드 밸런서를 액셀러레이터와 연결하도록 선택할 수 있습니다. 액셀러레이터 이름에는 a-z, A-Z, 0-9, .(마침표) 및 -(하이픈) 문자(최대 64자)가 포함될 수 있습니다. 액셀러레이터가 생성된 후에는 AWS Global Accelerator 콘솔을 사용하여 관리할 수 있습니다. 자세한 내용은 AWS Global Accelerator 개발자 안내서의 로드 밸런서를 생성할 때 액셀러레이터 추가를 참조하세요.

11. 태그 지정 및 생성

    a. (선택 사항) 태그를 추가하여 로드 밸런서를 분류합니다. 태그 키는 각 로드 밸런서에 대해 고유해야 합니다. 허용되는 문자는 문자, 공백, 숫자(UTF-8 형식) 및 특수 문자 + - = . _ : / @입니다. 선행 또는 후행 공백을 사용하면 안 됩니다. 태그 값은 대소문자를 구분합니다.

    b. 구성을 검토하고 로드 밸런서 생성(Create load balancer)을 선택합니다. 생성 중에 로드 밸런서에 몇 가지 기본 특성이 적용됩니다. 로드 밸런서를 생성한 후 이를 보고 편집할 수 있습니다. 자세한 내용은 로드 밸런서 속성 단원을 참조하십시오.

4단계: 로드 밸런서 테스트

로드 밸런서를 생성한 후에는 EC2 인스턴스가 초기 상태 확인을 통과했는지 확인할 수 있습니다. 그런 다음 로드 밸런서가 EC2 인스턴스로 트래픽을 전송하고 있는지 확인할 수 있습니다. 로드 밸런서를 삭제하려면 Application Load Balancer 삭제 섹션을 참조하세요.

로드 밸런서 테스트

1. 로드 밸런서가 생성된 후 [Close]를 선택합니다.
2. 탐색 창에서 대상 그룹을 선택합니다.
3. 새로 생성한 대상 그룹을 선택합니다.
4. [Targets]를 선택하고 인스턴스가 준비되었는지 확인합니다. 인스턴스의 상태가 `initial`인 경우, 일반적으로 인스턴스가 아직 등록 중이기 때문입니다. 이 상태는 인스턴스가 정상이라고 간주할 수 있는 최소 상태 확인 횟수를 통과하지 못했음을 나타낼 수도 있습니다. 하나 이상의 인스턴스 상태가 정상이어야 로드 밸런서를 테스트할 수 있습니다. 자세한 내용은 대상 상태 단원을 참조하십시오.
5. 탐색 창에서 로드 밸런서를 선택합니다.

6. 새로 생성한 로드 밸런서를 선택합니다.
7. 설명을 선택하고, 인터페이싱 또는 로드 밸런서의 DNS 이름(예: my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com)을 복사합니다.

   - 인터넷을 향하고 있는 로드 밸런서는 DNS 이름을 인터넷에 연결된 웹 브라우저의 주소 필드에 붙여 넣습니다.
   - 내부 로드 밸런서는 VPC에 비공개로 연결되는 웹 브라우저의 주소 필드에 DNS 이름을 붙여 넣습니다.

   모두 다 바르게 구성되면 브라우저에 서버 기본 페이지가 표시됩니다.
8. 웹 페이지가 표시되지 않는다면 다음 문서에서 추가 구성 도움말 및 문제 해결 단계를 참조하십시오.

   - DNS 관련 문제는 Amazon Route 53 개발자 안내서의 ELB 로드 밸런서로 트래픽 라우팅을 참조하세요.
   - Load Balancer 관련 문제에 대해서는 Application Load Balancer 문제 해결을 참조하십시오.

Application Load Balancer의 가용 영역 업데이트

로드 밸런서의 가용 영역을 언제든지 활성화 또는 비활성화할 수 있습니다. 가용 영역을 활성화하고 나면 로드 밸런서가 해당 가용 영역의 등록 대상으로 요청을 라우팅하기 시작합니다. Application Load Balancer에는 기본적으로 교차 영역 로드 밸런싱이 설정되어 있으므로 요청이 모든 가용 영역의 등록된 모든 대상으로 라우팅됩니다. 교차 영역 로드 밸런싱이 꺼져 있는 경우 로드 밸런서는 동일한 가용 영역의 대상으로만 요청을 라우팅합니다. 자세한 내용은 교차 영역 로드 밸런싱 단원을 참조하십시오. 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 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. Network mapping(네트워크 매핑) 탭에서 Edit subnets(서브넷 편집)을 선택합니다.

5. 가용 영역을 활성화하려면 해당 확인란을 선택하고 서브넷을 하나 선택합니다. 사용 가능한 서브넷이 하나뿐인 경우 해당 서브넷이 자동으로 선택됩니다.
6. 활성화된 가용 영역의 서브넷을 변경하려면 목록에서 다른 서브넷 중 하나를 선택합니다.
7. 가용 영역을 비활성화하려면 해당 확인란 선택을 취소합니다.
8. 변경 사항 저장을 선택합니다.

를 사용하여 가용 영역을 업데이트하려면 AWS CLI

set-subnets 명령을 사용합니다.

Application Load Balancer 보안 그룹

Application Load Balancer를 위한 보안 그룹은 로드 밸런서에 도달하고 나갈 수 있는 트래픽을 제어할 수 있습니다. 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. 인바운드 트래픽을 허용하는 포트 및 프로토콜을 선택할 수 있습니다. 예를 들어 로드 밸런서가 ping 요청에 응답하도록 ICMP(인터넷 제어 메시지 프로토콜) 연결을 열 수 있습니다(한편 ping 요청은 어떤 인스턴스에도 전달되지 않음).

권장 규칙

다음은 인터넷 경계 로드 밸런서에 대한 권장 규칙입니다.

Inbound		
소스	포트 범위	Comment
0.0.0.0/0	###	로드 밸런서 리스너 포트에서 모든 인바운드 트래픽을 허용
Outbound		
대상 주소	포트 범위	Comment

#### ## ##	*#### ###*	인스턴스 리스너 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다
#### ## ##	*## ##*	상태 확인 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다

다음은 내부 로드 밸런서에 대한 권장 규칙입니다.

Inbound

소스	포트 범위	Comment
VPC CIDR	*###*	로드 밸런서 리스너 포트에서 VPC CIDR에서 오는 인바운드 트래픽을 허용

Outbound

대상 주소	포트 범위	Comment
#### ## ##	*#### ###*	인스턴스 리스너 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다
#### ## ##	*## ##*	상태 확인 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다

Network Load Balancer의 대상으로 사용되는 Application Load Balancer에 대해 다음 규칙을 권장합니다.

Inbound

소스	포트 범위	설명

##### IP ##/CIDR	alb ###	로드 밸런서 리스너 포트에서 인바운드 트래픽을 허용합니다.
VPC CIDR	alb ###	로드 밸런서 리스너 포트 AWS PrivateLink 에서를 통한 인바운드 클라이언트 트래픽 허용
VPC CIDR	alb ###	Network Load Balancer로부터의 인바운드 상태 트래픽 허용

Outbound

대상	포트 범위	Comment
#### ## ##	#### ###	인스턴스 리스너 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다
#### ## ##	## ##	상태 확인 포트의 인스턴스로 아웃바운드 트래픽을 허용합니다

Application Load Balancer에 대한 보안 그룹은 연결 추적을 사용하여 Network Load Balancer로부터 들어오는 트래픽에 대한 정보를 추적합니다. 이는 Application Load Balancer에 설정된 보안 그룹 규칙에 관계없이 발생합니다. Amazon EC2 연결 추적에 대한 자세한 내용은 Amazon EC2 사용 설명서의 보안 그룹 연결 추적을 참조하세요.

대상이 로드 밸런서에서만 트래픽을 수신하도록 하려면 로드 밸런서로부터의 트래픽만 수락하도록 대상과 연결된 보안 그룹을 제한합니다. 대상의 보안 그룹의 수신 규칙에서 로드 밸런서의 보안 그룹을 소스로 설정하여 이 작업을 수행할 수 있습니다.

인바운드 ICMP 트래픽이 경로 MTU 검색을 지원하도록 허용하는 것이 좋습니다. 자세한 내용은 Amazon EC2 사용 설명서의 경로 MTU 검색을 참조하세요.

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 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 보안 탭에서 편집을 선택합니다.
5. 로드 밸런서에 보안 그룹을 연결하려면 보안 그룹을 선택합니다. 보안 그룹 연결을 제거하려면 해당 보안 그룹의 X 아이콘을 선택합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 보안 그룹을 업데이트하려면 AWS CLI

set-security-groups 명령을 사용합니다.

Application Load Balancer의 IP 주소 유형 업데이트

클라이언트가 로드 밸런서와 통신할 때 IPv4 주소만 사용하도록 하거나 IPv4 및 IPv6 주소를 둘 다 사용하도록(DualStack) Application Load Balancer를 구성할 수 있습니다. 로드 밸런서는 대상 그룹의 IP 주소 유형에 따라 대상과 통신합니다. 자세한 내용은 IP 주소 유형 단원을 참조하십시오.

DualStack 요구 사항

- 로드 밸런서를 만들고 업데이트할 때 언제든지 IP 주소 유형을 설정할 수 있습니다.
- 로드 밸런서용으로 지정하는 Virtual Private Cloud(VPC) 및 서브넷에는 연결된 IPv6 CIDR 블록이 있어야 합니다. 자세한 내용은 Amazon EC2 사용 설명서의의 IPv6 주소를 참조하세요.
- 로드 밸런서 서브넷의 라우팅 테이블은 IPv6 트래픽을 라우팅해야 합니다.
- 로드 밸런서의 보안 그룹은 IPv6 트래픽을 허용해야 합니다.
- 로드 밸런서 서브넷의 네트워크 ACL은 IPv6 트래픽을 허용해야 합니다.

생성 시 IP 주소 유형을 설정하려면

로드 밸런서 생성에 설명된 대로 설정을 구성합니다.

콘솔을 사용하여 IP 주소 유형을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.

3. 로드 밸런서를 선택합니다.
4. 네트워크 매핑 탭에서 IP 주소 유형 편집을 선택합니다.
5. IP 주소 유형에서 IPv4 주소만 지원하려면 IPv4를 선택하고, IPv4 및 IPv6 주소를 모두 지원하려면 듀얼 스택을 선택하고, IPv6 주소만 지원하려면 퍼블릭 IPv4가 없는 듀얼 스택을 선택합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 IP 주소 유형을 업데이트하려면 AWS CLI

set-ip-address-type 명령을 사용합니다.

Application Load Balancer의 IPAM IP 주소 풀 업데이트

IPAM IP 주소 풀은 먼저 IPAM 내에서 생성해야 Application Load Balancer에서 사용할 수 있습니다. 자세한 내용은 IPAM으로 IP 주소 가져오기를 참조하세요.

생성 시 IPAM IP 주소 풀을 설정하려면

로드 밸런서 생성에 설명된 대로 설정을 구성합니다.

콘솔을 사용하여 IPAM IP 주소 풀을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 네트워크 매핑 탭에서 IP 풀 편집을 선택합니다.
5. IP 풀에서 퍼블릭 IPv4 주소에 IPAM 풀 사용을 켭니다.
6. 퍼블릭 IPv4 IPAM 풀에서 사용할 IPAM 풀을 선택합니다.
7. 변경 사항 저장을 선택합니다.

를 사용하여 IPAM IP 주소 풀을 업데이트하려면 AWS CLI

modify-ip-pools 명령을 사용합니다.

Application Load Balancer 통합

다른 여러 AWS 서비스와 통합하여 Application Load Balancer 아키텍처를 최적화하여 애플리케이션의 성능, 보안 및 가용성을 향상시킬 수 있습니다.

로드 밸런서 통합

- Amazon Application Recovery Controller(ARC)
- Amazon CloudFront + AWS WAF
- AWS Global Accelerator
- AWS Config
- AWS WAF

Amazon Application Recovery Controller(ARC)

Amazon Application Recovery Controller(ARC)를 사용하면에서 실행되는 애플리케이션의 복구 작업을 준비하고 더 빠르게 수행할 수 있습니다 AWS. 영역 전환 및 영역 자동 전환은 Amazon Application Recovery Controller(ARC)의 기능입니다.

영역 전환을 사용하면 단일 작업으로 손상된 가용 영역에서 트래픽을 이동할 수 있습니다. 이러한 방법을 통해 AWS 리전의 다른 정상 가용 영역에서 계속 운영할 수 있습니다.

영역 자동 전환을 사용하면 이벤트 중에 애플리케이션의 리소스 트래픽을 사용자 대신 가용 영역에서 다른 AWS 곳으로 이동하여 복구 시간을 줄일 수 있는 권한을 부여합니다. 내부 모니터링에서 고객에게 잠재적으로 영향을 미칠 수 있는 가용 영역 장애가 있는 것으로 나타나면 자동 전환을 AWS 시작합니다. 가 자동 전환을 AWS 시작하면 영역 자동 전환을 위해 구성한 리소스에 대한 애플리케이션 트래픽이 가용 영역 밖으로 이동하기 시작합니다.

영역 전환을 시작하면 로드 밸런서가 해당 리소스에 대한 새 트래픽을 영향을 받는 가용 영역으로 보내는 것을 중단합니다. ARC는 영역 전환을 즉시 생성합니다. 그러나 클라이언트 동작과 연결 재사용에 따라 가용 영역에서 진행 중인 기존 연결이 완료되는 데 다소 시간이 걸릴 수 있습니다. DNS 설정 및 기타 요인에 따라 기존 연결이 몇 분 만에 완료되거나 더 오래 걸릴 수 있습니다. 자세한 내용은 Amazon Application Recovery Controller(ARC) 개발자 안내서의 클라이언트가 엔드포인트에 연결된 상태를 유지하는 시간 제한을 참조하세요.

Application Load Balancer에서 영역 전환 기능을 사용하려면 ARC 영역 전환 통합 속성이 활성화됨으로 설정되어 있어야 합니다.

Amazon Application Recovery Controller(ARC) 통합을 활성화하고 영역 전환을 활용하기 전에 다음을 검토합니다.

- 특정 로드 밸런서에 대한 영역 전환은 단일 가용 영역에 대해서만 시작할 수 있습니다. 여러 가용 영역에 대한 영역 전환은 시작할 수 없습니다.

- AWS 는 여러 인프라 문제가 서비스에 영향을 미칠 때 DNS에서 영역 로드 밸런서 IP 주소를 사전에 제거합니다. 영역 전환을 시작하기 전에 항상 현재 가용 영역 용량을 확인하세요. 로드 밸런서의 교차 영역 로드 밸런싱이 꺼져 있으며 영역 전환을 사용하여 영역 로드 밸런서 IP 주소를 제거하는 경우, 영역 전환의 영향을 받는 가용 영역도 대상 용량을 잃게 됩니다.

자세한 내용은 Amazon Application Recovery Controller(ARC) 개발자 안내서의 ARC의 영역 전환 모범 사례를 참조하세요.

교차 영역 지원 Application Load Balancer

영역 간 로드 밸런싱이 활성화된 Application Load Balancer에서 영역 전환이 시작되면 대상에 대한 모든 트래픽이 영향을 받는 가용 영역에서 차단되고 영역 IP 주소가 DNS에서 제거됩니다.

이점:

- 가용 영역 장애로부터 더 빠른 복구.
- 가용 영역에서 장애가 감지되면 트래픽을 정상 가용 영역으로 이동하는 기능입니다.
- 예상치 못한 가동 중지를 방지하기 위해 장애를 시뮬레이션하고 식별하여 애플리케이션 무결성을 테스트할 수 있습니다.

영역 전환 관리 재정의

Application Load Balancer에 속하는 대상에는 상태와 `AdministrativeOverride`독립적인 새 `TargetHealth` 상태가 포함됩니다.

Application Load Balancer에 대한 영역 전환이 시작되면 전환되는 영역 내의 모든 대상이 관리적으로 재정의된 것으로 간주됩니다. Application Load Balancer는 관리적으로 재정의된 대상으로 새 트래픽 라우팅을 중지하지만 기존 연결은 유기적으로 닫힐 때까지 그대로 유지됩니다.

가능한 `AdministrativeOverride` 상태는 다음과 같습니다.

unknown

내부 오류로 인해 상태를 전파할 수 없습니다.

no_override

대상에 대해 현재 활성 상태인 재정의가 없습니다.

zonal_shift_active

대상 가용 영역에서 영역 전환이 활성 상태입니다.

Amazon CloudFront + AWS WAF

Amazon CloudFront는를 사용하는 애플리케이션의 성능, 가용성 및 보안을 개선하는 데 도움이 되는 웹 서비스입니다 AWS. CloudFront는 Application Load Balancer를 사용하는 웹 애플리케이션의 분산된 단일 진입점 역할을 합니다. Application Load Balancer의 도달 범위를 전 세계로 확장하여 가까운 엣지 로케이션에서 사용자에게 효율적으로 서비스를 제공하여 콘텐츠 전송을 최적화하고 전 세계 사용자의 지연 시간을 줄일 수 있습니다. 이러한 엣지 로케이션의 자동 콘텐츠 캐싱은 Application Load Balancer의 부하를 크게 줄여 성능과 확장성을 개선합니다.

Elastic Load Balancing 콘솔에서 사용할 수 있는 원클릭 통합은 권장 AWS WAF 보안 보호 기능을 갖춘 CloudFront 배포를 생성하고 이를 Application Load Balancer에 연결합니다. 이 AWS WAF 보호 기능은 로드 밸런서에 도달하기 전에 일반적인 웹 악용을 차단합니다. 콘솔의 로드 밸런서 통합 탭에서 CloudFront 배포 및 해당 보안 대시보드에 액세스할 수 있습니다. 자세한 내용은 Amazon CloudFront 개발자 안내서의 CloudFront 보안 대시보드에서 보안 보호 관리 AWS WAF 및 aws.amazon.com/blogs 통합 CDN 및 보안 환경인 CloudFront 보안 대시보드 소개를 참조하세요. Amazon CloudFront

보안 모범 사례로 CloudFront의 AWS관리형 접두사 목록에서만 인바운드 트래픽을 허용하도록 인터넷 경계 Application Load Balancer의 보안 그룹을 구성하고 다른 인바운드 규칙을 제거합니다. 자세한 내용은 Amazon CloudFront 개발자 안내서>의 CloudFront 관리형 접두사 목록 사용, 요청에 사용자 지정 HTTP 헤더를 추가하도록 Amazon CloudFront 구성, 특정 헤더가 포함된 요청만 전달하도록 Application Load Balancer 구성을 참조하세요. CloudFront

> **Note**
>
> CloudFront는 미국 동부(버지니아 북부) us-east-1 리전에서만 ACM 인증서를 지원합니다. Application Load Balancer에 us-east-1 이외의 리전에서 ACM 인증서로 구성된 HTTPS 리스너가 있는 경우 CloudFront 오리진 연결을 HTTPS에서 HTTP로 변경하거나 미국 동부(버지니아 북부) 리전에서 ACM 인증서를 프로비저닝하여 CloudFront 배포에 연결해야 합니다.

AWS Global Accelerator

애플리케이션 가용성, 성능 및 보안을 최적화하려면 로드 밸런서를 위한 액셀러레이터를 생성합니다. 액셀러레이터는 AWS 글로벌 네트워크를 통해 트래픽을 클라이언트와 가장 가까운 리전에서 고정 엔드포인트 역할을 하는 고정 IP 주소로 보냅니다. AWS Global Accelerator 는 Shield Standard로 보호되므로 DDoS 공격으로 인한 애플리케이션 가동 중지 시간과 지연 시간을 최소화합니다.

자세한 내용은 AWS Global Accelerator 개발자 안내서의 로드 밸런서를 생성할 때 액셀러레이터 추가를 참조하세요.

AWS Config

로드 밸런서의 모니터링 및 규정 준수를 최적화하려면를 설정합니다 AWS Config.는 AWS 계정의 AWS 리소스 구성에 대한 자세한 보기를 AWS Config 제공합니다. 여기에는 리소스가 서로 어떻게 관련되어 있는지, 그리고 시간이 지남에 따라 구성 및 관계가 어떻게 변경되는지 확인할 수 있도록 과거에 어떻게 구성되었는지가 포함됩니다.는 감사, 규정 준수 및 문제 해결을 AWS Config 간소화합니다.

자세한 내용은 AWS Config 개발자 안내서의 What Is AWS Config?를 참조하세요.

AWS WAF

Application Load Balancer와 AWS WAF 함께를 사용하여 웹 액세스 제어 목록(웹 ACL)의 규칙을 기반으로 요청을 허용하거나 차단할 수 있습니다.

기본적으로 로드 밸런서가 응답을 받을 수 없는 경우 HTTP 500 오류를 AWS WAF반환하고 요청을 전달하지 않습니다. 연결할 수 없더라도 로드 밸런서가 대상에 요청을 전달해야 하는 경우 AWS WAF 실패 열기를 활성화 AWS WAF할 수 있습니다.

사전 정의된 웹 ACL

AWS WAF 통합을 활성화할 때 사전 정의된 규칙을 사용하여 새 웹 ACL을 자동으로 생성하도록 선택할 수 있습니다. 사전 정의된 웹 ACL에는 가장 일반적인 보안 위협에 대한 보호를 제공하는 세 가지 AWS 관리형 규칙이 포함되어 있습니다.

- `AWSManagedRulesAmazonIpReputationList` - Amazon IP 평판 목록 규칙 그룹은 일반적으로 봇 또는 기타 위협과 관련된 IP 주소를 차단합니다. 자세한 내용은 AWS WAF 개발자 안내서의 Amazon IP 평판 목록 관리형 규칙 그룹을 참조하세요.
- `AWSManagedRulesCommonRuleSet` - 핵심 규칙 집합(CRS) 규칙 그룹은 OWASP Top 10과 같은 OWASP 게시물에 설명된 자주 발생하고 위험성 높은 일부 취약성을 비롯한 광범위한 취약성 악용에 대한 보호를 제공합니다. 자세한 내용은 AWS WAF 개발자 안내서의 핵심 규칙 집합(CRS) 관리형 규칙 그룹을 참조하세요.
- `AWSManagedRulesKnownBadInputsRuleSet` - 알려진 잘못된 입력 규칙 그룹은 유효하지 않은 것으로 알려져 있으며 취약성의 악용 또는 검색과 관련된 요청 패턴을 차단합니다. 자세한 내용은 AWS WAF 개발자 안내서의 알려진 잘못된 입력 관리형 규칙 그룹을 참조하세요.

자세한 내용은 AWS WAF 개발자 안내서의 에서 웹 ACLs 사용을 참조하세요 AWS WAF.

Application Load Balancer의 속성 편집

Application Load Balancer를 생성한 후에는 속성을 편집할 수 있습니다.

로드 밸런서 속성

- 연결 유휴 제한 시간
- HTTP 클라이언트 연결 유지 기간
- 삭제 방지
- Desync Mitigation Mode
- Host header preservation

연결 유휴 제한 시간

연결 유휴 제한 시간은 로드 밸런서가 연결을 닫기 전에 송수신되는 데이터 없이 기존 클라이언트 또는 대상 연결이 비활성 상태로 유지될 수 있는 기간입니다.

파일 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. 또한 애플리케이션의 유휴 제한 시간을 로드 밸런서에 구성된 유휴 제한 시간보다 크게 설정하는 것이 좋습니다. 그렇게 하지 않으면 애플리케이션이 로드 밸런서에 대한 TCP 연결을 비정상적으로 닫을 경우, 로드 밸런서가 연결이 닫혔음을 가리키는 패킷을 받기 전에 애플리케이션에 요청을 전송할 수 있습니다. 이 경우 로드 밸런서는 HTTP 502 잘못된 게이트웨이 오류를 클라이언트에게 전송합니다.

Application Load Balancer는 HTTP/2 PING 프레임을 지원하지 않습니다. 연결 유휴 제한 시간은 재설정되지 않습니다.

기본적으로 Elastic Load Balancing에서는 로드 밸런서의 유휴 제한 시간 값을 60초(1분)로 설정합니다. 다른 유휴 제한 시간 값을 설정하려면 다음 절차를 따르세요.

콘솔을 사용하여 연결 유휴 제한 시간 값을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 트래픽 구성에서 연결 유휴 제한 시간 값을 입력합니다. 값의 범위는 1~4000초입니다.

6. 변경 사항 저장을 선택합니다.

를 사용하여 유휴 제한 시간 값을 업데이트하려면 AWS CLI

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

HTTP 클라이언트 연결 유지 기간

HTTP 클라이언트 연결 유지 기간은 Application Load Balancer가 클라이언트에 대한 지속적인 HTTP 연결을 유지하는 최대 시간입니다. 구성된 HTTP 클라이언트 연결 유지 기간이 경과하면 Application Load Balancer는 요청을 하나 더 수락한 다음 연결을 정상적으로 종료하는 응답을 반환합니다.

로드 밸런서가 전송하는 응답의 유형은 클라이언트 연결에서 사용하는 HTTP 버전에 따라 달라집니다.

- HTTP 1.x를 사용하여 연결된 클라이언트의 경우 로드 밸런서는 `Connection: close` 필드가 포함된 HTTP 헤더를 보냅니다.
- HTTP/2를 사용하여 연결된 클라이언트의 경우 로드 밸런서가 `GOAWAY` 프레임을 전송합니다.

기본적으로 Application Load Balancer는 로드 밸런서의 HTTP 클라이언트 연결 유지 기간 값을 3600초(1시간)로 설정합니다. HTTP 클라이언트 연결 유지 기간은 해제하거나 최소 60초 미만으로 설정할 수 없지만 HTTP 클라이언트 연결 유지 기간은 최대 604800초(7일)까지 늘릴 수 있습니다. Application Load Balancer는 클라이언트에 대한 HTTP 연결이 처음 설정될 때 HTTP 클라이언트 연결 유지 기간을 시작합니다. 트래픽이 없는 경우 유지 기간이 지속되며 새 연결이 설정될 때까지 재설정되지 않습니다.

영역 전환 또는 영역 자동 전환을 사용하여 로드 밸런서 트래픽이 장애가 발생한 가용 영역 외부로 전환되면 이미 활성 연결이 있는 클라이언트는 재연결 시까지 장애가 발생한 위치로 계속 요청을 전송할 수 있습니다. 더 빠른 복구를 지원하려면 더 낮은 연결 유지 기간 값을 설정하여 클라이언트가 로드 밸런서에 연결된 상태를 유지하는 시간을 제한하는 것이 좋습니다. 자세한 내용은 Amazon Application Recovery Controller(ARC) 개발자 안내서의 클라이언트가 엔드포인트에 연결된 상태를 유지하는 시간 제한을 참조하세요.

> **Note**
>
> 로드 밸런서가 Application Load Balancer의 IP 주소 유형을 `dualstack-without-public-ipv4`로 전환하면 로드 밸런서는 모든 활성 연결이 완료될 때까지 기다립니다. Application Load Balancer의 IP 주소 유형을 전환하는 데 걸리는 시간을 줄이려면 HTTP 클라이언트 연결 유지 기간을 줄이는 것이 좋습니다.

Application Load Balancer는 초기 연결 중에 HTTP 클라이언트 연결 유지 기간 값을 할당합니다. HTTP 클라이언트 연결 유지 기간을 업데이트하면 다른 HTTP 클라이언트 연결 유지 기간 값을 사용한 동시 연결이 발생할 수 있습니다. 기존 연결은 초기 연결 중에 적용된 HTTP 클라이언트 연결 유지 기간 값을 유지합니다. 새 연결은 업데이트된 HTTP 클라이언트 연결 유지 기간 값을 받습니다.

콘솔을 사용하여 클라이언트 연결 유지 기간 값을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 트래픽 구성에서 HTTP 클라이언트 유지 기간 값을 입력합니다. 값의 범위는 60~604800초입니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 클라이언트 연결 유지 기간 값을 업데이트하려면 AWS CLI

`client_keep_alive.seconds` 속성과 함께 modify-load-balancer-attributes 명령을 사용합니다.

삭제 방지

로드 밸런서가 실수로 삭제되지 않도록 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 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 구성에서 삭제 방지를 켭니다.
6. 변경 사항 저장을 선택합니다.

콘솔을 사용하여 삭제 방지 기능을 비활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 구성 페이지에서 삭제 방지를 끕니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 삭제 방지를 활성화 또는 비활성화하려면 AWS CLI

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

Desync Mitigation Mode

Desync Mitigation Mode는 HTTP Desync로 인한 문제로부터 애플리케이션을 보호합니다. 로드 밸런서는 위협 수준에 따라 각 요청을 분류하고 안전한 요청을 허용한 다음 지정한 완화 모드에서 지정한 대로 위험을 완화합니다. Desync Mitigation Mode는 Monitor, Defensive 또는 Strictest 모드입니다. 기본값은 Defensive 모드입니다. 이 모드는 애플리케이션의 가용성을 유지하면서 HTTP Desync에 대한 지속적인 완화를 제공합니다. 애플리케이션에서 RFC 7230을 준수하는 요청만 수신하도록 Strictest 모드로 전환할 수 있습니다.

http_desync_guardian 라이브러리는 HTTP Desync 공격을 방지하기 위해 HTTP 요청을 분석합니다. 자세한 내용은 GitHub의 HTTP Desync Guardian을 참조하세요.

분류

분류는 다음과 같습니다.

- 규정 준수 - 요청이 RFC 7230을 준수하며 알려진 보안 위협이 없습니다.
- 허용 가능 - 요청이 RFC 7230을 준수하지 않지만 알려진 보안 위협이 없습니다.
- 모호 - 요청이 RFC 7230을 준수하지 않지만 다양한 웹 서버와 프록시가 다르게 처리할 수 있으므로 위험을 초래합니다.
- 심각 - 요청이 높은 보안 위험을 초래합니다. 로드 밸런서는 요청을 차단하고 클라이언트에 대해 400 응답을 제공하고 클라이언트 연결을 종료합니다.

요청이 RFC 7230을 준수하지 않는 경우 로드 밸런서는 `DesyncMitigationMode_NonCompliant_Request_Count` 지표를 증가시킵니다. 자세한 내용은 Application Load Balancer 지표 단원을 참조하십시오.

각 요청에 대한 분류는 로드 밸런서 액세스 로그에 포함됩니다. 요청이 준수하지 않는 경우, 액세스 로그에 분류 사유 코드가 포함됩니다. 자세한 내용은 분류 이유 단원을 참조하십시오.

Modes

다음 표에서는 Application Load Balancer가 모드 및 분류를 기준으로 요청을 처리하는 방법에 대해 설명합니다.

Classification	Monitor 모드	Defensive 모드	Strictest 모드
규정 준수	Allowed	허용됨	Allowed
허용 가능	Allowed	Allowed	차단됨
모호	Allowed	허용[1]	차단됨
심각	Allowed	차단됨	차단됨

[1] 요청을 라우팅하지만 클라이언트 연결과 대상 연결을 종료합니다. 로드 밸런서가 Defensive 모드에서 모호한 요청을 대량으로 수신하는 경우 추가 요금이 발생할 수 있습니다. 이는 초당 새 연결 수가 증가하여 시간당 사용되는 LCU(로드 밸런서 용량 단위)에 기여하기 때문입니다. `NewConnectionCount` 지표를 사용하여 로드 밸런서가 Monitor 모드 및 Defensive 모드에서 새 연결을 설정하는 방법을 비교할 수 있습니다.

콘솔을 사용하여 Desync Mitigation Mode를 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 패킷 처리 아래의 비동기 완화 모드에서 방어적, 가장 엄격 또는 모니터링을 선택합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 비동기화 완화 모드를 업데이트하려면 AWS CLI

routing.http.desync_mitigation_mode 속성을 monitor, defensive 또는 strictest로 설정하여 modify-load-balancer-attributes 명령을 사용하세요.

Host header preservation

Preserve host header 속성을 활성화하면 Application Load Balancer가 HTTP 요청의 Host 헤더를 보존한 뒤 수정하지 않고 대상에 헤더를 보냅니다. Application Load Balancer가 여러 Host 헤더를 받는 경우 이를 모두 보존합니다. 리스너 규칙은 첫 번째 Host 헤더에만 적용됩니다.

기본적으로, Preserve host header 속성이 활성화되지 않은 경우 Application Load Balancer는 다음과 같은 방식으로 Host 헤더를 수정합니다:

호스트 헤더 보존이 활성화되어 있지 않고 리스너 포트가 기본 포트가 아닌 경우: 기본 포트(포트 80 또는 443)를 사용하지 않을 때 클라이언트가 호스트 헤더에 포트 번호를 아직 추가하지 않은 상태라면 포트 번호가 추가됩니다. 예를 들어, 리스너 포트가 8080 등의 기본 포트가 아닌 경우 Host: www.example.com를 포함하는 HTTP 요청의 Host 헤더가 Host: www.example.com:8080(으)로 수정됩니다.

호스트 헤더 보존이 활성화되어 있지 않고 리스너 포트가 기본 포트(포트 80 또는 443)인 경우: 기본 리스너 포트(포트 80 또는 443)의 경우 발신 호스트 헤더에 포트 번호가 추가되지 않습니다. 수신 호스트 헤더에 이미 있던 모든 포트 번호가 제거됩니다.

다음 표에 Application Load Balancer가 리스너 포트를 기반으로 HTTP 요청에서 호스트 헤더를 처리하는 방법에 대한 추가 예제가 나와 있습니다.

리스너 포트	요청 예제	요청의 호스트 헤더	호스트 헤더 보존이 비활성화됨(기본 동작)	호스트 헤더 보존이 활성화됨
요청이 기본 HTTP/HTTPS 리스너에서 전송됩니다.	`GET / index.html HTTP/1.1 Host: example.com`	example.com	example.com	example.com
요청이 기본 HTTP 리스너에서 전송되며 호스트 헤더에는 포트	`GET / index.html HTTP/1.1`	example.com:80	example.com	example.com:80

리스너 포트	요청 예제	요청의 호스트 헤더	호스트 헤더 보존이 비활성화됨(기본 동작)	호스트 헤더 보존이 활성화됨
(예: 80 또는 443)가 있습니다.	`Host: example.com:80`			
요청에 절대 경로가 있습니다.	`GET https://dns_name/index.html HTTP/1.1 Host: example.com`	example.com	dns_name	example.com
요청이 기본이 아닌 리스너 포트(예 8080)에서 전송됩니다.	`GET /index.html HTTP/1.1 Host: example.com`	example.com	example.com:8080	example.com
요청이 기본이 아닌 리스너 포트에서 전송되며 호스트 헤더에는 포트(예 8080)가 있습니다.	`GET /index.html HTTP/1.1 Host: example.com:8080`	example.com:8080	example.com:8080	example.com:8080

콘솔을 사용하여 호스트 헤더 보존을 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. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 패킷 처리에서 호스트 헤더 보존을 켭니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 호스트 헤더 보존을 활성화하려면 AWS CLI

`routing.http.preserve_host_header.enabled`로 설정된 `true` 속성과 함께 modify-load-balancer-attributes 명령을 사용합니다.

Application Load Balancer에 태그 지정

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

각 로드 밸런서에 여러 태그를 추가할 수 있습니다. 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 대소문자를 구분합니다. 허용되는 문자는 UTF-8로 표현할 수 있는 문자, 공백 및 숫자와 특수 문자 + - = . _ : / @입니다. 선행 또는 후행 공백을 사용하면 안 됩니다.
- 태그 이름 또는 값에 `aws:` 접두사를 사용하지 마세요. AWS 사용하기 위해 예약되어 있기 때문입니다. 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 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 태그 탭에서 태그 관리를 선택하고 다음 중 하나 이상의 작업을 수행합니다.
   a. 태그를 업데이트하려면 키 및 값 값을 수정합니다.
   b. 새로운 태그를 추가하려면 태그 추가를 선택한 다음 키 및 값에 값을 입력합니다.
   c. 태그를 삭제하려면 태그 옆의 제거(Remove) 버튼을 선택합니다.
5. 태그 업데이트를 마쳤으면 변경 사항 저장(Save changes)을 선택합니다.

를 사용하여 로드 밸런서의 태그를 업데이트하려면 AWS CLI

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 단원을 참조하십시오.

로드 밸런서를 삭제해도 등록된 대상에는 영향을 미치지 않습니다. 예를 들어 EC2 인스턴스는 계속 실행되고 대상 그룹에 계속 등록됩니다. 대상 그룹을 삭제하려면 Application Load Balancer 대상 그룹 삭제 단원을 참조하세요.

콘솔을 사용하여 로드 밸런서를 삭제하려면

1. 로드 밸런서를 가리키는 도메인을 위한 DNS 레코드가 있는 경우에는 새로운 위치를 가리키도록 하고 로드 밸런서를 삭제하기 전에 DNS 변경이 적용될 때까지 기다립니다.

   예시

   - 레코드가 300초 TTL(Time To Live)인 CNAME 레코드인 경우 다음 단계를 계속하기 전에 300초 이상 기다려야 합니다.
   - 레코드가 Route 53 Alias(A) 레코드인 경우 60초 이상 기다려야 합니다.
   - Route 53을 사용하는 경우 레코드 변경 사항이 모든 글로벌 Route 53 이름 서버에 전파되는 데 60초가 걸립니다. 업데이트 중인 레코드의 TTL 값에 이 시간을 더합니다.
2. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
3. 탐색 창에서 로드 밸런서를 선택합니다.
4. 로드 밸런서를 선택한 다음 작업, 로드 밸런서 삭제를 차례로 선택합니다.
5. 확인 메시지가 나타나면 **`confirm`**을 입력한 다음 삭제를 선택합니다.

를 사용하여 로드 밸런서를 삭제하려면 AWS CLI

delete-load-balancer 명령을 사용합니다.

Application Load Balancer 리소스 맵 보기

Application Load Balancer 리소스 맵은 연결된 리스너, 규칙, 대상 그룹 및 대상을 포함하여 로드 밸런서의 아키텍처를 대화형으로 표시합니다. 또한 리소스 맵은 모든 리소스 간의 관계 및 라우팅 경로를 강조 표시하여 로드 밸런서의 구성을 시각적으로 표현합니다.

콘솔을 사용하여 Application Load Balancer의 리소스 맵을 보려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리소스 맵 탭을 선택하여 로드 밸런서의 리소스 맵을 표시합니다.

리소스 맵 구성 요소

맵 보기

Application Load Balancer 리소스 맵에는 개요 및 비정상 대상 맵이라는 두 가지 보기가 있습니다. 개요는 기본적으로 선택되며 로드 밸런서의 모든 리소스를 표시합니다. 비정상 대상 맵 보기를 선택하면 비정상 대상과 연결된 리소스만 표시됩니다.

비정상 대상 맵 보기는 상태 확인에 실패한 대상의 문제를 해결하는 데 사용할 수 있습니다. 자세한 내용은 리소스 맵을 사용하여 비정상 대상 문제 해결 단원을 참조하십시오.

리소스 그룹

Application Load Balancer 리소스 맵에는 리소스 유형별로 하나씩 4개의 리소스 그룹이 있습니다. 리소스 그룹은 리스 , 규칙, 대상 그룹, 대상입니다.

리소스 타일

그룹 내의 각 리소스에는 해당 리소스의 세부 정보를 표시하는 자체 타일이 있습니다.

- 리소스 타일 위로 마우스를 가져가면 리소스와 다른 리소스 간의 관계가 강조 표시됩니다.
- 리소스 타일을 선택하면 리소스와 다른 리소스 간의 관계가 강조 표시되고 해당 리소스의 추가 세부 정보가 표시됩니다.
  - 규칙 조건: 각 규칙의 조건입니다.
  - 대상 그룹 상태 요약: 각 상태별로 등록된 대상 수입니다.
  - 대상 상태 대상의 현재 상태 및 설명입니다.

> **Note**
>
> 리소스 세부 정보 표시를 해제하여 리소스 맵 내에서 추가 세부 정보를 숨길 수 있습니다.

- 각 리소스 타일에는 선택할 경우 해당 리소스의 세부 정보 페이지로 이동하는 링크가 포함되어 있습니다.
  - 리스너 - 리스너 프로토콜:포트를 선택합니다. 예: `HTTP:80`
  - 규칙 - 규칙 작업을 선택합니다. 예: `Forward to target group`
  - 대상 그룹 - 대상 그룹 이름을 선택합니다. 예: `my-target-group`
  - 대상 - 대상 ID를 선택합니다. 예: `i-1234567890abcdef0`

리소스 맵 내보내기

내보내기를 선택하면 Application Load Balancer 리소스 맵의 현재 보기를 PDF로 내보낼 수 있는 옵션이 제공됩니다.

Application Load Balancer 용량 예약

로드 밸런서 용량 단위(LCU) 예약을 사용하면 로드 밸런서에 대한 정적 최소 용량을 예약할 수 있습니다. Application Load Balancer는 감지된 워크로드를 지원하고 용량 요구 사항을 충족하도록 자동으로 확장됩니다. 최소 용량이 구성된 경우 로드 밸런서는 수신된 트래픽에 따라 계속 확장 또는 축소되지만 구성된 최소 용량보다 용량이 낮아지지 않습니다.

다음과 같은 상황에서는 LCU 예약을 사용하는 것이 좋습니다.

- 갑작스럽고 비정상적인 트래픽이 많아지고 로드 밸런서가 이벤트 중에 갑작스런 트래픽 급증을 지원할 수 있도록 하려는 이벤트가 예정되어 있습니다.
- 단기간 동안 워크로드의 특성으로 인해 급증하는 트래픽이 예측할 수 없습니다.
- 특정 시작 시간에 서비스를 온보딩하거나 마이그레이션하도록 로드 밸런서를 설정하고 있으며 Auto Scaling이 적용될 때까지 기다리는 대신 대용량으로 시작해야 합니다.
- 서비스 수준 계약 또는 규정 준수 요구 사항을 충족하려면 최소 용량을 유지해야 합니다.
- 로드 밸런서 간에 워크로드를 마이그레이션하고 소스의 규모와 일치하도록 대상을 구성하려고 합니다.

필요한 용량 추정

로드 밸런서에 예약해야 하는 용량을 결정할 때는 로드 테스트를 수행하거나 예상되는 트래픽을 나타내는 과거 워크로드 데이터를 검토하는 것이 좋습니다. Elastic Load Balancing 콘솔을 사용하면 검토된 트래픽을 기반으로 예약해야 하는 용량을 추정할 수 있습니다.

또는 CloudWatch 지표를 활용하여 필요한 용량 수준을 PeakLCUs 결정할 수 있습니다. 지표는 로드 PeakLCUs 밸런서가 워크로드를 지원하기 위해 모든 조정 차원에 걸쳐 조정해야 하는 트래픽 패턴의 피크를 고려합니다. 지표는 트래픽의 결제 PeakLCUs 차원만 집계하는 ConsumedLCUs 지표와 다릅니다. 로드 PeakLCUs 밸런서 조정 중에 LCU 예약이 적절한지 확인하려면 지표를 사용하는 것이 좋습니다. 용량을 추정할 때는 분당 Sum을 사용합니다PeakLCUs.

참조할 과거 워크로드 데이터가 없고 로드 테스트를 수행할 수 없는 경우 LCU 예약 계산기를 사용하여 필요한 용량을 추정할 수 있습니다. LCU 예약 계산기는 과거 워크로드 AWS 관찰을 기반으로 데이터를 사용하며 특정 워크로드를 나타내지 않을 수 있습니다. 자세한 내용은 Load Balancer 용량 단위 예약 계산기를 참조하세요.

LCU 예약 할당량

계정에 LCUs. 자세한 내용은 the section called “Load Balancer서 용량 단위” 단원을 참조하십시오.

Application Load Balancer서 용량 단위 예약 요청

LCU 예약을 사용하기 전에 다음을 검토합니다.

- 용량은 리전 수준에서 예약되며 가용 영역에 균등하게 분산됩니다. LCU 예약을 켜기 전에 각 가용 영역에 균등하게 분산된 대상이 충분한지 확인합니다.
- LCU 예약 요청은 선착순으로 이행되며 해당 시점의 영역에 사용 가능한 용량에 따라 달라집니다. 대부분의 요청은 일반적으로 몇 분 내에 이행되지만 최대 몇 시간이 걸릴 수 있습니다.
- 기존 예약을 업데이트하려면 이전 요청을 프로비저닝하거나 실패해야 합니다. 필요한 만큼 예약 용량을 늘릴 수 있지만 예약 용량은 하루에 두 번만 줄일 수 있습니다.
- 예약되거나 프로비저닝된 용량에 대한 요금은 종료되거나 취소될 때까지 계속 발생합니다.

LCU 예약 요청

이 절차의 단계에서는 로드 밸런서에서 LCU 예약을 요청하는 방법을 설명합니다.

콘솔을 사용하여 LCU 예약을 요청하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.

3. 로드 밸런서를 선택합니다.
4. 용량 탭에서 LCU 예약 편집을 선택합니다.
5. 기록 참조 기반 추정치를 선택한 다음 드롭다운 목록에서 로드 밸런서를 선택합니다.
6. 참조 기간을 선택하여 권장 예약 LCU 수준을 확인합니다.
7. 과거 참조 워크로드가 없는 경우 수동 추정을 선택하고 예약할 LCUs 수를 입력할 수 있습니다.
8. 저장을 선택합니다.

를 사용하여 LCU 예약을 요청하려면 AWS CLI

modify-capacity-reservation 명령을 사용합니다.

Application Load Balancer 용량 단위 예약 업데이트 또는 종료

LCU 예약 업데이트 또는 종료

이 절차의 단계에서는 로드 밸런서에서 LCU 예약을 업데이트하거나 종료하는 방법을 설명합니다.

콘솔을 사용하여 LCU 예약을 업데이트하거나 종료하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 용량 탭에서 예약 상태가 프로비저닝됨인지 확인합니다.
    a. LCU 예약을 업데이트하려면 LCU 예약 편집을 선택합니다.
    b. LCU 예약을 종료하려면 용량 취소를 선택합니다.

를 사용하여 LCU 예약을 업데이트하거나 종료하려면 AWS CLI

modify-capacity-reservation 명령을 사용합니다.

Application Load Balancer에 대한 로드 밸런서 용량 단위 예약 모니터링 Application Load Balancer

예약 상태

LCU 예약에는 네 가지 사용 가능한 상태가 있습니다.

- pending - 프로비저닝 중인 예약을 나타냅니다.
- 프로비저닝됨 - 예약된 용량을 사용할 준비가 되었으며 사용할 수 있음을 나타냅니다.
- failed - 현재 요청을 완료할 수 없음을 나타냅니다.
- 리밸런싱 - 가용 영역이 추가 또는 제거되었고 로드 밸런서가 용량을 리밸런싱하고 있음을 나타냅니다.

예약 LCU

ReservedLCUs 지표는 분당 기준으로 보고됩니다. 용량은 시간 단위로 예약됩니다. 예를 들어 LCU 예약이 6,000인 경우의 총 1시간ReservedLCUs은 6,000이고 총 1분은 100입니다. 예약된 LCU 사용률을 확인하려면 PeakLCUs 지표를 참조하세요. CloudWatch 경보를 설정하여 Sum의 분당 PeakLCUs를 예약된 용량 값과 비교하거나 Sum의 시간당 ReservedLCUs를 비교하여 필요에 맞는 충분한 용량을 예약했는지 확인할 수 있습니다.

예약 용량 모니터링

이 프로세스의 단계에서는 로드 밸런서에서 LCU 예약의 상태를 확인하는 방법을 설명합니다.

콘솔을 사용하여 LCU 예약의 상태를 보려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 용량 탭에서 예약 상태 및 예약 LCU 값을 볼 수 있습니다.

를 사용하여 LCU 예약의 상태를 모니터링하려면 AWS CLI

describe-capacity-reservation 명령을 사용합니다.

Application Load Balancer를 위한 리스너

리스너는 구성한 프로토콜 및 포트를 사용하여 연결 요청을 확인하는 프로세스입니다. Application Load Balancer를 사용하기 전에 리스너를 최소 하나 이상 추가해야 합니다. 로드 밸런서에 리스너가 없는 경우 클라이언트로부터 트래픽을 수신할 수 없습니다. 리스너에 대해 정의한 규칙에 따라 로드 밸런서가 EC2 인스턴스와 같은 등록 대상에 요청을 라우팅하는 방법이 결정됩니다.

내용

- 리스너 구성
- 리스너 속성
- 리스너 규칙
- 규칙 작업 유형
- 규칙 조건 형식
- HTTP 헤더 및 Application Load Balancer
- Application Load Balancer용 HTTP 리스너 생성
- Application Load Balancer용 SSL 인증서
- Application Load Balancer 보안 정책
- Application Load Balancer용 HTTPS 리스너 생성
- Application Load Balancer를 위한 리스너 규칙
- Application Load Balancer용 HTTPS 리스너 업데이트
- Application Load Balancer에서 TLS를 사용한 상호 인증
- Application Load Balancer를 사용하여 사용자 인증
- Application Load Balancer 리스너 및 규칙에 대한 태그
- Application Load Balancer를 위한 리스너 삭제
- Application Load Balancer에 대한 HTTP 헤더 수정

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 내용은 Application Load Balancer용 HTTPS 리스너 생성 단원을 참조하십시오.

대상이 로드 밸런서 대신 HTTPS 트래픽을 해독하도록 하려면 포트 443에서 수신하는 TCP 리스너가 있는 Network Load Balancer를 생성합니다. TCP 리스너를 사용하여 로드 밸런서는 암호화된 트래픽을 해독하지 않고 대상으로 전달합니다. Network Load Balancer에 대한 자세한 정보는 Network Load Balancer 사용 설명서를 참조하세요.

WebSocket

Application Load Balancer는 WebSockets에 대한 기본 지원을 제공합니다. HTTP 연결 업그레이드를 사용하여 기존 HTTP/1.1 연결을 WebSocket(`ws` 또는 `wss`) 연결로 업그레이드할 수 있습니다. 업그레이드하면, (로드 밸런서 및 대상에 대한) 요청에 사용되는 TCP 연결이 로드 밸런서를 통해 클라이언트와 대상 간의 지속적인 WebSocket 연결이 됩니다. HTTP 및 HTTPS 리스너 모두에서 WebSockets를 사용할 수 있습니다. 리스너에 대해 선택한 옵션은 HTTP 트래픽뿐 아니라 WebSocket 연결에도 적용됩니다. 자세한 내용은 Amazon CloudFront 개발자 안내서의 WebSocket 프로토콜의 작동 방식을 참조하세요.

HTTP/2

Application Load Balancer는 HTTPS 리스너를 통해 HTTP/2에 대한 기본 지원을 제공합니다. 하나의 HTTP/2 연결을 이용해 최대 128개의 요청을 동시에 전송할 수 있습니다. 프로토콜 버전을 사용하면 HTTP/2를 사용하여 대상에 요청을 보낼 수 있습니다. 자세한 내용은 프로토콜 버전 단원을 참조하십시오. 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.

Application Load Balancer에 대한 상호 TLS 인증은 패스스루 및 확인 모드 모두에서 HTTP/2를 지원합니다. 자세한 내용은 Application Load Balancer에서 TLS를 사용한 상호 인증 단원을 참조하십시오.

자세한 내용은 Elastic Load Balancing 사용 설명서의 라우팅 요청을 참조하세요.

리스너 속성

다음은 Application Load Balancer의 리스너 속성입니다.

`routing.http.request.x_amzn_mtls_clientcert_serial_number.header_name`

X-Amzn-Mtls-Clientcert-Serial-Number HTTP 요청 헤더의 헤더 이름을 수정할 수 있습니다.

`routing.http.request.x_amzn_mtls_clientcert_issuer.header_name`

X-Amzn-Mtls-Clientcert-Issuer HTTP 요청 헤더의 헤더 이름을 수정할 수 있습니다.

`routing.http.request.x_amzn_mtls_clientcert_subject.header_name`

X-Amzn-Mtls-Clientcert-Subject HTTP 요청 헤더의 헤더 이름을 수정할 수 있습니다.

`routing.http.request.x_amzn_mtls_clientcert_validity.header_name`

X-Amzn-Mtls-Clientcert-Validity HTTP 요청 헤더의 헤더 이름을 수정할 수 있습니다.

`routing.http.request.x_amzn_mtls_clientcert_leaf.header_name`

X-Amzn-Mtls-Clientcert-Leaf HTTP 요청 헤더의 헤더 이름을 수정할 수 있습니다.

`routing.http.request.x_amzn_mtls_clientcert.header_name`

X-Amzn-Mtls-Clientcert HTTP 요청 헤더의 헤더 이름을 수정할 수 있습니다.

`routing.http.request.x_amzn_tls_version.header_name`

X-Amzn-Tls-Version HTTP 요청 헤더의 헤더 이름을 수정할 수 있습니다.

`routing.http.request.x_amzn_tls_cipher_suite.header_name`

X-Amzn-Tls-Cipher-Suite HTTP 요청 헤더의 헤더 이름을 수정할 수 있습니다.

`routing.http.response.server.enabled`

HTTP 응답 서버 헤더를 허용하거나 제거할 수 있습니다.

`routing.http.response.strict_transport_security.header_value`

브라우저에 사이트는 HTTPS를 통해서만 액세스해야 하며 향후 HTTP를 사용하여 사이트에 액세스하려는 모든 시도는 HTTPS로 자동 변환되어야 함을 알립니다.

`routing.http.response.access_control_allow_origin.header_value`

서버에 액세스할 수 있는 오리진을 지정합니다.

`routing.http.response.access_control_allow_methods.header_value`

다른 오리진에서 서버에 액세스할 때 허용되는 HTTP 메서드를 반환합니다.

`routing.http.response.access_control_allow_headers.header_value`

요청 중에 사용할 수 있는 헤더를 지정합니다.

`routing.http.response.access_control_allow_credentials.header_value`

요청 시 브라우저에 쿠키 또는 인증과 같은 자격 증명을 포함해야 하는지 여부를 나타냅니다.

`routing.http.response.access_control_expose_headers.header_value`

브라우저가 요청 클라이언트에 노출할 수 있는 헤더를 반환합니다.

`routing.http.response.access_control_max_age.header_value`

사전 요청의 결과를 캐시할 수 있는 시간을 초 단위로 지정합니다.

`routing.http.response.content_security_policy.header_value`

브라우저에서 적용되는 제한을 지정하여 특정 유형의 보안 위협의 위험을 최소화합니다.

`routing.http.response.x_content_type_options.header_value`

Content-Type 헤더에 광고된 MIME 유형을 따라야 하며 변경해서는 안 되는지 여부를 나타냅니다.

`routing.http.response.x_frame_options.header_value`

브라우저가 프레임, iframe, 임베드 또는 객체에서 페이지를 렌더링할 수 있는지 여부를 나타냅니다.

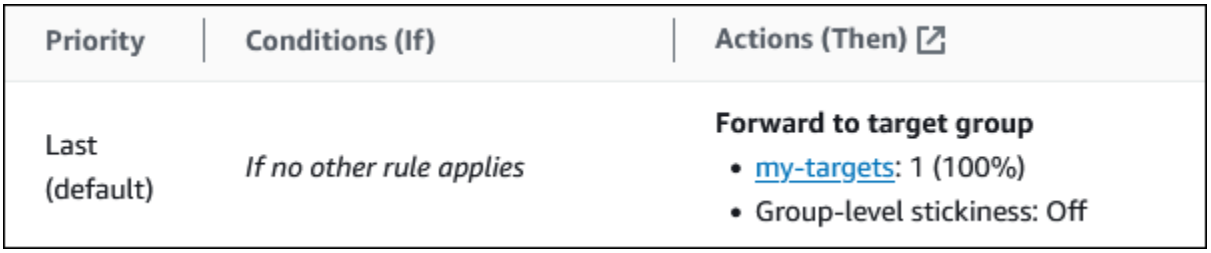
리스너 규칙

모든 리스너에는 기본 동작이 있으며, 이를 기본 규칙이라고도 합니다. 기본 규칙은 삭제할 수 없으며 항상 마지막에 수행됩니다. 추가 규칙을 생성할 수 있으며 추가 규칙은 우선 순위, 하나 이상의 작업, 하나 이상의 조건으로 구성됩니다. 언제든 규칙을 추가하거나 편집할 수 있습니다. 자세한 내용은 규칙 편집 단원을 참조하세요.

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 우선 순위 업데이트 단원을 참조하세요.

규칙 작업

각 규칙 작업에는 작업을 수행하는 데 필요한 유형, 우선 순위 및 정보가 있습니다. 자세한 내용은 규칙 작업 유형 단원을 참조하십시오.

규칙 조건

각 규칙 조건에는 유형과 구성 정보가 있습니다. 규칙에 대한 조건이 충족되면 작업이 수행됩니다. 자세한 내용은 규칙 조건 형식 단원을 참조하세요.

규칙 작업 유형

리스너 규칙에 대해 지원되는 작업 유형은 다음과 같습니다.

`authenticate-cognito`

[HTTPS 리스너] Amazon Cognito를 사용하여 사용자를 인증합니다. 자세한 내용은 Application Load Balancer를 사용하여 사용자 인증 단원을 참조하세요.

`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 단원을 참조하세요.

`forward`

요청을 지정된 대상 그룹으로 전달합니다. 자세한 내용은 전달 작업 단원을 참조하세요.

`redirect`

한 URL의 요청을 다른 URL로 리디렉션합니다. 자세한 내용은 리디렉션 작업 단원을 참조하십시오.

가장 낮은 우선 순위가 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.

프로토콜 버전이 gRPC 또는 HTTP/2인 경우, 지원되는 유일한 작업은 `forward` 작업입니다.

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 단원을 참조하세요. 성공한 `fixed-response` 작업의 개수는 `HTTP_Fixed_Response_Count` 지표에 보고됩니다. 자세한 내용은 Application Load Balancer 지표 단원을 참조하십시오.

Example 에 대한 고정 응답 작업의 예 AWS CLI

규칙을 만들거나 수정할 때 작업을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하세요. 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 하나 이상의 대상 그룹에 요청을 라우팅할 수 있습니다. `forward` 작업에 대해 여러 대상 그룹을 지정하는 경우 각 대상 그룹에 대해 가중치를 지정해야 합니다. 각 대상 그룹 가중치는 0과 999 사이의 값입니다. 가중 대상 그룹이 있는 리스너 규칙과 일치하는 요청은 가중치를 기준으로 이러한 대상 그룹에 배포됩니다. 예를 들어, 각각 가중치가 10인 두 개의 대상 그룹을 지정하면 각 대상 그룹은 요청을 절반씩 받습니다. 가중치가 10인 대상 그룹과 가중치가 20인 대상 그룹 두 개를 지정하면 가중치가 20인 대상 그룹이 다른 대상 그룹보다 두 배 많은 요청을 받습니다.

가중치가 적용된 대상 그룹 간에 트래픽을 분산하도록 규칙을 구성하고 대상 그룹 중 하나가 비어 있거나 비정상 대상만 있는 경우 로드 밸런서는 정상 대상이 있는 대상 그룹으로 자동으로 장애 조치하지 않습니다.

기본적으로 가중 대상 그룹 간에 트래픽을 배포하도록 규칙을 구성한다고 해서 고정 세션이 반드시 적용되는 것은 아닙니다. 고정 세션이 적용되도록 하려면 규칙에 대해 대상 그룹 고정을 활성화합니다. 로드 밸런서는 요청을 가중 대상 그룹에 처음 라우팅할 때 선택된 대상 그룹에 대한 정보를 인코딩하고 쿠키를 암호화하며 클라이언트에 대한 응답으로 쿠키를 포함하는 AWSALBTG라는 쿠키를 생성합니다. 클라이언트는 로드 밸런서에 대한 후속 요청에서 수신하는 쿠키를 포함해야 합니다. 로드 밸런서가 대상 그룹 고정 기능이 활성화된 규칙과 일치하고 쿠키를 포함하는 요청을 받으면 요청이 쿠키에 지정된 대상 그룹으로 라우팅됩니다.

Application Load Balancer는 URL로 인코딩된 쿠키 값을 지원하지 않습니다.

CORS(Cross-Origin Resource Sharing) 요청의 경우 고정을 활성화하려면 `SameSite=None;Secure`가 필요합니다. 이 경우 Elastic Load Balancing은 두 번째 쿠키인 AWSALBTGCORS를 생성합니다. 이 쿠키에는 원래 고정 쿠키와 동일한 정보와 `SameSite` 속성이 포함되어 있습니다. 클라이언트는 두 쿠키를 모두 수신합니다.

Example 하나의 대상 그룹이 있는 전달 작업의 예

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

```
[
  {
      "Type": "forward",
      "ForwardConfig": {
          "TargetGroups": [
              {
                  "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067"
              }
          ]
      }
  }
]
```

Example 두 개의 가중 대상 그룹이 있는 전달 작업의 예

다음 작업은 각 대상 그룹의 가중치를 기준으로 지정된 두 대상 그룹에 요청을 전달합니다.

```
[
  {
      "Type": "forward",
      "ForwardConfig": {
          "TargetGroups": [
              {
                  "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/blue-targets/73e2d6bc24d8a067",
                  "Weight": 10
              },
              {
                  "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/green-targets/09966783158cda59",
                  "Weight": 20
              }
          ]
      }
  }
]
```

Example 고정성이 활성화된 전달 작업의 예

대상 그룹이 여러 개인 전달 작업이 있고 하나 이상의 대상 그룹에 고정 세션이 활성화되어 있으면 대상 그룹 고정을 활성화해야 합니다.

다음 작업은 대상 그룹 고정 기능을 사용하여 요청을 지정된 두 대상 그룹으로 전달합니다. 고정 쿠키가 포함되지 않은 요청은 각 대상 그룹의 가중치에 따라 라우팅됩니다.

```
[
  {
      "Type": "forward",
      "ForwardConfig": {
          "TargetGroups": [
              {
                  "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/blue-targets/73e2d6bc24d8a067",
                  "Weight": 10
              },
              {
                  "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/green-targets/09966783158cda59",
                  "Weight": 20
              }
```

```
            ],
            "TargetGroupStickinessConfig": {
                "Enabled": true,
                "DurationSeconds": 1000
            }
        }
    }
]
```

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 최대 128자까지 가능하며 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 최대 길이는 128자입니다.

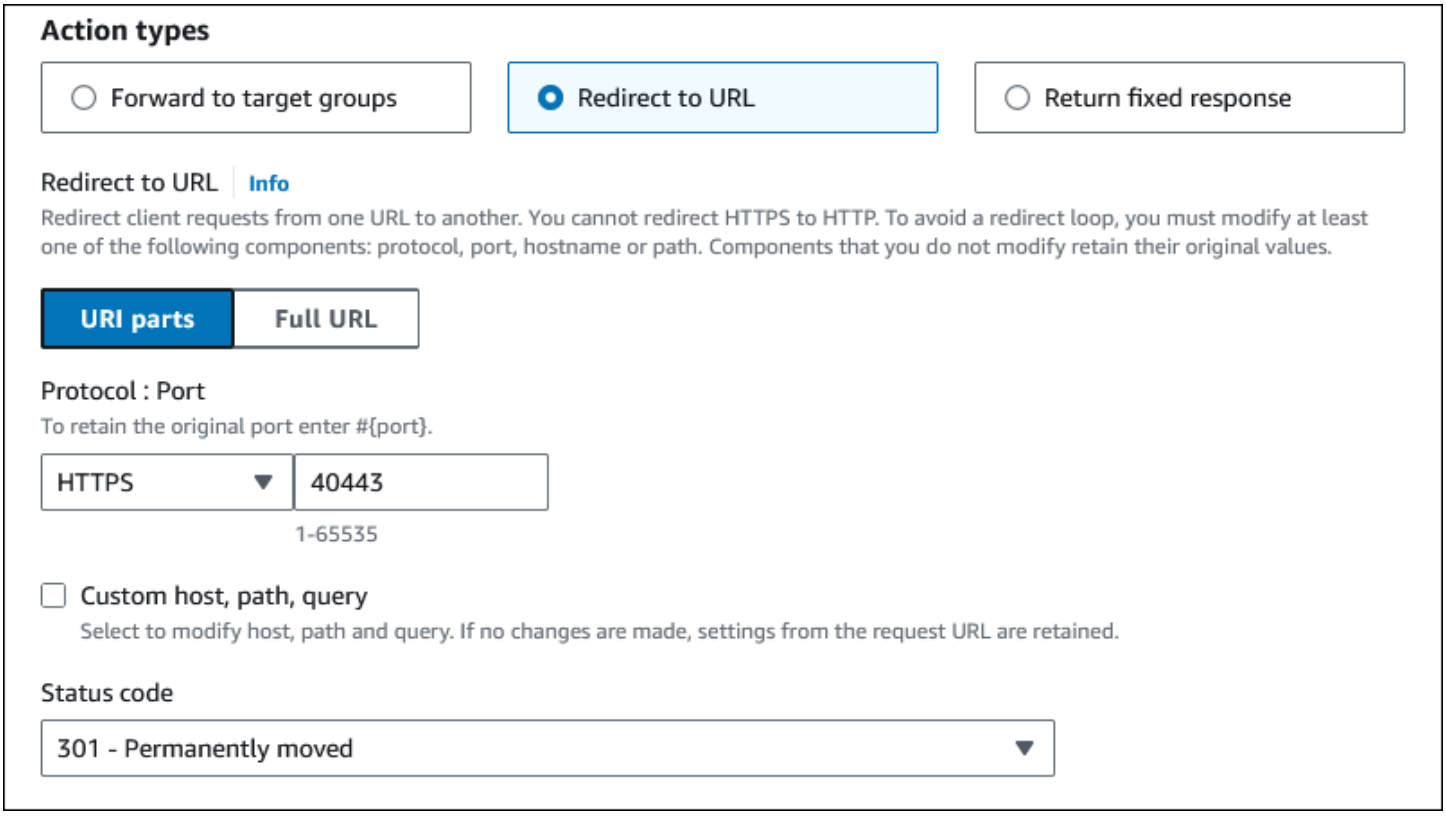
다음 예약 키워드를 사용하여 원래 URL의 URI 구성 요소를 대상 URL에 다시 사용할 수 있습니다.

- `#{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.

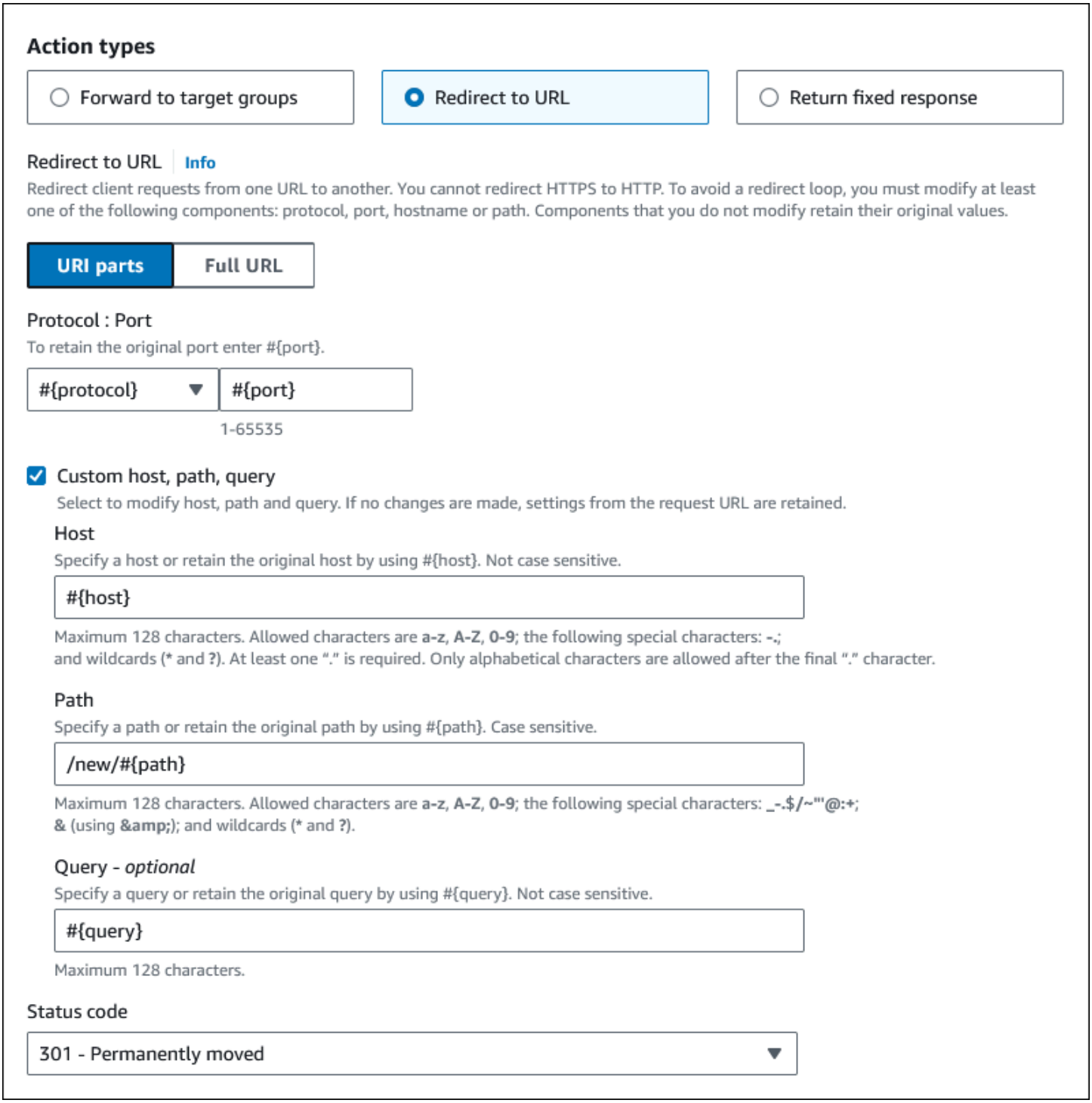
`redirect`작업이 수행되면 액세스 로그에 작업이 기록됩니다. 자세한 내용은 액세스 로그 항목 단원을 참조하세요. 성공한 `redirect` 작업의 개수는 `HTTP_Redirect_Count` 지표에 보고됩니다. 자세한 내용은 Application Load Balancer 지표 단원을 참조하세요.

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 에 대한 리디렉션 작업 예제 AWS CLI

규칙을 만들거나 수정할 때 작업을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하세요. 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 단원을 참조하세요.

`http-header`

각 요청의 HTTP 헤더를 기반으로 라우팅합니다. 자세한 내용은 HTTP 헤더 조건 단원을 참조하세요.

`http-request-method`

각 요청의 HTTP 요청 메서드를 기반으로 라우팅합니다. 자세한 내용은 HTTP 요청 메서드 조건 단원을 참조하세요.

`path-pattern`

요청 URL의 경로 패턴을 기반으로 라우팅합니다. 자세한 내용은 경로 조건 단원을 참조하세요.

`query-string`

쿼리 문자열의 키/값 페어 또는 값을 기반으로 라우팅합니다. 자세한 내용은 쿼리 문자열 조건 단원을 참조하세요.

`source-ip`

각 요청의 소스 IP 주소를 기반으로 라우팅합니다. 자세한 내용은 소스 IP 주소 조건 단원을 참조하세요.

각 규칙에는 선택적으로 `host-header`, `http-request-method`, `path-pattern` 및 `source-ip` 조건 중 하나 이상을 포함할 수 있습니다. 또한 각 규칙에는 선택적으로 `http-header` 및 `query-string` 조건 중 하나 이상을 포함할 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.

규칙은 표시되는 ASCII 문자에만 적용되며 제어 문자(0x00에서 0x1f 및 0x7f)는 제외됩니다.

데모는 고급 요청 라우팅을 참조하세요.

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.

Application Load Balancer 속성`routing.http.drop_invalid_header_fields`이 활성화되면 정규식()을 준수하지 않는 헤더 이름이 삭제됩니다`A-Z,a-z,0-9`. 정규식을 준수하지 않는 헤더 이름도 추가할 수 있습니다.

Example 에 대한 HTTP 헤더 조건 예제 AWS CLI

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하세요. 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 에 대한 HTTP 메서드 조건 예제 AWS CLI

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하세요. 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). 그러면 단일 로드 밸런서를 사용하여 여러 하위 도메인과 여러 최상위 도메인을 지원할 수 있습니다.

호스트 이름은 대/소문자를 구별하지 않고 최대 128자까지 가능하며 다음과 같은 문자를 포함할 수 있습니다.

- A~Z, a~z, 0~9
- - .
- * (0개 이상의 문자에 해당)
- ?(정확히 한 글자에 해당)

'.' 문자를 하나 이상 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 에 대한 호스트 헤더 조건 예제 AWS CLI

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하세요. 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. 표시되는 ASCII 문자에만 적용되며 제어 문자(0x00에서 0x1F 및 0x7f)는 제외됩니다.

규칙 평가는 URI 정규화가 발생한 후에만 수행됩니다.

경로 패턴은 대/소문자를 구별하고 최대 128자이며 다음과 같은 문자를 포함할 수 있습니다.

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

프로토콜 버전이 gRPC인 경우 조건은 패키지, 서비스 또는 메서드에 따라 달라질 수 있습니다.

HTTP 경로 패턴의 예

- `/img/*`
- `/img/*/pics`

gRPC 경로 패턴의 예

- /package
- /package.service
- /package.service/method

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

Example 에 대한 경로 패턴 조건 예제 AWS CLI

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하세요. 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 에 대한 쿼리 문자열 조건 예제 AWS CLI

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하세요. 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. 소스 IP 규칙 조건에 대한 `255.255.255.255/32` CIDR을 지정할 수 없습니다.

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 에 대한 소스 IP 조건 예제 AWS CLI

규칙을 만들거나 수정할 때 조건을 지정할 수 있습니다. 자세한 내용은 create-rule 및 modify-rule 명령을 참조하세요. 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"]
      }
  }
]
```

HTTP 헤더 및 Application Load Balancer

HTTP 요청 및 HTTP 응답은 헤더 필드를 사용하여 HTTP 메시지에 대한 정보를 전송합니다. HTTP 헤더가 자동으로 추가됩니다. 헤더 필드는 콜론으로 구분된 이름-값 페어이며 CR(캐리지 리턴) 및 LF(줄 바꿈)로 구분됩니다. HTTP 헤더 필드의 표준 집합은 RFC 2616, 메시지 헤더에 정의되어 있습니다. 자동으로 추가되고 애플리케이션에서 널리 사용되는 비표준 HTTP 헤더도 제공되고 있습니다. 일부 비표준 HTTP 헤더는 `X-Forwarded` 접두사를 가지고 있습니다. Application Load Balancer는 다음 `X-Forwarded` 헤더를 지원합니다.

HTTP 연결에 대한 자세한 내용은 Elastic Load Balancing 사용 설명서의 라우팅 요청을 참조하세요.

X-Forwarded 헤더

- X-Forwarded-For

- X-Forwarded-Proto
- X-Forwarded-Port

X-Forwarded-For

X-Forwarded-For 요청 헤더는 HTTP 또는 HTTPS 로드 밸런서를 사용할 때 클라이언트의 IP 주소를 식별하는 데 도움을 줍니다. 로드 밸런서가 클라이언트와 서버 간의 트래픽을 가로채기 때문에 서버 액세스 로그에 로드 밸런서의 IP 주소만 포함됩니다. 클라이언트의 IP 주소를 확인하려면 routing.http.xff_header_processing.mode 속성을 사용하십시오. 이 속성을 사용하여, Application Load Balancer가 대상에 요청을 보내기 전에 HTTP 요청의 X-Forwarded-For 헤더를 수정, 보존 또는 제거할 수 있습니다. 이 속성에 사용 가능한 값은 append, preserve 및 remove입니다. 이 속성의 기본값은 append입니다.

> **Important**
>
> 보안 위험이 발생할 수 있으므로 X-Forwarded-For 헤더는 신중하게 사용해야 합니다. 항목은 네트워크 내에서 적절하게 보호되는 시스템이 추가하는 경우에만 신뢰할 수 있는 것으로 간주될 수 있습니다.

Append

기본적으로, Application Load Balancer는 X-Forwarded-For 요청 헤더에 클라이언트의 IP 주소를 저장하고 이 헤더를 서버로 전달합니다. X-Forwarded-For 요청 헤더가 원본 요청에 포함되지 않은 경우, 로드 밸런서는 클라이언트 IP 주소를 요청 값으로 사용하여 하나를 생성합니다. 그러지 않으면, 로드 밸런서가 클라이언트 IP 주소를 기존 헤더에 추가하고 헤더를 서버로 전달합니다. X-Forwarded-For요청 헤더에는 쉼표로 구분된 여러 IP 주소가 포함될 수 있습니다.

X-Forwarded-For 요청 헤더의 형식은 다음과 같습니다.

```
X-Forwarded-For: client-ip-address
```

다음은 IP 주소가 203.0.113.7인 클라이언트의 X-Forwarded-For 요청 헤더입니다.

```
X-Forwarded-For: 203.0.113.7
```

다음은 IPv6 주소가 X-Forwarded-For인 클라이언트의 2001:DB8::21f:5bff:febf:ce22:8a2e 요청 헤더입니다.

```
X-Forwarded-For: 2001:DB8::21f:5bff:febf:ce22:8a2e
```

클라이언트 포트 보존 속성(routing.http.xff_client_port.enabled)이 로드 밸런서에서 활성화된 경우, X-Forwarded-For 요청 헤더에는 client-ip-address에 추가되는 client-port-number을(를) 포함합니다(콜론으로 구분). 이후 헤더의 형식은 다음과 같습니다.

```
IPv4 -- X-Forwarded-For: client-ip-address:client-port-number
```

```
IPv6 -- X-Forwarded-For: [client-ip-address]:client-port-number
```

IPv6의 경우 로드 밸런서가 client-ip-address을(를) 기존 헤더에 추가하면 주소를 대괄호로 묶습니다.

다음은 IPv4 주소가 12.34.56.78이고 포트 번호가 8080인 클라이언트의 X-Forwarded-For 요청 헤더입니다.

```
X-Forwarded-For: 12.34.56.78:8080
```

다음은 IPv6 주소가 2001:db8:85a3:8d3:1319:8a2e:370:7348이고 포트 번호가 8080인 클라이언트의 X-Forwarded-For 요청 헤더입니다.

```
X-Forwarded-For: [2001:db8:85a3:8d3:1319:8a2e:370:7348]:8080
```

Preserve

속성의 preserve 모드는 대상으로 전송되기 전에 HTTP 요청의 X-Forwarded-For 헤더가 어떤 방식으로도 수정되지 않도록 합니다.

제거

속성의 remove 모드는 대상으로 전송되기 전에 HTTP 요청의 X-Forwarded-For 헤더를 제거합니다.

> **Note**
>
> 클라이언트 포트 보존 속성(routing.http.xff_client_port.enabled)을 활성화하고 routing.http.xff_header_processing.mode 속성에 preserve 또는 remove을(를)

선택할 경우 Application Load Balancer는 클라이언트 포트 보존 속성을 재정의합니다. 선택하는 모드에 따라 대상으로 전송되기 전에 X-Forwarded-For 헤더가 변경되지 않거나 제거됩니다.

다음 표에 append, preserve, 또는 remove 모드 중 하나를 선택할 때 대상이 받는 X-Forwarded-For 헤더의 예제가 나와 있습니다. 이 예제에서 마지막 홉의 IP 주소는 127.0.0.1입니다.

요청 설명	요청 예제	XFF(**append** 모드)	XFF(**preserve** 모드)	XFF(**remove** 모드)
요청이 XFF 헤더 없이 전송됩니다	`GET / index.html HTTP/1.1 Host: example.com`	`X-Forwarded-For: 127.0.0.1`	존재하지 않음	존재하지 않음
요청이 XFF 헤더 및 클라이언트 IP 주소를 포함하여 전송됩니다.	`GET / index.html HTTP/1.1 Host: example.com X-Forwarded-For: 127.0.0.4`	`X-Forwarded-For: 127.0.0.4, 127.0.0.1`	`X-Forwarded-For: 127.0.0.4`	존재하지 않음
요청이 XFF 헤더 및 여러 클라이언트 IP 주소를 포함하여 전송됩니다.	`GET / index.html HTTP/1.1 Host: example.com X-Forwarded-For: 127.0.0.4, 127.0.0.8`	`X-Forwarded-For: 127.0.0.4, 127.0.0.8, 127.0.0.1`	`X-Forwarded-For: 127.0.0.4, 127.0.0.8`	존재하지 않음

콘솔을 사용한 X-Forwarded-For 헤더 수정, 유지 또는 제거

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 트래픽 구성 섹션의 패킷 처리에서 용X-Forwarded-For 헤더에 대해 추가(기본값),보존, 또는제거를 선택합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 X-Forwarded-For 헤더를 수정, 보존 또는 제거하려면 AWS CLI

`routing.http.xff_header_processing.mode` 속성과 함께 modify-load-balancer-attributes 명령을 사용합니다.

X-Forwarded-Proto

`X-Forwarded-Proto` 요청 헤더는 클라이언트가 로드 밸런서 연결에 사용한 프로토콜(HTTP 또는 HTTPS)을 식별하는 데 도움을 줍니다. 서버 액세스 로그에는 서버와 로드 밸런서 간에 사용된 프로토콜만 포함되어 있으며, 클라이언트와 로드 밸런서 간에 사용된 프로토콜에 대한 정보는 포함되어 있지 않습니다. 클라이언트와 로드 밸런서 간에 사용된 프로토콜을 확인하려면 `X-Forwarded-Proto` 요청 헤더를 사용하십시오. Elastic Load Balancing은 `X-Forwarded-Proto` 요청 헤더에 클라이언트와 로드 밸런서 간에 사용된 프로토콜을 저장하고 서버로 헤더를 전달합니다.

애플리케이션이나 웹 사이트는 `X-Forwarded-Proto` 요청 헤더에 저장된 프로토콜을 사용하여 해당 URL로 응답이 리디렉션 되도록 합니다.

`X-Forwarded-Proto` 요청 헤더의 형식은 다음과 같습니다.

```
X-Forwarded-Proto: originatingProtocol
```

다음 예제에는 HTTPS 요청으로서 클라이언트에서 시작된 요청에 대한 `X-Forwarded-Proto` 요청 헤더가 포함되어 있습니다.

```
X-Forwarded-Proto: https
```

X-Forwarded-Port

`X-Forwarded-Port` 요청 헤더는 클라이언트가 로드 밸런서 연결에 사용한 대상 포트를 식별하는 데 도움을 줍니다.

Application Load Balancer용 HTTP 리스너 생성

리스너는 연결 요청을 확인합니다. 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 섹션을 참조하세요.

사전 조건

- 기본 리스너 규칙에 전달 작업을 추가하려면 사용 가능한 대상 그룹을 지정해야 합니다. 자세한 내용은 Application Load Balancer 대상 그룹 생성 단원을 참조하십시오.
- 여러 리스너에서 동일한 대상 그룹을 지정할 수 있지만, 이러한 리스너는 동일한 로드 밸런서에 속해야 합니다. 대상 그룹을 로드 밸런서와 함께 사용하려면 대상 그룹이 다른 로드 밸런서용으로 리스너에서 사용되고 있지 않은지 확인해야 합니다.

HTTP 리스너 추가

리스너에서 클라이언트에서 로드 밸런서로의 연결을 위한 프로토콜 및 포트 번호와 기본 리스너 규칙에 대한 대상 그룹을 구성합니다. 자세한 내용은 리스너 구성 단원을 참조하세요.

콘솔을 사용하여 HTTP 리스너를 추가하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리스너 및 규칙 탭에서 리스너 추가를 선택합니다.
5. 프로토콜 : 포트에서 HTTP를 선택하고 기본 포트를 유지하거나 다른 포트를 입력합니다.
6. 기본 작업에서 다음 중 하나를 선택합니다.

    - 대상 그룹에 전달 – 트래픽을 전달할 대상 그룹을 하나 이상 선택합니다. 대상 그룹을 추가하려면 대상 그룹 추가를 선택합니다. 대상 그룹을 하나 이상 사용하는 경우, 각 대상 그룹의 가중치

를 선택하고 관련 비율을 검토합니다. 하나 이상의 대상 그룹에서 고정성을 활성화한 경우 규칙에 그룹 수준 고정성을 활성화해야 합니다.

- URL로 리디렉션 – 클라이언트 요청이 리디렉션될 URL을 지정합니다. URI 파트 탭에서 각 파트를 개별적으로 입력하거나 전체 URL 탭에서 전체 주소를 입력하여 이 작업을 수행합니다. 상태 코드의 경우 요구 사항에 따라 리디렉션을 임시(HTTP 302) 또는 영구(HTTP 301)로 구성할 수 있습니다.
- 고정 응답 반환 – 삭제된 클라이언트 요청으로 반환되는 응답 코드를 지정하십시오. 또한, 콘텐츠 유형 및 응답 본문을 지정할 수 있지만 필수는 아닙니다.

7. 추가를 선택합니다.

를 사용하여 HTTP 리스너를 추가하려면 AWS CLI

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

Application Load Balancer용 SSL 인증서

Application Load Balancer에 대한 보안 리스너를 생성할 때는 로드 밸런서에 하나 이상의 인증서를 배포해야 합니다. 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, 일련 번호, 발행자의 디지털 서명이 들어 있습니다.

로드 밸런서와 함께 사용할 인증서를 생성할 때 도메인 이름을 지정해야 합니다. 인증서의 도메인 이름은 사용자 지정 도메인 이름 레코드와 일치해야 TLS 연결을 확인할 수 있습니다. 두 값이 일치하지 않는 경우 트래픽이 암호화되지 않습니다.

인증서에 `www.example.com`과 같은 정규화된 도메인 이름(FQDN) 또는 `example.com`과 같은 apex 도메인 이름을 지정해야 합니다. 별표(*)를 와일드카드로 사용하여 동일한 도메인 내에서 여러 사이트 이름을 보호할 수도 있습니다. 와일드카드 인증서를 요청할 때 별표(*)는 도메인 이름의 맨 왼쪽에 와야 하며 하나의 하위 도메인 수준만 보호할 수 있습니다. 예를 들어 `*.example.com`은 `corp.example.com` 및 `images.example.com`은 보호하지만 `test.login.example.com`을 보호할 수는 없습니다. 또한 `*.example.com`은 `example.com`의 하위 도메인만 보호하고 베어 또는 apex 도메인(`example.com`)은 보호하지 못합니다. 와일드카드 이름은 주체 필드와 인증서의 주체 대체 이름 확장에 표시됩니다. 퍼블릭 인증서에 대한 자세한 내용은 AWS Certificate Manager 사용 설명서의 퍼블릭 인증서 요청을 참조하세요.

AWS Certificate Manager (ACM)을 사용해 로드 밸런서를 위한 인증서를 생성하는 것이 좋습니다. ACM은 2048, 3072, 4096비트 길이의 RSA 인증서와 모든 ECDSA 인증서를 지원합니다. ACM

은 Elastic Load Balancing과 통합하여 로드 밸런서에 인증서를 배포합니다. 자세한 내용은 AWS Certificate Manager 사용 설명서를 참조하십시오.

또는 SSL/TLS 도구를 사용하여 인증서 서명 요청(CSR)을 생성한 다음 CA가 서명한 CSR을 가져와 인증서를 생성한 다음 인증서를 ACM으로 가져오거나 인증서를 AWS Identity and Access Management (IAM)에 업로드할 수 있습니다. 인증서를 ACM으로 가져오는 방법에 대한 자세한 내용은 AWS Certificate Manager 사용 설명서의 인증서 가져오기 단원을 참조하세요. IAM으로 인증서를 업로드하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 서버 인증서 작업을 참조하세요.

기본 인증서

HTTPS 리스너를 생성할 때 인증서 하나를 꼭 지정해야 합니다. 이 인증서를 기본 인증서라고 합니다. HTTPS 리스너를 생성한 후 기본 인증서를 교체할 수 있습니다. 자세한 내용은 기본 인증서 교체 단원을 참조하세요.

인증서 목록에서 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 인증서 목록에 인증서를 추가할 수 있습니다. 를 사용하여 리스너를 생성한 경우 기본 인증서를 인증서 목록에 AWS Management Console추가했습니다. 그렇지 않으면 인증서 목록이 비어 있습니다. 인증서 목록을 사용하면 로드 밸런서가 동일한 포트의 여러 도메인을 지원하고 각 도메인에 대해 다른 인증서를 제공할 수 있습니다. 자세한 내용은 인증서 목록에 인증서 추가 단원을 참조하세요.

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)
- 만료(만료되지 않음 선호)

- 해싱 알고리즘(MD5보다 SHA 선호). SHA 인증서가 여러 개 있는 경우 가장 높은 SHA 번호를 선호합니다.
- 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 단원을 참조하세요.

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.

- 에서 제공하고 로드 밸런서에 AWS Certificate Manager 배포된 인증서는 자동으로 갱신할 수 있습니다. ACM은 인증서가 만료되기 전에 갱신을 시도합니다. 자세한 내용은 AWS Certificate Manager 사용 설명서에서 관리형 갱신을 참조하세요.
- ACM에 인증서를 가져온 경우에는 인증서의 만료일을 반드시 모니터링해서 만료되기 전에 인증서를 갱신해야 합니다. 자세한 내용은 AWS Certificate Manager 사용 설명서에서 인증서 가져오기를 참조하세요.
- IAM으로 인증서를 가져온 경우, 새 인증서를 만들어 ACM이나 IAM으로 가져온 후 로드 밸런서에 새 인증서를 추가하고, 만료된 인증서를 로드 밸런서에서 제거해야 합니다.

Application Load Balancer 보안 정책

Elastic Load Balancing은 보안 정책이라고 하는 Secure Socket Layer(SSL) 협상 구성을 사용해 클라이언트와 로드 밸런서 간의 연결을 협상합니다. 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.

고려 사항

- Application Load Balancer는 대상 연결에 대해서만 SSL 재협상을 지원합니다.
- HTTPS 리스너를 생성할 때 보안 정책을 선택해야 합니다.
- `ELBSecurityPolicy-TLS13-1-2-Res-2021-06` 정책은 AWS Management Console을 사용하여 생성한 리스너의 기본 보안 정책입니다. 이 정책은 TLS 1.3을 지원하며 TLS 1.2와 역호환됩니다.
- `ELBSecurityPolicy-2016-08` 정책은 AWS CLI을 사용하여 생성한 리스너의 기본 보안 정책입니다.
- Application Load Balancer는 사용자 지정 보안 정책을 지원하지 않습니다.
- 백엔드 연결이 아니라 프런트엔드 연결에서 사용되는 보안 정책을 선택할 수 있습니다.
  - 백엔드 연결의 경우 HTTPS 리스너 중 하나라도 TLS 1.3 보안 정책을 사용하면 `ELBSecurityPolicy-TLS13-1-0-2021-06` 보안 정책이 사용됩니다. 그러지 않으면, `ELBSecurityPolicy-2016-08` 보안 정책은 백엔드 연결에 사용됩니다.
  - 참고: HTTPS 리스너에서 FIPS TLS 정책을 사용하는 경우 백엔드 연결에 `ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04`가 사용됩니다.
- 특정 TLS 프로토콜 버전을 비활성화해야 하는 규정 준수 및 보안 표준을 충족하거나 더 이상 사용되지 않는 암호가 필요한 레거시 클라이언트를 지원하려면 `ELBSecurityPolicy-TLS-` 보안 정책 중 하나를 사용할 수 있습니다. Application Load Balancer에 대한 요청에서 TLS 프로토콜 버전을 확인하려면 로드 밸런서에서 액세스 로깅을 활성화하고 해당 액세스 로그 항목을 검사하세요. 자세한 내용은 Application Load Balancer 액세스 로그를 참조하세요.
- IAM AWS 계정 및 AWS Organizations 서비스 제어 정책(SCPs)의 Elastic Load Balancing 조건 키를 각각 사용하여 및 전체에서 사용자가 사용할 수 있는 보안 정책을 제한할 수 있습니다. 자세한 내용은 AWS Organizations 사용 설명서의 서비스 제어 정책(SCP)을 참조하세요.
- TLS 1.3만 지원하는 정책은 FS(순방향 보안)를 지원합니다. TLS_* 및 ECDHE_* 형식의 암호만 있는 TLS 1.3 및 TLS 1.2를 지원하는 정책도 FS를 제공합니다.
- Application Load Balancer는 PSK(TLS 1.3) 및 세션 ID/세션 티켓(TLS 1.2 이상)을 사용하여 TLS 재개를 지원합니다. 재개는 동일한 Application Load Balancer IP 주소에 대한 연결에서만 지원됩니다. 0-RTT 데이터 기능과 early_data 확장은 구현되지 않습니다.
- Application Load Balancer는 TLS 1.2에 대한 EMS(Extended Master Secret) 확장을 지원합니다.

describe-ssl-policies AWS CLI 명령을 사용하거나 아래 표를 참조하여 여 프로토콜 및 암호를 설명할 수 있습니다.

보안 정책

- TLS 보안 정책
  - 정책별 프로토콜
  - 정책별 암호
  - 암호별 정책
- FIPS 보안 정책
  - 정책별 프로토콜
  - 정책별 암호
  - 암호별 정책
- FS 지원 정책
  - 정책별 프로토콜
  - 정책별 암호
  - 암호별 정책

TLS 보안 정책

TLS 보안 정책을 사용하여 특정한 TLS 프로토콜 버전을 비활성화해야 하는 규정 준수 및 보안 표준을 충족하거나 암호 사용 중지가 필요한 기존 클라이언트를 지원할 수 있습니다.

TLS 1.3만 지원하는 정책은 FS(순방향 보안)를 지원합니다. TLS_* 및 ECDHE_* 형식의 암호만 있는 TLS 1.3 및 TLS 1.2를 지원하는 정책도 FS를 제공합니다.

내용

- 정책별 프로토콜
- 정책별 암호
- 암호별 정책

정책별 프로토콜

다음 표에서는 각 TLS 보안 정책이 지원하는 프로토콜을 설명합니다.

보안 정책	TLS 1.3	TLS 1.2	TLS 1.1	TLS 1.0
ELBSecurityPolicy-TLS13-1-3-2021-06	예	아니요	아니요	아니요
ELBSecurityPolicy-TLS13-1-2-2021-06	예	예	아니요	아니요
ELBSecurityPolicy-TLS13-1-2-Res-2021-06	예	예	아니요	아니요
ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06	예	예	아니요	아니요
ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06	예	예	아니요	아니요
ELBSecurityPolicy-TLS13-1-1-2021-06	예	예	예	아니요
ELBSecurityPolicy-TLS13-1-0-2021-06	예	예	예	예
ELBSecurityPolicy-TLS-1-2-Ext-2018-06	아니요	예	아니요	아니요
ELBSecurityPolicy-TLS-1-2-2017-01	아니요	예	아니요	아니요
ELBSecurityPolicy-TLS-1-1-2017-01	아니요	예	예	아니요
ELBSecurityPolicy-2016-08	아니요	예	예	예
ELBSecurityPolicy-2015-05	아니요	예	예	예

정책별 암호

다음 표에서는 각 TLS 보안 정책이 지원하는 암호를 설명합니다.

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS13-1-3-2021-06	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • TLS_CHACHA20_POLY1305_SHA256
ELBSecurityPolicy-TLS13-1-2-2021-06	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • TLS_CHACHA20_POLY1305_SHA256 • ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES128-SHA256 • ECDHE-RSA-AES128-SHA256 • ECDHE-ECDSA-AES256-GCM-SHA384 • ECDHE-RSA-AES256-GCM-SHA384 • ECDHE-ECDSA-AES256-SHA384 • ECDHE-RSA-AES256-SHA384
ELBSecurityPolicy-TLS13-1-2-Res-2021-06	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • TLS_CHACHA20_POLY1305_SHA256 • ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES256-GCM-SHA384 • ECDHE-RSA-AES256-GCM-SHA384
ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • TLS_CHACHA20_POLY1305_SHA256 • ECDHE-ECDSA-AES128-GCM-SHA256

보안 정책	암호(Ciphers)
	• 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-ECDSA-AES256-SHA • ECDHE-RSA-AES256-SHA • AES128-GCM-SHA256 • AES128-SHA256 • AES128-SHA • AES256-GCM-SHA384 • AES256-SHA256 • AES256-SHA

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • TLS_CHACHA20_POLY1305_SHA256 • ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES128-SHA256 • ECDHE-RSA-AES128-SHA256 • ECDHE-ECDSA-AES256-GCM-SHA384 • ECDHE-RSA-AES256-GCM-SHA384 • ECDHE-ECDSA-AES256-SHA384 • ECDHE-RSA-AES256-SHA384 • AES128-GCM-SHA256 • AES128-SHA256 • AES256-GCM-SHA384 • AES256-SHA256

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS13-1-1-2021-06	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • TLS_CHACHA20_POLY1305_SHA256 • ECDHE-ECDSA-AES128-GCM-SHA256 • 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-ECDSA-AES256-SHA • ECDHE-RSA-AES256-SHA • AES128-GCM-SHA256 • AES128-SHA256 • AES128-SHA • AES256-GCM-SHA384 • AES256-SHA256 • AES256-SHA

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS13-1-0-2021-06	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • TLS_CHACHA20_POLY1305_SHA256 • ECDHE-ECDSA-AES128-GCM-SHA256 • 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-ECDSA-AES256-SHA • ECDHE-RSA-AES256-SHA • AES128-GCM-SHA256 • AES128-SHA256 • AES128-SHA • AES256-GCM-SHA384 • AES256-SHA256 • AES256-SHA

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS-1-2-Ext-2018-06	• ECDHE-ECDSA-AES128-GCM-SHA256 • 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-ECDSA-AES256-SHA • ECDHE-RSA-AES256-SHA • AES128-GCM-SHA256 • AES128-SHA256 • AES128-SHA • AES256-GCM-SHA384 • AES256-SHA256 • AES256-SHA

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS-1-2-2017-01	• ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES128-SHA256 • ECDHE-RSA-AES128-SHA256 • ECDHE-ECDSA-AES256-GCM-SHA384 • ECDHE-RSA-AES256-GCM-SHA384 • ECDHE-ECDSA-AES256-SHA384 • ECDHE-RSA-AES256-SHA384 • AES128-GCM-SHA256 • AES128-SHA256 • AES256-GCM-SHA384 • AES256-SHA256

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS-1-1-2017-01	• ECDHE-ECDSA-AES128-GCM-SHA256 • 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-ECDSA-AES256-SHA • ECDHE-RSA-AES256-SHA • AES128-GCM-SHA256 • AES128-SHA256 • AES128-SHA • AES256-GCM-SHA384 • AES256-SHA256 • AES256-SHA

보안 정책	암호(Ciphers)
ELBSecurityPolicy-2016-08	• ECDHE-ECDSA-AES128-GCM-SHA256 • 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-ECDSA-AES256-SHA • ECDHE-RSA-AES256-SHA • AES128-GCM-SHA256 • AES128-SHA256 • AES128-SHA • AES256-GCM-SHA384 • AES256-SHA256 • AES256-SHA

보안 정책	암호(Ciphers)
ELBSecurityPolicy-2015-05	• ECDHE-ECDSA-AES128-GCM-SHA256 • 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-ECDSA-AES256-SHA • ECDHE-RSA-AES256-SHA • AES128-GCM-SHA256 • AES128-SHA256 • AES128-SHA • AES256-GCM-SHA384 • AES256-SHA256 • AES256-SHA

암호별 정책

다음 표에서는 각 암호를 지원하는 TLS 보안 정책을 설명합니다.

암호 이름	보안 정책	암호 그룹
OpenSSL – TLS_AES_128_GCM_SHA256 IANA – TLS_AES_128_GCM_SHA256	• ELBSecurityPolicy-TLS13-1-3-2021-06 • ELBSecurityPolicy-TLS13-1-2-2021-06	1301

암호 이름	보안 정책	암호 그룹
	• ELBSecurityPolicy-TLS13-1-2-Res-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06	
OpenSSL – TLS_AES_256_GCM_SHA384 IANA – TLS_AES_256_GCM_SHA384	• ELBSecurityPolicy-TLS13-1-3-2021-06 • ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Res-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06	1302

암호 이름	보안 정책	암호 그룹
OpenSSL – TLS_CHACHA20_POLY1305_SHA256 IANA – TLS_CHACHA20_POLY1305_SHA256	• ELBSecurityPolicy-TLS13-1-3-2021-06 • ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Res-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06	1303

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES128-GCM-SHA256 IANA – TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	• ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Res-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c02b

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-RSA-AES128-GCM-SHA256 IANA – TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	• ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Res-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c02f

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES128-SHA256 IANA – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	• ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c023
OpenSSL – ECDHE-RSA-AES128-SHA256 IANA – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	• ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c027

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES128-SHA IANA – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c009
OpenSSL – ECDHE-RSA-AES128-SHA IANA – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c013

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES256-GCM-SHA384 IANA – TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	• ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Res-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c02c

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-RSA-AES256-GCM-SHA384 IANA – TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	• ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Res-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c030

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES256-SHA384 IANA – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	• ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c024
OpenSSL – ECDHE-RSA-AES256-SHA384 IANA – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	• ELBSecurityPolicy-TLS13-1-2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c028

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES256-SHA IANA – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c00a
OpenSSL – ECDHE-RSA-AES256-SHA IANA – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	c014

암호 이름	보안 정책	암호 그룹
OpenSSL – AES128-GCM-SHA256 IANA – TLS_RSA_WITH_AES_128_GCM_SHA256	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	9c
OpenSSL – AES128-SHA256 IANA – TLS_RSA_WITH_AES_128_CBC_SHA256	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	3c

암호 이름	보안 정책	암호 그룹
OpenSSL – AES128-SHA IANA – TLS_RSA_WITH_AES_128_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	2f
OpenSSL – AES256-GCM-SHA384 IANA – TLS_RSA_WITH_AES_256_GCM_SHA384	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	9d

암호 이름	보안 정책	암호 그룹
OpenSSL – AES256-SHA256 IANA – TLS_RSA_WITH_AES_256_CBC_SHA256	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-2-2017-01 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	3d
OpenSSL – AES256-SHA IANA – TLS_RSA_WITH_AES_256_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06 • ELBSecurityPolicy-TLS13-1-1-2021-06 • ELBSecurityPolicy-TLS13-1-0-2021-06 • ELBSecurityPolicy-TLS-1-2-Ext-2018-06 • ELBSecurityPolicy-TLS-1-1-2017-01 • ELBSecurityPolicy-2016-08	35

FIPS 보안 정책

Important

Application Load Balancer에 연결된 모든 보안 리스너는 FIPS 보안 정책 또는 비 FIPS 보안 정책을 사용해야 합니다. 혼합할 수는 없습니다. 기존 Application Load Balancer에 비 FIPS 정책을 사용하는 리스너가 두 개 이상 있을 때 리스너가 대신 FIPS 보안 정책을 사용하도록 하려는

경우 리스너를 하나만 남을 때까지 모두 제거합니다. 리스너의 보안 정책을 FIPS로 변경한 다음 FIPS 보안 정책을 사용하여 추가 리스너를 생성합니다. 또는 FIPS 보안 정책만 사용하여 새 리스너로 새 Application Load Balancer를 생성할 수 있습니다.

Federal Information Processing Standard(FIPS)는 미국 및 캐나다 정부 보안 표준으로서, 기밀 정보를 보호하는 암호 모듈의 보안 요건을 규정하고 있습니다. 자세한 내용은 AWS 클라우드 보안 규정 준수 페이지의 Federal Information Processing Standard(FIPS) 140을 참조하세요.

모든 FIPS 정책은 AWS-LC FIPS 검증 암호화 모듈을 활용합니다. 자세한 내용은 NIST 암호화 모듈 검증 프로그램 사이트의 AWS-LC 암호화 모듈 페이지를 참조하세요.

Important

`ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04` 및 `ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04` 정책은 레거시 호환성을 위해서만 제공됩니다. 이들 정책은 FIPS140 모듈을 사용하는 FIPS 암호화를 활용하지만 TLS 구성에 대한 최신 NIST 지침을 준수하지 않을 수 있습니다.

내용

- 정책별 프로토콜
- 정책별 암호
- 암호별 정책

정책별 프로토콜

다음 표에서는 각 FIPS 보안 정책이 지원하는 프로토콜을 설명합니다.

보안 정책	TLS 1.3	TLS 1.2	TLS 1.1	TLS 1.0
ELBSecurityPolicy-TLS13-1-3-FIPS-2023-04	예	아니요	아니요	아니요
ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04	예	예	아니요	아니요

보안 정책	TLS 1.3	TLS 1.2	TLS 1.1	TLS 1.0
ELBSecurityPolicy-TLS13-1-2-Res-FIPS-2023-04	예	예	아니요	아니요
ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04	예	예	아니요	아니요
ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04	예	예	아니요	아니요
ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04	예	예	아니요	아니요
ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04	예	예	예	아니요
ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	예	예	예	예

정책별 암호

다음 표에서는 각 FIPS 보안 정책이 지원하는 암호를 설명합니다.

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS13-1-3-FIPS-2023-04	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384
ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES128-SHA256 • ECDHE-RSA-AES128-SHA256 • ECDHE-ECDSA-AES256-GCM-SHA384

보안 정책	암호(Ciphers)
	• ECDHE-RSA-AES256-GCM-SHA384 • ECDHE-ECDSA-AES256-SHA384 • ECDHE-RSA-AES256-SHA384
ELBSecurityPolicy-TLS13-1-2-Res-FIPS-2023-04	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES256-GCM-SHA384 • ECDHE-RSA-AES256-GCM-SHA384

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • ECDHE-ECDSA-AES128-GCM-SHA256 • 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

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES128-SHA256 • ECDHE-RSA-AES128-SHA256 • ECDHE-ECDSA-AES256-GCM-SHA384 • ECDHE-RSA-AES256-GCM-SHA384 • ECDHE-ECDSA-AES256-SHA384 • ECDHE-RSA-AES256-SHA384 • AES128-GCM-SHA256 • AES128-SHA256 • AES256-GCM-SHA384 • AES256-SHA256
ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • ECDHE-ECDSA-AES128-GCM-SHA256 • 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

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • ECDHE-ECDSA-AES128-GCM-SHA256 • 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

보안 정책	암호(Ciphers)
ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • ECDHE-ECDSA-AES128-GCM-SHA256 • 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

암호별 정책

다음 표에서는 각 암호를 지원하는 FIPS 보안 정책을 설명합니다.

암호 이름	보안 정책	암호 그룹
OpenSSL – TLS_AES_128_GCM_SHA256 IANA – TLS_AES_128_GCM_SHA256	• ELBSecurityPolicy-TLS13-1-3-FIPS-2023-04	1301

암호 이름	보안 정책	암호 그룹
	• ELBSecurityPolicy-TLS13-1-2-Res-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	
OpenSSL – TLS_AES_256_GCM_SHA384 IANA – TLS_AES_256_GCM_SHA384	• ELBSecurityPolicy-TLS13-1-3-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Res-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	1302

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES128-GCM-SHA256 IANA – TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	• ELBSecurityPolicy-TLS13-1-2-Res-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c02b
OpenSSL – ECDHE-RSA-AES128-GCM-SHA256 IANA – TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	• ELBSecurityPolicy-TLS13-1-2-Res-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c02f

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES128-SHA256 IANA – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	• ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c023
OpenSSL – ECDHE-RSA-AES128-SHA256 IANA – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	• ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c027

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES128-SHA IANA – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c009
OpenSSL – ECDHE-RSA-AES128-SHA IANA – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c013
OpenSSL – ECDHE-ECDSA-AES256-GCM-SHA384 IANA – TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	• ELBSecurityPolicy-TLS13-1-2-Res-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c02c

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-RSA-AES256-GCM-SHA384 IANA – TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	• ELBSecurityPolicy-TLS13-1-2-Res-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c030
OpenSSL – ECDHE-ECDSA-AES256-SHA384 IANA – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	• ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c024

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-RSA-AES256-SHA384 IANA – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	• ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c028
OpenSSL – ECDHE-ECDSA-AES256-SHA IANA – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c00a
OpenSSL – ECDHE-RSA-AES256-SHA IANA – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	c014

암호 이름	보안 정책	암호 그룹
OpenSSL – AES128-GCM-SHA256 IANA – TLS_RSA_WITH_AES_128_GCM_SHA256	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	9c
OpenSSL – AES128-SHA256 IANA – TLS_RSA_WITH_AES_128_CBC_SHA256	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	3c
OpenSSL – AES128-SHA IANA – TLS_RSA_WITH_AES_128_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	2f
OpenSSL – AES256-GCM-SHA384 IANA – TLS_RSA_WITH_AES_256_GCM_SHA384	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	9d

암호 이름	보안 정책	암호 그룹
OpenSSL – AES256-SHA256 IANA – TLS_RSA_WITH_AES_256_CBC_SHA256	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	3d
OpenSSL – AES256-SHA IANA – TLS_RSA_WITH_AES_256_CBC_SHA	• ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 • ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04	35

FS 지원 정책

FS(순방향 비밀성) 지원 보안 정책은 고유한 무작위 세션 키를 사용하여 암호화된 데이터를 도청할 수 없도록 추가적인 보호 기능을 제공합니다. 이렇게 하면 보안 암호 장기 키가 손상되더라도 캡처된 데이터의 디코딩이 방지됩니다.

이 섹션의 정책은 FS를 지원하며 "FS"가 이름에 포함됩니다. 그러나 이러한 정책이 FS를 지원하는 유일한 정책은 아닙니다. TLS 1.3만 지원하는 정책은 FS를 지원합니다. TLS_* 및 ECDHE_* 형식의 암호만 있는 TLS 1.3 및 TLS 1.2를 지원하는 정책도 FS를 제공합니다.

내용

- 정책별 프로토콜
- 정책별 암호
- 암호별 정책

정책별 프로토콜

다음 표에서는 각 FS 지원 보안 정책이 지원하는 프로토콜을 설명합니다.

보안 정책	TLS 1.3	TLS 1.2	TLS 1.1	TLS 1.0
ELBSecurityPolicy-FS-1-2-Res-2020-10	아니요	예	아니요	아니요
ELBSecurityPolicy-FS-1-2-Res-2019-08	아니요	예	아니요	아니요
ELBSecurityPolicy-FS-1-2-2019-08	아니요	예	아니요	아니요
ELBSecurityPolicy-FS-1-1-2019-08	아니요	예	예	아니요
ELBSecurityPolicy-FS-2018-06	아니요	예	예	예

정책별 암호

다음 표에서는 각 FS 지원 보안 정책이 지원하는 암호를 설명합니다.

보안 정책	암호(Ciphers)
ELBSecurityPolicy-FS-1-2-Res-2020-10	• ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES256-GCM-SHA384 • ECDHE-RSA-AES256-GCM-SHA384
ELBSecurityPolicy-FS-1-2-Res-2019-08	• ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES128-SHA256 • ECDHE-RSA-AES128-SHA256

보안 정책	암호(Ciphers)
	• ECDHE-ECDSA-AES256-GCM-SHA384 • ECDHE-RSA-AES256-GCM-SHA384 • ECDHE-ECDSA-AES256-SHA384 • ECDHE-RSA-AES256-SHA384
ELBSecurityPolicy-FS-1-2-2019-08	• ECDHE-ECDSA-AES128-GCM-SHA256 • 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
ELBSecurityPolicy-FS-1-1-2019-08	• ECDHE-ECDSA-AES128-GCM-SHA256 • 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

보안 정책	암호(Ciphers)
ELBSecurityPolicy-FS-2018-06	• ECDHE-ECDSA-AES128-GCM-SHA256 • 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

암호별 정책

다음 표에서는 각 암호를 지원하는 FS 지원 보안 정책을 설명합니다.

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-ECDSA-AES128-GCM-SHA256 IANA – TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	• ELBSecurityPolicy-FS-1-2-Res-2020-10 • ELBSecurityPolicy-FS-1-2-Res-2019-08 • ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c02b
OpenSSL – ECDHE-RSA-AES128-GCM-SHA256 IANA – TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	• ELBSecurityPolicy-FS-1-2-Res-2020-10 • ELBSecurityPolicy-FS-1-2-Res-2019-08	c02f

암호 이름	보안 정책	암호 그룹
	• ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	
OpenSSL – ECDHE-ECDSA-AES128-SHA256 IANA – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	• ELBSecurityPolicy-FS-1-2-Res-2019-08 • ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c023
OpenSSL – ECDHE-RSA-AES128-SHA256 IANA – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	• ELBSecurityPolicy-FS-1-2-Res-2019-08 • ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c027
OpenSSL – ECDHE-ECDSA-AES128-SHA IANA – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA	• ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c009
OpenSSL – ECDHE-RSA-AES128-SHA IANA – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	• ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c013
OpenSSL – ECDHE-ECDSA-AES256-GCM-SHA384 IANA – TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	• ELBSecurityPolicy-FS-1-2-Res-2020-10 • ELBSecurityPolicy-FS-1-2-Res-2019-08 • ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c02c

암호 이름	보안 정책	암호 그룹
OpenSSL – ECDHE-RSA-AES256-GCM-SHA384 IANA – TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	• ELBSecurityPolicy-FS-1-2-Res-2020-10 • ELBSecurityPolicy-FS-1-2-Res-2019-08 • ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c030
OpenSSL – ECDHE-ECDSA-AES256-SHA384 IANA – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	• ELBSecurityPolicy-FS-1-2-Res-2019-08 • ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c024
OpenSSL – ECDHE-RSA-AES256-SHA384 IANA – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	• ELBSecurityPolicy-FS-1-2-Res-2019-08 • ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c028
OpenSSL – ECDHE-ECDSA-AES256-SHA IANA – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA	• ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c00a
OpenSSL – ECDHE-RSA-AES256-SHA IANA – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	• ELBSecurityPolicy-FS-1-2-2019-08 • ELBSecurityPolicy-FS-1-1-2019-08 • ELBSecurityPolicy-FS-2018-06	c014

Application Load Balancer용 HTTPS 리스너 생성

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

HTTPS 리스너를 생성하려면 로드 밸런서에 한 개 이상의 SSL 서버 인증서를 반드시 배포해야 합니다. 로드 밸런서는 서버 인증서를 사용해 프런트 엔드 연결을 종료한 다음, 대상으로 전송하기 전에 클라이언트의 요청을 해독합니다. 클라이언트와 로드 밸런서 간의 보안 연결을 협상하는 데 사용되는 보안 정책도 지정해야 합니다.

암호화된 트래픽을 로드 밸런서의 해독 없이 대상으로 전달해야 하는 경우, 포트 443을 수신하는 TCP 리스너가 있는 Network Load Balancer 또는 Classic Load Balancer를 생성할 수 있습니다. TCP 리스너를 사용하여 로드 밸런서는 암호화된 트래픽을 해독하지 않고 대상으로 전달합니다.

이 페이지의 정보는 로드 밸런서용 HTTPS 리스너를 생성하는 데 도움이 됩니다. 로드 밸런서에 HTTP 리스너를 추가하려면 Application Load Balancer용 HTTP 리스너 생성 섹션을 참조하세요.

사전 조건

- 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.

  Application Load Balanca는 ED25519 키를 지원하지 않습니다.
- 기본 리스너 규칙에 전달 작업을 추가하려면 사용 가능한 대상 그룹을 지정해야 합니다. 자세한 내용은 Application Load Balancer 대상 그룹 생성 단원을 참조하십시오.
- 여러 리스너에서 동일한 대상 그룹을 지정할 수 있지만, 이러한 리스너는 동일한 로드 밸런서에 속해야 합니다. 대상 그룹을 로드 밸런서와 함께 사용하려면 대상 그룹이 다른 로드 밸런서용으로 리스너에서 사용되고 있지 않은지 확인해야 합니다.

HTTPS 리스너 추가

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

콘솔을 사용하여 HTTPS 리스너를 추가하는 방법

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

2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리스너 및 규칙 탭에서 리스너 추가를 선택합니다.
5. 프로토콜 : 포트에서 HTTPS를 선택하고 기본 포트를 유지하거나 다른 포트를 입력합니다.
6. (선택 사항) 인증을 활성화하려면 인증에서 오픈ID 또는 Amazon Cognito를 선택하고 요청된 정보를 제공하십시오. 자세한 내용은 Application Load Balancer를 사용하여 사용자 인증 단원을 참조하십시오.
7. 라우팅 작업에서 다음 중 하나를 수행합니다.

   - 대상 그룹으로 전달 - 트래픽을 전달할 대상 그룹을 선택합니다. 대상 그룹을 추가하려면 대상 그룹 추가를 선택합니다. 대상 그룹을 하나 이상 사용하는 경우, 각 대상 그룹의 가중치를 선택하고 관련 비율을 검토합니다. 하나 이상의 대상 그룹에서 고정성을 활성화한 경우 규칙에 그룹 수준 고정성을 활성화해야 합니다.
   - URL로 리디렉션 - 클라이언트 요청이 리디렉션될 URL을 입력합니다. URI 파트 탭에서 각 파트를 개별적으로 입력하거나 전체 URL 탭에서 전체 주소를 입력하여 이 작업을 수행합니다. 상태 코드의 경우 요구 사항에 따라 리디렉션을 임시(HTTP 302) 또는 영구(HTTP 301)로 구성할 수 있습니다.
   - 고정 응답 반환 - 응답 코드를 입력하여 삭제된 클라이언트 요청으로 돌아갑니다. 선택적으로 콘텐츠 유형 및 응답 본문을 지정할 수 있습니다.
8. 보안 정책의 경우 항상 최신 사전 정의 보안 정책을 사용하는 것이 좋습니다.
9. 기본 SSL/TLS 인증서에서 기본 인증서를 선택합니다. 또한 SNI 목록에 기본 인증서를 추가합니다. 다음 소스 중 하나에서 인증서를 선택할 수 있습니다.

   - 를 사용하여 인증서를 생성하거나 가져온 경우 ACM에서 를 AWS Certificate Manager선택한 다음 인증서(ACM에서)에서 인증서를 선택합니다.
   - IAM을 사용하여 인증서를 가져온 경우 IAM에서를 선택한 다음 인증서(IAM에서)에서 인증서를 선택합니다.
   - 인증서가 있는 경우 인증서 가져오기를 선택합니다. ACM으로 가져오기 또는 IAM으로 가져오기를 선택합니다. 인증서 프라이빗 키의 경우 프라이빗 키 파일(PEM 인코딩)의 내용을 복사하여 붙여 넣습니다. 인증서 본문의 경우 퍼블릭 키 인증서 파일(PEM 인코딩)의 내용을 복사하여 붙여 넣습니다. 자체 서명된 인증서를 사용하고 브라우저가 인증서를 암시적으로 수락하는 것이 중요하지 않은 경우를 제외하고 인증서 체인의 경우 인증서 체인 파일(PEM 인코딩)의 내용을 복사하여 붙여넣습니다.
10. (선택 사항) 상호 인증을 활성화하려면 클라이언트 인증서 처리에서 상호 인증(mTLS)을 활성화합니다.

활성화되면 기본 상호 TLS 모드는 패스스루입니다.

트러스트 스토어로 확인을 선택한 경우:

- 기본적으로 만료된 클라이언트 인증서와의 연결은 거부됩니다. 이 동작을 변경하려면 고급 mTLS 설정 을 확장한 다음 클라이언트 인증서 만료에서 만료된 클라이언트 인증서 허용을 선택합니다.
- 트러스트 스토어에서 기존 트러스트 스토어를 선택하거나 새 트러스트 스토어를 선택합니다.
  - 새 트러스트 스토어를 선택한 경우 트러스트 스토어 이름, S3 URI 인증 기관 위치, 선택적으로 S3 URI 인증서 해지 목록 위치를 제공합니다.
- (선택 사항) TrustStore CA 제목 이름 알림을 활성화할지 선택합니다.

11. 추가를 선택합니다.
12. 선택적 인증서 목록에 인증서를 추가하려면 섹션을 참조하세요인증서 목록에 인증서 추가.

를 사용하여 HTTPS 리스너를 추가하려면 AWS CLI

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

Application Load Balancer를 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 단원을 참조하십시오.

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 Application Load Balancer 대상 그룹 생성 단원을 참조하세요.

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 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택하면 세부 정보를 볼 수 있습니다.
4. 리스너 및 규칙 탭에서 다음 중 하나를 수행하십시오.

   a. 프로토콜: 포트 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다.

      규칙 탭에서 규칙 추가를 선택합니다.

   b. 규칙을 추가하려는 리스너를 선택합니다.

      규칙 관리를 선택한 다음, 규칙 추가를 선택합니다.
5. 이름 및 태그에서 규칙을 위한 이름을 지정할 수 있지만 필수는 아닙니다.

   추가 태그를 추가하려면 추가 태그 추가 텍스트를 선택합니다.
6. 다음을 선택합니다.
7. 조건 추가를 선택합니다.
8. 다음과 같은 조건을 하나 이상 추가합니다.

   - 호스트 헤더 -호스트 헤더를 정의합니다. 예를 들어 `*.example.com`입니다. 확인을 선택하여 조건을 저장합니다.

     최대 128자입니다. 대/소문자를 구분하지 않습니다. 허용되는 문자는 a-z, A-Z, 0-9이고 특수 문자는 -_., 와일드카드(* 및 ?) 입니다. '.' 문자를 하나 이상 포함해야 합니다. 마지막 "." 문자 다음에는 알파벳만 포함할 수 있습니다.
   - 경로 - 경로를 정의합니다. 예를 들어 `/item/*` 입니다. 확인을 선택하여 조건을 저장합니다.

최대 128자입니다. 대소문자 구분. 허용되는 문자는 a-z, A-Z, 0-9이고 특수 문자는 _-.$/~"'@:+; &; and wildcards (* 및 ?)입니다.

- HTTP 요청 방법 - HTTP 요청 방법을 정의합니다. 확인을 선택하여 조건을 저장합니다.

  최대 40자입니다. 대소문자 구분. 허용되는 문자는 A-Z이고 특수 문자는 -_입니다. 와일드카드는 지원되지 않습니다.

- 소스 IP - 소스 IP 주소를 CIDR 형식으로 정의합니다. 확인을 선택하여 조건을 저장합니다.

  IPv4 및 IPv6 CIDR 모두 허용됩니다. 와일드카드는 지원되지 않습니다.

- HTTP 헤더 - 헤더의 이름을 입력하고 비교 문자열을 하나 이상 추가합니다. 확인을 선택하여 조건을 저장합니다.
  - HTTP 헤더 이름— 규칙은 이 헤더가 포함된 요청을 평가하여 일치하는 값을 확인합니다.

    최대 40자입니다. 대/소문자를 구분하지 않습니다. 허용되는 문자는 a-z, A-Z, 0-9이며 특수 문자는 *?-!#$%&'+.^_`|~입니다. 와일드카드는 지원되지 않습니다.

  - HTTP 헤더 값— HTTP 헤더 값과 비교할 문자열을 입력합니다.

    최대 128자입니다. 대/소문자를 구분하지 않습니다. 허용되는 문자는 a-z, A-Z, 0-9, 공백, 특수 문자 !"#$%&'()+,./:;<=>@[]^_`{|}~-, 와일드카드(* 및 ?)입니다.

- 쿼리 문자열 - 쿼리 문자열의 키:값 페어 또는 값을 기반으로 요청을 라우팅합니다. 확인을 선택하여 조건을 저장합니다.

  최대 128자입니다. 대/소문자를 구분하지 않습니다. 허용되는 문자는 a-z, A-Z, 0-9이며 특수 문자는 _-.$/~"'@:+&()!,;=; 및 와일드카드(* 및 ?)입니다.

9. 다음을 선택합니다.
10. 규칙에 사용할 다음 작업 중 하나를 정의하십시오.

    - 대상 그룹에 전달 – 트래픽을 전달할 대상 그룹을 하나 이상 선택합니다. 대상 그룹을 추가하려면 대상 그룹 추가를 선택합니다. 대상 그룹을 하나 이상 사용하는 경우, 각 대상 그룹의 가중치를 선택하고 관련 비율을 검토합니다. 하나 이상의 대상 그룹에서 고정성을 활성화한 경우 규칙에 그룹 수준 고정성을 활성화해야 합니다.
    - URL로 리디렉션 – 클라이언트 요청이 리디렉션될 URL을 지정합니다. URI 파트 탭에서 각 파트를 개별적으로 입력하거나 전체 URL 탭에서 전체 주소를 입력하여 이 작업을 수행합니다. 상태 코드의 경우 요구 사항에 따라 리디렉션을 임시(HTTP 302) 또는 영구(HTTP 301)로 구성할 수 있습니다.

- 고정 응답 반환 – 삭제된 클라이언트 요청으로 반환되는 응답 코드를 지정하십시오. 또한, 콘텐츠 유형 및 응답 본문을 지정할 수 있지만 필수는 아닙니다.

11. 다음을 선택합니다.
12. 1~50000 사이의 값을 입력하여 규칙의 우선 순위를 지정합니다.
13. 다음을 선택합니다.
14. 새 규칙에 대해 현재 구성되어 있는 모든 세부 정보 및 설정을 검토하십시오. 선택 사항에 만족하면 생성을 선택합니다.

를 사용하여 규칙을 추가하려면 AWS CLI

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

규칙 편집

규칙에 대한 작업 및 조건은 언제든지 편집이 가능합니다. 규칙 업데이트는 즉시 적용되지 않으므로 규칙을 업데이트한 후 잠시 동안 이전 규칙 구성을 사용하여 요청을 라우팅할 수 있습니다. 모든 인플라이트 요청이 완료됩니다.

콘솔을 사용하여 규칙을 편집하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리스너 및 규칙 탭에서 다음 중 하나를 수행하십시오.

   - 프로토콜: 포트 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다.

     i. 규칙 탭의 리스너 규칙 섹션에서 편집할 규칙에 대한 이름 태그 열의 텍스트를 선택합니다.

        작업을 선택한 다음, 규칙 편집을 선택합니다.

     ii. 규칙 탭의 리스너 규칙 섹션에서 편집할 규칙을 선택합니다.

        작업을 선택한 다음, 규칙 편집을 선택합니다.

5. 필요에 따라 이름 및 태그를 수정합니다. 추가 태그를 추가하려면 추가 태그 추가 텍스트를 선택합니다.

6. 다음을 선택합니다.
7. 필요에 따라 조건을 수정합니다. 조건을 추가하거나 기존 조건을 편집 또는 삭제할 수 있습니다.
8. 다음을 선택합니다.
9. 필요에 따라 작업을 수정합니다.
10. 다음을 선택합니다.
11. 필요에 따라 규칙 우선 순위를 수정합니다. 1~50000 사이의 값을 입력할 수 있습니다.
12. 다음을 선택합니다.
13. 규칙에 구성되어 있는 모든 세부 정보 및 업데이트된 설정을 검토합니다. 원하는 대로 선택했다면 변경 사항 저장을 선택합니다.

를 사용하여 규칙을 편집하려면 AWS CLI

modify-rule 명령을 사용하세요.

규칙 우선 순위 업데이트

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.

콘솔을 사용하여 규칙 우선 순위를 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리스너 및 규칙 탭에서 다음 중 하나를 수행하십시오.
   a. 프로토콜: 포트 및 규칙 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다.
      i. 작업을 선택한 다음, 규칙의 우선 순위 재지정을 선택합니다.
      ii. 규칙 탭의 리스너 규칙 섹션에서 작업을 선택한 다음, 규칙의 우선 순위 재지정을 선택합니다.
   b. 리스너를 선택합니다.
      - 규칙 관리를 선택한 다음, 규칙의 우선 순위 재지정을 선택합니다

5. 리스너 규칙 섹션에서 우선 순위 열에는 현재 규칙의 우선 순위가 표시됩니다. 1~50000 사이의 값을 입력하여 규칙 우선 순위를 업데이트할 수 있습니다.
6. 변경 사항이 만족스러우면 변경사항 저장을 선택하십시오.

를 사용하여 규칙 우선 순위를 업데이트하려면 AWS CLI

set-rule-priorities 명령을 사용하세요.

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 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리스너 및 규칙 탭에서 다음 중 하나를 수행하십시오.
   a. 프로토콜: 포트 및 규칙 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다.
      i. 삭제할 규칙을 선택합니다.
      ii. 작업을 선택한 다음, 규칙 삭제를 선택합니다
      iii. 텍스트 필드에 `confirm`을(를) 입력하고 삭제를 선택합니다.
   b. 네임 태그 열에서 텍스트를 선택하여 규칙의 세부 정보 페이지를 엽니다.
      i. 작업을 선택한 다음, 규칙 삭제를 선택합니다.
      ii. 텍스트 필드에 `confirm`을(를) 입력하고 삭제를 선택합니다.

를 사용하여 규칙을 삭제하려면 AWS CLI

delete-rule 명령을 사용합니다.

Application Load Balancer용 HTTPS 리스너 업데이트

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

업무

- 기본 인증서 교체
- 인증서 목록에 인증서 추가
- 인증서 목록에서 인증서 제거
- 보안 정책 업데이트
- HTTP 헤더 수정

기본 인증서 교체

다음 절차에 따라 리스너의 기본 인증서를 교체할 수 있습니다. 자세한 내용은 Application Load Balancer용 SSL 인증서 단원을 참조하십시오.

콘솔을 사용하여 기본 인증서를 교체하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리스너 및 규칙 탭에서 프로토콜: 포트 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다.
5. 인증서 탭에서 기본값 변경을 선택합니다.
6. ACM 및 IAM 인증서 표에서 새 기본 인증서를 선택합니다.
7. 기본값으로 저장을 선택합니다.

를 사용하여 기본 인증서를 교체하려면 AWS CLI

modify-listener 명령을 사용하세요.

인증서 목록에 인증서 추가

다음 절차에 따라 리스너 인증서 목록에 인증서를 추가할 수 있습니다. 를 사용하여 리스너를 생성한 경우 기본 인증서를 인증서 목록에 AWS Management Console추가했습니다. 그렇지 않으면 인증서 목록이 비어 있습니다. 인증서 목록에 기본 인증서를 추가하면 기본 인증서로 대체되더라도이 인증서가 SNI 프로토콜과 함께 사용됩니다. 자세한 내용은 Application Load Balancer용 SSL 인증서 단원을 참조하십시오.

콘솔을 사용하여 인증서 목록에 인증서를 추가하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리스너 및 규칙 탭에서 프로토콜: 포트 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다.
5. 인증서 탭에서 인증서 추가를 선택합니다.
6. ACM 또는 IAM에서 이미 관리하는 인증서를 추가하려면 인증서의 확인란을 선택한 다음 아래 보류 중으로 포함을 선택합니다.
7. ACM 또는 IAM에서 관리하지 않는 인증서가 있는 경우 인증서 가져오기를 선택하고 양식을 작성한 다음 가져오기를 선택합니다.
8. 보류 중인 인증서 추가를 선택합니다.

를 사용하여 인증서 목록에 인증서를 추가하려면 AWS CLI

add-listener-certificates 명령을 사용합니다.

인증서 목록에서 인증서 제거

다음 절차에 따라 HTTPS 리스너의 인증서 목록에서 인증서를 제거할 수 있습니다. 인증서를 제거한 후에는 리스너가 더 이상 해당 인증서를 사용하여 연결을 생성할 수 없습니다. 클라이언트가 영향을 받지 않도록 하려면 목록에서 인증서를 제거하기 전에 목록에 새 인증서를 추가하고 연결이 작동하는지 확인합니다.

TLS 리스너의 기본 인증서를 제거하려면 기본 인증서 교체 섹션을 참조하세요.

콘솔을 사용하여 인증서 목록에서 인증서를 제거하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리스너 및 규칙 탭에서 프로토콜: 포트 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다.
5. 인증서 탭에서 인증서의 확인란을 선택하고 제거를 선택합니다.
6. 확인 메시지가 나타나면 **`confirm`**을 입력하고 제거를 선택합니다.

를 사용하여 인증서 목록에서 인증서를 제거하려면 AWS CLI

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 Application Load Balancer 보안 정책 단원을 참조하십시오.

로드 밸런서가 대량의 트래픽을 처리하는 경우 보안 정책을 업데이트하면 중단이 발생할 수 있습니다. 로드 밸런서가 대량의 트래픽을 처리할 때 중단 가능성을 줄이려면 트래픽을 처리하거나 LCU 예약을 요청하는 데 도움이 되는 추가 로드 밸런서를 생성합니다.

Application Load Balancer에서 FIPS 정책 사용

Application Load Balancer에 연결된 모든 보안 리스너는 FIPS 보안 정책 또는 비 FIPS 보안 정책을 사용해야 합니다. 혼합할 수는 없습니다. 기존 Application Load Balancer에 비 FIPS 정책을 사용하는 리스너가 두 개 이상 있을 때 리스너가 대신 FIPS 보안 정책을 사용하도록 하려는 경우 리스너를 하나만 남을 때까지 모두 제거합니다. 리스너의 보안 정책을 FIPS로 변경한 다음 FIPS 보안 정책을 사용하여 추가 리스너를 생성합니다. 또는 FIPS 보안 정책만 사용하여 새 리스너로 새 Application Load Balancer를 생성할 수 있습니다.

콘솔을 사용하여 보안 정책을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.
4. 리스너 및 규칙 탭에서 프로토콜: 포트 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다.
5. 세부 정보 페이지에서 작업을 선택한 다음, 편집 리스너를 선택합니다.
6. 보안 리스너 설정 섹션의 보안 정책에서 새 보안 정책을 선택합니다.
7. 변경 사항 저장을 선택합니다.

를 사용하여 보안 정책을 업데이트하려면 AWS CLI

modify-listener 명령을 사용합니다.

HTTP 헤더 수정

HTTP 헤더 수정을 통해 특정 로드 밸런서 생성 헤더의 이름을 바꾸고, 특정 응답 헤더를 삽입하고, 서버 응답 헤더를 비활성화할 수 있습니다. Application Load Balancer는 요청 헤더와 응답 헤더 모두에 대한 헤더 수정을 지원합니다.

자세한 내용은 Application Load Balancer에 대한 HTTP 헤더 수정 활성화 단원을 참조하십시오.

Application Load Balancer에서 TLS를 사용한 상호 인증

상호 TLS 인증은 Transport Layer Security(TLS)의 변형입니다. 기존 TLS는 서버와 클라이언트 간에 보안 통신을 설정합니다. 여기서 서버는 클라이언트에 ID를 제공해야 합니다. 상호 TLS를 사용하면 로드 밸런서는 TLS를 협상하는 동안 클라이언트와 서버 간의 상호 인증을 협상합니다. Application Load Balancer와 함께 상호 TLS를 사용하면 인증 관리를 간소화하고 애플리케이션의 부하를 줄일 수 있습니다.

Application Load Balancer와 함께 상호 TLS를 사용함으로써 로드 밸런서는 클라이언트 인증을 관리하여 신뢰할 수 있는 클라이언트만 백엔드 애플리케이션과 통신하도록 할 수 있습니다. 이 기능을 사용하면 Application Load Balancer는 타사 인증 기관(CA)의 인증서를 사용하거나 선택적으로 해지 확인을 통해 AWS Private Certificate Authority (PCA)를 사용하여 클라이언트를 인증합니다. Application Load Balancer는 클라이언트 인증서 정보를 백엔드로 전달하고, 애플리케이션은 이를 권한 부여에 사용할 수 있습니다. Application Load Balancer에서 상호 TLS를 사용하면 설정된 라이브러리를 사용하는 인증서 기반 엔터티에 대해 기본 제공되고 확장 가능하며 관리되는 인증을 받을 수 있습니다.

Application Load Balancer용 상호 TLS는 X.509v3 클라이언트 인증서를 검증하기 위한 다음 두 가지 옵션을 제공합니다.

참고: X.509v1 클라이언트 인증서는 지원되지 않습니다.

- 상호 TLS 패스스루: 상호 TLS 패스스루 모드를 사용하는 경우 Application Load Balancer는 HTTP 헤더를 사용하여 전체 클라이언트 인증서 체인을 대상으로 전송합니다. 그런 다음 클라이언트 인증서 체인을 사용하여 애플리케이션에서 해당 로드 밸런서 인증 및 대상 권한 부여 로직을 구현할 수 있습니다.
- 상호 TLS 확인: 상호 TLS 확인 모드를 사용하는 경우 Application Load Balancer는 로드 밸런서가 TLS 연결을 협상할 때 클라이언트에 대해 X.509 클라이언트 인증서 인증을 수행합니다.

Application Load Balancer에서 패스스루를 사용하여 상호 TLS를 시작하려면 클라이언트의 인증서를 수락하도록 리스너를 구성하기만 하면 됩니다. 상호 TLS를 확인 모드로 사용하려면 다음을 수행해야 합니다.

- 새 트러스트 스토어 리소스를 생성합니다.
- 인증 기관(CA) 번들 및 선택적으로 해지 목록을 업로드합니다.
- 클라이언트 인증서를 확인하도록 구성된 리스너에 트러스트 스토어를 연결합니다.

Application Load Balancer를 사용하여 상호 TLS 확인 모드를 구성하는 단계별 절차는 Application Load Balancer에서 상호 TLS 구성 섹션을 참조하세요.

Application Load Balancer에서 상호 TLS 구성을 시작하기 전에

Application Load Balancer에서 상호 TLS를 구성하기 전에 다음 사항에 유의하세요.

할당량

Application Load Balancer에는 AWS 계정 내에서 사용 중인 트러스트 스토어, CA 인증서 및 인증서 해지 목록의 양과 관련된 특정 제한이 포함됩니다.

자세한 내용은 Application Load Balancer에 대한 할당량을 참조하세요.

인증서 요구 사항

Application Load Balancer는 상호 TLS 인증에 사용되는 인증서에 대해 다음을 지원합니다.

- 지원되는 인증서: X.509v3
- 지원되는 퍼블릭 키: RSA 2K~8K 또는 ECDSA secp256r1, secp384r1, secp521r1
- 지원되는 서명 알고리즘: SHA256, 384, 512 with RSA/SHA256, 384, 512 with EC/SHA256, 384, 512 hash with RSASSA-PSS with MGF1

CA 인증서 번들

다음은 인증 기관(CA) 번들에 적용됩니다.

- Application Load Balancer는 각 인증 기관(CA) 인증서 번들을 배치로 업로드합니다. Application Load Balancer는 개별 인증서 업로드를 지원하지 않습니다. 새 인증서를 추가해야 하는 경우 인증서 번들 파일을 업로드해야 합니다.
- CA 인증서 번들을 교체하려면 ModifyTrustStore API를 사용합니다.

패스스루를 위한 인증서 순서

상호 TLS 패스스루를 사용하는 경우 Application Load Balancer는 헤더를 삽입하여 클라이언트 인증서 체인을 백엔드 대상에 표시합니다. 프레젠테이션 순서는 리프 인증서로 시작하고 루트 인증서로 끝납니다.

세션 재개

Application Load Balancer와 함께 상호 TLS 패스스루를 사용하거나 모드를 확인하는 동안에는 세션 재개가 지원되지 않습니다.

HTTP 헤더

Application Load Balancer는 상호 TLS를 사용하여 클라이언트 연결을 협상할 때 `X-Amzn-Mtls` 헤더를 사용하여 인증서 정보를 전송합니다. 자세한 내용 및 예제 헤더는 HTTP 헤더 및 상호 TLS 섹션을 참조하세요.

CA 인증서 파일

CA 인증서 파일은 다음 요구 사항을 충족해야 합니다.

- 인증서 파일은 PEM(Privacy Enhanced Mail) 형식을 사용해야 합니다.
- 인증서 콘텐츠는 `-----BEGIN CERTIFICATE-----` 및 `-----END CERTIFICATE-----` 경계 내에 묶어야 합니다.
- 주석 앞에는 # 문자를 붙여야 하며 - 문자를 포함해서는 안 됩니다.
- 빈 줄이 있으면 안 됩니다.

허용되지 않는(유효하지 않은) 인증서 예:

```
# comments

Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 01
    Signature Algorithm: ecdsa-with-SHA384
        Issuer: C=US, O=EXAMPLE, OU=EXAMPLE, CN=EXAMPLE
        Validity
            Not Before: Jan 11 23:57:57 2024 GMT
            Not After : Jan 10 00:57:57 2029 GMT
        Subject: C=US, O=EXAMPLE, OU=EXAMPLE, CN=EXAMPLE
        Subject Public Key Info:
```

```
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (384 bit)
                pub:
                    00:01:02:03:04:05:06:07:08
                ASN1 OID: secp384r1
                NIST CURVE: P-384
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment, Certificate Sign, CRL Sign
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Subject Key Identifier:
                00:01:02:03:04:05:06:07:08
            X509v3 Subject Alternative Name:
                URI:EXAMPLE.COM
    Signature Algorithm: ecdsa-with-SHA384
         00:01:02:03:04:05:06:07:08
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

허용되는(유효한) 인증서 예:

1. 단일 인증서(PEM 인코딩):

```
# comments
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

2. 여러 인증서(PEM 인코딩):

```
# comments
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
# comments
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

HTTP 헤더 및 상호 TLS

이 섹션에서는 Application Load Balancer가 상호 TLS를 사용하여 클라이언트와의 연결을 협상할 때 인증서 정보를 보내는 데 사용하는 HTTP 헤더에 대해 설명합니다. Application Load Balancer에서 사용하는 특정 `X-Amzn-Mtls` 헤더는 지정한 상호 TLS 모드, 즉 패스스루 모드 또는 확인 모드에 따라 달라집니다.

Application Load Balancer에서 지원하는 기타 HTTP 헤더에 대한 자세한 내용은 HTTP 헤더 및 Application Load Balancer 섹션을 참조하세요.

패스스루 모드용 HTTP 헤더

패스스루 모드의 상호 TLS의 경우 Application Load Balancer는 다음 헤더를 사용합니다.

X-Amzn-Mtls-Clientcert

이 헤더에는 연결에 제공된 전체 클라이언트 인증서 체인의 URL 인코딩 PEM 형식이 포함되어 있습니다(+=/를 안전 문자로 사용).

헤더 콘텐츠 예:

```
X-Amzn-Mtls-Clientcert: -----BEGIN%20CERTIFICATE-----%0AMIID<...reduced...>do0g
%3D%3D%0A-----END%20CERTIFICATE-----%0A-----BEGIN%20CERTIFICATE-----
%0AMIID1<...reduced...>3eZlyKA%3D%3D%0A-----END%20CERTIFICATE-----%0A
```

확인 모드용 HTTP 헤더

확인 모드의 상호 TLS의 경우 Application Load Balancer는 다음 헤더를 사용합니다.

X-Amzn-Mtls-Clientcert-Serial-Number

이 헤더에는 리프 인증서 일련 번호의 16진수 표현이 포함되어 있습니다.

헤더 콘텐츠 예:

```
X-Amzn-Mtls-Clientcert-Serial-Number: 03A5B1
```

X-Amzn-Mtls-Clientcert-Issuer

이 헤더에는 발급자 고유 이름(DN)의 RFC2253 문자열 표현이 포함되어 있습니다.

헤더 콘텐츠 예:

```
X-Amzn-Mtls-Clientcert-Issuer:
 CN=rootcamtls.com,OU=rootCA,O=mTLS,L=Seattle,ST=Washington,C=US
```

X-Amzn-Mtls-Clientcert-Subject

이 헤더에는 주체 고유 이름(DN)의 RFC2253 문자열 표현이 포함되어 있습니다.

헤더 콘텐츠 예:

```
X-Amzn-Mtls-Clientcert-Subject: CN=client_.com,OU=client-3,O=mTLS,ST=Washington,C=US
```

X-Amzn-Mtls-Clientcert-Validity

이 헤더에는 `notBefore` 및 `notAfter` 날짜의 ISO8601 형식이 포함되어 있습니다.

헤더 콘텐츠 예:

```
X-Amzn-Mtls-Clientcert-Validity:
 NotBefore=2023-09-21T01:50:17Z;NotAfter=2024-09-20T01:50:17Z
```

X-Amzn-Mtls-Clientcert-Leaf

이 헤더에는 리프 인증서의 URL 인코딩 PEM 형식이 포함되어 있습니다(+=/를 안전 문자로 사용).

헤더 콘텐츠 예:

```
X-Amzn-Mtls-Clientcert-Leaf: -----BEGIN%20CERTIFICATE-----%0AMIIG<...reduced...>NmrUlw
%0A-----END%20CERTIFICATE-----%0A
```

인증 기관(CA) 제목 이름 알리기

광고 인증 기관(CA) 주제 이름은 클라이언트가 상호 TLS 인증 중에 수락할 인증서를 결정할 수 있도록 지원하여 인증 프로세스를 개선합니다.

CA 주제 이름 알리기를 활성화하면 Application Load Balancer는 연결된 트러스트 스토어를 기반으로 신뢰할 수 있는 인증 기관(CAs) 주제 이름 목록을 알립니다. 클라이언트가 Application Load Balancer를 통해 대상에 연결되면 클라이언트는 신뢰할 수 있는 CA 주제 이름 목록을 수신합니다.

TLS 핸드셰이크 중에 Application Load Balancer가 클라이언트 인증서를 요청하면 인증서 요청 메시지에 신뢰할 수 있는 CA 고유 이름(DNs) 목록이 포함됩니다. 이를 통해 클라이언트는 알려진 CA 주체 이름과 일치하는 유효한 인증서를 선택하여 인증 프로세스를 간소화하고 연결 오류를 줄일 수 있습니다.

신규 및 기존 리스너에서 CA 주제 이름 알림을 활성화할 수 있습니다. 자세한 내용은 HTTPS 리스너 추가 단원을 참조하십시오.

Application Load Balancer에 대한 연결 로그

Elastic Load Balancing은 Application Load Balancer로 전송된 요청에 대한 속성을 캡처하는 연결 로그를 제공합니다. 연결 로그에는 클라이언트 IP 주소 및 포트, 클라이언트 인증서 정보, 연결 결과, 사용 중인 TLS 암호와 같은 정보가 포함됩니다. 이러한 연결 로그를 사용하여 요청 패턴 및 기타 추세를 검토할 수 있습니다.

연결 로그에 대한 자세한 내용은 Application Load Balancer의 연결 로그 섹션을 참조하세요.

Application Load Balancer에서 상호 TLS 구성

이 섹션에는 Application Load Balancer에서 인증하기 위한 상호 TLS 확인 모드를 구성하는 절차가 포함되어 있습니다.

상호 TLS 패스스루 모드를 사용하려면 클라이언트의 인증서를 수락하도록 리스너를 구성하기만 하면 됩니다. 상호 TLS 패스스루를 사용하는 경우 Application Load Balancer는 HTTP 헤더를 사용하여 전체 클라이언트 인증서 체인을 대상으로 전송하므로 애플리케이션에서 해당 인증 및 권한 부여 로직을 구현할 수 있습니다. 자세한 내용은 Application Load Balancer에 대한 HTTPS 리스너 생성을 참조하세요.

상호 TLS 확인 모드를 사용하는 경우 Application Load Balancer는 로드 밸런서가 TLS 연결을 협상할 때 클라이언트에 대해 X.509 클라이언트 인증서 인증을 수행합니다.

상호 TLS 확인 모드를 사용하려면 다음을 수행합니다.

- 새 트러스트 스토어 리소스를 생성합니다.
- 인증 기관(CA) 번들 및 선택적으로 해지 목록을 업로드합니다.
- 클라이언트 인증서를 확인하도록 구성된 리스너에 트러스트 스토어를 연결합니다.

AWS Management Console에서 이 섹션의 절차에 따라 Application Load Balancer에서 상호 TLS 확인 모드를 구성합니다. 콘솔 대신 API 작업을 사용하여 상호 TLS를 구성하려면 Application Load Balancer API 참조 안내서를 참조하세요.

업무

- 트러스트 스토어 생성

- 트러스트 스토어 연결
- 트러스트 스토어 세부 정보 보기
- 트러스트 스토어 수정
- 트러스트 스토어 삭제

트러스트 스토어 생성

트러스트 스토어를 생성할 수 있는 세 가지 방법은 Application Load Balancer를 생성할 때, 보안 리스너를 생성할 때, Trust Store 콘솔 사용입니다. 로드 밸런서 또는 리스너를 생성할 때 트러스트 스토어를 추가하면 트러스트 스토어가 새 리스너와 자동으로 연결됩니다. Trust Store 콘솔을 사용하여 트러스트 스토어를 생성할 때는 수동으로 리스너와 연결해야 합니다.

이 섹션에서는 Trust Store 콘솔을 사용하여 트러스트 스토어를 생성하는 방법을 다루지만 Application Load Balancer 또는 리스너를 생성할 때 사용되는 단계도 동일합니다. 자세한 내용은 로드 밸런서 및 리스너 구성 및 HTTPS 리스너 생성을 참조하세요.

사전 조건:

- 트러스트 스토어를 생성하려면 인증 기관(CA)의 인증서 번들이 있어야 합니다.

콘솔을 사용하여 트러스트 스토어를 생성하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어 생성을 선택합니다.
4. 트러스트 스토어 구성
   a. 트러스트 스토어 이름에 트러스트 스토어의 이름을 입력합니다.
   b. 인증 기관 번들에 트러스트 스토어가 사용할 인증서 번들의 Amazon S3 경로를 입력합니다.

      선택 사항: 객체 버전을 사용하여 이전 버전의 ca 인증서 번들을 선택합니다. 그러지 않으면 현재 버전이 사용됩니다.
5. 해지에서 선택적으로 트러스트 스토어에 인증서 해지 목록을 추가할 수 있습니다.
   - 인증서 해지 목록에 트러스트 스토어가 사용할 인증서 해지 목록의 Amazon S3 경로를 입력합니다.

선택 사항: 객체 버전을 사용하여 이전 버전의 인증서 해지 목록을 선택합니다. 그러지 않으면 현재 버전이 사용됩니다.

6. 트러스트 스토어 태그에서 필요에 따라 최대 50개의 태그를 입력하여 트러스트 스토어에 적용할 수 있습니다.
7. 트러스트 스토어 생성을 선택합니다.

트러스트 스토어 연결

트러스트 스토어를 생성한 후에는 리스너와 연결해야 Application Load Balancer가 트러스트 스토어를 사용할 수 있습니다. 각 보안 리스너에 하나의 트러스트 스토어만 연결할 수 있지만 여러 리스너에 하나의 트러스트 스토어를 연결할 수 있습니다.

이 섹션에서는 트러스트 스토어를 기존 리스너에 연결하는 방법을 다룹니다. Application Load Balancer 또는 리스너를 생성하는 동안에도 트러스트 스토어를 연결할 수 있습니다. 자세한 내용은 로드 밸런서 및 리스너 구성 및 HTTPS 리스너 생성을 참조하세요.

콘솔을 사용하여 트러스트 스토어를 연결하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택하여 세부 정보 페이지를 봅니다.
4. 리스너 및 규칙 탭에서 프로토콜:포트 열의 링크를 선택하여 보안 리스너에 대한 세부 정보 페이지를 엽니다.
5. 보안 탭에서 보안 리스너 설정 편집을 선택합니다.
6. (선택 사항) 상호 TLS가 활성화되지 않은 경우 클라이언트 인증서 처리에서 상호 인증(mTLS)을 선택한 다음 트러스트 스토어로 확인을 선택합니다.
7. 트러스트 스토어에서 생성한 트러스트 스토어를 선택합니다.
8. 변경 사항 저장을 선택합니다.

트러스트 스토어 세부 정보 보기

CA 인증서 번들

CA 인증서 번들은 트러스트 스토어의 필수 구성 요소입니다. 인증 기관에서 검증한 신뢰할 수 있는 루트 및 중간 인증서 모음입니다. 클라이언트는 이러한 검증된 인증서를 통해 로드 밸런서가 제시되는 인증서를 소유하고 있음을 신뢰할 수 있습니다.

언제든지 트러스트 스토어에서 현재 CA 인증서 번들의 내용을 볼 수 있습니다.

CA 인증서 번들 보기

콘솔을 사용하여 CA 인증서 번들을 보려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어를 선택하여 세부 정보 페이지를 봅니다.
4. 작업을 선택한 다음 CA 번들 가져오기를 선택합니다.
5. 링크 공유 또는 다운로드를 선택합니다.

인증서 해지 목록

필요에 따라 트러스트 스토어를 위한 인증서 해지 목록을 생성할 수 있습니다. 해지 목록은 인증 기관에서 릴리스하며 해지된 인증서에 대한 데이터를 포함합니다. Application Load Balancer는 PEM 형식의 인증서 해지 목록만 지원합니다.

인증서 해지 목록은 트러스트 스토어에 추가되면 해지 ID가 부여됩니다. 해지 ID는 트러스트 스토어에 추가된 모든 해지 목록에 대해 증가하며 변경할 수 없습니다. 인증서 해지 목록이 트러스트 스토어에서 삭제되면 해당 해지 ID도 삭제되고 트러스트 스토어 수명 동안 재사용되지 않습니다.

> **Note**
>
> Application Load Balancer는 인증서 해지 목록 내에서 음의 일련 번호가 있는 인증서를 해지할 수 없습니다.

인증서 해지 목록 보기

콘솔을 사용하여 해지 목록을 보려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어를 선택하여 세부 정보 페이지를 봅니다.
4. 인증서 해지 목록 탭에서 작업을 선택한 다음 해지 목록 가져오기를 선택합니다.
5. 링크 공유 또는 다운로드를 선택합니다.

트러스트 스토어 수정

트러스트 스토어에는 한 번에 하나의 CA 인증서 번들만 포함될 수 있지만 트러스트 스토어가 생성된 후 언제든지 CA 인증서 번들을 교체할 수 있습니다.

CA 인증서 번들 교체

콘솔을 사용하여 CA 인증서 번들을 교체하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어를 선택하여 세부 정보 페이지를 봅니다.
4. 작업을 선택한 다음 CA 번들 교체를 선택합니다.
5. CA 번들 교체 페이지의 인증 기관 번들에서 원하는 CA 번들의 Amazon S3 위치를 입력합니다.
6. (선택 사항) 객체 버전을 사용하여 이전 버전의 인증서 해지 목록을 선택합니다. 그러지 않으면 현재 버전이 사용됩니다.
7. CA 번들 교체를 선택합니다.

인증서 해지 목록 추가

콘솔을 사용하여 해지 목록을 추가하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어를 선택하여 세부 정보 페이지를 봅니다.
4. 인증서 해지 목록 탭에서 작업을 선택한 다음 해지 목록 추가를 선택합니다.
5. 해지 목록 추가 페이지의 인증서 해지 목록에서 원하는 인증서 해지 목록의 Amazon S3 위치를 입력합니다.
6. (선택 사항) 객체 버전을 사용하여 이전 버전의 인증서 해지 목록을 선택합니다. 그러지 않으면 현재 버전이 사용됩니다.
7. 해지 목록 추가를 선택합니다.

인증서 해지 목록 삭제

콘솔을 사용하여 해지 목록을 삭제하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어를 선택하여 세부 정보 페이지를 봅니다.
4. 인증서 해지 목록 탭에서 작업을 선택한 다음 해지 목록 삭제를 선택합니다.
5. `confirm`을 입력하여 삭제를 확인합니다.
6. 삭제를 선택합니다.

트러스트 스토어 삭제

더 이상 사용할 필요가 없는 트러스트 스토어를 삭제할 수 있습니다.

참고: 현재 리스너와 연결되어 있는 트러스트 스토어는 삭제할 수 없습니다.

콘솔을 사용하여 트러스트 스토어를 삭제하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어를 선택하여 세부 정보 페이지를 봅니다.
4. 작업을 선택한 다음 트러스트 스토어 삭제를 선택합니다.
5. `confirm`을 입력하여 삭제를 확인합니다.
6. 삭제를 선택합니다.

Application Load Balancer용 Elastic Load Balancing 트러스트 스토어 공유

Elastic Load Balancing은 AWS Resource Access Manager (AWS RAM)와 통합되어 트러스트 스토어 공유를 활성화합니다. AWS RAM 는 조직 또는 조직 단위(OU) 간에 AWS 계정 Elastic Load Balancing 트러스트 스토어 리소스를 안전하게 공유할 수 있는 서비스입니다.OUs 계정이 여러 개인 경우 트러스트 스토어를 한 번 생성하고 AWS RAM 를 사용하여 다른 계정에서도 사용하도록 할 수 있습니다. 에서 계정을 관리하는 경우 AWS Organizations트러스트 스토어를 조직의 모든 계정 또는 지정된 조직 단위(OUs.

를 사용하면 리소스 AWS RAM공유를 생성하여 소유한 리소스를 공유할 수 있습니다. 리소스 공유는 공유할 리소스와 공유 대상 소비자를 지정합니다. 이 모델에서 트러스트 스토어를 AWS 계정 소유한 (소유자)는 이를 다른 AWS 계정 (소비자)과 공유합니다. 소비자는 자신의 계정에서 트러스트 스토어를 연결하는 것과 동일한 방식으로 공유 트러스트 스토어를 Application Load Balancer 리스너에 연결할 수 있습니다.

트러스트 스토어 소유자는 트러스트 스토어를 다음과 공유할 수 있습니다.

- 에서 조직 AWS 계정 내부 또는 외부에 특정 AWS Organizations
- 의 조직 내 조직 단위 AWS Organizations
- 의 전체 조직 AWS Organizations

내용

- 트러스트 스토어 공유를 위한 사전 조건
- 공유 트러스트 스토어에 대한 권한
- 트러스트 스토어 공유
- 트러스트 스토어 공유 중지
- 결제 및 측정

트러스트 스토어 공유를 위한 사전 조건

- 를 사용하여 리소스 공유를 생성해야 합니다 AWS Resource Access Manager. 자세한 내용은 AWS RAM 사용 설명서의 리소스 공유 생성을 참조하세요.
- 트러스트 스토어를 공유하려면에서 트러스트 스토어를 소유해야 합니다 AWS 계정. 내게 공유된 트러스트 스토어를 공유할 수 없습니다.
- AWS Organizations의 조직 또는 조직 단위와 트러스트 스토어를 공유하려면 AWS Organizations와의 공유를 활성화해야 합니다. 자세한 내용은 AWS RAM 사용 설명서의 AWS Organizations과(와) 공유 활성화를 참조하세요.

공유 트러스트 스토어에 대한 권한

트러스트 스토어 소유자

- 트러스트 스토어 소유자는 트러스트 스토어를 생성할 수 있습니다.

- 트러스트 스토어 소유자는 동일한 계정의 로드 밸런서와 함께 트러스트 스토어를 사용할 수 있습니다.
- 트러스트 스토어 소유자는 트러스트 스토어를 다른 AWS 계정 또는와 공유할 수 있습니다 AWS Organizations.
- 트러스트 스토어 소유자는 모든 AWS 계정 또는에서 트러스트 스토어를 공유 해제할 수 있습니다 AWS Organizations.
- 트러스트 스토어 소유자는 로드 밸런서가 동일한 계정의 트러스트 스토어를 사용하는 것을 방지할 수 없습니다.
- 트러스트 스토어 소유자는 공유 트러스트 스토어를 사용하는 모든 Application Load Balancer를 나열할 수 있습니다.
- 트러스트 스토어 소유자는 현재 연결이 없는 경우 트러스트 스토어를 삭제할 수 있습니다.
- 트러스트 스토어 소유자는 공유 트러스트 스토어와의 연결을 삭제할 수 있습니다.
- 트러스트 스토어 소유자는 공유 트러스트 스토어가 사용될 때 CloudTrail 로그를 수신합니다.

트러스트 스토어 소비자

- 트러스트 스토어 소비자는 공유 트러스트 스토어를 볼 수 있습니다.
- 트러스트 스토어 소비자는 동일한 계정의 트러스트 스토어를 사용하여 리스너를 생성하거나 수정할 수 있습니다.
- 트러스트 스토어 소비자는 공유 트러스트 스토어를 사용하여 리스너를 생성하거나 수정할 수 있습니다.
- 트러스트 스토어 소비자는 더 이상 공유되지 않는 트러스트 스토어를 사용하여 리스너를 생성할 수 없습니다.
- 트러스트 스토어 소비자는 공유 트러스트 스토어를 수정할 수 없습니다.
- 트러스트 스토어 소비자는 리스너와 연결된 공유 트러스트 스토어 ARN을 볼 수 있습니다.
- 트러스트 스토어 소비자는 공유 트러스트 스토어를 사용하여 리스너를 생성하거나 수정할 때 CloudTrail 로그를 수신합니다.

관리형 권한

트러스트 스토어를 공유할 때 리소스 공유는 관리형 권한을 사용하여 트러스트 스토어 소비자가 허용하는 작업을 제어합니다. 사용 가능한 모든 권한이 포함된 기본 관리형 권한 `AWSRAMPermissionElasticLoadBalancingTrustStore`를 사용하거나 자체 고객 관리형 권

한을 생성할 수 있습니다. `DescribeTrustStores`, `DescribeTrustStoreRevocations` 및 `DescribeTrustStoreAssociations` 권한은 항상 활성화되어 있으며 제거할 수 없습니다.

트러스트 스토어 리소스 공유에는 다음 권한이 지원됩니다.

elasticloadbalancing:CreateListener

공유 트러스트 스토어를 새 리스너에 연결할 수 있습니다.

elasticloadbalancing:ModifyListener

공유 트러스트 스토어를 기존 리스너에 연결할 수 있습니다.

elasticloadbalancing:GetTrustStoreCaCertificatesBundle

공유 트러스트 스토어와 연결된 ca 인증서 번들을 다운로드할 수 있습니다.

elasticloadbalancing:GetTrustStoreRevocationContent

공유 트러스트 스토어와 연결된 해지 파일을 다운로드할 수 있습니다.

elasticloadbalancing:DescribeTrustStores(기본)

소유 및 계정과 공유되는 모든 트러스트 스토어를 나열할 수 있습니다.

elasticloadbalancing:DescribeTrustStoreRevocations(기본)

특정 트러스트 스토어 arn에 대한 모든 해지 콘텐츠를 나열할 수 있습니다.

elasticloadbalancing:DescribeTrustStoreAssociations(기본)

공유 트러스트 스토어와 연결된 트러스트 스토어 소비자 계정의 모든 리소스를 나열할 수 있습니다.

트러스트 스토어 공유

트러스트 스토어를 공유하려면 리소스 공유에 추가해야 합니다. 리소스 공유는 AWS 계정전반에서 리소스를 공유할 수 있게 해주는 AWS RAM 리소스입니다. 리소스 공유는 공유할 리소스와 공유 대상 소비자 그리고 보안 주체가 수행할 수 있는 작업을 지정합니다. Amazon EC2 콘솔을 사용하여 트러스트 스토어를 공유하면 기존 리소스 공유에 추가하는 것입니다. 새 리소스 공유에 트러스트 스토어를 추가하려면, 우선 AWS RAM 콘솔을 사용해 리소스 공유를 생성해야 합니다.

소유한 트러스트 스토어를 다른 계정과 공유하는 경우 AWS 계정해당 계정이 Application Load Balancer 리스너를 계정의 트러스트 스토어와 연결할 수 있습니다.

의 조직에 속 AWS Organizations 해 있고 조직 내 공유가 활성화된 경우 조직의 소비자에게 공유 트러스트 스토어에 대한 액세스 권한이 자동으로 부여됩니다. 그러지 않으면 소비자는 리소스 공유에 가입하라는 초대장을 받고 초대를 수락한 후 공유된 트러스트 스토어에 대한 액세스 권한을 받습니다.

Amazon EC2 콘솔, AWS RAM 콘솔 또는 AWS CLI를 사용하여 소유하는 트러스트 스토어를 공유할 수 있습니다.

Amazon EC2 콘솔을 사용하여 소유하는 트러스트 스토어를 공유하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱 아래에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어 이름을 선택하여 세부 정보 페이지를 봅니다.
4. 공유 탭에서 트러스트 스토어 공유를 선택합니다.
5. 트러스트 스토어 공유 페이지의 리소스 공유에서 트러스트 스토어를 공유할 리소스 공유를 선택합니다.
6. (선택 사항) 새 리소스 공유를 생성해야 하는 경우 RAM 콘솔에서 리소스 공유 생성 링크를 선택하세요.
7. 트러스트 스토어 공유를 선택합니다.

AWS RAM 콘솔을 사용하여 소유한 트러스트 스토어를 공유하려면

AWS RAM 사용 설명서의 리소스 공유 생성을 참조하세요.

를 사용하여 소유한 트러스트 스토어를 공유하려면 AWS CLI

create-resource-share 명령을 사용합니다.

트러스트 스토어 공유 중지

소유하는 트러스트 스토어의 공유를 중지하려면 리소스 공유에서 제거해야 합니다. 기존 연결은 트러스트 스토어 공유를 중지한 후에도 유지되지만 이전에 공유된 트러스트 스토어에 대한 새 연결은 허용되지 않습니다. 트러스트 스토어 소유자 또는 트러스트 스토어 소비자가 연결을 삭제하면 두 계정 모두에서 삭제됩니다. 트러스트 스토어 소비자가 리소스 공유에서 탈퇴하려는 경우 리소스 공유 소유자에게 계정을 제거해 달라고 요청해야 합니다.

⚠ 연결 삭제

트러스트 스토어 소유자는 DeleteTrustStoreAssociation 명령을 사용하여 기존 트러스트 스토어 연결을 강제로 삭제할 수 있습니다. 연결이 삭제되면 트러스트 스토어를 사용하는 로드 밸런서 리스너는 더 이상 클라이언트 인증서를 확인할 수 없으며 TLS 핸드셰이크에 실패합니다.

Amazon EC2 콘솔, AWS RAM 콘솔 또는 AWS CLI를 사용하여 트러스트 스토어 공유를 중지할 수 있습니다.

Amazon EC2 콘솔을 사용하여 소유하는 트러스트 스토어의 공유를 중지하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱 아래에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어 이름을 선택하여 세부 정보 페이지를 봅니다.
4. 공유 탭의 리소스 공유에서 공유를 중지할 리소스 공유를 선택합니다.
5. 제거를 선택합니다.

AWS RAM 콘솔을 사용하여 소유한 트러스트 스토어 공유를 중지하려면

AWS RAM 사용 설명서에서 리소스 공유 업데이트를 참조하세요.

를 사용하여 소유한 트러스트 스토어 공유를 중지하려면 AWS CLI

disassociate-resource-share 명령을 사용합니다.

결제 및 측정

공유 트러스트 스토어에는 Application Load Balancer와의 트러스트 스토어 연결당 시간 단위로 청구되는 동일한 표준 트러스트 스토어 요금이 발생합니다.

리전별 요금을 포함한 자세한 내용은 Elastic Load Balancing 요금을 참조하세요.

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 소셜 IdP를 사용하여 사용자를 인증합니다.
- Amazon Cognito에서 지원되는 사용자 풀을 통해 SAML, OpenID Connect(OIDC) 또는 OAuth를 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. IdP의 DNS는 공개적으로 확인할 수 있어야 합니다.
- 클라이언트 ID와 클라이언트 암호를 구성해야 합니다.
- IdP가 게시하는 권한 부여, 토큰 및 사용자 정보와 같은 엔드포인트를 가져옵니다. 구성에서 이 정보를 찾을 수 있습니다.
- IdP 엔드포인트 인증서는 신뢰할 수 있는 공개 인증 기관에서 발급해야 합니다.
- 엔드포인트의 DNS 항목은 프라이빗 IP 주소로 확인되더라도 공개적으로 확인할 수 있어야 합니다.
- IdP 앱에서 리디렉션 URL 중 하나를 사용자가 사용하도록 허용합니다. 여기서 DNS는 로드 밸런서의 도메인 이름이고 CNAME은 애플리케이션의 DNS 별칭입니다.
  - https://*DNS*/oauth2/idpresponse
  - https://*CNAME*/oauth2/idpresponse

Amazon Cognito 사용 준비

사용 가능한 리전

Application Load Balancer용 Amazon Cognito 통합은 다음 리전에서 사용할 수 있습니다.

- 미국 동부(버지니아 북부)
- 미국 동부(오하이오)
- 미국 서부(캘리포니아 북부)
- 미국 서부(오레곤)
- 캐나다(중부)
- 캐나다 서부(캘거리)

- 유럽(스톡홀름)
- 유럽(밀라노)
- 유럽(프랑크푸르트)
- 유럽(취리히)
- 유럽(아일랜드)
- 유럽(런던)
- 유럽(파리)
- 유럽(스페인)
- 남아메리카(상파울루)
- 아시아 태평양(홍콩)
- 아시아 태평양(도쿄)
- 아시아 태평양(서울)
- 아시아 태평양(오사카)
- 아시아 태평양(뭄바이)
- 아시아 태평양(하이데라바드)
- 아시아 태평양(싱가포르)
- 아시아 태평양(시드니)
- 아시아 태평양(자카르타)
- 아시아 태평양(멜버른)
- 중동(UAE)
- 중동(바레인)
- 아프리카(케이프타운)
- 이스라엘(텔아비브)

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

- 사용자 풀을 생성합니다. 자세한 내용은 Amazon Cognito 개발자 안내서의 Amazon Cognito 사용자 풀을 참조하세요.
- 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 안내서의 사용자 풀 앱 클라이언트 구성을 참조하세요.

- 사용자 풀 도메인을 생성합니다. 자세한 내용은 Amazon Cognito 개발자 안내서의 사용자 풀 도메인 구성을 참조하세요.
- 요청된 범위가 ID 토큰을 반환하는지 확인하세요. 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 안내서의 타사 자격 증명 공급자를 사용한 사용자 풀 로그인을 참조하세요.
- Amazon Cognito의 콜백 URL 필드에서 다음 리디렉션 URL을 허용합니다. 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 허용합니다. IdP의 형식을 사용합니다. 다음 예를 참조하세요.
  - https://*domain-prefix*.auth.*region*.amazoncognito.com/saml2/idpresponse
  - https://*user-pool-domain*/saml2/idpresponse

앱 클라이언트 설정의 콜백 URL은 모두 소문자를 사용해야 합니다.

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

Amazon CloudFront 사용 준비

Application Load Balancer의 프런트에 CloudFront 배포를 사용하는 경우 다음 설정을 활성화합니다.

- 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.
- Amazon CloudFront와 함께 OpenID Connect(OIDC) 인증을 구성할 때 HTTPS 포트 443이 전체 연결 경로에서 일관되게 사용되는지 확인합니다. 그렇지 않으면 클라이언트 OIDC 리디렉션 URLs 원래 생성된 URI의 포트 번호와 일치하지 않기 때문에 인증 실패가 발생할 수 있습니다.

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 참조하세요.

로드 밸런서는 인증 상태를 유지하기 위해 클라이언트에 세션 쿠키를 보냅니다. 사용자 인증에는 HTTPS 리스너가 필요하므로 이 쿠키에는 항상 `secure` 속성이 포함됩니다. 이 쿠키에는 CORS(Cross-Origin Resource Sharing) 요청이 있는 `SameSite=None` 속성이 포함됩니다.

독립적인 클라이언트 인증이 필요한 여러 애플리케이션을 지원하는 로드 밸런서의 경우 인증 작업이 있는 각 리스너 규칙에는 고유한 쿠키 이름이 있어야 합니다. 이를 통해 클라이언트는 규칙에 지정된 대상 그룹으로 라우팅되기 전에 항상 IdP로 인증됩니다.

Application Load Balancer는 URL로 인코딩된 쿠키 값을 지원하지 않습니다.

기본적으로 `SessionTimeout` 필드는 7일로 설정됩니다. 더 짧은 세션이 필요한 경우 세션 제한 시간을 1초까지 짧게 구성할 수 있습니다. 자세한 내용은 세션 제한 시간 단원을 참조하세요.

애플리케이션에 적절하게 `OnUnauthenticatedRequest` 필드를 구성합니다. 다음 예를 참조하세요.

- 사용자가 소셜 또는 기업 자격 증명을 사용하여 로그인해야 하는 애플리케이션 - 이 작업은 기본 옵션인 `authenticate`에서 지원됩니다. 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 사용합니다. 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 JavaScript가 로드되는 단일 페이지 애플리케이션 - `deny` 옵션을 사용하는 경우 로드 밸런서는 인증 정보가 없는 AJAX 호출에 대해 HTTP 401 권한 없음 오류를 반환합니다. 그러나 사용자가 인증 정보를 만료한 경우 클라이언트를 IdP 권한 부여 엔드포인트로 리디렉션합니다.

로드 밸런서는 IdP 토큰 엔드포인트(`TokenEndpoint`) 및 IdP 사용자 정보 엔드포인트(`UserInfoEndpoint`)와 통신할 수 있어야 합니다. Application Load Balancer는 이러한 엔드포인트와 통신할 때 IPv4만 지원합니다. IdP가 퍼블릭 주소를 사용하는 경우 로드 밸런서의 보안 그룹과 VPC

의 네트워크 ACL이 엔드포인트에 대한 액세스를 허용하는지 확인하세요. 내부 로드 밸런서 또는 IP 주소 유형 dualstack-without-public-ipv4를 사용하는 경우 NAT 게이트웨이를 통해 로드 밸런서가 엔드포인트와 통신할 수 있습니다. 자세한 내용은 Amazon VPC 사용 설명서의 NAT 게이트웨이 기본 사항 단원을 참조하세요.

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 예제입니다. AuthenticationRequestExtraParams는 인증 동안 IdP에 추가 파라미터를 전달할 수 있도록 허용합니다. 지원되는 필드인지 확인하려면 자격 증명 공급자가 제공하는 설명서를 참조하세요.

```
[{
    "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
}]
```

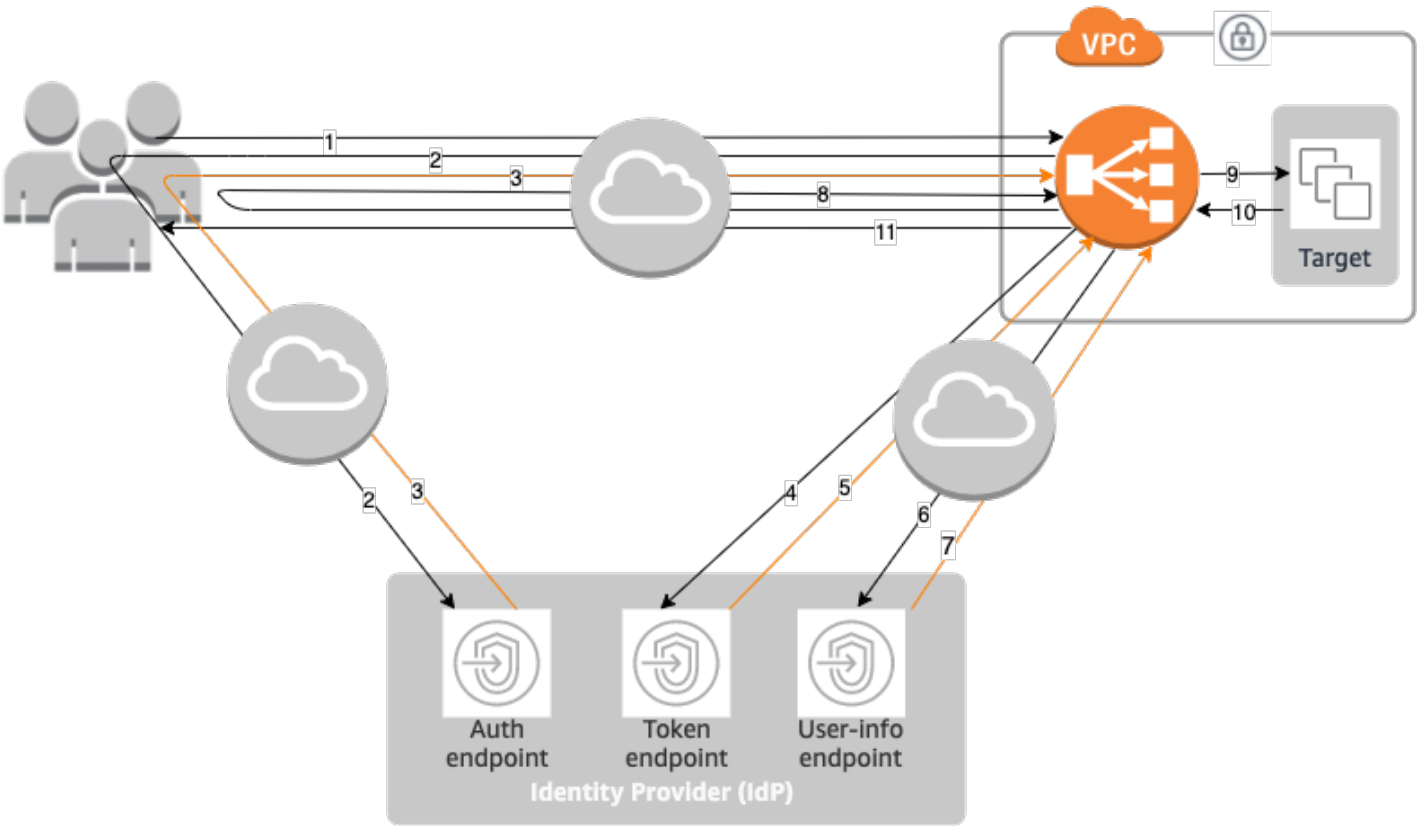
다음은 actions.json 작업 및 authenticate-cognito 작업을 지정하는 forward 파일의 예입니다.

```
[{
    "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 단원을 참조하세요.

인증 흐름

다음 네트워크 다이어그램은 Application Load Balancer가 OIDC를 사용하여 사용자를 인증하는 방법을 시각적으로 보여 줍니다.



아래의 번호가 매겨진 항목은 위의 네트워크 다이어그램에 표시된 요소를 강조 표시하고 설명합니다.

1. 사용자는 Application Load Balancer 뒤에 호스팅되는 웹 사이트에 HTTPS 요청을 보냅니다. 인증 작업이 포함된 규칙에 대한 조건이 충족되면 로드 밸런서는 요청 헤더에서 인증 세션 쿠키를 확인합니다.
2. 쿠키가 없는 경우 로드 밸런서는 IdP가 사용자를 인증할 수 있도록 사용자를 IdP 권한 부여 엔드포인트로 리디렉션합니다.
3. 사용자가 인증된 후 IdP는 권한 부여 코드를 사용하여 사용자를 로드 밸런서로 다시 전송합니다.
4. 로드 밸런서는 IdP 토큰 엔드포인트에 권한 부여 코드를 제공합니다.
5. 유효한 권한 부여 코드를 받으면 IdP가 Application Load Balancer에 ID 토큰 및 액세스 토큰을 제공합니다.
6. 그런 다음 Application Load Balancer가 액세스 토큰을 사용자 정보 엔드포인트로 전송합니다.
7. 사용자 정보 엔드포인트는 사용자 클레임을 액세스 토큰으로 교환합니다.
8. Application Load Balancer는 `AWSELB` 인증 세션 쿠키를 사용하는 사용자를 원래 URI로 리디렉션합니다. 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.

9. Application Load Balancer가 쿠키를 검증하고 사용자 정보를 `X-AMZN-OIDC-*` HTTP 헤더 세트의 대상으로 전송합니다. 자세한 내용은 사용자 클레임 인코딩 및 서명 확인 단원을 참조하십시오.
10. 대상은 Application Load Balancer에 응답을 전송합니다.
11. Application Load Balancer가 최종 응답을 사용자에게 전송합니다.

모든 새 요청은 1~11단계를 거치며 후속 요청은 9~11단계를 거칩니다. 즉, 모든 후속 요청은 쿠키가 만료되지 않은 한 9단계에서 시작됩니다.

`AWSALBAuthNonce` 쿠키는 사용자가 IdP에서 인증한 후 요청 헤더에 추가됩니다. Application Load Balancer가 IdP에서 리디렉션 요청을 처리하는 방식은 변경되지 않습니다.

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 세션 제한 시간 단원을 참조하십시오.

> **Note**
>
> 쿠키 만료는 인증 세션 만료와 다릅니다. 쿠키 만료는 쿠키의 속성이며 7일로 설정됩니다. 인증 세션의 실제 길이는 인증 기능의 Application Load Balancer에서 구성된 세션 제한 시간에 따라 결정됩니다. 이 세션 제한 시간은 마찬가지로 암호화한 인증 쿠키 값에 포함됩니다.

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.

참고: 하위 클레임은 특정 사용자를 식별하는 가장 좋은 방법입니다.

`x-amzn-oidc-data`

사용자 클레임, JSON 웹 토큰(JWT) 형식.

액세스 토큰 및 사용자 클레임은 ID 토큰과 다릅니다. 액세스 토큰 및 사용자 클레임은 서버 리소스에 대한 액세스만 허용하지만, ID 토큰은 사용자를 인증하기 위한 추가 정보를 전달합니다. Application Load Balancer는 사용자를 인증할 때 새 액세스 토큰을 생성하고 액세스 토큰 및 클레임만 백엔드에 전달할 뿐, ID 토큰 정보는 전달하지 않습니다.

이러한 토큰은 JWT 형식을 따르지만 ID 토큰이 아닙니다. JWT 형식에는 base64 URL 방식으로 인코딩된 헤더, 페이로드 및 서명이 포함되고 끝에 패딩 문자가 포함됩니다. Application Load Balancer는 ES256(P-256 및 SHA256을 사용하는 ECDSA)를 사용하여 JWT 서명을 생성합니다.

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 밸런서가 사용자 클레임을 암호화하도록 하려면 HTTPS를 사용하도록 대상 그룹을 구성해야 합니다. 또한 보안 모범 사례로 Application Load Balancer의 트래픽만 수신하도록 대상을 제한하는 것이 좋습니다. 로드 밸런서의 보안 그룹 ID를 참조하도록 대상의 보안 그룹을 구성하여 이 작업을 수행할 수 있습니다.

보안을 보장하려면 클레임을 기반으로 권한을 부여하기 전에 서명을 확인하고 JWT 헤더의 `signer` 필드에 필요한 Application Load Balancer ARN이 포함되어 있는지 확인해야 합니다.

퍼블릭 키를 가져오려면 JWT 헤더에서 키 ID를 가져오고 이 정보를 사용하여 엔드포인트에서 퍼블릭 키를 조회합니다. 각 AWS 리전에 대한 엔드포인트는 다음과 같습니다.

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

AWS GovCloud (US)의 경우 엔드포인트는 다음과 같습니다.

```
https://s3-us-gov-west-1.amazonaws.com/aws-elb-public-keys-prod-us-gov-west-1/key-id
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: Validate the signer
expected_alb_arn = 'arn:aws:elasticloadbalancing:region-code:account-id:loadbalancer/
app/load-balancer-name/load-balancer-id'

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)
received_alb_arn = decoded_json['signer']

assert expected_alb_arn == received_alb_arn, "Invalid Signer"

# Step 2: Get the key id from JWT headers (the kid field)
kid = decoded_json['kid']

# Step 3: 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 4: 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: Validate the signer
expected_alb_arn = 'arn:aws:elasticloadbalancing:region-code:account-id:loadbalancer/
app/load-balancer-name/load-balancer-id'

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)
received_alb_arn = decoded_json['signer']

assert expected_alb_arn == received_alb_arn, "Invalid Signer"

# Step 2: Get the key id from JWT headers (the kid field)
kid = decoded_json['kid']

# Step 3: 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 4: Get the payload
payload = jwt.decode(encoded_jwt, pub_key, algorithms=['ES256'])
```

고려 사항

- 이 예제에서는 토큰의 서명을 사용하여 발행자의 서명을 검증하는 방법을 다루지 않습니다.
- 표준 라이브러리는 JWT 형식의 Application Load Balancer 인증 토큰에 포함된 패딩과 호환되지 않습니다.

제한 시간

세션 제한 시간

새로 고침 토큰과 세션 제한 시간은 다음과 같이 연계됩니다.

- 세션 제한 시간이 액세스 토큰 만료 시간보다 짧은 경우 로드 밸런서는 세션 제한 시간을 준수합니다. 사용자에게 IdP가 포함된 활성 세션이 있는 경우 다시 로그인하라는 메시지가 표시되지 않을 수 있습니다. 그러지 않으면 사용자가 로그인하도록 리디렉션됩니다.
  - IdP 세션 제한 시간이 Application Load Balancer 세션 제한 시간보다 긴 경우, 사용자는 다시 로그인하기 위해 자격 증명을 제공할 필요가 없습니다. 대신, IdP는 새 권한 부여 코드를 사용하여 Application Load Balancer로 다시 리디렉션합니다. 다시 로그인하지 않는 경우에도 인증 코드는 한 번만 사용합니다.
  - IdP 세션 제한 시간이 Application Load Balancer 세션 제한 시간 이하인 경우 사용자는 다시 로그인하기 위해 자격 증명을 제공해야 합니다. 다시 로그인한 후 IdP는 새 권한 부여 코드를 사용하여 Application Load Balancer로 다시 리디렉션하고 나머지 인증 흐름은 요청이 백엔드에 도달할 때까지 계속됩니다.
- 세션 제한 시간이 액세스 토큰 만료 시간보다 길고 IdP가 새로 고침 토큰을 지원하지 않는 경우 로드 밸런서는 제한 시간이 초과될 때까지 인증 세션을 유지합니다. 그런 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.

클라이언트 로그인 시간 초과

클라이언트는 15분 이내에 인증 프로세스를 시작하고 완료해야 합니다. 클라이언트가 15분 한도 내에서 인증을 완료하지 못하면 로드 밸런서에서 HTTP 401 오류가 발생합니다. 이 제한 시간은 변경하거나 제거할 수 없습니다.

예를 들어 사용자가 Application Load Balancer를 통해 로그인 페이지를 로드하는 경우 15분 이내에 로그인 프로세스를 완료해야 합니다. 15분 시간 초과가 만료된 후 사용자가 기다렸다가 로그인을 시도하면 로드 밸런서는 HTTP 401 오류를 반환합니다. 사용자는 페이지를 새로 고치고 다시 로그인을 시도해야 합니다.

인증 로그아웃

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 토큰이 있는 만료된 액세스 토큰이 있는 세션 쿠키를 로드 밸런서에 제공하는 경우 로드 밸런서는 IdP에 연락하여 사용자가 여전히 로그인되어 있는지 확인합니다.

클라이언트 로그아웃 랜딩 페이지는 인증되지 않습니다. 즉, 인증이 필요한 Application Load Balancer 규칙 뒤에는 있을 수 없습니다.

- 요청이 대상에 전송되면 애플리케이션은 모든 인증 쿠키에 대해 만료를 -1로 설정해야 합니다. Application Load Balancer는 최대 16K 크기의 쿠키를 지원하므로 클라이언트로 전송하는 샤드를 최대 4개까지 생성할 수 있습니다.
  - IdP에 로그아웃 엔드포인트가 있는 경우 IdP 로그아웃 엔드포인트(예: Amazon Cognito 개발자 안내서에 설명된 LOGOUT 엔드포인트)로 리디렉션을 실행해야 합니다.
  - IdP에 로그아웃 엔드포인트가 없는 경우 요청은 클라이언트 로그아웃 랜딩 페이지로 돌아가고 로그인 프로세스가 다시 시작됩니다.
- IdP에 로그아웃 엔드포인트가 있다고 가정하면 IdP는 액세스 토큰과 새로 고침 토큰을 만료하고 사용자를 클라이언트 로그아웃 랜딩 페이지로 다시 리디렉션해야 합니다.
- 후속 요청은 원래의 인증 흐름을 따릅니다.

Application Load Balancer 리스너 및 규칙에 대한 태그

태그를 사용하면 리스너와 규칙을 다양한 방식으로 분류할 수 있습니다. 예를 들어 용도, 소유자 또는 환경별로 리소스를 태깅할 수 있습니다.

각 리스너 및 규칙에 여러 태그를 추가할 수 있습니다. 태그 키는 리스너 및 규칙마다 고유해야 합니다. 리스너 및 규칙에 이미 연결된 키로 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 + - = . _ : / @입니다. 선행 또는 후행 공백을 사용하면 안 됩니다.
- 태그 이름 또는 값에 `aws:` 접두사를 사용하지 마세요. AWS 사용하기 위해 예약되어 있기 때문입니다. 이 접두사가 지정된 태그 이름이나 값은 편집하거나 삭제할 수 없습니다. 이 접두사가 지정된 태그는 리소스당 태그 수 제한에 포함되지 않습니다.

리스너 태그를 업데이트합니다

콘솔을 사용하여 리스너용 태그를 업데이트합니다

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 로드 밸런서를 선택합니다.
3. 업데이트하려는 리스너가 포함된 로드 밸런서의 이름을 선택하면 해당 리스너의 세부 정보 페이지가 열립니다.
4. 리스너 및 규칙 탭에서 다음 중 하나를 수행하십시오.

   a. 프로토콜: 포트 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다.

      태그 탭에서 태그 관리를 선택합니다.

   b. 태그를 업데이트하려는 리스너를 선택합니다.

      리스너 관리를 선택한 다음, 태그 관리를 선택합니다.

   c. 태그 열에 있는 텍스트를 선택하여 태그 탭에서 리스너 세부 정보 페이지를 엽니다.

      태그 관리를 선택합니다.
5. 태그 관리 페이지에서 하나 이상의 작업을 수행하십시오.

   a. 태그를 업데이트하려면 키 및 값에 새 값을 입력합니다.

   b. 태그를 추가하려면 새 태그 추가를 선택하고 키 및 값에 값을 입력합니다.

   c. 태그를 삭제하려면 태그 옆의 제거를 선택합니다.
6. 태그 업데이트를 마쳤으면 변경 사항 저장(Save changes)을 선택합니다.

를 사용하여 리스너의 태그를 업데이트하려면 AWS CLI

add-tags 및 remove-tags 명령을 사용합니다.

규칙 태그를 업데이트합니다

콘솔을 사용하여 규칙용 태그 업데이트하기

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 로드 밸런서를 선택합니다.
3. 업데이트하려는 규칙이 포함된 로드 밸런서의 이름을 선택하여 해당 세부 정보 페이지를 엽니다.
4. 리스너 및 규칙 탭에서 업데이트할 규칙이 포함된 리스너에 대한 프로토콜: 포트 열의 텍스트를 선택하여 리스너에 대한 세부 정보 페이지를 엽니다
5. 리스너 세부 정보 페이지에서 다음 중 하나를 수행하십시오.

   a. 네임 태그 열에서 텍스트를 선택하여 규칙의 세부 정보 페이지를 엽니다.

      규칙 세부 정보 페이지에서 태그 관리를 선택합니다.

   b. 업데이트하려는 규칙용 태그 열에 있는 텍스트를 선택합니다.

      태그 요약 팝업에서 태그 관리를 선택합니다.
6. 태그 관리 페이지에서 하나 이상의 작업을 수행하십시오.

   a. 태그를 업데이트하려면 키 및 값에 새 값을 입력합니다.
   b. 태그를 추가하려면 새 태그 추가를 선택하고 키 및 값에 값을 입력합니다.
   c. 태그를 삭제하려면 태그 옆의 제거를 선택합니다.
7. 태그 업데이트를 마쳤으면 변경 사항 저장(Save changes)을 선택합니다.

를 사용하여 규칙의 태그를 업데이트하려면 AWS CLI

add-tags 및 remove-tags 명령을 사용합니다.

Application Load Balancer를 위한 리스너 삭제

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

콘솔을 사용하여 리스너를 삭제하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.
3. 로드 밸런서를 선택합니다.

4. 리스너 및 규칙 탭에서 리스너의 확인란을 선택하고 작업, 리스너 삭제를 선택합니다.
5. 확인 메시지가 나타나면 **confirm**을 입력하고 Delete(삭제)를 선택합니다.

를 사용하여 리스너를 삭제하려면 AWS CLI

delete-listener 명령을 사용하세요.

Application Load Balancer에 대한 HTTP 헤더 수정

HTTP 헤더 수정은 요청 헤더와 응답 헤더 모두에 대해 Application Load Balancer에서 지원됩니다. 애플리케이션 코드를 업데이트할 필요 없이 헤더 수정을 통해 애플리케이션의 트래픽과 보안을 더 잘 제어할 수 있습니다.

헤더 수정을 활성화하려면 섹션을 참조하세요헤더 수정 활성화.

mTLS/TLS 헤더 이름 바꾸기

헤더 이름 바꾸기 기능을 사용하면 Application Load Balancer가 생성하여 요청에 추가하는 mTLS 및 TLS 헤더의 이름을 구성할 수 있습니다.

HTTP 헤더를 수정하는 이러한 기능을 통해 Application Load Balancer는 특정 형식의 요청 및 응답 헤더를 사용하는 애플리케이션을 쉽게 지원할 수 있습니다.

헤더	설명	
X-Amzn-Mtls-Clientcert-Serial-Number	대상이 TLS 핸드셰이크 중에 클라이언트가 제공하는 특정 인증서를 식별하고 확인할 수 있는지 확인합니다.	
X-Amzn-Mtls-Clientcert-Issuer	대상은 인증서를 발급한 인증 기관을 식별하여 클라이언트 인증서를 검증하고 인증할 수 있습니다.	
X-Amzn-Mtls-Clientcert-Subject	대상에 클라이언트 인증서가 발급된 엔터티에 대한 자세한 정보를 제공하여 mTLS 인증	

헤더	설명	
	중에 식별, 인증, 권한 부여 및 로깅에 도움이 됩니다.	
X-Amzn-Mtls-Clientcert-Validity	대상에서 사용 중인 클라이언트 인증서가 정의된 유효 기간 내에 있는지 확인할 수 있도록 허용하여 인증서가 만료되거나 조기에 사용되지 않도록 합니다.	
X-Amzn-Mtls-Clientcert-Leaf	mTLS 핸드셰이크에 사용되는 클라이언트 인증서를 제공하여 서버가 클라이언트를 인증하고 인증서 체인을 검증할 수 있도록 합니다. 이렇게 하면 연결이 안전하고 인증됩니다.	
X-Amzn-Mtls-Clientcert	전체 클라이언트 인증서를 처리합니다. 대상이 mTLS 핸드셰이크 프로세스 중에 인증서의 신뢰성을 확인하고, 인증서 체인을 검증하고, 클라이언트를 인증하도록 허용합니다.	
X-Amzn-TLS-Version	연결에 사용되는 TLS 프로토콜의 버전을 나타냅니다. 이를 통해 통신의 보안 수준을 확인하고, 연결 문제를 해결하고, 규정 준수를 보장할 수 있습니다.	

헤더	설명	
X-Amzn-TLS-Cipher-Suite	TLS에서 연결을 보호하는 데 사용되는 암호화 알고리즘의 조합을 나타냅니다. 이를 통해 서버는 연결의 보안을 평가하여 호환성 문제 해결을 지원하고 보안 정책 준수를 보장할 수 있습니다.	

응답 헤더 추가

삽입 헤더를 사용하여 응답에 보안 관련 헤더를 추가하도록 Application Load Balancer를 구성할 수 있습니다. 이러한 속성을 사용하면 HSTS, CORS 및 CSP를 포함한 헤더를 삽입할 수 있습니다.

기본적으로 이러한 헤더는 비어 있습니다. 이 경우 Application Load Balancer는이 응답 헤더를 수정하지 않습니다.

응답 헤더를 활성화하면 Application Load Balancer는 구성된 값이 있는 헤더를 모든 응답에 추가합니다. 대상의 응답에 HTTP 응답 헤더가 포함된 경우 로드 밸런서는 헤더 값을 구성된 값으로 업데이트합니다. 그렇지 않으면 로드 밸런서가 HTTP 응답 헤더를 구성된 값으로 응답에 추가합니다.

헤더	설명
Strict-Transport-Security	지정된 기간 동안 브라우저에 의한 HTTPS 전용 연결을 적용하여 man-in-the-middle 공격, 프로토콜 다운그레이드 및 사용자 오류로부터 보호하는 데 도움이 됩니다. 클라이언트와 대상 간의 모든 통신이 암호화되도록 합니다.
Access-Control-Allow-Origin	대상의 리소스에 다른 오리진에서 액세스할 수 있는지 여부를 제어합니다. 이렇게 하면 무단 액세스를 방지하면서 안전한 교차 오리진 상호 작용이 가능합니다.
Access-Control-Allow-Methods	대상에 교차 오리진 요청을 할 때 허용되는 HTTP 메서드를 지정합니다. 서로 다른 오리진

헤더	설명
	에서 수행할 수 있는 작업을 제어할 수 있습니다.
Access-Control-Allow-Headers	교차 오리진 요청에 포함할 수 있는 사용자 지정 헤더 또는 단순하지 않은 헤더를 지정합니다. 이 헤더를 통해 대상은 서로 다른 오리진의 클라이언트가 전송할 수 있는 헤더를 제어할 수 있습니다.
Access-Control-Allow-Credentials	클라이언트가 쿠키, HTTP 인증 또는 클라이언트 인증서와 같은 자격 증명을 교차 오리진 요청에 포함할지 여부를 지정합니다.
Access-Control-Expose-Headers	대상이 클라이언트가 교차 오리진 요청에서 액세스할 수 있는 추가 응답 헤더를 지정할 수 있도록 허용합니다.
Access-Control-Max-Age	브라우저가 사전 요청의 결과를 캐싱할 수 있는 기간을 정의하여 반복적인 사전 확인의 필요성을 줄입니다. 이를 통해 특정 교차 오리진 요청에 필요한 OPTIONS 요청 수를 줄여 성능을 최적화할 수 있습니다.
Content-Security-Policy	웹 사이트에서 로드하고 실행할 수 있는 스크립트, 스타일, 이미지 등과 같은 리소스를 제어하여 XSS와 같은 코드 삽입 공격을 방지하는 보안 기능입니다.
X-Content-Type-Options	No-Sniff 지시문을 사용하면 브라우저가 리소스의 MIME 유형을 추측하지 못하도록 하여 웹 보안을 강화합니다. 브라우저가 선언된 콘텐츠 유형에 따라 콘텐츠만 해석하도록 합니다.

헤더	설명
X-Frame-Options	웹 페이지를 프레임에 포함할 수 있는지 여부를 제어하여 클릭 잭팅 공격을 방지하는 데 도움이 되는 헤더 보안 메커니즘입니다. DENY 및 SAMEORIGIN과 같은 값은 콘텐츠가 악성 또는 신뢰할 수 없는 웹 사이트에 임베드되지 않도록 할 수 있습니다.

헤더 비활성화

헤더 비활성화를 사용하여 응답에서 `server:awselb/2.0` 헤더를 비활성화하도록 Application Load Balancer를 구성할 수 있습니다. 이렇게 하면 애플리케이션에 추가 보호 계층을 추가하는 동시에 서버별 정보의 노출이 줄어듭니다.

속성 이름은 입니다`routing.http.response.server.enabled`. 사용 가능한 값은 `true` 또는 입니다`false`. 기본값은 `true`입니다.

제한 사항

- 헤더 값에는 다음 문자가 포함될 수 있습니다.
  - 영숫자: `a-z`, `A-Z`및 `0-9`
  - 특수 문자: `_ :;.,\/'?!(){}[]@<>=-+*#&`|~^%`
- 속성 값은 크기가 1K바이트를 초과할 수 없습니다.
- Elastic Load Balancing은 기본 입력 검증을 수행하여 헤더 값이 유효한지 확인합니다. 그러나 검증은 값이 특정 헤더에 대해 지원되는지 확인할 수 없습니다.
- 속성에 빈 값을 설정하면 Application Load Balancer가 기본 동작으로 되돌아갑니다.

Application Load Balancer에 대한 HTTP 헤더 수정 활성화

헤더 수정은 기본적으로 꺼져 있으며 각 리스너에서를 활성화해야 합니다.

콘솔을 사용하여 헤더 수정을 활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 로드 밸런서를 선택합니다.

3. Application Load Balancer를 선택합니다.
4. 리스너 및 규칙 탭에서 리스너를 선택합니다.
5. Attributes(속성) 탭에서 Edit(편집)을 선택합니다.

   참고: 리스너 속성은 그룹으로 구성됩니다. 활성화하려는 기능의 수를 선택합니다.
6. [HTTPS 리스너] 수정 가능한 mTLS/TLS 헤더 이름

   a. 수정 가능한 mTLS/TLS 헤더 이름을 확장합니다.

   b. 수정하려는 모든 요청 헤더의 이름을 활성화하고 제공합니다. 자세한 내용은 the section called “mTLS/TLS 헤더 이름 바꾸기” 단원을 참조하십시오.
7. 응답 헤더 추가

   a. 응답 헤더 추가를 확장합니다.

   b. 추가하려는 모든 응답 헤더의 값을 활성화하고 제공합니다. 자세한 내용은 the section called “응답 헤더 추가” 단원을 참조하십시오.
8. ALB 서버 응답 헤더

   • 서버 헤더를 활성화하거나 비활성화합니다.
9. 변경 사항 저장을 선택합니다.

를 사용하여 헤더 수정을 활성화하려면 AWS CLI

modifymodify-listener-attributes 명령을 다음 속성과 함께 사용합니다.

`routing.http.request.x_amzn_mtls_clientcert_serial_number.header_name`

X-Amzn-Mtls-Clientcert-Serial-Number의 헤더 이름을 수정합니다.

`routing.http.request.x_amzn_mtls_clientcert_issuer.header_name`

X-Amzn-Mtls-Clientcert-Issuer의 헤더 이름을 수정합니다.

`routing.http.request.x_amzn_mtls_clientcert_subject.header_name`

X-Amzn-Mtls-Clientcert-Subject의 헤더 이름을 수정합니다.

`routing.http.request.x_amzn_mtls_clientcert_validity.header_name`

X-Amzn-Mtls-Clientcert-Validity의 헤더 이름을 수정합니다.

`routing.http.request.x_amzn_mtls_clientcert_leaf.header_name`

X-Amzn-Mtls-Clientcert-Leaf의 헤더 이름을 수정합니다.

`routing.http.request.x_amzn_mtls_clientcert.header_name`

X-Amzn-Mtls-Clientcert의 헤더 이름을 수정합니다.

`routing.http.request.x_amzn_tls_version.header_name`

X-Amzn-Tls-Version의 헤더 이름을 수정합니다.

`routing.http.request.x_amzn_tls_cipher_suite.header_name`

X-Amzn-Tls-Cipher-Suite의 헤더 이름을 수정합니다.

`routing.http.response.server.enabled`

HTTP 응답 서버 헤더를 허용할지 또는 제거할지를 나타냅니다.

`routing.http.response.strict_transport_security.header_value`

Strict-Transport-Security 헤더를 추가하여 브라우저에 사이트는 HTTPS를 통해서만 액세스해야 하며 향후 HTTP를 사용하여 사이트에 액세스하려는 모든 시도는 자동으로 HTTPS로 변환되어야 함을 알립니다.

`routing.http.response.access_control_allow_origin.header_value`

Access-Control-Allow-Origin 헤더를 추가하여 서버에 액세스할 수 있는 오리진을 지정합니다.

`routing.http.response.access_control_allow_methods.header_value`

Access-Control-Allow-Methods 헤더를 추가하여 다른 오리진에서 서버에 액세스할 때 허용되는 HTTP 메서드를 지정합니다.

`routing.http.response.access_control_allow_headers.header_value`

Access-Control-Allow-Headers 헤더를 추가하여 교차 오리진 요청 중에 허용되는 헤더를 지정합니다.

`routing.http.response.access_control_allow_credentials.header_value`

Access-Control-Allow-Credentials 헤더를 추가하여 브라우저가 교차 오리진 요청에 쿠키 또는 인증과 같은 자격 증명을 포함해야 하는지 여부를 나타냅니다.

`routing.http.response.access_control_expose_headers.header_value`

Access-Control-Expose-Headers 헤더를 추가하여 브라우저가 요청 클라이언트에 노출할 수 있는 헤더를 지정합니다.

`routing.http.response.access_control_max_age.header_value`

Access-Control-Max-Age 헤더를 추가하여 사전 요청의 결과를 캐시할 수 있는 기간을 초 단위로 지정합니다.

`routing.http.response.content_security_policy.header_value`

Content-Security-Policy 헤더를 추가하여 브라우저에서 적용되는 제한을 지정하면 특정 유형의 보안 위협 위험을 최소화하는 데 도움이 됩니다.

`routing.http.response.x_content_type_options.header_value`

X-Content-Type-Options 헤더를 추가하여 Content-Type 헤더에 광고된 MIME 유형을 따라야 하는지 여부와 변경해서는 안 되는지 여부를 나타냅니다.

`routing.http.response.x_frame_options.header_value`

X-Frame-Options 헤더를 추가하여 브라우저가 프레임, iframe, 임베드 또는 객체에서 페이지를 렌더링할 수 있는지 여부를 나타냅니다.

Application Load Balancer 대상 그룹

대상 그룹은 지정한 프로토콜과 포트 번호를 사용하여 EC2 인스턴스 같은 개별 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.

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. 하나의 로드 밸런서에만 각 대상 그룹을 사용할 수 있습니다. 자세한 내용은 Application Load Balancer 구성 요소 단원을 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
- 대상 유형
- IP 주소 유형
- 프로토콜 버전
- 등록된 대상
- 대상 그룹 속성
- 라우팅 알고리즘
- 대상 그룹 상태
- Application Load Balancer 대상 그룹 생성
- Application Load Balancer 대상 그룹의 상태 설정 업데이트
- Application Load Balancer 대상 그룹의 상태 확인
- Application Load Balancer 대상 그룹 속성 편집

- Application Load Balancer 대상 그룹에 대상 등록
- Lambda 함수를 Application Load Balancer의 대상으로 사용
- Application Load Balancer 대상 그룹에 대한 태그
- Application Load Balancer 대상 그룹 삭제

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 구성되었거나 HTTPS 상태 확인을 사용하는 경우 HTTPS 리스너가 TLS 1.3 보안 정책을 사용하면 `ELBSecurityPolicy-TLS13-1-0-2021-06` 보안 정책이 대상 연결에 사용됩니다. 그러지 않으면 `ELBSecurityPolicy-2016-08` 보안 정책이 사용됩니다. 로드 밸런서는 대상에 설치된 인증서를 사용하여 대상과의 TLS 연결을 설정합니다. 로드 밸런서는 이러한 인증서를 검증하지 않습니다. 따라서 자체 서명된 인증서 또는 만료된 인증서를 사용할 수 있습니다. 로드 밸런서 및 그 대상이 가상 프라이빗 클라우드(VPC)에 있으므로 로드 밸런서와 대상 간의 트래픽은 패킷 수준에서 인증됩니다. 따라서 대상의 인증서가 유효하지 않더라도 중간자 공격이나 스푸핑이 발생할 위험이 없습니다. 에서 나가는 트래픽 AWS 에는 동일한 보호 기능이 없으며 트래픽을 추가로 보호하기 위해 추가 단계가 필요할 수 있습니다.

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)

> **Important**
>
> 공개적으로 라우팅 가능한 IP 주소는 지정할 수 없습니다.

지원되는 모든 CIDR 블록을 사용하여 다음 대상을 대상 그룹에 등록할 수 있습니다.

- 로드 밸런서 VPC(동일한 리전 또는 다른 리전)로 피어링된 VPC의 인스턴스.
- AWS IP 주소 및 포트로 주소를 지정할 수 있는 리소스(예: 데이터베이스).
- AWS Direct Connect 또는 Site-to-Site VPN 연결을 AWS 통해에 연결된 온프레미스 리소스입니다.

> **Note**
>
> Local Zone 내에 배포된 Application Load Balancer의 경우 트래픽을 수신하려면 `ip` 대상이 동일한 로컬 영역에 있어야 합니다.
> 자세한 내용은 AWS 로컬 영역이란 무엇입니까?를 참조하세요.

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 Lambda 함수를 Application Load Balancer의 대상으로 사용 단원을 참조하십시오.

Amazon Elastic Container Service(Amazon ECS)를 Application Load Balancer의 대상으로 구성할 수 있습니다. 자세한 내용은 Amazon Elastic Container Service 개발자 안내서의 Amazon ECS용 Application Load Balancer 사용을 참조하세요.

IP 주소 유형

새 대상 그룹 생성 시 대상 그룹의 IP 주소 유형을 선택할 수 있습니다. 이는 대상과 통신하고 상태를 확인하는 데 사용되는 IP 버전을 제어합니다.

Application Load Balancer의 대상 그룹은 다음 IP 주소 유형을 지원합니다.

`ipv4`

로드 밸런서는 IPv4를 사용하여 대상과 통신합니다.

`ipv6`

로드 밸런서는 IPv6를 사용하여 대상과 통신합니다.

고려 사항

- 로드 밸런서는 대상 그룹의 IP 주소 유형에 따라 대상과 통신합니다. IPv4 대상 그룹의 대상은 로드 밸런서의 IPv4 트래픽을 수락하고 IPv6 대상 그룹의 대상은 로드 밸런서의 IPv6 트래픽을 수락해야 합니다.
- `ipv4` 로드 밸런서에는 IPv6 대상 그룹을 사용할 수 없습니다.
- IPv6 대상 그룹에 Lambda 함수를 등록할 수 없습니다.

프로토콜 버전

기본적으로 Application Load Balancer는 HTTP/1.1을 사용하여 대상에 요청을 보냅니다. 프로토콜 버전을 사용하면 HTTP/2 또는 gRPC를 사용하여 대상에 요청을 보낼 수 있습니다.

다음 표에는 요청 프로토콜과 대상 그룹 프로토콜 버전의 조합에 대한 결과가 요약되어 있습니다.

요청 프로토콜	프로토콜 버전	결과
HTTP/1.1	HTTP/1.1	성공
HTTP/2	HTTP/1.1	성공
gRPC	HTTP/1.1	오류
HTTP/1.1	HTTP/2	오류
HTTP/2	HTTP/2	성공
gRPC	HTTP/2	대상이 gRPC를 지원하는 경우 성공
HTTP/1.1	gRPC	오류
HTTP/2	gRPC	POST 요청 시 성공
gRPC	gRPC	성공

gRPC 프로토콜 버전에 대한 고려 사항

- 지원되는 유일한 리스너 프로토콜은 HTTPS입니다.
- 리스너 규칙에 대해 지원되는 유일한 작업 유형은 `forward`입니다.
- 지원되는 유일한 대상 유형은 `instance` 및 `ip`입니다.
- 로드 밸런서는 gRPC 요청을 구문 분석하고 패키지, 서비스 및 메서드를 기반으로 gRPC 호출을 적절한 대상 그룹으로 라우팅합니다.
- 로드 밸런서는 단항, 클라이언트 측 스트리밍, 서버 측 스트리밍, 양방향 스트리밍을 지원합니다.
- 사용자 지정 상태 확인 방법을 `/package.service/method` 형식으로 제공해야 합니다.
- 대상으로부터 응답 성공을 확인할 때 사용할 gRPC 상태 코드를 지정해야 합니다.
- Lambda 함수를 대상으로 사용할 수 없습니다.

HTTP/2 프로토콜 버전에 대한 고려 사항

- 지원되는 유일한 리스너 프로토콜은 HTTPS입니다.
- 리스너 규칙에 대해 지원되는 유일한 작업 유형은 `forward`입니다.

- 지원되는 유일한 대상 유형은 `instance` 및 `ip`입니다.
- 로드 밸런서는 클라이언트에서 오는 스트리밍을 지원합니다. 로드 밸런서는 대상으로 가는 스트리밍을 지원하지 않습니다.

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.

애플리케이션에 대한 요구가 증가하면 이를 처리하기 위해 하나 이상의 대상 그룹에 추가 대상을 등록할 수 있습니다. 로드 밸런서는 등록 프로세스가 완료되고 대상이 구성된 임계값에 관계없이 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 Scaling은 대상을 시작할 때 대상 그룹에 해당 대상을 등록합니다. 자세한 내용은 Amazon EC2 Auto Scaling 사용 설명서에서 로드 밸런서를 Auto Scaling 그룹에 연결을 참조하세요.

Limits

- 동일한 VPC에서 다른 Application Load Balancer의 IP 주소는 등록할 수 없습니다. 다른 Application Load Balancer가 로드 밸런서 VPC에 피어링된 VPC에 있는 경우 해당 IP 주소를 등록할 수 있습니다.
- 로드 밸런서 VPC(동일한 리전 또는 다른 리전)로 피어링된 VPC의 인스턴스는 인스턴스 ID로 등록할 수 없습니다. 이러한 인스턴스는 IP 주소로 등록할 수 있습니다.

대상 그룹 속성

속성을 편집하여 대상 그룹을 구성할 수 있습니다. 자세한 내용은 대상 그룹 속성 편집 단원을 참조하십시오.

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. 기본 값은 300초입니다.

`load_balancing.algorithm.type`

로드 밸런싱 알고리즘은 요청을 라우팅할 때 로드 밸런서가 대상을 선택하는 방법을 결정합니다. 값은 `round_robin`, `least_outstanding_requests` 또는 `weighted_random`입니다. 기본값은 `round_robin`입니다.

`load_balancing.algorithm.anomaly_mitigation`

`load_balancing.algorithm.type`이 `weighted_random`인 경우에만 사용할 수 있습니다. 이상 완화가 활성화되었는지 여부를 나타냅니다. 값은 `on` 또는 `off`입니다. 기본값은 `off`입니다.

`load_balancing.cross_zone.enabled`

교차 영역 로드 밸런싱의 활성화 여부를 나타냅니다. 값은 `true`, `false` 또는 `use_load_balancer_configuration`입니다. 기본값은 `use_load_balancer_configuration`입니다.

`slow_start.duration_seconds`

로드 밸런서가 대상 그룹에 대해 선형으로 증가하는 트래픽 공유를 새로 등록된 대상에 보내는 시간(초 단위). 범위는 30~900초입니다(15분). 기본값은 0초입니다(비활성화).

`stickiness.enabled`

고정 세션을 활성화할지 여부를 나타냅니다. 값은 `true` 또는 `false`입니다. 기본값은 `false`입니다.

`stickiness.app_cookie.cookie_name`

애플리케이션 쿠키의 이름입니다. 애플리케이션 쿠키 이름에는 `AWSALB`, `AWSALBAPP`, 또는 `AWSALBTG` 접두사를 사용할 수 없습니다. 이 단어는 로드 밸런서에서 사용하도록 예약되어 있습니다.

`stickiness.app_cookie.duration_seconds`

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

`stickiness.lb_cookie.duration_seconds`

기간 기반 쿠키 만료 기간(초)입니다. 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`과 `app_cookie`입니다.

`target_group_health.dns_failover.minimum_healthy_targets.count`

정상 상태로 유지되어야 하는 최소 대상 수. 정상 대상 수가 이 값보다 낮으면 DNS에서 영역을 비정상으로 표시하여 트래픽이 정상 영역으로만 라우팅되도록 합니다. 가능한 값은 `off` 또는 1부터 최대 대상 수까지의 정수입니다. `off`인 경우 DNS 장애 조치가 비활성화됩니다. 즉, 대상 그룹의 모든 대상이 비정상인 경우에도 노드가 DNS에서 제거되지 않습니다. 기본 값은 1입니다.

`target_group_health.dns_failover.minimum_healthy_targets.percentage`

정상 상태로 유지되어야 하는 대상의 최소 백분율. 정상 대상 백분율이 이 값보다 낮으면 DNS에서 노드를 비정상으로 표시하여 트래픽이 정상 노드로만 라우팅되도록 합니다. 가능한 값은 `off` 또는 1부터 100까지의 정수입니다. `off`인 경우 DNS 장애 조치가 비활성화됩니다. 즉, 대상 그룹의 모든 대상이 비정상인 경우에도 노드가 DNS에서 제거되지 않습니다. 기본값은 `off`입니다.

`target_group_health.unhealthy_state_routing.minimum_healthy_targets.count`

정상 상태로 유지되어야 하는 최소 대상 수. 정상 대상 수가 이 값보다 낮으면 비정상 대상을 포함한 모든 대상으로 트래픽을 전송합니다. 범위는 1에서 최대 대상 수까지입니다. 기본 값은 1입니다.

`target_group_health.unhealthy_state_routing.minimum_healthy_targets.percentage`

정상 상태로 유지되어야 하는 대상의 최소 백분율. 정상 대상의 백분율이 이 값보다 낮으면 비정상 대상을 포함한 모든 대상으로 트래픽을 전송합니다. 가능한 값은 `off` 또는 1부터 100까지의 정수입니다. 기본값은 `off`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. 기본 값은 `false`입니다. 자세한 내용은 다중 값 헤더 단원을 참조하십시오.

라우팅 알고리즘

라우팅 알고리즘은 로드 밸런서가 요청을 수신할 대상을 결정할 때 사용하는 방법입니다. 기본적으로 라운드 로빈 라우팅 알고리즘이 대상 그룹 레벨에서 요청을 라우팅하는 데 사용됩니다. 애플리케이션의 요구 사항에 따라 최소 미해결 요청 및 가중치 기반 무작위 라우팅 알고리즘도 사용할 수 있습니다.

대상 그룹에는 한 번에 하나의 활성 라우팅 알고리즘만 있을 수 있지만 필요할 때마다 라우팅 알고리즘을 업데이트할 수 있습니다.

스티키 세션을 활성화하면 선택한 라우팅 알고리즘이 초기 대상 선택에 사용됩니다. 동일한 클라이언트의 향후 요청은 선택한 라우팅 알고리즘을 우회하여 동일한 대상으로 전달됩니다.

라운드 로빈

- 라운드 로빈 라우팅 알고리즘은 요청을 대상 그룹 내 정상 대상 간에 순차적으로 균등하게 라우팅합니다.
- 이 알고리즘은 수신되는 요청의 복잡성이 유사하거나, 등록된 대상의 처리 용량이 유사하거나, 요청을 대상 간에 균등하게 분산해야 하는 경우에 일반적으로 사용됩니다.

최소 미해결 요청

- 최소 미해결 요청 라우팅 알고리즘은 진행 중인 요청 수가 가장 적은 대상에 요청을 라우팅합니다.
- 이 알고리즘은 수신되는 요청의 복잡성이 다양하고 등록된 대상의 처리 용량이 다양할 때 일반적으로 사용됩니다.
- HTTP/2를 지원하는 로드 밸런서가 HTTP/1.1만 지원하는 대상을 사용하는 경우 요청을 여러 HTTP/1.1 요청으로 변환합니다. 이 구성에서 최소 미해결 요청 알고리즘은 각 HTTP/2 요청을 여러 요청으로 처리합니다.
- WebSockets를 사용하는 경우 최소 미해결 요청 알고리즘을 사용하여 대상이 선택됩니다. 대상이 선택되면 로드 밸런서는 해당 대상에 대한 연결을 생성하고 이 연결을 통해 모든 메시지를 보냅니다.
- 최소 미해결 요청 라우팅 알고리즘은 느린 시작 모드에서 사용할 수 없습니다.

가중치 기반 무작위

- 가중치 기반 무작위 라우팅 알고리즘은 요청을 대상 그룹 내 정상 대상 간에 무작위 순서로 균등하게 라우팅합니다.
- 이 알고리즘은 자동 대상 가중치(ATW) 이상 완화를 지원합니다.
- 가중치 기반 무작위 라우팅 알고리즘은 느린 시작 모드에서 사용할 수 없습니다.
- 가중치 기반 무작위 라우팅 알고리즘은 스티키 세션에서 사용할 수 없습니다.

대상 그룹의 라우팅 알고리즘 수정

대상 그룹의 라우팅 알고리즘은 언제든지 수정할 수 있습니다.

새 콘솔을 사용하여 라우팅 알고리즘을 수정하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 대상 그룹 세부 정보 페이지의 속성 섹션에서 편집을 선택합니다.
5. 대상 그룹 속성 편집 페이지에 있는 트래픽 구성 섹션의 로드 밸런싱 알고리즘에서 라운드 로빈, 최소 미해결 요청 또는 가중치 기반 무작위를 선택합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 라우팅 알고리즘을 수정하려면 AWS CLI

`load_balancing.algorithm.type` 속성과 함께 modify-target-group-attributes 명령을 사용합니다.

대상 그룹 상태

기본적으로 대상 그룹은 그룹에 정상 대상이 하나 이상 있는 한 정상 그룹으로 간주됩니다. 플릿이 크면 트래픽을 처리하는 정상 대상이 하나만 있는 것으로는 충분하지 않습니다. 대신, 정상이어야 하는 대상의 최소 개수 또는 백분율을 지정하고 정상 대상이 지정된 임계값 아래로 떨어질 경우 로드 밸런서가 취하는 조치를 지정할 수 있습니다. 이렇게 하면 애플리케이션의 가용성이 향상됩니다.

내용

- 비정상 상태 작업
- 요구 사항 및 고려 사항
- 모니터링
- 예제
- 로드 밸런서에 대한 Route 53 DNS 장애 조치 사용

비정상 상태 작업

다음 작업에 대해 정상 임계값을 구성할 수도 있습니다.

- DNS 장애 조치 - 영역의 정상 대상이 임계값 아래로 떨어지면 DNS에서 해당 영역에 대한 로드 밸런서 노드의 IP 주소를 비정상으로 표시합니다. 따라서 클라이언트가 로드 밸런서 DNS 이름을 확인하면 트래픽이 정상 영역으로만 라우팅됩니다.

- 라우팅 장애 조치 - 영역의 정상 대상이 임계값 아래로 떨어지면 로드 밸런서는 비정상 대상을 포함하여 로드 밸런서 노드에 사용할 수 있는 모든 대상으로 트래픽을 전송합니다. 이렇게 하면 특히 대상이 일시적으로 상태 확인을 통과하지 못하는 경우 클라이언트 연결이 성공할 가능성이 높아지고 정상 대상에 과부하가 걸릴 위험이 줄어듭니다.

요구 사항 및 고려 사항

- 대상이 Lambda 함수인 대상 그룹에는 이 기능을 사용할 수 없습니다. Application Load Balancer가 Network Load Balancer 또는 Global Accelerator의 대상인 경우 DNS 장애 조치의 임계값을 구성하지 마십시오.
- 작업에 대해 두 가지 유형의 임계값(개수 및 백분율)을 모두 지정하는 경우 로드 밸런서는 두 임계값 중 하나가 위반될 때 조치를 취합니다.
- 두 작업 모두에 대해 임계값을 지정하는 경우 DNS 장애 조치의 임계값은 라우팅 장애 조치 임계값보다 크거나 같아야 합니다. 그래야 라우팅 장애 조치 시 또는 라우팅 장애 조치 전에 DNS 장애 조치가 발생할 수 있습니다.
- 임계값을 백분율로 지정하면 대상 그룹에 등록된 총 대상 수를 기준으로 값이 동적으로 계산됩니다.
- 총 대상 수는 교차 영역 로드 밸런싱의 활성화 여부를 기반으로 합니다. 교차 영역 로드 밸런싱이 해제된 경우 각 노드는 자체 영역의 대상에만 트래픽을 전송합니다. 즉, 임계값은 활성화된 각 영역의 대상 수에 개별적으로 적용됩니다. 교차 영역 로드 밸런싱이 해제된 경우 각 노드는 활성화된 모든 영역의 모든 대상에 트래픽을 전송합니다. 즉, 지정된 임계값은 활성화된 모든 영역의 총 대상 수에 적용됩니다. 자세한 내용은 Application Load Balancer 대상 그룹에 대한 교차 영역 로드 밸런싱 단원을 참조하십시오.
- DNS 장애 조치가 발생하면 로드 밸런서와 연결된 모든 대상 그룹에 영향을 미칩니다. 나머지 영역에 이러한 추가 트래픽을 처리할 수 있는 충분한 용량이 있는지 확인하세요. 특히 교차 영역 로드 밸런싱이 꺼져 있는 경우에는 더욱 확인하세요.
- DNS 장애 조치를 사용하면 로드 밸런서의 DNS 호스트 이름에서 비정상 영역의 IP 주소를 제거합니다. 하지만 로컬 클라이언트 DNS 캐시에는 DNS 레코드의 TTL(time-to-live)이 만료될 때까지(60초) 이러한 IP 주소가 포함될 수도 있습니다.
- DNS 장애 조치를 사용하면 Application Load Balancer에 연결된 대상 그룹이 여러 개 있고 한 영역에서 대상 그룹이 비정상인 경우 해당 영역에서 다른 대상 그룹이 하나 이상 정상인 경우 DNS 상태 확인이 성공합니다.
- DNS 장애 조치를 사용하면 모든 로드 밸런서 영역이 비정상인 것으로 간주되면 로드 밸런서는 비정상 영역을 포함한 모든 영역으로 트래픽을 전송합니다.

- DNS 장애 조치로 이어질 수 있는 정상 대상이 충분한지 여부와는 다른 요인(예: 영역 상태)이 있습니다.

모니터링

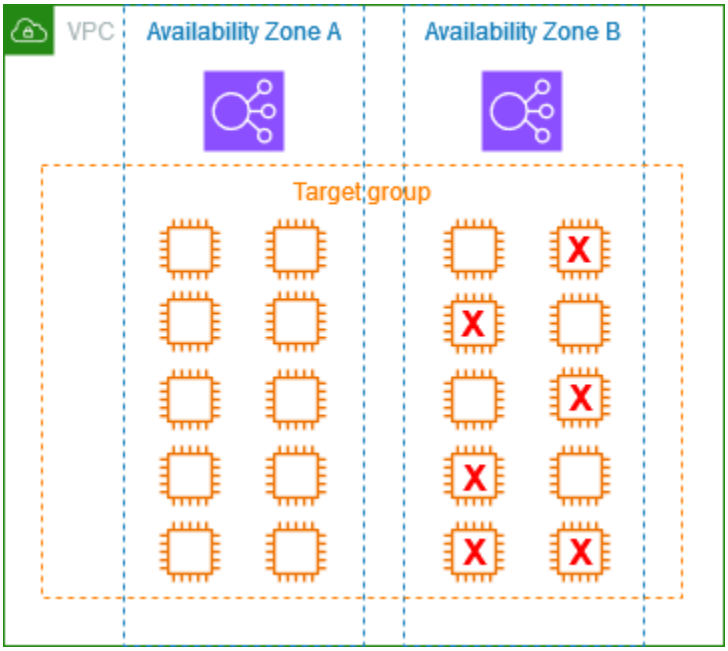
대상 그룹의 상태를 모니터링하려면 대상 그룹 상태에 대한 CloudWatch 지표를 참조하세요.

예제

다음 예는 대상 그룹 상태 설정을 적용하는 방법을 보여줍니다.

시나리오

- 두 개의 가용 영역 A와 B를 지원하는 로드 밸런서
- 각 가용 영역에는 10개의 등록된 대상이 포함됨
- 대상 그룹에는 다음과 같은 대상 그룹 상태 설정이 있습니다.
  - DNS 장애 조치 - 50%
  - 라우팅 장애 조치 - 50%
- 가용 영역 B에서 6개의 대상 장애



교차 영역 로드 밸런싱이 꺼져 있는 경우

- 각 가용 영역에 있는 로드 밸런서 노드는 가용 영역에 있는 10개 대상에만 트래픽을 전송할 수 있습니다.
- 가용 영역 A에는 10개의 정상 대상이 있으며, 이는 정상 대상의 필수 백분율을 충족합니다. 로드 밸런서는 10개의 정상 대상 간에 트래픽을 계속 분산합니다.
- 가용 영역 B에는 정상 대상이 4개뿐이며, 이는 가용 영역 B의 로드 밸런서 노드 대상의 40%에 해당합니다. 이는 정상 대상의 필수 백분율보다 적으므로 로드 밸런서는 다음 작업을 수행합니다.
  - DNS 장애 조치 - 가용 영역 B가 DNS에서 비정상으로 표시됩니다. 클라이언트가 가용 영역 B의 로드 밸런서 노드에 대한 로드 밸런서 이름을 확인할 수 없고 가용 영역 A가 정상이므로 클라이언트는 가용 영역 A에 새 연결을 보냅니다.
  - 라우팅 장애 조치 - 새 연결이 가용 영역 B로 명시적으로 전송되면 로드 밸런서는 비정상 대상을 포함하여 가용 영역 B의 모든 대상으로 트래픽을 분산합니다. 이는 나머지 정상 대상 간의 중단을 방지할 수 있습니다.

교차 영역 로드 밸런싱이 켜져 있는 경우

- 각 로드 밸런서 노드는 두 가용 영역에서 등록된 20개 대상 모두에 트래픽을 전송할 수 있습니다.
- 가용 영역 A에는 10개의 정상 대상이 있고 가용 영역 B에는 4개의 정상 대상이 있으므로 총 14개의 정상 대상이 있습니다. 이는 두 가용 영역 모두에 있는 로드 밸런서 노드에 대한 대상의 70%이며, 정상 대상 중 필수 백분율을 충족합니다.
- 로드 밸런서는 두 가용 영역 모두에서 14개의 정상 대상 간에 트래픽을 계속 분산합니다.

로드 밸런서에 대한 Route 53 DNS 장애 조치 사용

Route 53을 사용하여 로드 밸런서에 DNS 요청을 라우팅하는 경우, Route 53을 사용하여 로드 밸런서에 대한 DNS 장애 조치를 구성할 수도 있습니다. 장애 조치 구성에서 Route 53은 로드 밸런서에 대한 대상 그룹 대상의 상태를 확인하여 가용 여부를 결정합니다. 로드 밸런서에 정상 상태의 대상이 등록되어 있지 않거나 로드 밸런서 자체가 정상 상태가 아니면 Route 53은 정상 상태 로드 밸런서나 Amazon S3의 정적 웹 사이트 같은 또 다른 가용 리소스로 트래픽을 라우팅합니다.

예를 들어 `www.example.com`에 대한 웹 사이트가 있고 서로 다른 리전에 상주하는 두 개의 로드 밸런서에서 중복 인스턴스를 실행하고 싶다고 가정합시다. 한 리전의 로드 밸런서에 트래픽을 주로 라우팅하고 다른 리전의 로드 밸런서는 장애 시 백업으로 사용하고 싶을 수 있습니다. DNS 장애 조치를 구성하면 주 및 보조(백업) 로드 밸런서를 지정할 수 있습니다. Route 53은 주 로드 밸런서가 사용 가능한 상태일 때는 여기로 트래픽을 라우팅하고, 그러지 않으면 보조 로드 밸런서로 라우팅합니다.

대상 상태 평가 작동 방식

- Application Load Balancer의 별칭 레코드Yes에서 대상 상태 평가가 로 설정된 경우 Route 53는 `alias target` 값으로 지정된 리소스의 상태를 평가합니다. Route 53는 대상 그룹 상태 확인을 사용합니다.
- Application Load Balancer에 연결된 모든 대상 그룹이 정상이면 Route 53는 별칭 레코드를 정상으로 표시합니다. 대상 그룹에 대한 임계값을 구성하고 임계값을 충족하는 경우 상태 확인을 통과합니다. 그렇지 않으면 대상 그룹에 정상 대상이 하나 이상 포함된 경우 상태 확인을 통과합니다. 상태 확인이 통과하면 Route 53는 라우팅 정책에 따라 레코드를 반환합니다. 장애 조치 라우팅 정책을 사용하는 경우 Route 53는 기본 레코드를 반환합니다.
- Application Load Balancer에 연결된 대상 그룹이 비정상인 경우 별칭 레코드는 Route 53 상태 확인(fail-open)에 실패합니다. 대상 상태 평가를 사용하는 경우 장애 조치 라우팅 정책은 트래픽을 보조 리소스로 리디렉션합니다.
- Application Load Balancer에 연결된 모든 대상 그룹이 비어 있는 경우(대상 없음) Route 53는 레코드를 비정상으로 간주합니다(fail-open). 대상 상태 평가를 사용하는 경우 장애 조치 라우팅 정책은 트래픽을 보조 리소스로 리디렉션합니다.

자세한 내용은 AWS 블로그의 로드 밸런서 대상 그룹 상태 임계값 사용 및 Amazon Route 53 개발자 안내서의 DNS 장애 조치 구성을 참조하세요.

Application Load Balancer 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 단원을 참조하십시오. 여러 리스너에서 동일한 대상 그룹을 지정할 수 있지만 이러한 리스너는 동일한 Application Load Balancer에 속해야 합니다. 대상 그룹을 로드 밸런서와 함께 사용하려면 대상 그룹이 다른 로드 밸런서용으로 리스너에서 사용되고 있지 않은지 확인해야 합니다.

언제든지 대상 그룹에서 대상을 추가하거나 삭제할 수 있습니다. 자세한 내용은 Application Load Balancer 대상 그룹에 대상 등록 단원을 참조하십시오. 대상 그룹에 대한 상태 확인 설정을 변경할 수

도 있습니다. 자세한 내용은 Application Load Balancer 대상 그룹의 상태 확인 설정 업데이트 단원을 참조하십시오.

콘솔을 사용하여 대상 그룹을 생성하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱(Load Balancing) 아래에서 대상 그룹(Target Groups)을 선택합니다.
3. [대상 그룹 생성(Create target group)]을 선택합니다.
4. 대상 유형 선택(Choose a target type)에서 인스턴스 ID로 대상을 등록하려면 인스턴스(Instances)를, IP 주소로 대상을 등록하려면 IP 주소(IP addresses)를, Lambda 함수를 대상으로 등록하려면 Lambda 함수(Lambda function)를 선택합니다.
5. [대상 그룹 이름(Target group name)]에 대상 그룹의 이름을 입력합니다. 이 이름은 계정당 리전당 고유해야 하고, 최대 32자여야 하며, 알파벳 문자 또는 하이픈만 포함해야 하고, 하이픈으로 시작하거나 끝나지 않아야 합니다.
6. (선택 사항) [프로토콜(Protocol)]과 [포트(Port)]에서 필요에 따라 기본값을 변경합니다.
7. 대상 유형이 인스턴스 또는 IP 주소인 경우 IPv4 또는 IPv6로 IP 주소 유형을 선택하고, 그 외의 경우에는 다음 단계로 건너뜁니다.

   선택한 IP 주소 유형을 가진 대상만 이 대상 그룹에 포함될 수 있습니다. 대상 그룹을 생성한 후에는 IP 주소 유형을 변경할 수 없습니다.
8. VPC에서 Virtual Private Cloud(VPC)를 선택합니다. IP 주소 대상 유형에 대하여 선택할 수 있는 VPC는 이전 단계에서 선택한 IP 주소 유형을 지원하는 VPC입니다.
9. (선택 사항) 프로토콜 버전의 경우 필요시 기본값을 수정합니다.
10. (선택 사항) 상태 확인 섹션에서 필요에 따라 기본 설정을 수정합니다.
11. 대상 유형이 Lambda 함수(Lambda function)인 경우 상태 확인(Health checks) 섹션에서 활성화(Enable)를 선택하여 상태 확인을 활성화할 수 있습니다.
12. (선택 사항) 다음과 같이 하나 이상의 태그를 추가합니다.

    a. 태그 섹션을 확장합니다.

    b. [태그 추가(Add tag)]를 선택합니다.

    c. 태그 키와 태그 값을 입력합니다.
13. 다음을 선택합니다.
14. (선택 사항) 다음과 같이 하나 이상의 대상을 추가합니다.
    - 대상 유형이 인스턴스인 경우 하나 이상의 인스턴스를 선택하고 하나 이상의 포트를 입력한 다음 아래에 보류 중인 것으로 포함을 선택합니다.

참고: IPv6 대상 그룹에 등록하려면 인스턴스에 할당된 기본 IPv6 주소가 있어야 합니다.

- 대상 유형이 IP 주소인 경우 다음을 수행합니다.

  a. 네트워크 VPC를 목록에서 선택하거나 기타 프라이빗 IP 주소를 선택합니다.

  b. IP 주소를 수동으로 입력하거나 인스턴스 세부 정보를 사용하여 IP 주소를 찾습니다. 한 번에 최대 5개의 IP 주소를 입력할 수 있습니다.

  c. 지정된 IP 주소로 트래픽을 라우팅할 포트를 입력합니다.

  d. 아래에서 보류 중인 것으로 포함을 선택합니다.

- 대상 유형이 Lambda 함수인 경우 단일 Lambda 함수를 지정하거나 이 단계를 생략하고 나중에 Lambda 함수를 지정합니다.

15. 대상 그룹 생성을 선택합니다.
16. (선택 사항) 리스너 규칙에서 대상 그룹을 지정할 수 있습니다. 자세한 내용은 리스너 규칙을 참조하십시오.

를 사용하여 대상 그룹을 생성하려면 AWS CLI

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

Application Load Balancer 대상 그룹의 상태 설정 업데이트

다음과 같이 대상 그룹에 대한 대상 그룹 상태 설정을 변경할 수도 있습니다.

콘솔을 사용하여 대상 그룹의 상태 설정을 변경하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 교차 영역 로드 밸런싱이 켜져 있는지 또는 꺼져 있는지 확인합니다. 필요에 따라 이 설정을 업데이트하여 영역에 장애가 발생할 경우 추가 트래픽을 처리할 수 있는 충분한 용량이 있는지 확인하세요.
6. Target group health requirements(대상 그룹 상태 요구 사항)을 확장합니다.
7. Configuration type(구성 유형)의 경우 두 작업에 대해 동일한 임계값을 설정하는 Unified configuration(통합 구성)을 선택하는 것이 좋습니다.

8. Healthy state requirements(정상 상태 요구 사항)의 경우 다음 중 하나를 실시합니다.
   - Minimum healthy target count(최소 정상 대상 개수)를 선택한 다음 1부터 대상 그룹의 최대 대상 수까지의 숫자를 입력합니다.
   - Minimum healthy target percentage(최소 정상 대상 백분율)을 선택한 다음 1부터 100까지의 숫자를 입력합니다.
9. 변경 사항 저장을 선택합니다.

를 사용하여 대상 그룹 상태 설정을 수정하려면 AWS CLI

modify-target-group-attributes 명령을 사용합니다. 다음 예에서는 두 비정상 상태 동작 모두에 대한 정상 임계값을 50%로 설정합니다.

```
aws elbv2 modify-target-group-attributes \
--target-group-arn arn:aws:elasticloadbalancing:region:123456789012:targetgroup/my-targets/73e2d6bc24d8a067 \
--attributes
 Key=target_group_health.dns_failover.minimum_healthy_targets.percentage,Value=50 \

 Key=target_group_health.unhealthy_state_routing.minimum_healthy_targets.percentage,Value=50
```

Application Load Balancer 대상 그룹의 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 밸런서는 상태와 관계없이 모든 해당 대상에 요청을 라우팅합니다. 즉 모든 대상이 활성화된 모든 가용 영역에서 동시에 상태 확인에 실패하면 로드 밸런서가 열리지 않습니다. 오류 시 열림이 적용되면 상태에 관계없이 로드 밸런싱 알고리즘에 따라 활성화된 모든 가용 영역의 모든 대상에 대한 트래픽이 허용됩니다.

WebSockets에서는 상태 확인이 지원되지 않습니다.

자세한 내용은 the section called “대상 그룹 상태” 단원을 참조하십시오.

상태 확인 설정

다음 표에 설명된 대로 대상 그룹의 대상에 대한 상태 확인을 구성합니다. 테이블에 사용되는 설정 이름은 API에 사용되는 이름입니다. 로드 밸런서는 지정된 포트, 프로토콜 및 상태 확인 경로를 사용하여 HealthCheckIntervalSeconds 초마다 모든 등록 대상에 상태 확인 요청을 전송합니다. 각 상태 확인 요청은 독립적이며 결과는 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.

대상 등록을 취소하면 HealthyHostCount는 감소하지만 UnhealthyHostCount는 증가하지 않습니다.

설정	설명
HealthCheckProtocol	대상에 대한 상태 확인을 수행할 때 로드 밸런서가 사용하는 프로토콜입니다. Application Load Balancer의 경우 가능한 프로토콜은 HTTP와 HTTPS입니다. HTTP 프로토콜이 기본 설정값입니다. 이러한 프로토콜은 HTTP GET 메서드를 사용하여 상태 확인 요청을 전송합니다.
HealthCheckPort	대상에 대한 상태 확인을 수행할 때 로드 밸런서가 사용하는 포트입니다. 각 대상이 로드 밸런서에서 트래픽을 수신하는 포트를 사용하도록 기본 설정되어 있습니다.
HealthCheckPath	대상에 대한 상태 확인을 위한 대상입니다. 프로토콜 버전이 HTTP/1.1 또는 HTTP/2인 경우 유효한 URI(/path?query)를 참조하세요. 기본값은 /입니다. 프로토콜 버전이 gRPC인 경우, 사용자 지정 상태 확인 방법의 경로를 `/package.service/method` 형식으로 지정합니다. 기본값은 `/AWS.ALB/healthcheck` 입니다.

설정	설명
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 `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 2~10회입니다. 기본값은 5입니다.
UnhealthyThresholdCount	대상을 비정상 상태로 간주하기까지 필요한 연속적인 상태 확인 실패 횟수입니다. 범위는 2~10회입니다. 기본값은 2입니다.
Matcher	대상으로부터 응답 성공을 확인할 때 사용하는 코드입니다. 이를 콘솔에서 성공 코드라고 합니다. 프로토콜 버전이 HTTP/1.1 또는 HTTP/2인 경우 가능한 값은 200~499입니다. 값 범위(예: "200-299")에서 여러 값(예: "200,202")을 지정할 수 있습니다. 기본값은 200입니다. 프로토콜 버전이 gRPC인 경우 가능한 값은 0~99입니다. 여러 값(예: "0,1") 또는 값 범위(예: "0-5")를 지정할 수 있습니다. 기본값은 12입니다.

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. 관련 사유 코드: `Elb.RegistrationInProgress` | `Elb.InitialHealthChecking`
`healthy`	대상이 정상 상태입니다. 관련 사유 코드: 없음
`unhealthy`	대상이 상태 확인에 응답하지 않았거나 상태 확인에 실패했습니다. 관련 사유 코드: `Target.ResponseCodeMismatch` | `Target.Timeout` | `Target.FailedHealthChecks` | `Elb.InternalError`
`unused`	대상이 대상 그룹에 등록되어 있지 않거나, 대상 그룹이 리스너 규칙에서 사용되지 않거나, 대상이 활성화되지 않은 가용 영역에 있거나, 대상이 중지 상태 또는 종료 상태입니다. 관련 사유 코드: `Target.NotRegistered` | `Target.NotInUse` | `Target.InvalidState` | `Target.IpUnusable`
`draining`	대상이 등록 취소되고 있으며 Connection Draining이 진행 중입니다.

값	설명
	관련 사유 코드: `Target.DeregistrationInProgress`
`unavailable`	대상 그룹에 대한 상태 확인이 비활성화되었습니다. 관련 사유 코드: `Target.HealthCheckDisabled`

상태 확인 사유 코드

대상의 상태가 `Healthy` 이외의 값인 경우에는 API가 문제에 대한 사유 코드와 설명을 반환하고 콘솔이 동일한 설명을 표시합니다. `Elb`로 시작되는 사유 코드는 로드 밸런서 측에서 호출되고, `Target`으로 시작되는 사유 코드는 대상 측에서 호출됩니다. 상태 확인 실패의 다양한 원인에 대한 자세한 내용은 문제 해결을 참조하세요.

사유 코드	설명
`Elb.InitialHealthChecking`	초기 상태 확인이 진행 중
`Elb.InternalError`	내부 오류로 인한 상태 확인 실패
`Elb.RegistrationInProgress`	대상 등록이 진행 중
`Target.DeregistrationInProgress`	대상 등록 취소가 진행 중
`Target.FailedHealthChecks`	상태 확인 실패
`Target.HealthCheckDisabled`	상태 확인 비활성화됨
`Target.InvalidState`	대상이 중지 상태에 있음 대상이 종료 상태에 있음 대상이 종료 또는 중지 상태에 있음

사유 코드	설명
	대상이 잘못된 상태에 있음
`Target.IpUnusable`	로드 밸런서에서 사용 중인 IP 주소이므로 대상으로 사용할 수 없음
`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

Application Load Balancer 대상의 상태 확인

대상 그룹에 등록된 대상의 상태를 확인할 수 있습니다.

콘솔을 사용하여 대상의 상태를 확인하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 대상 탭에서 상태 열은 각 대상의 상태를 나타냅니다.
5. 상태가 `Healthy` 이외의 값인 경우에는 상태 세부 정보 열에 자세한 정보가 포함됩니다. 상태 확인 실패에 대한 도움말은 문제 해결을 참조하세요.

를 사용하여 대상의 상태를 확인하려면 AWS CLI

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

비정상 대상에 대한 이메일 알림을 받으려면

CloudWatch 경보를 통해 Lambda 함수를 트리거하여 비정상 대상에 대한 세부 정보를 전송합니다. 단계별 지침은 블로그 게시물 로드 밸런서의 비정상 대상 식별을 참조하십시오.

Application Load Balancer 대상 그룹의 상태 확인 설정 업데이트

대상 그룹의 상태 확인 설정을 언제든지 업데이트할 수 있습니다.

콘솔을 사용하여 대상 그룹의 상태 확인 설정을 업데이트하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 그룹 세부 정보 탭의 상태 확인 설정 섹션에서 편집을 선택합니다.
5. 상태 확인 설정 편집(Edit health check settings) 페이지에서 필요에 따라 설정을 수정한 다음 변경 사항 저장(Save changes)을 선택합니다.

를 사용하여 대상 그룹의 상태 확인 설정을 수정하려면 AWS CLI

modify-target-group 명령을 사용합니다.

Application Load Balancer 대상 그룹 속성 편집

Application Load Balancer 대상 그룹을 생성한 후 대상 그룹 속성을 편집할 수 있습니다.

대상 그룹 속성

- 등록 취소 지연
- 느린 시작 모드
- Application Load Balancer 대상 그룹에 대한 교차 영역 로드 밸런싱
- 자동 대상 가중치(ATW)
- Application Load Balancer에 대한 고정 세션

등록 취소 지연

Elastic Load Balancing은 등록 취소 중인 대상으로 요청을 전송하는 것을 중지합니다. 기본적으로 Elastic Load Balancing은 등록 취소 프로세스를 완료하기 전에 300초 동안 대기하는데, 이는 대상에

대해 진행 중인 요청을 완료하는 데 도움이 될 수 있습니다. Elastic Load Balancing이 대기하는 시간을 변경하려면 등록 취소 지연 값을 업데이트합니다.

등록 취소하는 대상의 초기 상태는 `draining`입니다. 등록 취소 지연이 경과한 후 등록 취소 프로세스가 완료되며 대상 상태는 `unused`입니다. 대상이 Auto Scaling 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 지연 제한 시간이 초과될 때까지 `draining`으로 표시됩니다. 제한 시간이 초과되면 대상은 `unused` 상태로 전환됩니다.

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 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 그룹 세부 정보 탭의 속성 섹션에서 편집을 선택합니다.
5. 속성 편집 페이지에서 필요에 따라 등록 취소 지연 값을 변경합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 등록 취소 지연 값을 업데이트하려면 AWS CLI

`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 그룹에 대해 느린 시작을 활성화한 후 대상 그룹이 정상으로 간주하면 해당 대상이 느린 시작 모드로 들어갑니다. 느린 시작 모드의 대상은 구성된 느린 시작 기간이 경과하거나 대상이 비정상 상태가 되면 느린 시작 모드를 종료합니다. 느린 시작 모드에서는 로드 밸런서가 대상으로 보낼 수 있는 요청

의 수를 선형으로 증가시킵니다. 정상 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 정상 대상은 느린 시작 모드를 시작하지 않습니다.
- 비어있는 대상 그룹을 위해 느린 시작을 활성화한 다음 단일 등록 작업을 사용하여 대상을 등록하면, 이러한 대상들은 느린 시작 모드를 시작하지 않습니다. 느린 시작 모드 상태가 아닌 정상 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 등록할 경우 대상 그룹이 정상으로 간주하면 느린 시작 모드로 전환됩니다.
- 느린 시작 모드의 대상이 비정상 상태가 되면 대상이 느린 시작 모드를 종료합니다. 대상이 정상 상태가 되면 다시 느린 시작 모드로 전환됩니다.
- 최소 미해결 요청 또는 가중치 기반 무작위 라우팅 알고리즘을 사용할 때는 느린 시작 모드를 활성화할 수 없습니다.

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 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 그룹 세부 정보 탭의 속성 섹션에서 편집을 선택합니다.
5. 속성 편집 페이지에서 필요에 따라 느린 시작 기간의 값을 변경합니다. 느린 시작 모드를 비활성화하려면 지속시간을 0으로 설정합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 느린 시작 기간 값을 업데이트하려면 AWS CLI

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

Application Load Balancer 대상 그룹에 대한 교차 영역 로드 밸런싱

로드 밸런서의 노드는 클라이언트로부터 요청을 가져와서 등록된 대상에 분산합니다. 교차 영역 로드 밸런싱이 켜진 경우 각 로드 밸런서 노드가 등록된 모든 가용 영역에 있는 등록된 대상 간에 트래픽을 분산합니다. 교차 영역 로드 밸런싱이 꺼진 경우 각 로드 밸런서 노드가 해당 가용 영역에 있는 등록된

대상 간에만 트래픽을 분산합니다. 이는 영역 장애 도메인이 지역보다 선호되는 경우 정상 영역이 비정상 영역의 영향을 받지 않도록 하거나 전반적인 지연 시간을 개선하기 위한 것일 수 있습니다.

Application Load Balancer를 사용하면 교차 영역 로드 밸런싱이 로드 밸런서 수준에서 항상 켜져 있으며 끌 수 없습니다. 대상 그룹의 경우 기본적으로 로드 밸런서 설정을 사용하지만 대상 그룹 수준에서 교차 영역 로드 밸런싱을 명시적으로 해제하여 기본값을 재정의할 수 있습니다.

고려 사항

- 교차 영역 로드 밸런싱이 꺼져 있는 경우 대상 고정성이 지원되지 않습니다.
- 교차 영역 로드 밸런싱이 꺼져 있는 경우 대상으로서의 Lambda 함수는 지원되지 않습니다.
- 대상에 매개 변수 `AvailabilityZone`이(가) `all`(으)로 설정된 경우 `ModifyTargetGroupAttributes` API를 통해 교차 영역 로드 밸런싱을 끄려고 하면 오류가 발생합니다.
- 대상을 등록할 때는 `AvailabilityZone` 파라미터가 필요합니다. 교차 영역 로드 밸런싱이 꺼진 경우에만 특정 가용 영역 값을 사용할 수 있습니다. 그러지 않으면 파라미터가 무시되고 `all`(으)로 처리됩니다.

모범 사례

- 대상 그룹별로 활용할 것으로 예상되는 모든 가용 영역에서 충분한 대상 용량을 계획하세요. 참여하는 모든 가용 영역에서 충분한 용량을 계획할 수 없다면 교차 영역 로드 밸런싱을 유지하는 것이 좋습니다.
- 여러 대상 그룹으로 Application Load Balancer를 구성할 때는 모든 대상 그룹이 구성된 지역 내의 동일한 가용 영역에 속해 있는지 확인하세요. 이는 교차 영역 로드 밸런싱이 꺼져 있는 동안 가용 영역이 비어 있는 것을 방지하기 위한 것입니다. 빈 가용 영역으로 들어오는 모든 HTTP 요청에 대해 503 오류를 트리거하기 때문입니다.
- 빈 서브넷을 생성하지 마십시오. Application Load Balancer는 빈 서브넷의 DNS를 통해 영역 IP 주소를 노출하며, 이로 인해 HTTP 요청에 대해 503 오류가 발생합니다.
- 교차 영역 로드 밸런싱이 해제된 대상 그룹의 가용 영역당 계획된 대상 용량이 충분하지만 가용 영역의 모든 대상이 비정상 상태가 되는 경우가 있을 수 있습니다. 비정상 대상이 모두 포함된 대상 그룹이 하나 이상 있는 경우 로드 밸런서 노드의 IP 주소가 DNS에서 제거됩니다. 대상 그룹에 정상 대상이 하나 이상 있으면 IP 주소가 DNS로 복원됩니다.

교차 영역 로드 밸런싱 해제

Application Load Balancer 대상 그룹에 대해 언제든지 교차 영역 로드 밸런싱을 끌 수 있습니다.

콘솔을 사용하여 교차 영역 로드 밸런싱을 해제하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing(로드 밸런싱)에서 Target Groups(대상 그룹)을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. Attributes(속성) 탭에서 Edit(편집)을 선택합니다.
5. Edit target group attributes(대상 그룹 속성 편집) 페이지에서 Cross-zone load balancing(교차 영역 로드 밸런싱)에 대해 Off(해제)를 선택합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 교차 영역 로드 밸런싱을 끄려면 AWS CLI

modify-target-group-attributes 명령을 사용하고 `load_balancing.cross_zone.enabled` 속성을 `false`(으)로 설정합니다.

```
aws elbv2 modify-target-group-attributes --target-group-arn my-targetgroup-arn --
attributes Key=load_balancing.cross_zone.enabled,Value=false
```

다음은 응답의 예입니다.

```
{
    "Attributes": [
        {
            "Key": "load_balancing.cross_zone.enabled",
            "Value": "false"
        },
    ]
}
```

교차 영역 로드 밸런싱 설정

Application Load Balancer 대상 그룹에 대해 언제든지 교차 영역 로드 밸런싱을 켤 수 있습니다. 대상 그룹 수준의 교차 영역 로드 밸런싱 설정은 로드 밸런서 수준의 설정을 재정의합니다.

콘솔을 사용하여 교차 영역 로드 밸런싱을 설정하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing(로드 밸런싱)에서 Target Groups(대상 그룹)을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. Attributes(속성) 탭에서 Edit(편집)을 선택합니다.
5. Edit target group attributes(대상 그룹 속성 편집) 페이지에서 Cross-zone load balancing(교차 영역 로드 밸런싱)에 대해 On(사용)을 선택합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 교차 영역 로드 밸런싱을 켜려면 AWS CLI

modify-target-group-attributes 명령을 사용하고 `load_balancing.cross_zone.enabled` 속성을 `true`(으)로 설정합니다.

```
aws elbv2 modify-target-group-attributes --target-group-arn my-targetgroup-arn --
attributes Key=load_balancing.cross_zone.enabled,Value=true
```

다음은 응답의 예입니다.

```
{
    "Attributes": [
        {
            "Key": "load_balancing.cross_zone.enabled",
            "Value": "true"
        },
    ]
}
```

자동 대상 가중치(ATW)

자동 대상 가중치(ATW)는 애플리케이션을 실행하는 대상을 지속적으로 모니터링하여 이상이라고 하는 심각한 성능 편차를 감지합니다. ATW는 실시간 데이터 이상 탐지를 통해 대상으로 라우팅되는 트래픽의 양을 동적으로 조정할 수 있는 기능을 제공합니다.

자동 대상 가중치(ATW)는 계정의 모든 Application Load Balancer에서 이상 탐지를 자동으로 수행합니다. 이상 대상이 식별되면 ATW는 라우팅되는 트래픽의 양을 줄여 자동으로 안정화를 시도할 수 있습

니다. 이를 이상 완화라고 합니다. ATW는 트래픽 배포를 지속적으로 최적화하여 대상별 성공률을 극대화하는 동시에 대상 그룹 실패율을 최소화합니다.

고려 사항:

- 이상 탐지는 대상이 반환하는 HTTP 5xx 응답 코드와 대상에 대한 연결 실패를 모니터링합니다. 이상 탐지는 항상 켜지며 끌 수 없습니다.
- Lambda를 대상으로 사용하는 경우 ATW는 지원되지 않습니다.

이상 탐지

ATW 이상 탐지는 대상 그룹 내 다른 대상과 비교하여 동작에서 상당한 편차를 나타내는 모든 대상을 모니터링합니다. 이러한 편차, 즉 이상은 한 대상의 오류 백분율과 대상 그룹 내 다른 대상의 오류 백분율을 비교하여 결정됩니다. 이러한 오류는 연결 오류와 HTTP 오류 코드 모두일 수 있습니다. 피어보다 훨씬 더 많은 오류를 보고하는 대상이 이상으로 간주됩니다.

이상 탐지를 위해서는 대상 그룹에 최소 3개의 정상 대상이 필요합니다. 대상이 대상 그룹에 등록되면 먼저 상태 확인을 통과하여 트래픽 수신을 시작해야 합니다. 대상이 트래픽을 수신하면 ATW는 대상 모니터링을 시작하고 이상 결과를 지속적으로 게시합니다. 이상이 없는 대상의 경우 이상 결과는 `normal`입니다. 이상이 있는 대상의 경우 이상 결과는 `anomalous`입니다.

ATW 이상 탐지는 대상 그룹 상태 확인과 독립적으로 작동합니다. 대상은 모든 대상 그룹 상태 확인을 통과할 수 있지만 오류율 증가로 인해 여전히 비정상으로 표시될 수 있습니다. 비정상으로 분류되는 대상은 대상 그룹 상태 확인 상태에 영향을 주지 않습니다.

이상 탐지 상태

ATW는 대상에서 수행하는 이상 탐지의 상태를 지속적으로 게시합니다. AWS Management Console 또는를 사용하여 언제든지 현재 상태를 볼 수 있습니다 AWS CLI.

콘솔을 사용하여 이상 탐지 상태를 보려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 대상 그룹 세부 정보 페이지에서 대상 탭을 선택합니다.
5. 등록된 대상 표의 이상 탐지 결과 열에서 각 대상의 이상 상태를 볼 수 있습니다.

   이상이 탐지되지 않은 경우 결과는 `normal`입니다.

이상이 탐지된 경우 결과는 anomalous입니다.

를 사용하여 이상 탐지 결과를 보려면 AWS CLI

Include.member.N 속성 값을 AnomalyDetection으로 설정하여 describe-target-health 명령을 사용합니다.

이상 완화

> **Important**
>
> ATW의 이상 완화 기능은 가중치 기반 무작위 라우팅 알고리즘을 사용하는 경우에만 사용할 수 있습니다.

ATW 이상 완화는 자동으로 이상 대상을 배제하고 트래픽을 라우팅하여 복구 기회를 제공합니다.

완화 도중:

- ATW는 이상 대상으로 라우팅되는 트래픽의 양을 주기적으로 조정합니다. 현재 이 주기는 5초입니다.
- ATW는 이상 대상에 라우팅되는 트래픽 양을 이상 완화를 수행하는 데 필요한 최소 양으로 줄입니다.
- 더 이상 이상으로 탐지되지 않는 대상은 대상 그룹의 다른 정상 대상과 동등해질 때까지 점차적으로 더 많은 트래픽이 라우팅됩니다.

ATW 이상 완화 활성화

언제든지 이상 완화 기능을 켤 수 있습니다.

콘솔을 사용하여 이상 완화를 켜려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 대상 그룹 세부 정보 페이지의 속성 섹션에서 편집을 선택합니다.

5. 대상 그룹 속성 편집 페이지에 있는 트래픽 구성 섹션의 로드 밸런싱 알고리즘 아래에서 가중치 기반 무작위가 선택되었는지 확인합니다.

   참고: 가중치 기반 무작위 알고리즘을 처음 선택하면 이상 탐지가 기본적으로 켜집니다.
6. 이상 완화에서 이상 완화 켜기가 선택되어 있는지 확인합니다.
7. 변경 사항 저장을 선택합니다.

를 사용하여 이상 완화를 켜려면 AWS CLI

`load_balancing.algorithm.anomaly_mitigation` 속성과 함께 modify-target-group-attributes 명령을 사용합니다.

이상 완화 상태

ATW가 대상에 대한 완화를 수행할 때마다 AWS Management Console 또는를 사용하여 언제든지 현재 상태를 볼 수 있습니다 AWS CLI.

콘솔을 사용하여 이상 완화 상태를 보려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 대상 그룹 세부 정보 페이지에서 대상 탭을 선택합니다.
5. 등록된 대상 표의 완화 적용 중 열에서 각 대상의 이상 완화 상태를 볼 수 있습니다.

   완화가 진행 중이지 않은 경우 상태는 `yes`입니다.

   완화가 진행 중인 경우 상태는 no입니다.

를 사용하여 이상 완화 상태를 보려면 AWS CLI

`Include.member.N` 속성 값을 `AnomalyDetection`으로 설정하여 describe-target-health 명령을 사용합니다.

Application Load Balancer에 대한 고정 세션

기본적으로, Application Load Balancer는 선택한 로드 밸런싱 알고리즘에 따라 각 요청을 등록된 대상으로 독립적으로 라우팅합니다. 한편, 고정 세션 기능(세션 어피니티라고도 함)을 사용해 로드 밸런서

가 사용자의 세션을 특정 대상에 바인딩하도록 할 수 있습니다. 이렇게 하면 세션 중에 사용자로부터 들어오는 모든 요청이 동일한 대상으로 전송됩니다. 이 기능은 클라이언트에게 지속적인 경험을 제공하기 위해 상태 정보를 유지하는 서버에 유용합니다. 고정 세션을 사용하려면 클라이언트가 쿠키를 지원해야 합니다.

Application Load Balancer는 기간 기반 쿠키와 애플리케이션 기반 쿠키를 둘 다 지원합니다. 고정 세션은 대상 그룹 레벨에서 활성화됩니다. 전체 대상 그룹에 대해 기간 기반 고정, 애플리케이션 기반 고정, 고정 없음을 조합하여 사용할 수 있습니다.

고정 세션 관리에서 핵심은 로드 밸런서가 얼마나 오래 사용자 요청을 동일한 대상으로 일관되게 라우팅하는지를 결정하는 것입니다. 애플리케이션이 자체 세션 쿠키를 가지고 있는 경우에는 애플리케이션 기반 고정을 사용할 수 있으며 로드 밸런서 세션 쿠키는 애플리케이션의 세션 쿠키에 지정된 기간을 따릅니다. 애플리케이션이 자체 세션 쿠키를 가지고 있지 않은 경우에는 기간 기반 고정을 사용하여 원하는 기간이 지정된 로드 밸런서 세션 쿠키를 생성할 수 있습니다.

로드 밸런서 생성 쿠키의 내용은 회전 키를 사용하여 암호화됩니다. 로드 밸런서가 생성한 쿠키는 해독하거나 변경할 수 없습니다.

두 고정 유형 모두에서, Application Load Balancer는 모든 요청 후에 생성하는 쿠키의 만료 기간을 재설정합니다. 쿠키가 만료되면 세션은 더 이상 고정되지 않으며 클라이언트는 쿠키 저장소에서 쿠키를 제거해야 합니다.

요구 사항

- HTTP/HTTPS 로드 밸런서입니다.
- 각 가용 영역에 있는 하나 이상의 정상 상태 인스턴스입니다.

고려 사항

- 교차 영역 로드 밸런싱이 비활성화된 경우 스티키 세션이 지원되지 않습니다. 교차 영역 로드 밸런싱이 비활성화된 상태에서 고정 세션을 활성화하려고 시도하면 실패합니다.
- 애플리케이션 기반 쿠키의 경우, 각 대상 그룹에 대해 쿠키 이름을 개별적으로 지정해야 합니다. 그러나 기간 기반 쿠키의 경우, `AWSALB`은(는) 모든 대상 그룹에서 사용되는 유일한 이름입니다.
- Application Load Balancer의 여러 계층을 사용하는 경우, 애플리케이션 기반 쿠키를 사용하여 모든 계층에서 고정 세션을 활성화할 수 있습니다. 그러나 기간 기반 쿠키를 사용하면 하나의 레이어에서만 고정 세션을 활성화할 수 있습니다. `AWSALB`이(가) 사용할 수 있는 유일한 이름이기 때문입니다.
- Application Load Balancer가 `AWSALBCORS` 및 `AWSALB` 기간 기반 고정성 쿠키를 모두 수신하면 `AWSALBCORS`의 값이 우선합니다.

- 애플리케이션 기반 고정은 가중 대상 그룹에서 작동하지 않습니다.
- 대상 그룹이 여러 개인 전달 작업이 있고 하나 이상의 대상 그룹에 고정 세션이 활성화되어 있으면 대상 그룹 레벨에서 고정을 활성화해야 합니다.
- 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.
- Application Load Balancer는 쿠키 헤더에서 `Max-Age` 속성 대신 `Expires` 속성을 사용합니다.
- Application Load Balancer는 URL로 인코딩된 쿠키 값을 지원하지 않습니다.
- 등록 취소로 인해 대상이 드레이닝되는 동안 Application Load Balancer가 새 요청을 수신하면 요청이 정상 대상으로 라우팅됩니다.

기간 기반 고정

기간 기반 고정은 로드 밸런서 생성 쿠키(`AWSALB`)를 사용하여 대상 그룹의 동일한 대상으로 요청을 라우팅합니다. 쿠키는 세션을 대상에 매핑하는 데 사용됩니다. 애플리케이션에 자체 세션 쿠키가 없는 경우, 고유한 고정 기간을 지정하고 로드 밸런서가 사용자 요청을 동일한 대상으로 일관되게 라우팅하는 기간을 관리할 수 있습니다.

처음 클라이언트의 요청을 받으면 로드 밸런서는 (선택한 알고리즘을 기반으로) 요청을 대상으로 라우팅하고 `AWSALB`(이)라는 쿠키를 생성합니다. 선택한 대상에 대한 정보를 인코딩하고, 쿠키를 암호화하고, 클라이언트에 대한 응답에 쿠키를 포함합니다. 로드 밸런서에서 생성된 쿠키는 만료 기간이 7일이며 이는 구성할 수 없습니다.

후속 요청에서 클라이언트는 `AWSALB` 쿠키를 포함해야 합니다. 로드 밸런서는 쿠키를 포함하는 클라이언트로부터 요청을 수신하면 이를 감지하고 해당 요청을 동일한 대상으로 라우팅합니다. 쿠키가 있지만 디코딩할 수 없거나 등록이 해제되었거나 비정상 상태인 대상을 참조하는 경우 로드 밸런서는 새 대상을 선택하고 새 대상에 대한 정보로 쿠키를 업데이트합니다.

크로스 오리진 리소스 공유(CORS) 요청의 경우 고정성을 활성화하려면 `SameSite=None; Secure`가 필요합니다. 이러한 브라우저를 지원하기 위해 로드 밸런서는 항상 원래 고정성 쿠키와 동일한 정보 및 `SameSite` 속성을 포함하는 두 번째 고정성 쿠키 `AWSALBCORS`를 생성합니다. 클라이언트는 CORS가 아닌 요청을 포함하여 두 쿠키를 모두 수신합니다.

콘솔을 사용하여 기간 기반 고정을 활성화하는 방법

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

2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 그룹 세부 정보 탭의 속성 섹션에서 편집을 선택합니다.
5. [Edit attributes] 페이지에서 다음을 수행합니다.
   a. 고정성을 선택합니다.
   b. 고정 유형에서 로드 밸런서 생성 쿠키를 선택합니다.
   c. [Stickiness duration]에서 1초에서 7일 사이의 값을 지정합니다.
   d. 변경 사항 저장을 선택합니다.

를 사용하여 지속 시간 기반 고정을 활성화하려면 AWS CLI

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

다음 명령을 사용하여 기간 기반 고정을 활성화합니다.

```
aws elbv2 modify-target-group-attributes --target-group-arn ARN --attributes
 Key=stickiness.enabled,Value=true
 Key=stickiness.lb_cookie.duration_seconds,Value=time-in-seconds
```

다음 예와 유사하게 출력되어야 합니다:

```
    {
       "Attributes": [
          ...
           {
                "Key": "stickiness.enabled",
                "Value": "true"
             },
             {
                 "Key": "stickiness.lb_cookie.duration_seconds",
                 "Value": "86500"

             },
          ...
        ]
     }
```

애플리케이션 기반 고정

애플리케이션 기반 고정은 클라이언트 대상 고정에 대한 사용자 고유의 기준을 설정할 수 있는 유연성을 제공합니다. 애플리케이션 기반 고정을 활성화하면 로드 밸런서는 선택한 알고리즘을 기반으로 첫 번째 요청을 대상 그룹 내의 대상으로 라우팅합니다. 대상은 고정을 활성화하기 위해 로드 밸런서에 구성된 쿠키와 일치하는 사용자 지정 애플리케이션 쿠키를 설정해야 합니다. 이 사용자 지정 쿠키에는 애플리케이션에 필요한 모든 쿠키 속성이 포함될 수 있습니다.

Application Load Balancer가 대상으로부터 사용자 지정 애플리케이션 쿠키를 수신하면, 애플리케이션 쿠키라는 암호화된 새 쿠키가 자동으로 생성되어 고정 정보를 캡처합니다. 이 로드 밸런서에서 생성한 애플리케이션 쿠키는 애플리케이션 기반 고정성이 활성화된 각 대상 그룹에 대한 고정 정보를 캡처합니다.

로드 밸런서에서 생성한 애플리케이션 쿠키는 대상에 의해 설정된 사용자 지정 쿠키의 속성을 복사하지 않습니다. 여기에는 구성할 수 없는 7일의 자체 만료 기간이 있습니다. 클라이언트에 대한 응답에서 Application Load Balancer는 사용자 지정 쿠키의 값 또는 만료 속성이 아니라, 대상 그룹 수준에서 사용자 지정 쿠키가 구성된 이름만 검증합니다. 이름이 일치하는 한, 로드 밸런서는 클라이언트에 대한 응답에서 로드 밸런서가 생성한 애플리케이션 쿠키 및 대상에서 설정한 사용자 지정 쿠키와 같이 두 쿠키를 모두 전송합니다.

후속 요청에서, 클라이언트는 고정을 유지하기 위해 쿠키를 둘 다 다시 보내야 합니다. 로드 밸런서는 애플리케이션 쿠키를 해독하고 구성된 고정 기간이 여전히 유효한지 여부를 확인합니다. 그런 다음 쿠키에 있는 정보를 사용하여 대상 그룹 내의 동일한 대상으로 요청을 전송하여 고정을 유지합니다. 또한, 로드 밸런서는 사용자 지정 애플리케이션 쿠키를 검사하거나 수정하지 않고 대상으로 프록시합니다. 후속 응답에서 로드 밸런서가 생성한 애플리케이션 쿠키의 만료 및 로드 밸런서에 구성된 고정 기간이 재설정됩니다. 클라이언트와 대상 사이의 고정성을 유지하려면 쿠키의 만료 및 고정 기간이 경과하지 않아야 합니다.

대상이 실패하거나 비정상 상태가 되면 로드 밸런서는 해당 대상으로의 요청 라우팅을 중지하고 선택한 로드 밸런싱 알고리즘을 기반으로 정상 상태의 대상을 새로 선정합니다. 로드 밸런서는 세션을 정상 상태의 새 대상에 “고정”된 것으로 간주하고, 실패한 대상이 다시 오더라도 요청을 정상 상태의 새 대상으로 계속 라우팅합니다.

Cross-Origin Resource Sharing(CORS) 요청을 사용하는 경우 고정성을 활성화하려면 사용자 에이전트 버전이 Chromium80 이상인 경우에만 로드 밸런서가 `SameSite=None; Secure` 속성을 로드 밸런서가 생성한 애플리케이션 쿠키에 추가합니다.

대부분의 브라우저는 쿠키를 4K로 제한하기 때문에 로드 밸런서는 애플리케이션 쿠키를 4K보다 큰 여러 쿠키로 분할합니다. Application Load Balancer는 최대 16K 크기의 쿠키를 지원하므로 클라이언트로 전송하는 샤드를 최대 4개까지 생성할 수 있습니다. 클라이언트에 표시되는 애플리케이션 쿠

키 이름은 "AWSALBAPP-"로 시작하며 조각 번호를 포함합니다. 예를 들어, 쿠키 크기가 0-4K이면 클라이언트에 AWSALBAPP-0이 표시됩니다. 쿠키 크기가 4-8K인 경우 클라이언트에 AWSALBAPP-0, AWSALBAPP-1 등으로 표시됩니다.

콘솔을 사용하여 애플리케이션 기반 고정을 활성화하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 그룹 세부 정보 탭의 속성 섹션에서 편집을 선택합니다.
5. [Edit attributes] 페이지에서 다음을 수행합니다.
   a. 고정성을 선택합니다.
   b. 고정 유형에서 애플리케이션 기반 쿠키를 선택합니다.
   c. [Stickiness duration]에서 1초에서 7일 사이의 값을 지정합니다.
   d. 앱 쿠키 이름에서 애플리케이션 기반 쿠키의 이름을 입력합니다.

      쿠키 이름에는 AWSALB, AWSALBAPP 또는 AWSALBTG을(를) 사용하지 마세요. 이 단어는 로드 밸런서에서 사용하도록 예약되어 있습니다.
   e. 변경 사항 저장을 선택합니다.

를 사용하여 애플리케이션 기반 고정을 활성화하려면 AWS CLI

modify-target-group-attributes 명령을 다음 속성과 함께 사용합니다.

- `stickiness.enabled`
- `stickiness.type`
- `stickiness.app_cookie.cookie_name`
- `stickiness.app_cookie.duration_seconds`

애플리케이션 기반 고정을 활성화하려면 다음 명령을 사용합니다.

```
aws elbv2 modify-target-group-attributes --target-group-arn ARN --attributes
 Key=stickiness.enabled,Value=true Key=stickiness.type,Value=app_cookie
 Key=stickiness.app_cookie.cookie_name,Value=my-cookie-name
 Key=stickiness.app_cookie.duration_seconds,Value=time-in-seconds
```

다음 예와 유사하게 출력되어야 합니다:

```
{
     "Attributes": [
         ...
         {
             "Key": "stickiness.enabled",
             "Value": "true"
         },
         {
             "Key": "stickiness.app_cookie.cookie_name",
             "Value": "MyCookie"
         },
         {
             "Key": "stickiness.type",
             "Value": "app_cookie"
         },
         {
             "Key": "stickiness.app_cookie.duration_seconds",
             "Value": "86500"
         },
         ...
     ]
 }
```

수동 재조정

확장 시 대상 수가 크게 증가하면 고정으로 인해 하중이 불균등하게 분산될 가능성이 있습니다. 이 시나리오에서는 다음 두 옵션을 사용하여 대상에 대한 부하를 재조정할 수 있습니다.

- 애플리케이션에 의해 생성된 쿠키에 대한 만료일을 현재 날짜 및 시간 이전으로 설정합니다. 이렇게 하면 클라이언트가 쿠키를 Application Load Balancer에 보내지 못하게 되어, 고정을 설정하는 프로세스가 다시 시작됩니다.
- 로드 밸런서의 애플리케이션 기반 고정 구성에서 매우 짧은 기간을 설정합니다(예: 1초). 이렇게 하면 대상에 의해 설정된 쿠키가 만료되지 않은 경우에도 Application Load Balancer가 고정을 다시 설정합니다.

Application Load Balancer 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 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 Connection Draining이라고 합니다. Connection Draining이 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. Auto Scaling 그룹에서 대상 그룹을 분리하면 인스턴스가 대상 그룹에서 자동으로 등록 취소됩니다. 자세한 내용은 Amazon EC2 Auto Scaling 사용 설명서에서 로드 밸런서를 Auto Scaling 그룹에 연결을 참조하세요.

대상에서 애플리케이션을 종료할 때는 먼저 대상 그룹에서 대상을 등록 취소하고 기존 연결이 드레이닝될 때까지 기다려야 합니다. describe-target-health CLI 명령을 사용하거나에서 대상 그룹 보기를 새로 고쳐 등록 취소 상태를 모니터링할 수 있습니다 AWS Management Console. 대상이 등록 취소되었는지 확인한 후 애플리케이션 중지 또는 종료를 진행할 수 있습니다. 이 시퀀스는 트래픽을 처리하는 동안 애플리케이션이 종료될 때 사용자에게 5XX 오류가 발생하지 않도록 합니다.

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

소스	포트 범위	Comment
## ### ## ##	#### ###	Allow traffic from the load balancer on the instance listener port
## ### ## ##	## ##	Allow traffic from the load balancer on the health check port

인바운드 ICMP 트래픽이 경로 MTU 검색을 지원하도록 허용하는 것이 좋습니다. 자세한 내용은 Amazon EC2 사용 설명서의 경로 MTU 검색을 참조하세요.

공유 서브넷

참여자는 공유 VPC에서 Application Load Balancer를 생성할 수 있습니다. 참여자는 자신과 공유되지 않은 서브넷에서 실행되는 대상을 등록할 수 없습니다.

대상 등록 또는 등록 취소

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

목차

- 인스턴스 ID로 대상 등록 또는 등록 취소
- IP 주소로 대상 등록 또는 등록 취소
- Lambda 함수 등록 또는 등록 취소
- AWS CLI를 사용하여 대상 등록 또는 등록 취소

인스턴스 ID로 대상 등록 또는 등록 취소

> **Note**
>
> IPv6 대상 그룹의 인스턴스 ID로 대상을 등록하는 경우 대상에 할당된 기본 IPv6 주소가 있어야 합니다. 자세히 알아보려면 Amazon EC2 사용 설명서의 IPv6 주소를 참조하세요.

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

콘솔을 사용하여 인스턴스 ID별로 대상을 등록 또는 등록 취소하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 대상 탭을 선택합니다.
5. 인스턴스를 등록하려면 대상 등록을 선택합니다. 하나 이상의 인스턴스를 선택하고 필요에 따라 기본 인스턴스 포트를 입력한 다음 아래에 보류 중인 것으로 포함을 선택합니다. 인스턴스 추가를 마쳤으면 보류 중인 대상 등록(Register pending targets)을 선택합니다.

   참고:

   - IPv6 대상 그룹에 등록하려면 인스턴스에 할당된 기본 IPv6 주소가 있어야 합니다.
   - AWS GovCloud (US) Region은 콘솔을 사용하여 기본 IPv6 주소를 할당하는 것을 지원하지 않습니다. API를 사용하여 AWS GovCloud (US) Region에서 기본 IPv6 주소를 할당해야 합니다.
6. 인스턴스의 등록을 취소하려면 인스턴스를 선택한 다음 등록 취소를 선택합니다.

IP 주소로 대상 등록 또는 등록 취소

IPv4 대상

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)

동일한 VPC에서 다른 Application Load Balancer의 IP 주소는 등록할 수 없습니다. 다른 Application Load Balancer가 로드 밸런서 VPC에 피어링된 VPC에 있는 경우 해당 IP 주소를 등록할 수 있습니다.

IPv6 대상

- 사용자가 등록하는 IP 주소는 VPC CIDR 블록 내에 있거나 피어링된 VPC CIDR 블록 내에 있어야 합니다.

콘솔을 사용하여 IP 주소로 대상을 등록 또는 등록 취소하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 대상 탭을 선택합니다.
5. IP 주소를 등록하려면 대상 등록을 선택합니다. 각 IP 주소에 대해 네트워크를 선택하고 IP 주소 및 포트를 입력한 다음 아래에 보류 중인 것으로 포함을 선택합니다.
6. 선택 사항: IP 주소가 선택한 VPC 외부에 있는 경우 가용 영역을 지정해야 합니다.
7. 주소 지정을 마치면 보류 중인 대상 등록(Register pending targets)을 선택합니다.
8. IP 주소의 등록을 취소하려면 IP 주소를 선택한 다음 등록 취소를 선택합니다. 등록 취소된 IP 주소가 많은 경우 필터를 추가하거나 정렬 순서를 변경하는 것이 유용할 수 있습니다.

Lambda 함수 등록 또는 등록 취소

각 대상 그룹에 단일 Lambda 함수를 등록할 수 있습니다. Elastic Load Balancing에 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 실패합니다. Lambda 함수를 바꾸려면 그 대신 새 대상 그룹을 생성하는 것이 더 좋습니다. 자세한 내용은 Lambda 함수를 Application Load Balancer의 대상으로 사용 단원을 참조하십시오.

콘솔을 사용하여 Lambda 함수를 등록 또는 등록 취소하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.

3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 대상 탭을 선택합니다.
5. 등록된 Lambda 함수가 없는 경우 Register(등록)를 선택합니다. Lambda 함수를 선택한 다음 Register(등록)를 선택합니다.
6. Lambda 함수의 등록을 취소하려면 등록 취소를 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.

Lambda 함수를 Application Load Balancer의 대상으로 사용

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.

Limits

- Lambda 함수와 대상 그룹은 동일한 계정 및 동일한 리전에 있어야 합니다.
- Lambda 함수에 전송할 수 있는 요청 본문의 최대 크기는 1MB입니다. 관련 크기 제한은 HTTP 헤더 제한을 참조하세요.
- Lambda 함수가 전송할 수 있는 응답 JSON의 최대 크기는 1MB입니다.
- WebSocket은 지원되지 않습니다. 업그레이드 요청은 HTTP 400 코드와 함께 거부됩니다.
- 로컬 영역은 지원되지 않습니다.
- 자동 대상 가중치(ATW)는 지원되지 않습니다.

내용

- Lambda 함수 준비
- Lambda 함수에 대한 대상 그룹 생성
- 로드 밸런서에서 이벤트 수신
- 로드 밸런서에 응답
- 다중 값 헤더

- 상태 확인 활성화
- Lambda 함수 등록 취소

데모는 Application Load Balancer의 Lambda 대상을 참조하세요.

Lambda 함수 준비

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

Lambda 함수를 호출할 권한

AWS Management Console을 사용하여 대상 그룹을 생성하고 Lambda 함수를 등록하면 콘솔은 사용자를 대신하여 필수 권한을 Lambda 함수 정책에 추가합니다. 그렇지 않으면 대상 그룹을 생성하고를 사용하여 함수를 등록한 후 add-permission 명령을 사용하여 Elastic Load Balancing에 Lambda 함수를 호출할 수 있는 권한을 부여 AWS CLI해야 합니다. `aws:SourceAccount` 및 `aws:SourceArn` 조건 키를 사용하여 함수 호출을 지정된 대상 그룹으로 제한하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 혼동된 대리자 문제를 참조하세요.

```
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 \
--source-account target-group-account-id
```

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 개발자 안내서의 AWS Lambda 함수 별칭을 참조하세요.

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 변경 방법에 대한 자세한 내용은 Lambda 함수 제한 시간 구성을 참조하세요. 구성할 수 있는 최대 제한 시간 값에 대한 자세한 내용은 AWS Lambda 할당량을 참조하세요.

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.

콘솔을 사용하여 대상 그룹을 생성하고 Lambda 함수를 등록하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 로드 밸런싱(Load Balancing) 아래에서 대상 그룹(Target Groups)을 선택합니다.
3. [대상 그룹 생성(Create target group)]을 선택합니다.
4. 대상 유형에서 Lambda 함수를 선택합니다.
5. [대상 그룹 이름(Target group name)]에 대상 그룹의 이름을 입력합니다.
6. (선택 사항) 상태 확인을 활성화하려면 상태 확인(Health checks) 섹션에서 활성화(Enable)를 선택합니다.
7. (선택 사항) 다음과 같이 하나 이상의 태그를 추가합니다.
   a. 태그 섹션을 확장합니다.
   b. [태그 추가(Add tag)]를 선택합니다.
   c. 태그 키와 태그 값을 입력합니다.
8. 다음을 선택합니다.
9. 단일 Lambda 함수를 지정하거나 이 단계를 생략하고 나중에 Lambda 함수를 지정합니다.
10. 대상 그룹 생성을 선택합니다.

AWS CLI를 사용하여 대상 그룹을 생성하고 Lambda 함수를 등록하는 방법

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.

`content-encoding` 헤더가 있으면 로드 밸런서는 본문을 Base64로 인코딩하고 `isBase64Encoded`를 `true`로 설정합니다.

content-encoding 헤더가 없으면 Base64 인코딩은 콘텐츠 유형에 따라 다릅니다. 콘텐츠 유형이 text/*, application/json, application/javascript 및 application/xml인 경우 로드 밸런서는 본문을 그대로 전송하고 isBase64Encoded를 false로 설정합니다. 그러지 않으면 로드 밸런서는 본문을 Base64로 인코딩하고 isBase64Encoded를 true로 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 및 헤더가 포함됩니다. 본문을 생략할 수 있습니다.

응답의 본문에 바이너리 콘텐츠를 포함시키려면 콘텐츠를 Base64로 인코딩하고 isBase64Encoded를 true를 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.

다음 사항은 nodejs 기반 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 참조하세요. 아니면 Lambda 콘솔을 열고 애플리케이션, 애플리케이션 생성을 선택한 다음 에서 AWS Serverless Application Repository 중 하나를 선택합니다.

- 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 포함되거나, 동일한 키에 대한 값을 여러 개 가진 쿼리 파라미터가 포함되는 경우 다중 값 헤더 구문에 대한 지원을 활성화할 수 있습니다. 다중 값 헤더를 활성화한 후에는 로드 밸런서와 Lambda 함수 간에 교환되는 헤더 및 쿼리 파라미터에서 문자열 대신 배열이 사용됩니다. 다중 값 헤더 구문을 사용하지 않고 헤더 또는 쿼리 파라미터에 값이 여러 개인 경우 로드 밸런서는 마지막으로 수신된 값을 사용합니다.

목차

- 다중 값 헤더가 있는 요청
- 다중 값 헤더가 있는 응답
- 다중 값 헤더 활성화

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. 다음 예를 참조하십시오.

```
"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. 다음 예를 참조하십시오.

```
"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. 다음 예를 참조하십시오.

```
"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. 다음 예를 참조하십시오.

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

```
    ...
},
```

쿼리 파라미터가 URL 인코딩된 경우 로드 밸런서는 해당 파라미터를 디코딩하지 않습니다. 사용자가 람다 함수에서 직접 디코딩해야 합니다.

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 지정해야 합니다.

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

로드 밸런서는 Lambda 응답 페이로드에 지정된 순서와 다른 순서로 클라이언트에 헤더를 전송할 수도 있습니다. 따라서 헤더가 특정 순서로 반환될 것이라고 믿으면 안 됩니다.

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 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 그룹 세부 정보 탭의 속성 섹션에서 편집을 선택합니다.
5. 다중 값 헤더를 선택하거나 선택 취소합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 다중 값 헤더를 활성화하려면 AWS CLI

`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
}
```

콘솔을 사용하여 대상 그룹에 대한 상태 확인을 활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 그룹 세부 정보 탭의 상태 확인 설정 섹션에서 편집을 선택합니다.
5. 상태 확인에서 활성화를 선택합니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 대상 그룹에 대한 상태 확인을 활성화하려면 AWS CLI

`--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.

콘솔을 사용하여 Lambda 함수의 등록을 취소하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 대상 탭에서 등록 취소를 선택합니다.
5. 확인 메시지가 나타나면 [Deregister]를 선택합니다.

를 사용하여 Lambda 함수의 등록을 취소하려면 AWS CLI

deregister-targets 명령을 사용합니다.

Application Load Balancer 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 + - = . _ : / @입니다. 선행 또는 후행 공백을 사용하면 안 됩니다.
- 태그 이름 또는 값은 사용하기 위해 예약되어 있으므로 `aws:` 접두사를 AWS 사용하지 마십시오. 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 업데이트하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹의 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 태그(Tags) 탭에서 태그 관리(Manage tags)를 선택하고 다음 중 하나 이상의 작업을 수행합니다.
   a. 태그를 업데이트하려면 키 및 값에 새 값을 입력합니다.
   b. 태그를 추가하려면 태그 추가를 선택하고 키 및 값에 값을 입력합니다.
   c. 태그를 삭제하려면 태그 옆의 제거를 선택합니다.
5. 태그 업데이트를 마쳤으면 변경 사항 저장(Save changes)을 선택합니다.

를 사용하여 대상 그룹의 태그를 업데이트하려면 AWS CLI

add-tags 및 remove-tags 명령을 사용합니다.

Application Load Balancer 대상 그룹 삭제

리스너 규칙의 전달 작업에서 참조하지 않는 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 삭제하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 Load Balancing 아래에서 대상 그룹을 선택합니다.
3. 대상 그룹을 선택하고 작업, 삭제를 차례로 선택합니다.
4. 확인 메시지가 나타나면 예, 삭제합니다를 선택합니다.

를 사용하여 대상 그룹을 삭제하려면 AWS CLI

delete-target-group 명령을 사용합니다.

Application Load Balancer 모니터링

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

CloudWatch 지표

Amazon CloudWatch를 사용하여 로드 밸런서 및 대상에 대한 데이터 포인트에 대한 통계를 정렬된 시계열 데이터 집합으로 검색할 수 있습니다(지표라고 함). 이러한 지표를 사용하여 시스템이 예상대로 수행되고 있는지 확인할 수 있습니다. 자세한 내용은 Application Load Balancer의 CloudWatch 지표 단원을 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 Application Load Balancer에 대한 액세스 로그 단원을 참조하십시오.

연결 로그

연결 로그를 사용하여 로드 밸런서에 보낸 요청의 속성을 캡처하고 Amazon S3에 로그 파일로 저장할 수 있습니다. 이러한 연결 로그를 사용하여 사용할 클라이언트 IP 주소 및 포트, 클라이언트 인증서 정보, 연결 결과 및 TLS 암호를 확인할 수 있습니다. 이러한 연결 로그를 사용하여 요청 패턴 및 기타 추세를 검토할 수 있습니다. 자세한 내용은 Application Load Balancer의 연결 로그 단원을 참조하십시오.

요청 추적

요청 추적을 사용하여 HTTP 요청을 추적할 수 있습니다. 로드 밸런서는 수신한 각 요청에 트레이스 식별자가 있는 헤더를 추가합니다. 자세한 내용은 Application Load Balancer에 대한 요청 추적 단원을 참조하십시오.

CloudTrail 로그

AWS CloudTrail 를 사용하여 Elastic Load Balancing API에 대한 호출에 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 CloudTrail을 사용하여 Elastic Load Balancing에 대한 API 직접 호출 로깅을 참조하세요.

Application Load Balancer의 CloudWatch 지표

Elastic Load Balancing은 로드 밸런서와 대상을 위해 Amazon CloudWatch에 데이터 포인트를 게시합니다. CloudWatch를 사용하면 이러한 데이터 포인트에 대한 통계를 정렬된 시계열 데이터 세트로 검색할 수 있습니다. 이러한 통계를 지표라고 합니다. 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 Balancing은 요청이 로드 밸런서를 통과하는 경우에만 지표를 CloudWatch에 보고합니다. 로드 밸런서를 통과하는 요청이 있는 경우 Elastic Load Balancing은 60초 간격으로 지표를 측정하고 전송합니다. 로드 밸런서를 통과하고 있는 요청이 없는 경우나 지표에 대한 데이터가 없는 경우에는 지표가 보고되지 않습니다.

자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하십시오.

목차

- Application Load Balancer 지표
- Application Load Balancer의 지표 차원
- Application Load Balancer 지표에 대한 통계
- 로드 밸런서에 대한 CloudWatch 지표 보기

Application Load Balancer 지표

- 로드 밸런서
- 대상
- 대상 그룹 상태
- Lambda 함수
- 사용자 인증

AWS/ApplicationELB 네임스페이스에는 다음 로드 밸런서 지표가 포함되어 있습니다.

측정치	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 • AvailabilityZone , LoadBalancer
AnomalousHostCount	이상이 탐지된 호스트 수입니다. 보고 기준: 항상 보고 통계: 가장 유용한 통계는 Average, Minimum 및 Maximum입니다. Dimensions • TargetGroup , LoadBalancer • TargetGroup , AvailabilityZone , LoadBalancer
BYoIPUtilPercentage	IP 풀의 사용량 비율입니다. 보고 기준: 로드 밸런서에서 BYoIP가 활성화됩니다. 통계: 유일하게 의미 있는 통계는 Average입니다. Dimensions • LoadBalancer , TargetGroup • LoadBalancer , TargetGroup , AvailabilityZone
ClientTLSNegotiationErrorCount	TLS 오류로 인해 로드 밸런서와 세션을 구성하지 않은 클라이언트에서 시작된 TLS 연결 수. 가능한 원인으로는 암호 또는 프로토콜 불일치나 클라이언트의 서버 인증서 확인 실패 및 연결 종료가 있습니다. 보고 기준: 0이 아닌 값이 있을 때

측정치	설명
	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. LCU 예약이 활성화되면 ConsumedLCUs는 사용량이 예약 용량 미만인 0 경우를 보고하고 사용량이 예약 LCUs를 초과하는 0 경우 위의 값을 보고합니다. 자세한 내용은 Elastic Load Balancing 요금을 참조하세요. 보고 기준: 항상 보고 통계: 모두 Dimensions • LoadBalancer
PeakLCUs	지정된 시점에 로드 밸런서가 사용하는 로드 밸런서 용량 단위(LCU)의 최대 수입니다. LCU 예약을 사용하는 경우에만 적용됩니다. 보고 기준: 항상 통계: 가장 유용한 통계는 Sum 및 Max입니다. Dimensions • LoadBalancer

측정치	설명
ReservedLCUs	분당 예약 용량을 보고하는 결제 지표입니다. 일정 기간 동안의 총 ReservedLCUs는 청구되는 LCUs의 양입니다. 예를 들어 500LCUs 한 시간 동안 예약된 경우 분당 지표는 8.33LCUs. 자세한 내용은 예약 모니터링 단원을 참조하십시오. 보고 기준: 0이 아닌 값이 있을 때 통계: 모두 Dimensions • LoadBalancer
DesyncMitigationMode_NonCompliant_Request_Count	RFC 7230을 준수하지 않는 요청 수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer
DroppedInvalidHeaderRequestCount	로드 밸런서가 요청을 라우팅하기 전에 유효하지 않은 헤더 필드가 있는 HTTP 헤더를 제거한 요청 수입니다. 로드 밸런서는 routing.http.drop_invalid_header_fields.enabled 속성이 true로 설정된 경우에만 이러한 헤더를 제거합니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 모두 Dimensions • AvailabilityZone , LoadBalancer

측정치	설명
MitigatedHostCount	완화가 진행 중인 대상 수입니다. 보고 기준: 항상 보고 통계: 가장 유용한 통계는 Average, Minimum 및 Maximum입니다. Dimensions • TargetGroup , LoadBalancer • TargetGroup , AvailabilityZone , LoadBalancer
ForwardedInvalidHeaderRequestCount	유효하지 않은 헤더 필드가 있는 HTTP 헤더가 포함된 로드 밸런서가 라우팅한 요청 수입니다. 로드 밸런서는 routing.http.drop_invalid_header_fields.enabled 속성이 false로 설정된 경우에만 이러한 헤더와 함께 요청을 전달합니다. 보고 기준: 항상 보고 통계: 모두 Dimensions • AvailabilityZone , LoadBalancer
GrpcRequestCount	IPv4 및 IPv6를 통해 처리된 gRPC 요청 수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Minimum, Maximum 및 Average은(는) 모두 1을 반환합니다. Dimensions • LoadBalancer , TargetGroup • AvailabilityZone , LoadBalancer , TargetGroup • TargetGroup • AvailabilityZone , TargetGroup

측정치	설명
HTTP_Fixed_Response_Count	성공한 고정 응답 작업의 수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer
HTTP_Redirect_Count	성공한 리디렉션 작업의 수입니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer
HTTP_Redirect_Url_Limit_Exceeded_Count	응답 위치 헤더의 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 • AvailabilityZone , LoadBalancer

측정치	설명
HTTPCode_ELB_3XX_Count	로드 밸런서에서 생성되는 HTTP 3XX 리디렉션 코드의 수입니다. 단, 대상에서 생성된 응답 코드 수는 여기에 포함되지 않습니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 유일하게 의미 있는 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer
HTTPCode_ELB_4XX_Count	로드 밸런서에서 생성된 HTTP 4XX 클라이언트 오류 코드 수. 단, 대상에서 생성된 응답 코드 수는 여기에 포함되지 않습니다. 클라이언트 오류는 요청 형식이 잘못되었거나 불완전할 때 생성됩니다. 로드 밸런서가 HTTP 460 오류 코드를 반환하는 경우를 제외하고 대상에서는 이러한 요청이 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 Average은(는) 모두 1을 반환합니다. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer

측정치	설명
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 Average은(는) 모두 1을 반환합니다. Dimensions • LoadBalancer • AavailabilityZone , 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. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer
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. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer

측정치	설명
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. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer
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. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer
IPv6ProcessedBytes	로드 밸런서에서 IPv6를 통해 처리된 총 바이트 수. 이 수는 ProcessedBytes 에 포함됩니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer

측정치	설명
IPv6RequestCount	로드 밸런서가 수신한 IPv6 요청 수. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Minimum, Maximum 및 Average은(는) 모두 1을 반환합니다. Dimensions • LoadBalancer • 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 • AvailabilityZone , LoadBalancer

측정치	설명
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. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer
ProcessedBytes	로드 밸런서에서 IPv4 및 IPv6를 통해 처리된 총 바이트 수(HTTP 헤더 및 HTTP 페이로드)입니다. 이 수에는 클라이언트와 Lambda 함수로 송수신하는 트래픽과 IdP(ID 공급자)가 보내는 트래픽(사용자 인증이 활성화된 경우)이 포함됩니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer

측정치	설명
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. 이 지표는 로드 밸런서 노드가 대상을 선택할 수 있었던 요청에 대해서만 증가합니다. 대상이 선택되기 전에 거부된 요청은 이 지표에 반영되지 않습니다. 보고 기준: 등록된 대상이 있는 경우 보고됩니다. 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • LoadBalancer , AvailabilityZone • LoadBalancer , TargetGroup • LoadBalancer , AvailabilityZone , TargetGroup
RuleEvaluations	요청을 처리하는 동안 로드 밸런서가 평가한 규칙 수입니다. 기본 규칙은 계수되지 않습니다. 요청당 10개의 무료 규칙 평가가 이 수에 포함됩니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer

측정치	설명
ZonalShiftedHostCount	영역 전환으로 인해 비활성화된 것으로 간주되는 대상 수입니다. 보고 기준: 값이 있을 때 보고됨 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer , TargetGroup . • AvailabilityZone , LoadBalancer , TargetGroup .

AWS/ApplicationELB 네임스페이스에는 다음 대상 지표가 포함되어 있습니다.

측정치	설명
HealthyHostCount	정상 상태로 간주되는 대상 수 보고 기준: 등록된 대상이 있는 경우 보고됩니다. 통계: 가장 유용한 통계는 Average, Minimum 및 Maximum입니다. Dimensions • LoadBalancer , TargetGroup • LoadBalancer , AvailabilityZone , TargetGroup
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 코드 수는 여기에 포함되지 않습니다. 보고 기준: 등록된 대상이 있는 경우 보고됩니다. 통계: 가장 유용한 통계는 Sum입니다. Minimum, Maximum 및 Average은(는) 모두 1을 반환합니다. Dimensions • LoadBalancer

측정치	설명
	• AvailabilityZone , LoadBalancer • TargetGroup , LoadBalancer • TargetGroup , AvailabilityZone , LoadBalancer
RequestCountPerTarget	대상 그룹의 대상당 평균 요청 수입니다. 대상 그룹은 TargetGroup 차원을 사용하여 지정해야 합니다. 대상이 Lambda 함수인 경우 이 지표는 적용되지 않습니다. 이 수는 대상 그룹이 수신한 총 요청 수를 대상 그룹의 정상 대상 수로 나눈 값을 사용합니다. 대상 그룹에 정상 대상이 없는 경우 등록된 총 대상 수로 나눕니다. 보고 기준: 항상 보고 통계: 유일하게 유효한 통계는 Sum입니다. 이는 합계가 아니라 평균을 의미합니다. Dimensions • TargetGroup • TargetGroup , AvailabilityZone • LoadBalancer , TargetGroup • LoadBalancer , AvailabilityZone , TargetGroup

측정치	설명
TargetConnectionErrorCount	로드 밸런서와 대상 사이에 성공적으로 구성되지 않은 연결 수 대상이 Lambda 함수인 경우 이 지표는 적용되지 않습니다. 실패한 상태 확인 연결에는이 지표가 증가하지 않습니다. 보고 기준: 0이 아닌 값이 있을 때 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer • TargetGroup , LoadBalancer • TargetGroup , AvailabilityZone , LoadBalancer
TargetResponseTime	로드 밸런서가 요청을 전송한 후 대상이 응답 헤더 전송을 시작할 때까지 경과한 시간(초)입니다. 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

측정치	설명
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. 대상 등록을 취소하면이 감소HealthyHostCount 하지만는 증가하지 않습니다UnhealthyHostCount . 보고 기준: 등록된 대상이 있는 경우 보고됩니다. 통계: 가장 유용한 통계는 Average, Minimum 및 Maximum입니다. Dimensions • LoadBalancer , TargetGroup • LoadBalancer , AvailabilityZone , TargetGroup

AWS/ApplicationELB 네임스페이스에는 다음 대상 그룹 상태에 대한 지표가 포함되어 있습니다. 자세한 내용은 the section called “대상 그룹 상태” 단원을 참조하십시오.

측정치	설명
HealthyStateDNS	DNS 정상 상태 요구 사항을 충족하는 영역 수.

측정치	설명
	통계: 가장 유용한 통계는 Max입니다. Dimensions • LoadBalancer , TargetGroup • AvailabilityZone , LoadBalancer , TargetGroup
HealthyStateRouting	라우팅 정상 상태 요구 사항을 충족하는 영역 수. 통계: 가장 유용한 통계는 Max입니다. Dimensions • LoadBalancer , TargetGroup • AvailabilityZone , LoadBalancer , TargetGroup
UnhealthyRoutingRequestCount	라우팅 장애 조치 작업(페일 오픈)을 사용하여 라우팅된 요청 수. 통계: 가장 유용한 통계는 Sum입니다. Dimensions • LoadBalancer , TargetGroup • AvailabilityZone , LoadBalancer , TargetGroup
UnhealthyStateDNS	DNS 정상 상태 요구 사항을 충족하지 않아 DNS에서 비정상으로 표시된 영역 수. 통계: 가장 유용한 통계는 Min입니다. Dimensions • LoadBalancer , TargetGroup • AvailabilityZone , LoadBalancer , TargetGroup

측정치	설명
UnhealthyStateRouting	라우팅 정상 상태 요구 사항을 충족하지 않아 로드 밸런서가 비정상 대상을 포함한 영역 내 모든 대상으로 트래픽을 분산하는 영역 수. 통계: 가장 유용한 통계는 Min입니다. Dimensions • LoadBalancer , TargetGroup • AvailabilityZone , LoadBalancer , TargetGroup

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

측정치	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. Dimensions • TargetGroup • TargetGroup , LoadBalancer
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.

측정치	설명
	Dimensions • 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. Dimensions • TargetGroup • TargetGroup , LoadBalancer

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

측정치	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. Dimensions • LoadBalancer

측정치	설명
	• AvailabilityZone , 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. Dimensions • LoadBalancer • AvailabilityZone , 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. Dimensions • LoadBalancer • AvailabilityZone , 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. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer

측정치	설명
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. Dimensions • LoadBalancer • AvailabilityZone , 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. Dimensions • LoadBalancer • AvailabilityZone , LoadBalancer

Application Load Balancer의 지표 차원

Application Load Balancer 지표를 필터링하려면 다음 차원을 사용하세요.

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)

차원	설명
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 Balancing에서 게시한 지표 데이터 포인트를 기반으로 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 각 샘플링 창에서 개별 로드 밸런서 노드가 보고한 최소 및 최대 데이터 포인트 값을 반영합니다. 예를 들어 Application Load Balancer를 구성하는 로드 밸런서 노드가 2개라고 가정해 보겠습니다. 하나의 노드에는 HealthyHostCount이 2, Minimum이 10, Maximum가 6인 Average가 있으며 다른 노드에는 HealthyHostCount이 1, Minimum이 5, Maximum가 3인 Average가 있습니다. 따라서 로드 밸런서의 Minimum은 1, Maximum은 10, Average는 4입니다.

UnHealthyHostCountMinimum 통계에서 0이 아닌 값을 모니터링하고 둘 이상의 데이터 요소에 대해 0이 아닌 값에 대해 경보를 표시하는 것이 좋습니다. Minimum을 사용하면 로드 밸런서의 모든 노드와 가용 영역에서 대상이 비정상으로 간주되는 시점을 감지할 수 있습니다. Average 또는 Maximum 알람은 잠재적인 문제에 대한 알림을 받고 싶을 때 유용하며, 고객은 이 지표를 검토하여 0이 아닌 발생 항목을 조사하는 것이 좋습니다. Amazon EC2 Auto Scaling 또는 Amazon Elastic Container Service(Amazon ECS)에서 로드 밸런서 상태 확인을 사용하는 모범 사례에 따라 장애를 자동으로 완화할 수 있습니다.

Sum 통계는 모든 로드 밸런서 노드의 집계 값입니다. 지표에는 기간별 보고서가 여러 개 있기 때문에 Sum은 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 HealthyHostCount에 대해 SampleCount는 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 사용하여 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 사용하여 로드 밸런서에 대한 지표를 볼 수 있습니다.

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 사용하여 지표를 보려면

1. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 [지표(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 가용 영역에 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 가용 영역에 대한 지표를 보려면 검색 필드에 해당되는 이름을 입력합니다.

를 사용하여 지표를 보려면 AWS CLI

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

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

를 사용하여 지표에 대한 통계를 가져오려면 AWS CLI

지정된 지표 및 차원에 대한 통계를 구하려면 아래 get-metric-statistics 명령을 사용하세요. 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"
}
```

Application Load Balancer에 대한 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 로그는Elastic Load Balancing의 옵션 기능으로, 기본적으로 비활성화되어 있습니다. 로드 밸런서에 대해 액세스 로그를 활성화하면 Elastic Load Balancing이 로그를 캡처하여 압축 파일을 지정한 Amazon S3 버킷에 저장합니다. 액세스 로그는 언제든지 비활성화할 수 있습니다.

Amazon S3의 스토리지 비용은 청구되지만, Amazon S3로 로그 파일을 전송하기 위해 Elastic Load Balancing에서 사용하는 대역폭에 대해서는 요금이 부과되지 않습니다. 스토리지 비용에 대한 자세한 내용은 Amazon S3 요금을 참조하세요.

목차

- 액세스 로그 파일
- 액세스 로그 항목
- 로그 항목 예제
- 액세스 로그 파일 처리
- Application Load Balancer 액세스 로그 활성화
- Application Load Balancer 액세스 로그 비활성화

액세스 로그 파일

Elastic Load Balancing은 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_app.load-balancer-id_end-time_ip-address_random-string.log.gz
```

bucket

S3 버킷의 이름.

접두사

(선택 사항) 버킷의 접두사(논리적 계층 구조)입니다. 지정하는 접두사에는 문자열 `AWSLogs`가 포함되지 않아야 합니다. 자세한 내용은 접두사를 사용한 객체 구성을 참조하세요.

`AWSLogs`

`AWSLogs`로 시작하는 파일 이름의 일부가 지정하는 버킷 이름과 선택적 접두사 뒤에 추가됩니다.

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 UTC 또는 Zulu 시간으로 23:35과 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 접두사가 있는 로그 파일 이름의 예입니다.

```
s3://amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/
elasticloadbalancing/us-east-2/2022/05/01/123456789012_elasticloadbalancing_us-
east-2_app.my-
loadbalancer.1234567890abcdef_20220215T2340Z_172.160.001.192_20sg8hgm.log.gz
```

다음은 접두사가 없는 로그 파일 이름의 예입니다.

```
s3://amzn-s3-demo-logging-bucket/AWSLogs/123456789012/elasticloadbalancing/
us-east-2/2022/05/01/123456789012_elasticloadbalancing_us-east-2_app.my-
loadbalancer.1234567890abcdef_20220215T2340Z_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 S3 사용 설명서의 객체 수명 주기 관리를 참조하세요.

액세스 로그 항목

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

내용

- 구문
- 수행된 작업
- 분류 이유
- 오류 이유 코드

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 • `https` - TLS를 통한 HTTP • `h2` - TLS를 통한 HTTP/2 • `grpcs` - TLS를 통한 gRPC • `ws` — WebSockets • `wss` - TLS를 통한 WebSockets
시간	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. 로드 밸런서 앞에 프록시가 있는 경우 이 필드에는 프록시의 IP 주소가 포함됩니다.
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.

필드	설명
	대상이 Lambda 함수인 경우 이 값은 -로 설정됩니다. 요청이에 의해 차단되면 AWS WAF이 값은 -로 설정됩니다.
request_processing_time	로드 밸런서가 요청을 수신한 시간부터 대상으로 요청을 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. 10초 TCP 연결 제한 시간에 도달할 때까지 대상과 TCP 연결을 설정할 수 없는 경우에도 이 값을 -1로 설정할 수 있습니다. Application Load Balancer에 AWS WAF 가 활성화되어 있거나 대상 유형이 Lambda 함수인 경우 클라이언트가 POST 요청에 필요한 데이터를 전송하는 데 걸리는 시간은 로 계산됩니다`request_processing_time` .
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. Application Load Balancer에 AWS WAF 가 활성화되지 않은 경우 클라이언트가 POST 요청에 필요한 데이터를 전송하는 데 걸리는 시간은 로 계산됩니다`target_processing_time` .

필드	설명
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 대상으로부터 응답을 받지 못하면 이 값은 -1로 설정됩니다. 대상이 유휴 제한 시간 전에 연결을 종료하거나 클라이언트가 잘못된 요청을 보내는 경우에 이런 상황이 발생할 수 있습니다.
elb_status_code	블록 작업에 대한 로드 밸런서, 고정 응답 규칙 또는 AWS WAF 사용자 지정 응답 코드에서 생성된 응답의 상태 코드입니다.
target_status_code	대상의 응답 상태 코드입니다. 대상으로 연결이 설정되고 대상이 응답을 전송한 경우에만 이 값이 기록됩니다. 그러지 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.
sent_bytes	클라이언트(요청자)에게 보낸 응답의 크기(바이트)입니다. HTTP 요청의 경우 여기에는 응답 헤더와 본문이 포함됩니다. WebSockets의 경우에는 연결 시 클라이언트에 전송된 총 바이트 수입니다. TCP 헤더 및 TLS 핸드셰이크 페이로드는 계산되지 않으며 `DataTransfer-Out-Bytes` 의와 상관 관계가 없습니다 AWS Cost Explorer.
"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.

필드	설명
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. 리스너가 HTTPS 리스너가 아닌 경우 이 값은 -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 수행된 작업에서 설명하는 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 코드에서 설명하는 오류 코드 중 하나입니다. 수행한 작업에 인증 작업이 포함되지 않거나 대상이 Lambda 함수가 아닌 경우, 이 값은 -로 설정됩니다.

필드	설명
"target:port_list"	이 요청을 처리한 대상에 대한 IP 주소 및 포트를 공백으로 구분한 목록이며 큰따옴표로 묶여 있습니다. 현재 이 목록에는 하나의 항목이 포함될 수 있으며 target:port 필드와 일치합니다. 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. 요청이에 의해 차단되면 AWS WAF이 값은 -로 설정됩니다.
"target_status_code_list"	대상의 응답에서 공백으로 구분된 상태 코드 목록이며 큰따옴표로 묶여 있습니다. 현재 이 목록에는 하나의 항목이 포함될 수 있으며 target_status_code 필드와 일치합니다. 대상으로 연결이 설정되고 대상이 응답을 전송한 경우에만 이 값이 기록됩니다. 그러지 않으면 -에 설정됩니다.
"classification"	동기화 해제 완화에 대한 분류로 큰따옴표로 묶여 있습니다. 요청이 RFC 7230을 준수하지 않는 경우 가능한 값은 허용 가능, 모호 및 심각입니다. 요청이 RFC 7230을 준수하는 경우 이 값은 -로 설정됩니다.
"classification_reason"	큰 따옴표로 묶인 분류 사유 코드입니다. 요청이 RFC 7230을 준수하지 않는 경우 이 코드는 분류 이유에서 설명하는 분류 코드 중 하나입니다. 요청이 RFC 7230을 준수하는 경우 이 값은 -로 설정됩니다.
conn_trace_id	연결 추적 가능성 ID는 각 연결을 식별하는 데 사용되는 고유한 불투명 ID입니다. 클라이언트와의 연결이 설정되면 이 클라이언트의 후속 요청에서 이 ID가 해당 액세스 로그 항목에 포함됩니다. 이 ID는 외부 키 역할을 하여 연결 로그와 액세스 로그 간의 링크를 생성합니다.

수행된 작업

로드 밸런서는 수행하는 작업을 액세스 로그의 actions_ecuted 필드에 저장합니다.

- `authenticate` - 로드 밸런서는 규칙 구성에 지정된 대로 세션 유효성을 검사하고, 사용자를 인증한 다음 요청 헤더에 사용자 정보를 추가했습니다 .
- `fixed-response` - 로드 밸런서가 규칙 구성에 지정된 대로 고정 응답을 실행했습니다 .
- `forward` - 로드 밸런서는 규칙 구성에 지정된 대로 대상에 요청을 전달했습니다.
- `redirect` - 로드 밸런서는 규칙 구성에 지정된 대로 요청을 다른 URL로 리디렉션했습니다.
- `waf` - 로드 밸런서는 요청을 대상으로 전달해야 하는지 여부를 결정하기 위해 AWS WAF 로 요청을 전달했습니다. 이것이 최종 작업인 경우 요청을 거부하도록 AWS WAF 결정했습니다. 기본적으로에서 거부한 요청은 `elb_status_code` 필드에 "403"으로 기록 AWS WAF 됩니다. AWS WAF 가 사용자 지정 응답 코드로 요청을 거부하도록 구성된 경우 `elb_status_code` 필드에 구성된 응답 코드가 반영됩니다.
- `waf-failed` - 로드 밸런서가 요청을 전달하려고 AWS WAF했지만이 프로세스가 실패했습니다.

분류 이유

요청이 RFC 7230을 준수하지 않는 경우 로드 밸런서는 액세스 로그의 classification_reason 필드에 다음 코드 중 하나를 저장합니다. 자세한 내용은 Desync Mitigation Mode 단원을 참조하십시오.

코드	설명	Classification
`AmbiguousUri`	요청 URI에 제어 문자가 포함되어 있습니다.	모호
`BadContentLength`	Content-Length 헤더에 구문 분석할 수 없거나 유효한 숫자가 아닌 값이 포함되어 있습니다.	심각
`BadHeader`	헤더에 null 문자 또는 캐리지 리턴이 포함되어 있습니다.	심각
`BadTransferEncoding`	Transfer-Encoding 헤더에 잘못된 값이 포함되어 있습니다.	심각
`BadUri`	요청 URI에 null 문자 또는 캐리지 리턴이 포함되어 있습니다.	심각
`BadMethod`	요청 메서드의 형식이 잘못되었습니다.	심각
`BadVersion`	요청 버전의 형식이 잘못되었습니다.	심각

코드	설명	Classification
BothTeClPresent	요청에 Transfer-Encoding 헤더와 Content-Length 헤더가 모두 포함되어 있습니다.	모호
DuplicateContentLength	값이 동일한 Content-Length 헤더가 여러 개 있습니다.	모호
EmptyHeader	헤더가 비어 있거나 공백만 있는 줄이 있습니다.	모호
GetHeadZeroContentLength	GET 또는 HEAD 요청에 대한 값이 0인 Content-Length 헤더가 있습니다.	허용 가능
MultipleContentLength	값이 서로 다른 Content-Length 헤더가 여러 개 있습니다.	심각
MultipleTransferEncodingChunked	여러 Transfer-Encoding이 있습니다: chunked 헤더.	심각
NonCompliantHeader	헤더에 비 ASCII 또는 제어 문자가 포함되어 있습니다.	허용 가능
NonCompliantVersion	요청 버전에 잘못된 값이 포함되어 있습니다.	허용 가능
SpaceInUri	요청 URI에 URL이 인코딩되지 않은 공백이 포함되어 있습니다.	허용 가능
SuspiciousHeader	일반적인 텍스트 정규화 기술을 사용하여 Transfer-Encoding 또는 Content-Length로 정규화할 수 있는 헤더가 있습니다.	모호
SuspiciousTeClPresent	요청에는 Transfer-Encoding 헤더와 Content-Length 헤더가 모두 포함되며, 그 중 하나 이상이 의심스럽습니다.	심각

코드	설명	Classification
`UndefinedContentLengthSemantics`	GET 또는 HEAD 요청에 대해 정의된 Content-Length 헤더가 있습니다.	모호
`UndefinedTransferEncodingSemantics`	GET 또는 HEAD 요청에 대한 정의된 Transfer-Encoding 헤더가 있습니다.	모호

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 단원을 참조하십시오.

코드	설명	측정치
`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`

코드	설명	측정치
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 없거나 토큰 엔드포인트가 5초 이내에 응답하지 않습니다.	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 없거나 사용자 정보 엔드포인트가 5초 이내에 응답하지 않습니다.	ELBAuthError
AuthUserinfoResponseSizeExceeded	IdP에서 반환한 클레임의 크기가 11K 바이트를 초과했습니다.	ELBAuthUserClaimsSizeExceeded

가중 대상 그룹에 대한 요청이 실패하면 로드 밸런서는 다음 오류 코드 중 하나를 액세스 로그의 error_reason 필드에 저장합니다.

코드	설명
AWSALBTGCookieInvalid	가중 대상 그룹과 함께 사용되는 AWSALBTG 쿠키는 유효하지 않습니다. 예를 들어, 로드 밸런서는 쿠키 값이 URL로 인코딩될 때 이 오류를 반환합니다.
WeightedTargetGroupsUnhandledException	로드 밸런서에 처리할 수 없는 예외가 발생했습니다.

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 참조하세요.

코드	설명	측정치
LambdaAccessDenied	로드 밸런서는 Lambda 함수를 호출할 권한이 없습니다.	LambdaUserError
LambdaBadRequest	클라이언트 요청 헤더 또는 본문에 UTF-8 문자만 포함되어 있지 않아 Lambda 호출에 실패했습니다.	LambdaUserError
LambdaConnectionError	로드 밸런서에서 Lambda에 연결할 수 없습니다.	LambdaInternalError
LambdaConnectionTimeout	Lambda에 연결하려는 시도가 시간 초과되었습니다.	LambdaInternalError
LambdaEC2AccessDeniedException	Amazon EC2가 함수 초기화 중 Lambda에 대한 액세스를 거부했습니다.	LambdaUserError
LambdaEC2ThrottledException	Amazon EC2가 함수 초기화 중 Lambda를 제한했습니다.	LambdaUserError

코드	설명	측정치
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
LambdaInvalidSecurityGroupIDException	Lambda 함수의 구성에 지정된 보안 그룹 ID가 유효하지 않습니다.	LambdaUserError
LambdaInvalidSubnetIDException	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

코드	설명	측정치
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	Lambda에서 내부 오류가 발생했습니다.	LambdaInternalError
LambdaSubnetIPAddressLimitReachedException	하나 이상의 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

코드	설명	측정치
LambdaUnhandledException	로드 밸런서에 처리할 수 없는 예외가 발생했습니다.	LambdaInternalError
LambdaWebsocketNotSupported	WebSocket은 Lambda를 통해 지원되지 않습니다.	LambdaUserError

요청을 전달할 때 로드 밸런서에 오류가 발생하면 액세스 로그의 error_reason 필드에 다음 오류 코드 중 하나를 AWS WAF저장합니다.

코드	설명
WAFConnectionError	로드 밸런서는 연결할 수 없습니다 AWS WAF.
WAFConnectionTimeout	에 대한 연결이 AWS WAF 시간 초과되었습니다.
WAFResponseReadTimeout	AWS WAF 시간 초과에 대한 요청입니다.
WAFServiceError	AWS WAF 에서 5XX 오류를 반환했습니다.
WAFUnhandledException	로드 밸런서에 처리할 수 없는 예외가 발생했습니다.

로그 항목 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" "-" "-" "10.0.0.1:80" "200" "-" "-"
```

HTTPS 항목 예제

다음은 HTTPS 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" "-" "-" "10.0.0.1:80" "200" "-"
 "-" TID_123456
```

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/" "-" "10.0.0.66:9000"
 "200" "-" "-"
```

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" "-" "-" "10.0.1.192:8010" "101" "-" "-"
```

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" "-" "-" "10.0.0.171:8010" "101" "-" "-"
```

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" "-" "-" "-" "-"
```

액세스 로그 파일 처리

액세스 로그 파일은 압축이 됩니다. 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 Athena는 표준 SQL을 사용해 Amazon S3에 저장된 데이터를 간편하게 분석할 수 있는 대화식 쿼리 서비스입니다. 자세한 내용은 Amazon Athena 사용 설명서의 Application Load Balancer 로그 쿼리 방법을 참조하세요.
- Loggly
- Splunk
- Sumo Logic

Application Load Balancer 액세스 로그 활성화

로드 밸런서에 대한 액세스 로그를 활성화할 때는 로드 밸런서가 로그를 저장할 S3 버킷의 이름을 지정해야 합니다. 버킷에 액세스 로그를 쓰는 Elastic Load Balancing 권한을 부여하는 버킷 정책이 이 버킷에 있어야 합니다.

업무

- 1단계: S3 버킷 생성
- 2단계: S3 버킷에 정책 연결
- 3단계: 액세스 로그 구성
- 4단계: 버킷 권한 확인
- 문제 해결

1단계: S3 버킷 생성

액세스 로그를 활성화할 때는 반드시 액세스 로그에 대한 S3 버킷을 지정해야 합니다. 기존 버킷을 사용하거나 액세스 로그 전용 버킷을 생성할 수 있습니다. 버킷은 다음 요구 사항을 충족해야 합니다.

요구 사항

- 버킷은 로드 밸런서와 같은 리전에 있어야 합니다. 서로 다른 계정에서 버킷과 로드 밸런서를 소유할 수 있습니다.
- 지원되는 유일한 서버 측 암호화 옵션은 Amazon S3 관리형 키(SSE-S3)입니다. 자세한 내용을 Amazon S3 관리형 암호화 키(SSE-S3) 섹션을 참조하세요.

Amazon S3 콘솔을 사용하여 S3 버킷에 폴더를 생성하려면

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 버킷 만들기를 선택합니다.
3. [Create a bucket] 페이지에서 다음과 같이 실행합니다.

   a. [Bucket Name]에서 버킷 이름을 입력합니다. 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 S3 사용 설명서의 버킷 규제 및 제한을 참조하세요.
   b. AWS 리전의 경우 로드 밸런서를 생성한 리전을 선택합니다.
   c. 기본 암호화에서 Amazon S3 관리형 키(SSE-S3)를 선택합니다.
   d. 버킷 생성을 선택합니다.

2단계: S3 버킷에 정책 연결

버킷에 액세스 로그를 쓰는 Elastic Load Balancing 권한을 부여하는 버킷 정책이 S3 버킷에 있어야 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.

연결된 정책이 이미 있는 기존 버킷을 사용하는 Elastic Load Balancing 액세스 로그에 대한 문을 정책에 추가할 수 있습니다. 그렇게 하는 경우, 결과적인 액세스 권한 집합을 평가하여 해당 집합이 액세스 로그에 대한 버킷에 액세스해야 하는 사용자에게 적절한 권한인지 확인하는 것이 좋습니다.

사용 가능한 버킷 정책

사용할 버킷 정책은 AWS 리전 및 영역 유형에 따라 다릅니다.

이용 가능한 리전(2022년 8월 이후 기준)

이 정책은 지정된 로그 전송 서비스에 권한을 부여합니다. 다음 리전의 로드 밸런서에이 정책을 사용합니다.

- 아시아 태평양(하이데라바드)
- 아시아 태평양(말레이시아)
- 아시아 태평양(멜버른)
- 아시아 태평양(타이베이)
- 아시아 태평양(태국)
- 캐나다 서부(캘거리)

- 유럽(스페인)
- 유럽(취리히)
- 이스라엘(텔아비브)
- 중동(UAE)
- 멕시코(중부)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "logdelivery.elasticloadbalancing.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix/AWSLogs/123456789012/*"
    }
  ]
}
```

에 예제 정책에 표시된 형식을 사용하여 액세스 로그 위치의 ARN을 `Resource`입력합니다. 항상 로드 밸런서가 있는 계정의 계정 ID를 S3 버킷 ARN의 리소스 경로에 포함합니다. 이렇게 하면 지정된 계정의 로드 밸런서만 S3 버킷에 액세스 로그를 쓸 수 있습니다.

지정하는 ARN은 3단계에서 액세스 로그를 활성화할 때 접두사를 포함할 계획인지 여부에 따라 다릅니다.

접두사가 있는 S3 버킷 ARN 예제

S3 버킷 이름은 amzn-s3-demo-logging-bucket이고 접두사는 입니다logging-prefix.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/*
```

접두사가 없는 S3 버킷 ARN 예제

S3 버킷 이름은 입니다amzn-s3-demo-logging-bucket. S3 버킷 ARN에는 접두사 부분이 없습니다.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/AWSLogs/123456789012/*
```

이용 가능한 리전(2022년 8월 이전 기준)

이 정책은 지정된 Elastic Load Balancing 계정 ID에 권한을 부여합니다. 아래 나열된 리전의 로드 밸런서에이 정책을 사용합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::elb-account-id:root"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix/AWSLogs/123456789012/*"
    }
  ]
}
```

의 경우 *elb-account-id*를 로드 밸런서의 리전에 대한 Elastic Load Balancing 계정 ID로 `Principal`바꿉니다.

- 미국 동부(버지니아 북부) – 127311923021
- 미국 동부(오하이오) – 033677994240
- 미국 서부(캘리포니아 북부) – 027434742980
- 미국 서부(오레곤) – 797873946194
- 아프리카(케이프타운) – 098369216593
- 아시아 태평양(홍콩) – 754344448648
- 아시아 태평양(자카르타) – 589379963580
- 아시아 태평양(뭄바이) – 718504428378
- 아시아 태평양(오사카) – 383597477331
- 아시아 태평양(서울) – 600734575887
- 아시아 태평양(싱가포르) – 114774131450
- 아시아 태평양(시드니) – 783225319266
- 아시아 태평양(도쿄) – 582318560864
- 캐나다(중부) – 985666609251
- 유럽(프랑크푸르트) – 054676820928

- 유럽(아일랜드) – 156460612806
- 유럽(런던) – 652711504416
- 유럽(밀라노) – 635631232127
- 유럽(파리) – 009996457667
- 유럽(스톡홀름) – 897822967062
- 중동(바레인) – 076674570225
- 남아메리카(상파울루) – 507241528517

에 예제 정책에 표시된 형식을 사용하여 액세스 로그 위치의 ARN을 `Resource`입력합니다. 항상 로드 밸런서가 있는 계정의 계정 ID를 S3 버킷 ARN의 리소스 경로에 포함합니다. 이렇게 하면 지정된 계정의 로드 밸런서만 S3 버킷에 액세스 로그를 쓸 수 있습니다.

지정하는 ARN은 3단계에서 액세스 로그를 활성화할 때 접두사를 포함할 계획인지 여부에 따라 다릅니다.

접두사가 있는 S3 버킷 ARN 예제

S3 버킷 이름은 amzn-s3-demo-logging-bucket이고 접두사는 입니다logging-prefix.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/*
```

접두사가 없는 S3 버킷 ARN 예제

S3 버킷 이름은 입니다amzn-s3-demo-logging-bucket. S3 버킷 ARN에는 접두사 부분이 없습니다.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/AWSLogs/123456789012/*
```

AWS GovCloud (US) 리전

이 정책은 지정된 Elastic Load Balancing 계정 ID에 권한을 부여합니다. AWS GovCloud (US) 리전의 로드 밸런서에이 정책을 사용합니다.

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

```
        "AWS": "arn:aws-us-gov:iam::elb-account-id:root"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws-us-gov:s3:::amzn-s3-demo-bucket/prefix/AWSLogs/123456789012/
*"
    }
  ]
}
```

의 경우 *elb-account-id*를 로드 밸런서의 리전에 대한 Elastic Load Balancing 계정 ID로 `Principal`바꿉니다.

- AWS GovCloud(미국 서부) – 048591011584
- AWS GovCloud(미국 동부) – 190560391635

에 예제 정책에 표시된 형식을 사용하여 액세스 로그 위치의 ARN을 `Resource`입력합니다. 항상 로드 밸런서가 있는 계정의 계정 ID를 S3 버킷 ARN의 리소스 경로에 포함합니다. 이렇게 하면 지정된 계정의 로드 밸런서만 S3 버킷에 액세스 로그를 쓸 수 있습니다.

지정하는 S3 버킷 ARN은 액세스 로그 링크 3단계를 활성화할 때 접두사를 포함할 계획인지 여부에 따라 달라집니다.

접두사가 있는 S3 버킷 ARN 예제

S3 버킷 이름은 amzn-s3-demo-logging-bucket이고 접두사는 입니다logging-prefix.

```
arn:aws-us-gov:s3:::amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/*
```

접두사가 없는 S3 버킷 ARN 예제

S3 버킷 이름은 입니다amzn-s3-demo-logging-bucket. S3 버킷 ARN에는 접두사 부분이 없습니다.

```
arn:aws-us-gov:s3:::amzn-s3-demo-logging-bucket/AWSLogs/123456789012/*
```

Outposts 영역

다음 정책은 지정된 로그 전송 서비스에 권한을 부여합니다. Outposts 영역의 로드 밸런서에 이 정책을 사용하세요.

```
{
```

```
    "Effect": "Allow",
    "Principal": {
        "Service": "logdelivery.elb.amazonaws.com"
    },
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix/AWSLogs/123456789012/*",
    "Condition": {
        "StringEquals": {
            "s3:x-amz-acl": "bucket-owner-full-control"
        }
    }
}
```

에 예제 정책에 표시된 형식을 사용하여 액세스 로그 위치의 ARN을 Resource입력합니다. 항상 로드 밸런서가 있는 계정의 계정 ID를 S3 버킷 ARN의 리소스 경로에 포함합니다. 이렇게 하면 지정된 계정의 로드 밸런서만 S3 버킷에 액세스 로그를 쓸 수 있습니다.

지정하는 S3 버킷 ARN은 3단계에서 액세스 로그를 활성화할 때 접두사를 포함할 계획인지 여부에 따라 달라집니다.

접두사가 있는 S3 버킷 ARN 예제

S3 버킷 이름은 amzn-s3-demo-logging-bucket이고 접두사는 입니다logging-prefix.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/*
```

접두사가 없는 S3 버킷 ARN 예제

S3 버킷 이름은 입니다amzn-s3-demo-logging-bucket. S3 버킷 ARN에는 접두사 부분이 없습니다.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/AWSLogs/123456789012/*
```

⚠ 보안 강화

다음 제안을 사용하여 S3 버킷의 보안을 강화합니다.

버킷 정책 검토

- S3 버킷 ARN의 계정 ID 부분을 포함하여 전체 리소스 경로를 사용합니다. S3 버킷 ARN의 계정 ID 부분에 와일드카드(*)를 사용하지 마세요.

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix/AWSLogs/123456789012/*"
```

- aws:SourceArn를 사용하여 지정된 리전 및 계정의 로드 밸런서만 버킷을 사용할 수 있도록 합니다.

```
"Condition": {
    "ArnLike": {
        "aws:SourceArn":
 "arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/*"
    }
}
```

- 와 aws:SourceOrgId 함께 aws:SourceArn를 사용하여 지정된 조직의 로드 밸런서만 버킷을 사용할 수 있도록 합니다.

```
"Condition": {
    "StringEquals": {
        "aws:SourceOrgId": "o-1234567890"
    },
    "ArnLike": {
        "aws:SourceArn": "arn:aws:elasticloadbalancing:*:*:loadbalancer/*"
    }
}
```

- 명시적으로 허용된 보안 주체를 제외한 서비스 보안 주체에 대한 액세스를 방지하는 Deny 명령문이 있는 경우 허용된 서비스 보안 주체 목록에 logdelivery.elasticloadbalancing.amazonaws.com를 추가해야 합니다. 예를 들어 aws:PrincipalServiceNamesList 조건을 사용한 경우 다음과 logdelivery.elasticloadbalancing.amazonaws.com 같이를 추가합니다.

```
{
  "Effect": "Deny",
  "Principal": "*",
  "Condition": {
      "StringNotEqualsIfExists": {
          "aws:PrincipalServiceNamesList": [
              "logdelivery.elasticloadbalancing.amazonaws.com",
              "service.amazonaws.com"
          }
      }
  }
```

```
}
```

NotPrincipal 요소를 사용한 경우 다음과 logdelivery.elasticloadbalancing.amazonaws.com 같이를 추가합니다. NotPrincipal 요소를 사용하는 대신 aws:PrincipalServiceName 또는 aws:PrincipalServiceNamesList 조건 키를 사용하여 서비스 보안 주체를 명시적으로 허용하는 것이 좋습니다. 자세한 내용은 NotPrincipal을 참조하세요.

```
{
  "Effect": "Deny",
  "NotPrincipal": {
    "Service": [
       "logdelivery.elasticloadbalancing.amazonaws.com",
       "service.amazonaws.com"
    ]
  }
},
```

Amazon S3 콘솔을 사용하여 버킷에 액세스 로그의 버킷 정책 연결

1. https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 버킷 이름을 선택하여 세부 정보 페이지를 엽니다.
3. 권한을 선택한 다음에 버킷 정책, 편집을 선택합니다.
4. 버킷 정책을 업데이트하여 필수 권한을 부여합니다.
5. 변경 사항 저장을 선택합니다.

3단계: 액세스 로그 구성

다음 절차에 따라 요청 정보를 캡처하고 로그 파일을 S3 버킷에 전송하도록 액세스 로그를 구성합니다.

요구 사항

버킷은 1단계에 설명된 요구 사항을 충족해야 하며 2단계의 설명에 따라 버킷 정책을 연결해야 합니다. 접두사를 포함하는 경우 접두사에 'AWSLogs' 문자열이 포함되지 않아야 합니다.

콘솔을 사용하여 로드 밸런서에 대한 액세스 로그를 활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 [Load Balancers]를 클릭합니다.
3. 로드 밸런서 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 모니터링에서 액세스 로그를 켭니다.
6. S3 URI로 로그 파일의 S3 URI를 입력합니다. 지정하는 URI는 접두사 사용 여부에 따라 달라집니다.
   - 접두사가 있는 URI: s3://*amzn-s3-demo-logging-bucket*/*logging-prefix*
   - 접두사가 없는 URI: s3://*amzn-s3-demo-logging-bucket*
7. 변경 사항 저장을 선택합니다.

를 사용하여 액세스 로그를 활성화하려면 AWS CLI

modify-load-balancer-attributes 명령을 사용합니다.

액세스 로그에 대해 S3 버킷을 관리하려면

액세스 로그용으로 구성한 버킷을 삭제하기 전에 액세스 로그를 비활성화해야 합니다. 이렇게 하지 않으면 이름이 동일한 새로운 버킷이 있지만 필요한 버킷 정책이 다른 AWS 계정 에 생성된 경우, Elastic Load Balancing이 내 로드 밸런서의 액세스 로그를 이 새로운 버킷에 쓸 수 있습니다.

4단계: 버킷 권한 확인

로드 밸런서에 대해 액세스 로그가 활성화되면 Elastic Load Balancing에서는 S3 버킷을 검증하고 버킷 정책에서 필수 권한을 지정하는지 확인하는 테스트 파일을 생성합니다. Amazon S3 콘솔을 사용하여 테스트 파일이 생성되었는지 확인할 수 있습니다. 테스트 파일은 실제 액세스 로그 파일이 아니며, 예제 레코드가 포함되어 있지 않습니다.

Amazon S3 콘솔을 사용하여 버킷에서 테스트 파일이 생성되었는지 확인하려면

1. https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 액세스 로그에 대해 지정한 버킷의 이름을 선택합니다.
3. 테스트 파일인 `ELBAccessLogTestFile`로 이동합니다. 위치는 접두사 사용 여부에 따라 달라집니다.

- 접두사가 있는 위치: *amzn-s3-demo-logging-bucket*/*logging-prefix*/AWSLogs/*123456789012*/ELBAccessLogTestFile
- 접두사가 없는 위치: *amzn-s3-demo-logging-bucket*/AWSLogs/*123456789012*/ELBAccessLogTestFile

문제 해결

액세스 거부 오류가 발생한 경우, 가능한 원인은 다음과 같습니다.

- 버킷 정책이 버킷에 액세스 로그를 쓰도록 허용하는 Elastic Load Balancing 권한을 부여하지 않습니다. 리전에 올바른 버킷 정책을 사용하고 있는지 확인합니다. 액세스 로그를 활성화할 때 지정한 것과 동일한 버킷 이름을 리소스 ARN에서 사용하는지 확인하세요. 액세스 로그를 활성화할 때 접두사를 지정하지 않은 경우 리소스 ARN에 접두사가 포함되어 있지 않은지 확인합니다.
- 버킷이 지원되지 않는 서버 측 암호화 옵션을 사용합니다. 버킷이 Amazon S3 관리형 키(SSE-S3)를 사용해야 합니다.

Application Load Balancer 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 S3 사용 설명서의 S3 버킷 생성, 구성 및 작업을 참조하세요. Amazon S3

콘솔을 사용하여 액세스 로그를 비활성화하는 방법

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 [Load Balancers]를 클릭합니다.
3. 로드 밸런서 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 모니터링에서 액세스 로그를 끕니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 액세스 로그를 비활성화하려면 AWS CLI

modify-load-balancer-attributes 명령을 사용합니다.

Application Load Balancer의 연결 로그

Elastic Load Balancing은 로드 밸런서에 전송된 요청에 대한 자세한 정보를 캡처하는 연결 로그를 제공합니다. 각 로그에는 클라이언트의 IP 주소 및 포트, 리스너 포트, 사용된 TLS 암호 및 프로토콜, TLS 핸드셰이크 지연 시간, 연결 상태, 클라이언트 인증서 세부 정보와 같은 정보가 포함됩니다. 이러한 연결 로그를 사용하여 요청 패턴을 분석하고 문제를 해결할 수 있습니다.

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

Amazon S3의 스토리지 비용은 청구되지만, Amazon S3로 로그 파일을 전송하기 위해 Elastic Load Balancing에서 사용하는 대역폭에 대해서는 요금이 부과되지 않습니다. 스토리지 비용에 대한 자세한 내용은 Amazon S3 요금을 참조하세요.

내용

- 연결 로그 파일
- 연결 로그 항목
- 로그 항목 예제
- 연결 로그 파일 처리
- Application Load Balancer에서 연결 로그 활성화
- Application Load Balancer에서 연결 로그 비활성화

연결 로그 파일

Elastic Load Balancing은 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.

연결 로그의 파일 이름은 다음 형식을 사용합니다.

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

bucket

S3 버킷의 이름.

접두사

(선택 사항) 버킷의 접두사(논리적 계층 구조)입니다. 지정하는 접두사에는 문자열 AWSLogs가 포함되지 않아야 합니다. 자세한 내용은 접두사를 사용한 객체 구성을 참조하세요.

AWSLogs

AWSLogs로 시작하는 파일 이름의 일부가 지정하는 버킷 이름과 선택적 접두사 뒤에 추가됩니다.

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 UTC 또는 Zulu 시간으로 23:35과 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 접두사가 있는 로그 파일 이름의 예입니다.

```
s3://amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/
elasticloadbalancing/us-east-2/2022/05/01/
conn_log.123456789012_elasticloadbalancing_us-east-2_app.my-
loadbalancer.1234567890abcdef_20220215T2340Z_172.160.001.192_20sg8hgm.log.gz
```

다음은 접두사가 없는 로그 파일 이름의 예입니다.

```
s3://amzn-s3-demo-logging-bucket/AWSLogs/123456789012/elasticloadbalancing/us-
east-2/2022/05/01/conn_log.123456789012_elasticloadbalancing_us-east-2_app.my-
loadbalancer.1234567890abcdef_20220215T2340Z_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 S3 사용 설명서의 객체 수명 주기 관리를 참조하세요.

연결 로그 항목

각 연결 시도마다 연결 로그 파일에 항목이 있습니다. 클라이언트 요청이 전송되는 방법은 연결이 영구적 또는 비영구적인지에 따라 결정됩니다. 비영구적 연결에는 단일 요청이 있으며, 이는 액세스 로그 및 연결 로그에 단일 항목을 생성합니다. 영구적 연결에는 여러 요청이 있으며, 이는 액세스 로그에 여러 항목을 생성하고 연결 로그에 단일 항목을 생성합니다.

내용

- 구문
- 오류 이유 코드

구문

연결 로그 항목은 다음 형식을 사용합니다.

```
[timestamp] [client_ip] [client_port] [listener_port] [tls_protocol] [tls_cipher]
 [tls_handshake_latency] [leaf_client_cert_subject] [leaf_client_cert_validity]
 [leaf_client_cert_serial_number] [tls_verify_status]
```

다음 표에서는 연결 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
타임스탬프	로드 밸런서가 연결을 성공적으로 설정한 또는 설정하지 못한 시간(ISO 8601 형식)입니다.
client_ip	요청을 하는 클라이언트의 IP 주소입니다.

필드	설명
client_port	요청을 전송하는 클라이언트의 포트입니다.
listener_port	클라이언트 요청을 수신하는 로드 밸런서 리스너의 포트입니다.
tls_protocol	[HTTPS 리스너] 핸드셰이크 중에 사용되는 SSL/TLS 프로토콜입니다. 비 SSL/TLS 요청에서는 이 필드가 -로 설정됩니다.
tls_cipher	[HTTPS 리스너] 핸드셰이크 중에 사용되는 SSL/TLS 프로토콜입니다. 비 SSL/TLS 요청에서는 이 필드가 -로 설정됩니다.
tls_handshake_latency	[HTTPS 리스너] 성공적인 핸드셰이크를 설정하는 동안 경과된 총 시간(초 단위, 밀리초 정밀도)입니다. 다음의 경우 이 필드는 -로 설정됩니다. • 수신 요청이 SSL/TLS 요청이 아닙니다. • 핸드셰이크가 성공적으로 설정되지 않았습니다.
leaf_client_cert_subject	[HTTPS 리스너] 리프 클라이언트 인증서의 주제 이름입니다. 다음의 경우 이 필드는 -로 설정됩니다. • 수신 요청이 SSL/TLS 요청이 아닙니다. • 로드 밸런서 리스너의 구성에서 mTLS가 활성화되지 않았습니다. • 서버가 리프 클라이언트 인증서를 로드/파싱할 수 없습니다.
leaf_client_cert_validity	[HTTPS 리스너] 리프 클라이언트 인증서의 유효성입니다(ISO 8601 형식의 `not-before` 및 `not-after` 사용). 다음의 경우 이 필드는 -로 설정됩니다. • 수신 요청이 SSL/TLS 요청이 아닙니다. • 로드 밸런서 리스너의 구성에서 mTLS가 활성화되지 않았습니다. • 서버가 리프 클라이언트 인증서를 로드/파싱할 수 없습니다.

필드	설명
leaf_client_cert_serial_number	[HTTPS 리스너] 리프 클라이언트 인증서의 일련 번호입니다. 다음의 경우 이 필드는 -로 설정됩니다. • 수신 요청이 SSL/TLS 요청이 아닙니다. • 로드 밸런서 리스너의 구성에서 mTLS가 활성화되지 않았습니다. • 서버가 리프 클라이언트 인증서를 로드/파싱할 수 없습니다.
tls_verify_status	[HTTPS 리스너] 연결 요청의 상태입니다. 연결이 성공적으로 설정되면 값이 Success입니다. 연결이 실패하면 값이 Failed:$error_code 입니다.
conn_trace_id	연결 추적 가능성 ID는 각 연결을 식별하는 데 사용되는 고유한 불투명 ID입니다. 클라이언트와의 연결이 설정되면 이 클라이언트의 후속 요청에서 이 ID가 해당 액세스 로그 항목에 포함됩니다. 이 ID는 외부 키 역할을 하여 연결 로그와 액세스 로그 간의 링크를 생성합니다.

오류 이유 코드

로드 밸런서가 연결을 설정할 수 없는 경우 로드 밸런서는 연결 로그에 다음 사유 코드 중 하나를 저장합니다.

코드	설명
ClientCertMaxChainDepthExceeded	최대 클라이언트 인증서 체인 깊이를 초과했습니다.
ClientCertMaxSizeExceeded	최대 클라이언트 인증서 크기를 초과했습니다.
ClientCertCrlHit	CA가 클라이언트 인증서를 취소했습니다.

코드	설명	
ClientCertCrlProcessingError	CRL 처리 오류	
ClientCertUntrusted	클라이언트 인증서를 신뢰할 수 없습니다.	
ClientCertNotYetValid	클라이언트 인증서가 아직 유효하지 않습니다.	
ClientCertExpired	인증서가 만료되었습니다.	
ClientCertTypeUnsupported	클라이언트 인증서 유형이 지원되지 않습니다.	
ClientCertInvalid	인증서가 유효하지 않습니다.	
ClientCertPurposeInvalid	클라이언트 인증서 목적이 잘못되었습니다.	
ClientCertRejected	클라이언트 인증서가 사용자 지정 서버 검증에서 거부되었습니다.	
UnmappedConnectionError	매핑되지 않은 런타임 연결 오류	

로그 항목 예제

다음은 예제 연결 로그 항목입니다.

다음은 포트 443에서 상호 TLS 확인 모드가 활성화된 HTTPS 리스너와의 연결 성공에 대한 예제 로그 항목입니다.

```
2023-10-04T17:05:15.514108Z 203.0.113.1 36280 443 TLSv1.2 ECDHE-RSA-AES128-GCM-
SHA256 4.036 "CN=amazondomains.com,O=endEntity,L=Seattle,ST=Washington,C=US"
 NotBefore=2023-09-21T22:43:21Z;NotAfter=2026-06-17T22:43:21Z FEF257372D5C14D4 Success
```

다음은 포트 443에서 상호 TLS 확인 모드가 활성화된 HTTPS 리스너와의 연결 실패에 대한 예제 로그 항목입니다.

```
2023-10-04T17:05:15.514108Z 203.0.113.1 36280 443 TLSv1.2 ECDHE-RSA-AES128-
GCM-SHA256 - "CN=amazondomains.com,O=endEntity,L=Seattle,ST=Washington,C=US"
 NotBefore=2023-09-21T22:43:21Z;NotAfter=2026-06-17T22:43:21Z FEF257372D5C14D4
 Failed:ClientCertUntrusted
```

연결 로그 파일 처리

연결 로그 파일은 압축된 상태입니다. Amazon S3 콘솔을 사용하여 파일을 열면 파일이 압축되지 않고 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 연결 로그를 분석 및 처리할 수 있습니다.

- Amazon Athena는 표준 SQL을 사용해 Amazon S3에 저장된 데이터를 간편하게 분석할 수 있는 대화식 쿼리 서비스입니다.
- Loggly
- Splunk
- Sumo Logic

Application Load Balancer에서 연결 로그 활성화

로드 밸런서에 대한 연결 로그를 활성화할 때는 로드 밸런서가 로그를 저장할 S3 버킷의 이름을 지정해야 합니다. 버킷에 액세스 로그를 쓰는 Elastic Load Balancing 권한을 부여하는 버킷 정책이 이 버킷에 있어야 합니다.

업무

- 1단계: S3 버킷 생성

- 2단계: S3 버킷에 정책 연결
- 3단계: 연결 로그 구성
- 4단계: 버킷 권한 확인
- 문제 해결

1단계: S3 버킷 생성

연결 로그를 활성화할 때는 반드시 연결 로그용 S3 버킷을 지정해야 합니다. 기존 버킷을 사용하거나 연결 로그 전용 버킷을 생성할 수 있습니다. 버킷은 다음 요구 사항을 충족해야 합니다.

요구 사항

- 버킷은 로드 밸런서와 같은 리전에 있어야 합니다. 서로 다른 계정에서 버킷과 로드 밸런서를 소유할 수 있습니다.
- 지원되는 유일한 서버 측 암호화 옵션은 Amazon S3 관리형 키(SSE-S3)입니다. 자세한 내용을 Amazon S3 관리형 암호화 키(SSE-S3) 섹션을 참조하세요.

Amazon S3 콘솔을 사용하여 S3 버킷에 폴더를 생성하려면

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 버킷 만들기를 선택합니다.
3. [Create a bucket] 페이지에서 다음과 같이 실행합니다.
    a. [Bucket Name]에서 버킷 이름을 입력합니다. 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 S3 사용 설명서의 버킷 규제 및 제한을 참조하세요.
    b. AWS 리전의 경우 로드 밸런서를 생성한 리전을 선택합니다.
    c. 기본 암호화에서 Amazon S3 관리형 키(SSE-S3)를 선택합니다.
    d. 버킷 생성을 선택합니다.

2단계: S3 버킷에 정책 연결

버킷에 연결 로그를 쓰는 Elastic Load Balancing 권한을 부여하는 버킷 정책이 S3 버킷에 있어야 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.

연결된 정책이 이미 있는 기존 버킷을 사용하는 Elastic Load Balancing 연결 로그에 대한 문을 정책에 추가할 수 있습니다. 그렇게 하는 경우, 결과적인 액세스 권한 집합을 평가하여 해당 집합이 연결 로그에 대한 버킷에 액세스해야 하는 사용자에게 적절한 권한인지 확인하는 것이 좋습니다.

사용 가능한 버킷 정책

사용할 버킷 정책은 AWS 리전 및 영역 유형에 따라 다릅니다.

⚠ 정확한 S3 버킷 ARNs.

- S3 버킷 ARN뿐만 아니라 전체 리소스 경로를 사용합니다.
- S3 버킷 ARN의 계정 ID 부분을 포함합니다.
- S3 버킷 ARN의 계정 ID 부분에 와일드카드(*)를 사용하지 마세요.

이용 가능한 리전(2022년 8월 이후 기준)

이 정책은 지정된 로그 전송 서비스에 권한을 부여합니다. 다음 리전의 로드 밸런서에이 정책을 사용합니다.

- 아시아 태평양(하이데라바드)
- 아시아 태평양(말레이시아)
- 아시아 태평양(멜버른)
- 아시아 태평양(태국)
- 캐나다 서부(캘거리)
- 유럽(스페인)
- 유럽(취리히)
- 이스라엘(텔아비브)
- 중동(UAE)
- 멕시코(중부)

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

```
      "Principal": {
        "Service": "logdelivery.elasticloadbalancing.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix/AWSLogs/123456789012/*"
    }
  ]
}
```

에 예제 정책에 표시된 형식을 사용하여 액세스 로그 위치의 ARN을 Resource입력합니다. 항상 로드 밸런서가 있는 계정의 계정 ID를 S3 버킷 ARN의 리소스 경로에 포함합니다. 이렇게 하면 지정된 계정의 로드 밸런서만 S3 버킷에 액세스 로그를 쓸 수 있습니다.

지정하는 S3 버킷 ARN은 3단계에서 액세스 로그를 활성화할 때 접두사를 포함할 계획인지 여부에 따라 달라집니다.

접두사가 있는 S3 버킷 ARN 예제

S3 버킷 이름은 amzn-s3-demo-logging-bucket이고 접두사는 입니다logging-prefix.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/*
```

접두사가 없는 S3 버킷 ARN 예제

S3 버킷 이름은 입니다amzn-s3-demo-logging-bucket. S3 버킷 ARN에는 접두사 부분이 없습니다.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/AWSLogs/123456789012/*
```

Effect가 Deny인 경우 NotPrincipal 사용

Amazon S3 버킷 정책이 아래 예제와 같이 Effect 값의 Deny를 사용하고 NotPrincipal을 포함하는 경우 Service 목록에 logdelivery.elasticloadbalancing.amazonaws.com이 포함되어야 합니다.

```
{
  "Effect": "Deny",
  "NotPrincipal": {
    "Service": [
       "logdelivery.elasticloadbalancing.amazonaws.com",
       "example.com"
    ]
  }
```

```
},
```

이용 가능한 리전(2022년 8월 이전 기준)

이 정책은 지정된 Elastic Load Balancing 계정에 권한을 부여합니다. 아래 나열된 리전의 로드 밸런서에이 정책을 사용합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::elb-account-id:root"
      },
      "Action": "s3:PutObject",
       "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix/AWSLogs/123456789012/*"
    }
  ]
}
```

의 경우 *elb-account-id*를 로드 밸런서의 리전에 대한 Elastic Load Balancing 계정 ID로 `Principal`바꿉니다.

- 미국 동부(버지니아 북부) – 127311923021
- 미국 동부(오하이오) – 033677994240
- 미국 서부(캘리포니아 북부) – 027434742980
- 미국 서부(오레곤) – 797873946194
- 아프리카(케이프타운) – 098369216593
- 아시아 태평양(홍콩) – 754344448648
- 아시아 태평양(자카르타) – 589379963580
- 아시아 태평양(뭄바이) – 718504428378
- 아시아 태평양(오사카) – 383597477331
- 아시아 태평양(서울) – 600734575887
- 아시아 태평양(싱가포르) – 114774131450
- 아시아 태평양(시드니) – 783225319266
- 아시아 태평양(도쿄) – 582318560864

- 캐나다(중부) – 985666609251
- 유럽(프랑크푸르트) – 054676820928
- 유럽(아일랜드) – 156460612806
- 유럽(런던) – 652711504416
- 유럽(밀라노) – 635631232127
- 유럽(파리) – 009996457667
- 유럽(스톡홀름) – 897822967062
- 중동(바레인) – 076674570225
- 남아메리카(상파울루) – 507241528517

에 예제 정책에 표시된 형식을 사용하여 액세스 로그 위치의 ARN을 `Resource`입력합니다. 항상 로드 밸런서가 있는 계정의 계정 ID를 S3 버킷 ARN의 리소스 경로에 포함합니다. 이렇게 하면 지정된 계정의 로드 밸런서만 S3 버킷에 액세스 로그를 쓸 수 있습니다.

지정하는 S3 버킷 ARN은 3단계에서 액세스 로그를 활성화할 때 접두사를 포함할 계획인지 여부에 따라 달라집니다.

접두사가 있는 S3 버킷 ARN 예제

S3 버킷 이름은 amzn-s3-demo-logging-bucket이고 접두사는 입니다logging-prefix.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/*
```

접두사가 없는 S3 버킷 ARN 예제

S3 버킷 이름은 입니다amzn-s3-demo-logging-bucket. S3 버킷 ARN에는 접두사 부분이 없습니다.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/AWSLogs/123456789012/*
```

AWS GovCloud (US) 리전

이 정책은 지정된 Elastic Load Balancing 계정에 권한을 부여합니다. 아래 목록에 있는 AWS GovCloud (US) 리전의 가용 영역 또는 로컬 영역의 로드 밸런서에이 정책을 사용합니다.

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

```
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws-us-gov:iam::elb-account-id:root"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws-us-gov:s3:::amzn-s3-demo-bucket/prefix/AWSLogs/123456789012/
*"
    }
  ]
}
```

*elb-account-id*를 로드 밸런서의 리전에 대한 Elastic Load Balancing 계정 ID로 `Principal` 바꾸려면:

- AWS GovCloud(미국 서부) – 048591011584
- AWS GovCloud(미국 동부) – 190560391635

에 예제 정책에 표시된 형식을 사용하여 액세스 로그 위치의 ARN을 `Resource`입력합니다. 항상 로드 밸런서가 있는 계정의 계정 ID를 S3 버킷 ARN의 리소스 경로에 포함합니다. 는 지정된 계정에서 로드 밸런서가 S3 버킷에 액세스 로그를 쓸 수 있도록 합니다.

지정하는 S3 버킷 ARN은 액세스 로그를 활성화할 때 접두사를 포함할 계획인지 여부에 따라 달라집니다.

접두사가 있는 S3 버킷 ARN 예제

S3 버킷 이름은 amzn-s3-demo-logging-bucket이고 접두사는 입니다logging-prefix.

```
arn:aws-us-gov:s3:::amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/*
```

접두사가 없는 S3 버킷 ARN 예제

S3 버킷 이름은 입니다amzn-s3-demo-logging-bucket. S3 버킷 ARN에는 접두사 부분이 없습니다.

```
arn:aws-us-gov:s3:::amzn-s3-demo-logging-bucket/AWSLogs/123456789012/*
```

Outposts 영역

다음 정책은 지정된 로그 전송 서비스에 권한을 부여합니다. Outposts 영역의 로드 밸런서에 이 정책을 사용하세요.

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "logdelivery.elb.amazonaws.com"
    },
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix/AWSLogs/123456789012/*"
    "Condition": {
        "StringEquals": {
            "s3:x-amz-acl": "bucket-owner-full-control"
        }
    }
}
```

에 액세스 로그 위치의 ARN을 Resource입력합니다. 항상 로드 밸런서가 있는 계정의 계정 ID를 S3 버킷 ARN의 리소스 경로에 포함합니다. 이렇게 하면 지정된 계정의 로드 밸런서만 S3 버킷에 액세스 로그를 쓸 수 있습니다.

지정하는 ARN은 3단계에서 액세스 로그를 활성화할 때 접두사를 포함할 계획인지 여부에 따라 다릅니다.

접두사가 있는 S3 버킷 ARN 예제

S3 버킷 이름은 amzn-s3-demo-logging-bucket이고 접두사는 입니다logging-prefix.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/logging-prefix/AWSLogs/123456789012/*
```

접두사가 없는 S3 버킷 ARN 예제

S3 버킷 이름은 입니다amzn-s3-demo-logging-bucket. S3 버킷 ARN에는 접두사 부분이 없습니다.

```
arn:aws:s3:::amzn-s3-demo-logging-bucket/AWSLogs/123456789012/*
```

Effect가 Deny인 경우 NotPrincipal 사용

Amazon S3 버킷 정책이 아래 예제와 같이 Effect 값의 Deny를 사용하고 NotPrincipal을 포함하는 경우 Service 목록에 logdelivery.elasticloadbalancing.amazonaws.com이 포함되어야 합니다.

```
{
```

```
  "Effect": "Deny",
  "NotPrincipal": {
    "Service": [
       "logdelivery.elasticloadbalancing.amazonaws.com",
       "example.com"
    ]
  }
},
```

Amazon S3 콘솔을 사용하여 연결 로그의 버킷 정책을 버킷에 연결하려면

1. https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 버킷 이름을 선택하여 세부 정보 페이지를 엽니다.
3. 권한을 선택한 다음에 버킷 정책, 편집을 선택합니다.
4. 버킷 정책을 업데이트하여 필수 권한을 부여합니다.
5. 변경 사항 저장을 선택합니다.

3단계: 연결 로그 구성

다음 절차에 따라 이벤트를 캡처하고 로그 파일을 S3 버킷에 전송하도록 연결 로그를 구성합니다.

요구 사항

버킷은 1단계에 설명된 요구 사항을 충족해야 하며 2단계의 설명에 따라 버킷 정책을 연결해야 합니다. 접두사를 지정하는 경우 접두사에 'AWSLogs' 문자열이 포함되지 않아야 합니다.

콘솔을 사용하여 로드 밸런서에서 연결 로그를 활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 [Load Balancers]를 클릭합니다.
3. 로드 밸런서 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 모니터링에서 연결 로그를 켭니다.
6. S3 URI로 로그 파일의 S3 URI를 입력합니다. 지정하는 URI는 접두사 사용 여부에 따라 달라집니다.

   - 접두사가 있는 URI: `s3://bucket-name/prefix`
   - 접두사가 없는 URI: `s3://bucket-name`

7. 변경 사항 저장을 선택합니다.

를 사용하여 연결 로그를 활성화하려면 AWS CLI

modify-load-balancer-attributes 명령을 사용합니다.

연결 로그용 S3 버킷을 관리하려면

연결 로그용으로 구성한 버킷을 삭제하기 전에 연결 로그를 비활성화해야 합니다. 이렇게 하지 않으면 이름이 동일한 새로운 버킷이 있지만 필요한 버킷 정책이 다른 AWS 계정 에 생성된 경우, Elastic Load Balancing이 내 로드 밸런서의 연결 로그를 이 새로운 버킷에 쓸 수 있습니다.

4단계: 버킷 권한 확인

로드 밸런서에 대해 연결 로그가 활성화되면 Elastic Load Balancing에서는 S3 버킷을 검증하고 버킷 정책에서 필수 권한을 지정하는지 확인하는 테스트 파일을 생성합니다. Amazon S3 콘솔을 사용하여 테스트 파일이 생성되었는지 확인할 수 있습니다. 테스트 파일은 실제 연결 로그 파일이 아니며, 예제 레코드가 포함되어 있지 않습니다.

S3 버킷에서 Elastic Load Balancing이 테스트 파일을 생성했는지 확인하려면

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 연결 로그에 대해 지정한 버킷의 이름을 선택합니다.
3. 테스트 파일인 `ELBConnectionLogTestFile`로 이동합니다. 위치는 접두사 사용 여부에 따라 달라집니다.

   - 접두사가 있는 위치: *amzn-s3-demo-logging-bucket*/*prefix*/AWSLogs/*123456789012*/ELBConnectionLogTestFile
   - 접두사가 없는 위치: *amzn-s3-demo-logging-bucket*/AWSLogs/*123456789012*/ELBConnectionLogTestFile

문제 해결

액세스 거부 오류가 발생한 경우, 가능한 원인은 다음과 같습니다.

- 버킷 정책이 버킷에 연결 로그를 쓰도록 허용하는 Elastic Load Balancing 권한을 부여하지 않습니다. 리전에 올바른 버킷 정책을 사용하고 있는지 확인합니다. 연결 로그를 활성화할 때 지정한 것과 동일한 버킷 이름을 리소스 ARN에서 사용하는지 확인하세요. 연결 로그를 활성화할 때 접두사를 지정하지 않은 경우 리소스 ARN에 접두사가 포함되어 있지 않은지 확인합니다.

- 버킷이 지원되지 않는 서버 측 암호화 옵션을 사용합니다. 버킷이 Amazon S3 관리형 키(SSE-S3)를 사용해야 합니다.

Application Load Balancer에서 연결 로그 비활성화

언제든지 로드 밸런서에서 연결 로그를 비활성화할 수 있습니다. 연결 로그를 비활성화하면 연결 로그는 사용자가 삭제할 때까지 S3 버킷에 남아 있습니다. 자세한 내용은 Amazon S3 사용 설명서의 버킷 생성, 구성 및 작업을 참조하세요.

콘솔을 사용하여 연결 로그를 비활성화하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 [Load Balancers]를 클릭합니다.
3. 로드 밸런서 이름을 선택하여 세부 정보 페이지를 엽니다.
4. 속성성(Attributes) 탭에서 편집(Edit)을 선택합니다.
5. 모니터링에서 연결 로그를 끕니다.
6. 변경 사항 저장을 선택합니다.

를 사용하여 연결 로그를 비활성화하려면 AWS CLI

modify-load-balancer-attributes 명령을 사용합니다.

Application Load Balancer에 대한 요청 추적

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.

요청 추적을 사용하여 클라이언트에서 대상 또는 기타 서비스로 가는 HTTP 요청을 추적할 수 있습니다. 액세스 로그를 활성화하면 X-Amzn-Trace-Id 헤더의 콘텐츠가 기록됩니다. 자세한 내용은 Application Load Balancer에 대한 액세스 로그 단원을 참조하십시오.

구문

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

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

필드

필드의 이름입니다. 지원되는 값은 Root 및 Self입니다.

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

version

버전 번호입니다. 이 값은 1입니다.

시간

epoch 시간(초)입니다. 이 값은 8자리 16진수입니다.

id

트레이스 식별자입니다. 이 값은 24자리 16진수입니다.

예시

X-Amzn-Trace-Id 헤더가 들어오는 요청에 존재하지 않는 경우에는 로드 밸런서가 Root 필드를 가진 헤더를 생성하고 요청을 전달합니다. 다음 예를 참조하십시오.

```
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. 다음 예를 참조하십시오.

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

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

제한 사항

- 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.

Application Load Balancer 문제 해결

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

문제

- 등록된 대상은 서비스되지 않고 있습니다.
- 클라이언트가 인터넷 경계 로드 밸런서에 연결할 수 없음
- 사용자 지정 도메인으로 전송된 요청은 로드 밸런서에 수신되지 않음
- 로드 밸런서로 전송된 HTTPS 요청은 “NET: :ERR_CERT_COMMON_NAME_INVALID”를 반환합니다.
- 로드 밸런서가 높은 처리 시간을 표시합니다
- 로드 밸런서가 응답 코드 000을 보냅니다.
- 로드 밸런서가 HTTP 오류 코드를 생성
- 대상이 HTTP 오류 코드를 생성
- AWS Certificate Manager 인증서를 사용할 수 없음
- 여러 줄의 헤더는 지원되지 않습니다.
- 리소스 맵을 사용하여 비정상 대상 문제 해결

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 Application Load Balancer 대상 그룹의 상태 확인 단원을 참조하십시오.

인스턴스가 상태 확인에 실패하고 있는지 확인한 다음, 다음 문제를 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.

대상 응답 코드의 형식이 잘못되었거나 대상에 연결하는 동안 오류가 발생했습니다.

애플리케이션이 로드 밸런서의 상태 확인 요청에 응답하는지 확인합니다. 일부 애플리케이션에서는 로드 밸런서가 보낸 HTTP 호스트 헤더에 응답하기 위한 가상 호스트 구성과 같이 상태 확인에 응답하기 위해 추가 구성이 필요합니다. 호스트 헤더 값에는 대상의 프라이빗 IP 주소가 포함되며 기본 포트를 사용하지 않는 경우 상태 확인 포트가 뒤에 옵니다. 대상이 기본 상태 확인 포트를 사용하는 경우 호스트 헤더 값에는 대상의 프라이빗 IP 주소만 포함됩니다. 예를 들어, 대상의 프라이빗 IP 주소가 `10.0.0.10`이고 상태 확인 포트가 `8080`인 경우 상태 확인에서 로드 밸런서가 보내는 HTTP 호스트 헤더는 `Host: 10.0.0.10:8080`입니다. 대상의 프라이빗 IP 주소가 `10.0.0.10`이고 상태 확인 포트가 `80`인 경우 상태 확인에서 로드 밸런서가 보내는 HTTP 호스트 헤더는 `Host: 10.0.0.10`입니다. 애플리케이션 상태를 확인하려면 해당 호스트에 응답하는 가상 호스트 구성 또는 기본 구성이 필요할 수 있습니다. 상태 확인 요청에는 다음 속성이 포함됩니다. `User-Agent`는 `ELB-HealthChecker/2.0`로 설정되고, 메시지 헤더 필드의 행 종결자는 시퀀스 CRLF이며, 헤더는 첫 번째 빈 행에서 종료되고 그 뒤에 CRLF가 옵니다.

클라이언트가 인터넷 경계 로드 밸런서에 연결할 수 없음

로드 밸런서가 요청에 응답하지 않는 경우에는 다음 문제를 점검하세요.

인터넷 경계 로드 밸런서가 프라이빗 서브넷에 연결됩니다.

로드 밸런서를 위한 퍼블릭 서브넷을 지정해야 합니다. 퍼블릭 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.

사용자 지정 도메인으로 전송된 요청은 로드 밸런서에 수신되지 않음

로드 밸런서가 커스텀 도메인에 보낸 요청을 받지 않는 경우에는 다음 문제를 점검하세요.

사용자 지정 도메인 이름이 로드 밸런서 IP 주소로 확인되지 않음

- 명령줄 인터페이스를 사용하여 사용자 지정 도메인 이름이 어떤 IP 주소로 변환되는지 확인합니다.
  - Linux, macOS 또는 Unix — 터미널에서 `dig` 명령을 사용할 수 있습니다. Ex.`dig example.com`
  - Windows — 명령 프롬프트에서 `nslookup` 명령을 사용할 수 있습니다. Ex.`nslookup example.com`
- 명령줄 인터페이스를 사용하여 로드 밸런서 DNS 이름이 어떤 IP 주소로 변환되는지 확인합니다.
- 두 출력의 결과를 비교합니다. IP 주소는 일치해야 합니다.

Route 53을 사용하여 사용자 지정 도메인을 호스팅하는 경우 Amazon Route 53 개발자 안내서의 인터넷에서 내 도메인을 사용할 수 없음을 참조하세요.

로드 밸런서로 전송된 HTTPS 요청은 “NET: :ERR_CERT_COMMON_NAME_INVALID”를 반환합니다.

HTTPS 요청이 로드 밸런서에서 `NET::ERR_CERT_COMMON_NAME_INVALID`을 받는 경우 다음과 같은 가능한 원인을 확인하세요.

- HTTPS 요청에 사용된 도메인 이름이 리스너 관련 ACM 인증서에 지정된 대체 이름과 일치하지 않습니다.
- 로드 밸런서의 기본 DNS 이름이 사용되고 있습니다. `*.amazonaws.com` 도메인에 공개 인증서를 요청할 수 없으므로 기본 DNS 이름을 사용하여 HTTPS 요청을 할 수 없습니다.

로드 밸런서가 높은 처리 시간을 표시합니다

로드 밸런서가 구성에 따라 처리 시간을 다르게 계산합니다.

- AWS WAF 가 Application Load Balancer와 연결되어 있고 클라이언트가 HTTP POST 요청을 보내는 경우 POST 요청에 대한 데이터를 보내는 시간은 로드 밸런서 액세스 로그의 `request_processing_time` 필드에 반영됩니다. 이는 HTTP POST 요청에 대해 예상된 동작입니다.
- AWS WAF 가 Application Load Balancer와 연결되지 않고 클라이언트가 HTTP POST 요청을 보내는 경우 POST 요청에 대한 데이터를 보내는 시간은 로드 밸런서 액세스 로그의 `target_processing_time` 필드에 반영됩니다. 이는 HTTP POST 요청에 대해 예상된 동작입니다.

로드 밸런서가 응답 코드 000을 보냅니다.

HTTP/2 연결을 사용하면 한 연결을 통해 제공되는 요청 수가 10,000개를 초과하는 경우 로드 밸런서는 GOAWAY 프레임을 보내고 TCP FIN과의 연결을 닫습니다.

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
- HTTP 401: 권한 없음
- HTTP 403: 금지됨
- HTTP 405: 허용되지 않은 메서드
- HTTP 408: 요청 제한 시간
- HTTP 413: 페이로드가 너무 큼
- HTTP 414: URI가 너무 길
- HTTP 460
- HTTP 463
- HTTP 464
- HTTP 500: 내부 서버 오류
- HTTP 501: 구현되지 않음
- HTTP 502: 잘못된 게이트웨이
- HTTP 503: 서비스 사용 불가
- HTTP 504: 게이트웨이 제한 시간
- HTTP 505: 버전이 지원되지 않습니다.
- HTTP 507: 스토리지 부족
- HTTP 561: 권한 없음

HTTP 400: 잘못된 요청

가능한 원인:

- 클라이언트가 HTTP 사양을 충족하지 않는 잘못된 형식의 요청을 전송했습니다.
- 요청 헤더가 요청 줄당 16K, 단일 헤더당 16K 또는 전체 요청 헤더에서 64K를 초과했습니다.
- 클라이언트가 전체 요청 본문을 보내기 전에 연결을 종료했습니다.

HTTP 401: 권한 없음

사용자를 인증하도록 리스너 규칙을 구성했지만, 다음 중 하나가 true입니다.

- 인증되지 않은 사용자를 거부하도록 `OnUnauthenticatedRequest`를 구성했거나 IdP가 액세스를 거부했습니다.

- 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.
- 요청된 범위가 ID 토큰을 반환하지 않습니다.
- 클라이언트 로그인 제한 시간이 만료되기 전에 로그인 프로세스를 완료하지 않았습니다. 자세한 내용은 클라이언트 로그인 시간 초과 단원을 참조하세요.

HTTP 403: 금지됨

Application Load Balancer에 대한 요청을 모니터링하도록 AWS WAF 웹 액세스 제어 목록(웹 ACL)을 구성했으며 요청을 차단했습니다.

HTTP 405: 허용되지 않은 메서드

클라이언트가 사용한 TRACE 방법은 Application Load Balancer에서 지원하지 않습니다.

HTTP 408: 요청 제한 시간

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

가능한 원인:

- 대상이 Lambda 함수이고 요청 본문이 1MB를 초과합니다.
- 요청 헤더가 요청 줄당 16K, 단일 헤더당 16K 또는 전체 요청 헤더에서 64K를 초과했습니다.

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 너무 많은 IP 주소를 가진 X-Forwarded-For 요청 헤더를 받았습니다. IP 주소의 상한은 30입니다.

HTTP 464

로드 밸런서가 대상 그룹 프로토콜의 버전 구성과 호환되지 않는 수신 요청 프로토콜을 받았습니다.

가능한 원인:

- 요청 프로토콜은 HTTP/1.1이지만, 대상 그룹 프로토콜 버전은 gRPC 또는 HTTP/2입니다.
- 요청 프로토콜은 GRPC이지만, 대상 그룹 프로토콜 버전은 HTTP/1.1입니다.
- 요청 프로토콜은 HTTP/2이고 요청은 POST가 아니지만, 대상 그룹 프로토콜 버전은 gRPC입니다.

HTTP 500: 내부 서버 오류

가능한 원인:

- AWS WAF 웹 액세스 제어 목록(웹 ACL)을 구성했으며 웹 ACL 규칙을 실행하는 동안 오류가 발생했습니다.
- 로드 밸런서가 IdP 토큰 엔드포인트 또는 IdP 사용자 정보 엔드포인트와 통신할 수 없습니다.
  - IdP의 DNS를 공개적으로 확인할 수 있는지 확인합니다.
  - 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.
- IdP로부터 받은 사용자 클레임의 크기가 11KB를 초과합니다.
- IdP 토큰 엔드포인트 또는 IdP 사용자 정보 엔드포인트가 응답하는 데 5초 이상 걸립니다.

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.
- 전체 응답 헤더의 대상 응답 헤더가 32K를 초과했습니다.
- 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.
- 대상이 오류를 반환하는 Lambda 함수이거나 Lambda 서비스에 의해 스로틀된 함수입니다.
- 로드 밸런서가 대상에 연결할 때 SSL 핸드셰이크 오류가 발생했습니다.

자세한 내용은 AWS 지원 지식 센터에서 Application Load Balancer HTTP 502 오류 문제를 해결하려면 어떻게 해야 합니까?를 참조하세요.

HTTP 503: 서비스 사용 불가

로드 밸런서의 대상 그룹에 등록된 대상이 없거나 등록된 모든 대상이 `unused` 상태에 있습니다.

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.
- 네트워크 ACL 또는 SecurityGroup 정책이 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 Lambda 함수이고 Lambda 서비스는 연결 제한 시간이 만료되기 전에 응답하지 않았습니다.
- 로드 밸런서가 대상에 연결할 때 SSL 핸드셰이크 시간 초과(10초)가 발생했습니다.

HTTP 505: 버전이 지원되지 않습니다.

로드 밸런서가 예상치 못한 HTTP 버전 요청을 받았습니다. 예를 들어, 로드 밸런서가 HTTP/1 연결을 설정했지만 HTTP/2 요청을 받았습니다.

HTTP 507: 스토리지 부족

리디렉션 URL이 너무 깁니다.

HTTP 561: 권한 없음

사용자를 인증하도록 리스너 규칙을 구성했지만, 사용자를 인증할 때 IdP가 오류 코드를 반환했습니다. 액세스 로그에 관련 오류 원인 코드가 있는지 확인하십시오.

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.

AWS Certificate Manager 인증서를 사용할 수 없음

Application Load Balancer에서 HTTPS 리스너를 사용하기로 결정할 때 인증서를 발급하기 전에 도메인 소유권을 검증해야 AWS Certificate Manager 합니다. 설치 중에 이 단계를 빠뜨렸다면 인증서는 `Pending Validation` 상태로 유지되며 유효성이 확인될 때까지 사용할 수 없습니다.

- 이메일 검증을 사용하는 경우 AWS Certificate Manager 사용 설명서의 이메일 검증을 참조하세요.
- DNS 검증을 사용하는 경우AWS Certificate Manager 사용 설명서의 DNS 검증을 참조하십시오.

여러 줄의 헤더는 지원되지 않습니다.

Application Load Balancer는`message/http` 미디어 유형 헤더를 비롯한 여러 줄 헤더를 지원하지 않습니다. 여러 줄의 헤더가 제공되면 Application Load Balancer는 `":"` 콜론 문자를 추가한 다음 대상에 전달합니다.

리소스 맵을 사용하여 비정상 대상 문제 해결

Application Load Balancer 대상이 상태 확인에 실패하는 경우 리소스 맵을 사용하여 비정상 대상을 찾고 실패 원인 코드를 기반으로 조치를 취할 수 있습니다. 자세한 내용은 Application Load Balancer 리소스 맵 보기 단원을 참조하십시오.

리소스 맵은 개요 및 비정상 대상 맵의 두 가지 보기를 제공합니다. 개요는 기본적으로 선택되며 로드 밸런서의 모든 리소스를 표시합니다. 비정상 대상 맵 보기를 선택하면 Application Load Balancer와 연결된 각 대상 그룹에서 비정상 대상만 표시됩니다.

> **Note**
>
> 리소스 세부 정보 표시를 활성화하여 리소스 맵 내의 모든 관련 리소스에 대한 상태 확인 요약 및 오류 메시지를 확인해야 합니다. 활성화되지 않은 경우 각 리소스를 선택하여 세부 정보를 확인해야 합니다.

대상 그룹 열에는 각 대상 그룹의 정상 및 비정상 대상에 대한 요약이 표시됩니다. 이는 모든 대상이 상태 확인에 실패하는지 또는 특정 대상만 실패하는지 확인하는 데 도움이 될 수 있습니다. 대상 그룹의 모든 대상이 상태 확인에 실패하는 경우 대상 그룹의 구성을 확인합니다. 대상 그룹 이름을 선택하여 새 탭에서 세부 정보 페이지를 엽니다.

대상 열에는 TargetID와 각 대상의 현재 상태 확인 상태가 표시됩니다. 대상이 비정상이면 상태 확인 실패 사유 코드가 표시됩니다. 단일 대상이 상태 확인에 실패하면 대상에 충분한 리소스가 있는지 확인하고 대상에서 실행되는 애플리케이션을 사용할 수 있는지 확인합니다. 대상 ID를 선택하여 새 탭에서 세부 정보 페이지를 엽니다.

내보내기를 선택하면 Application Load Balancer 리소스 맵의 현재 보기를 PDF로 내보낼 수 있는 옵션이 제공됩니다.

인스턴스가 상태 확인에 실패하는지 확인한 다음, 실패 사유 코드를 기반으로 다음 문제를 점검합니다.

- 비정상: HTTP 응답 불일치
  - 대상에서 실행되는 애플리케이션이 Application Load Balancer의 상태 확인 요청에 올바른 HTTP 응답을 보내는지 확인합니다.
  - 또는 대상에서 실행되는 애플리케이션의 응답과 일치하도록 Application Load Balancer의 상태 확인 요청을 업데이트할 수 있습니다.
- 비정상: 요청 시간 초과
  - 대상 및 Application Load Balancer와 연결된 보안 그룹 및 네트워크 액세스 제어 목록(ACL)이 연결을 차단하지 않는지 확인합니다.
  - 대상에 Application Load Balancer의 연결을 수락할 수 있는 충분한 리소스가 있는지 확인합니다.
  - 대상에서 실행 중인 애플리케이션의 상태를 확인합니다.
  - Application Load Balancer의 상태 확인 응답은 각 대상의 애플리케이션 로그에서 볼 수 있습니다. 자세한 정보는 상태 확인 사유 코드를 참조하세요.
- 비정상: 상태 확인 실패
  - 대상에서 실행 중인 애플리케이션의 상태를 확인합니다.
  - 대상이 상태 확인 포트에서 트래픽을 수신하는지 확인합니다.

    HTTPS 리스너를 사용하는 경우

    프런트엔드 연결에 사용되는 보안 정책을 선택합니다. 백엔드 연결에 사용되는 보안 정책은 사용 중인 프런트엔드 보안 정책에 따라 자동으로 선택됩니다.

    - HTTPS 리스너가 프런트엔드 연결에 TLS 1.3 보안 정책을 사용하면 백엔드 연결에 `ELBSecurityPolicy-TLS13-1-0-2021-06` 보안 정책이 사용됩니다.
    - 백엔드 연결의 경우 HTTPS 리스너가 TLS 1.3 보안 정책을 사용하면 `ELBSecurityPolicy-2016-08` 보안 정책이 사용됩니다.

    자세한 내용은 보안 정책을 참조하세요.
  - 대상이 보안 정책에 지정된 올바른 형식으로 서버 인증서 및 키를 제공하는지 확인합니다.
  - 대상이 하나 이상의 일치하는 암호와 Application Load Balancer에서 TLS 핸드셰이크를 설정하기 위해 제공하는 프로토콜을 지원하는지 확인합니다.

Application Load Balancer에 대한 할당량

AWS 계정에는 각 AWS 서비스에 대한 이전 제한이라고 하는 기본 할당량이 있습니다. 다르게 표시되지 않는 한 리전별로 각 할당량이 적용됩니다. 일부 할당량에 대한 증가를 요청할 수 있으며 다른 할당량은 늘릴 수 없습니다.

Application Load Balancer에 대한 할당량을 보려면 Service Quotas 콘솔을 엽니다. 탐색 창에서 AWS 서비스를 선택하고 Elastic Load Balancing을 선택합니다. Elastic Load Balancing에 대해 describe-account-limits(AWS CLI) 명령을 사용할 수도 있습니다.

할당량 증가를 요청하려면 Service Quotas 사용 설명서의 할당량 증가 요청을 참조하세요. Service Quotas에서 아직 할당량을 사용할 수 없는 경우 서비스 할당량 증가 요청을 제출합니다.

할당량

- 로드 밸런서
- 대상 그룹
- 규칙
- 트러스트 스토어
- 인증서
- HTTP 헤더
- Load Balancer서 용량 단위

로드 밸런서

AWS 계정에는 Application Load Balancer와 관련된 다음과 같은 할당량이 있습니다.

명칭	기본값	조정 가능
리전당 Application Load Balancer	50	예
Application Load Balancer당 인증서(기본 인증서 제외)	25	예
Application Load Balancer당 리스너	50	예

명칭	기본값	조정 가능
Application Load Balancer당 작업당 대상 그룹	5	아니요
Application Load Balancer당 대상 그룹	100	아니요
Application Load Balancer당 대상	1,000	예

대상 그룹

다음 할당량은 대상 그룹용입니다.

명칭	기본값	조정 가능
리전당 대상 그룹	3,000*	예
리전별 대상 그룹당 대상(인스턴스 또는 IP 주소)	1,000	예
리전별 대상 그룹당 대상(Lambda 함수)	1	아니요
대상 그룹당 로드 밸런서	1	아니요

* 이 할당량은 Application Load Balancer 및 Network Load Balancer에서 공유됩니다.

규칙

다음 할당량은 규칙용입니다.

명칭	기본값	조정 가능
Application Load Balancer당 규칙(기본 규칙 제외)	100	예
규칙당 조건 값	5	아니요
규칙당 조건 와일드카드	5	아니요
규칙당 일치 평가	5	아니요

트러스트 스토어

다음은 트러스트 스토어에 대한 할당량입니다.

명칭	기본값	조정 가능
계정당 트러스트 스토어	20	예
로드 밸런서당 확인 모드에서 mTLS를 사용하는 리스너 수입니다.	2	아니요

인증서

CA 인증서 이름 알림 및 인증서 해지 목록을 포함한 다음 할당량이 인증서에 적용됩니다.

명칭	기본값	조정 가능
CA 인증서 크기	16KB	아니요
트러스트 스토어당 CA 인증서 수	25	예
트러스트 스토어당 CA 인증서 주체 크기	10,000	예
최대 인증서 체인 깊이	4	아니요
트러스트 스토어당 해지 항목 수	500,000	예
해지 목록 파일 크기	50MB	아니요
트러스트 스토어당 해지 목록 수	30	예
TLS 메시지 크기	64K	아니요

HTTP 헤더

HTTP 헤더에는 다음과 같이 크기 제한이 있습니다.

명칭	기본값	조정 가능
요청 라인	16K	아니요
단일 헤더	16K	아니요
전체 응답 헤더	32K	아니요
전체 요청 헤더	64K	아니요

Load Balancer서 용량 단위

다음 할당량은 Load Balancer 용량 단위(LCU)에 대한 할당량입니다.

명칭	기본값	조정 가능
Application Load Balancer당 예약 Application Load Balancer 용량 단위(LCUs)	1500	예
리전당 예약 Application Load Balancer 용량 단위(LCU)	0	예

Application Load Balancer 문서 기록

다음 표에서는 Application Load Balancer의 릴리스에 대해 설명합니다.

변경 사항	설명	날짜
HTTP 헤더 수정	이 릴리스에는 모든 응답 코드에 대한 HTTP 헤더 수정 지원이 추가되었습니다. 이전에는이 기능이 응답 코드 2xx 및 3xx로 제한되었습니다.	2025년 2월 28일
용량 단위 예약	이 릴리스에서는 로드 밸런서의 최소 용량을 설정하는 지원이 추가되었습니다.	2024년 11월 20일
리소스 맵	이 릴리스에는 로드 밸런서 리소스 및 관계를 시각적 형식으로 볼 수 있는 지원이 추가되었습니다.	2024년 3월 8일
원클릭 WAF	이 릴리스에서는 로드 밸런서가 클릭 한 번으로 통합되는 경우 로드 밸런서의 동작을 구성하기 위한 지원을 추가합니다 AWS WAF.	2024년 2월 6일
상호 TLS	이 릴리스에는 상호 TLS 인증에 대한 지원이 추가되었습니다.	2023년 11월 26일
자동 대상 가중치	이 릴리스에는 자동 대상 가중치 알고리즘에 대한 지원이 추가되었습니다.	2023년 11월 26일
FIPS 140-3 TLS 종료	이 릴리스에서는 TLS 연결을 종료할 때 FIPS 140-3 암호 모	2023년 11월 20일

	듈을 사용하는 보안 정책이 추가되었습니다.	
IPv6를 사용하여 대상 등록	이 릴리스에는 IPv6 주소를 사용할 때 인스턴스를 대상으로 등록할 수 있는 지원이 추가되었습니다.	2023년 10월 2일
TLS 1.3을 지원하는 보안 정책	이 릴리스에서는 TLS 1.3 사전 정의 보안 정책에 대한 지원이 추가되었습니다.	2023년 3월 22일
영역 전환	이 릴리스에는 Amazon Application Recovery Controller (ARC)와의 통합을 통해 장애가 있는 단일 가용 영역에서 다른 가용 영역으로 트래픽을 라우팅할 수 있는 지원이 추가되었습니다.	2022년 11월 28일
교차 영역 로드 밸런싱 해제	이 릴리스에는 교차 영역 로드 밸런싱을 비활성화할 수 있는 지원이 추가되었습니다.	2022년 11월 28일
대상 그룹 상태	이 릴리스에는 정상이어야 하는 대상의 최소 개수 또는 백분율, 임계값이 충족되지 않은 경우 로드 밸런서가 취하는 작업을 구성할 수 있는 지원이 추가되었습니다.	2022년 11월 28일
교차 영역 로드 밸런싱	이 릴리스에는 대상 그룹 수준에서 교차 영역 로드 밸런싱을 구성할 수 있는 지원이 추가되었습니다.	2022년 11월 17일

IPv6 대상 그룹	이 릴리스에서는 Application Load Balancer의 IPv6 대상 그룹 구성에 대한 지원이 추가되었습니다.	2021년 11월 23일
IPv6 내부 로드 밸런서	이 릴리스에서는 Application Load Balancer의 IPv6 대상 그룹 구성에 대한 지원이 추가되었습니다.	2021년 11월 23일
AWS PrivateLink 및 고정 IP 주소	이 릴리스에서는 Network Load Balancer에서 Application Load Balancer로 트래픽을 직접 전달하여 고정 IP 주소를 사용하고 AWS PrivateLink 노출하는 지원이 추가되었습니다.	2021년 9월 27일
클라이언트 포트 보존	이 릴리스에서는 클라이언트가 로드 밸런서 연결에 사용한 소스 포트를 보존하는 속성이 추가되었습니다.	2021년 7월 29일
TLS 헤더	이 릴리스에는 협상된 TLS 버전과 암호 그룹에 대한 정보를 포함하는 TLS 헤더가 대상에 전송되기 전에 클라이언트 요청에 추가되었음을 나타내는 속성이 추가되었습니다.	2021년 7월 21일
추가적인 ACM 인증서	이 릴리스에서는 2048, 3072, 4096비트 키 길이의 RSA 인증서와 모든 ECDSA 인증서를 지원합니다.	2021년 7월 14일
애플리케이션 기반 고정	이 릴리스에서는 로드 밸런서의 고정 세션을 지원하기 위해 애플리케이션 기반 쿠키를 추가합니다.	2021년 2월 8일

TLS 버전 1.2를 지원하는 FS에 대한 보안 정책	이 릴리스에는 TLS 버전 1.2를 지원하는 FS(Forward Secrecy)에 대한 보안 정책이 추가되었습니다.	2020년 11월 24일
WAF 페일 오픈 지원	이 릴리스에서는 로드 밸런서가 통합되는 경우 로드 밸런서의 동작을 구성하기 위한 지원을 추가합니다 AWS WAF.	2020년 11월 13일
gRPC 및 HTTP/2 지원	이번 릴리스에서는 gRPC 워크로드 및 엔드 투 엔드 HTTP/2에 대한 지원이 추가되었습니다.	2020년 10월 29일
Outpost 지원	AWS Outposts에 Application Load Balancer를 프로비저닝할 수 있습니다.	2020년 9월 8일
Desync Mitigation Mode	이 릴리스에서는 Desync Mitigation Mode에 대한 지원이 추가되었습니다.	2020년 8월 17일
최소 미해결 요청	이 릴리스에서는 최소 미해결 요청 알고리즘에 대한 지원이 추가되었습니다.	2019년 11월 25일
가중 대상 그룹	이 릴리스에는 여러 대상 그룹이 있는 전달 작업에 대한 지원이 추가되었습니다. 요청은 각 대상 그룹에 대해 지정한 가중치를 기준으로 이러한 대상 그룹에 배포됩니다.	2019년 11월 19일
New attribute	이 릴리스에서는 routing.http.drop_invalid_header_fields.enabled 속성에 대한 지원이 추가되었습니다.	2019년 11월 15일

FS 보안 정책	이 릴리스에는 세 개의 추가 사전 정의 순방향 비밀성 보안 정책에 대한 지원이 추가되었습니다.	2019년 10월 8일
고급 요청 라우팅	이 릴리스에서는 리스너 규칙의 추가 조건 형식에 대한 지원을 추가합니다.	2019년 3월 27일
대상으로서 Lambda 함수	이 릴리스에는 대상으로서 Lambda 함수를 등록하는 작업에 대한 지원이 추가됩니다.	2018년 11월 29일
리디렉션 작업	이 릴리스에는 로드 밸런서가 요청을 다른 URL로 리디렉션하기 위한 지원이 추가되었습니다.	2018년 7월 25일
고정 응답 작업	이 릴리스에는 로드 밸런서가 사용자 지정 HTTP 응답을 반환하기 위한 지원이 추가되었습니다.	2018년 7월 25일
FS 및 TLS 1.2 보안 정책	이 릴리스에서는 두 개의 추가 사전 정의 보안 정책에 대한 지원이 추가되었습니다.	2018년 6월 6일
사용자 인증	이 릴리스에서는 요청을 라우팅하기 전에 기업 또는 소셜 자격 증명을 사용하여 애플리케이션의 사용자를 인증할 수 있도록 로드 밸런서에 대한 지원이 추가되었습니다.	2018년 5월 30일
리소스 수준 권한	이 릴리스에서는 리소스 레벨 권한 및 태깅 조건 키에 대한 지원이 추가되었습니다.	2018년 5월 10일

느린 시작 모드	이 릴리스는 느린 시작 모드(로드 밸런서가 워밍업 동안 새로 등록된 대상으로 보내는 요청 공유를 점차 증가시킴)를 위한 지원을 추가합니다.	2018년 3월 24일
SNI 지원	이번 릴리스에는 SNI(Server Name Indication)에 대한 지원이 추가되었습니다.	2017년 10월 10일
IP 주소를 대상으로 사용	이 릴리스에서는 IP 주소를 대상으로 등록에 대한 지원이 추가되었습니다.	2017년 8월 31일
호스트 기반 라우팅	이 릴리스에서는 호스트 헤더의 호스트 이름을 기반으로 하는 라우팅 요청에 대한 지원이 추가되었습니다.	2017년 4월 5일
TLS 1.1 및 TLS 1.2 보안 정책	이 릴리스에는 TLS 1.1 및 TLS 1.2. 보안 정책이 추가되었습니다.	2017년 2월 6일
IPv6 지원	이 릴리스에는 IPv6 주소에 대한 지원이 추가되었습니다.	2017년 1월 25일
요청 추적	이 릴리스에는 요청 추적에 대한 지원이 추가되었습니다.	2016년 11월 22일
TargetResponseTime 지표 백분위수 지원	이 릴리스에는 Amazon CloudWatch가 지원하는 새로운 백분위수 통계 지원이 추가되었습니다.	2016년 11월 17일
새로운 로드 밸런서 유형	이번 Elastic Load Balancing 릴리스에는 Application Load Balancer가 도입되었습니다.	2016년 8월 11일

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