개발자 가이드

# Amazon CloudFront

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

# Amazon CloudFront: 개발자 가이드

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 상표 및 트레이드 드레스는 Amazon 외 제품 또는 서비스와 함께 사용되어서는 안되며, 고객에게 혼동을 일으키거나 Amazon 브랜드 이미지를 떨어뜨리고 폄하하는 방식으로 이용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 해당 상표의 소유자의 자산이며, 해당 상표의 소유자가 Amazon의 계열사이거나 Amazon과 제휴 관계에 있거나 Amazon의 후원을 받는 업체일 수 있습니다.

# Table of Contents

Amazon CloudFront란 무엇입니까? ..... 1
콘텐츠를 전송하도록 CloudFront를 설정하는 방법 ..... 2
표준 배포 또는 다중 테넌트 배포 중에서 선택 ..... 4
요금 ..... 4
CloudFront 사용 방법 ..... 4
정적 웹 사이트 콘텐츠 전송 속도 향상 ..... 5
온디맨드 비디오 또는 라이브 스트리밍 비디오 제공 ..... 5
시스템 처리를 통해 특정 필드 암호화 ..... 6
엣지에서의 사용자 지정 ..... 6
Lambda@Edge 사용자 지정을 사용하여 프라이빗 콘텐츠 제공 ..... 6
CloudFront에서 콘텐츠를 제공하는 방법 ..... 7
CloudFront에서 사용자에게 콘텐츠를 제공하는 방법 ..... 7
CloudFront에서 리전 에지 캐시를 사용하는 방식 ..... 8
CloudFront 엣지 서버 ..... 10
CloudFront 관리형 접두사 목록 사용 ..... 11
AWS SDK 작업 ..... 11
CloudFront 기술 리소스 ..... 12
시작하기 ..... 14
AWS 계정 설정 ..... 14
AWS 계정에 가입 ..... 14
관리자 액세스 권한이 있는 사용자 생성 ..... 15
CloudFront에 액세스하는 방법 선택 ..... 16
표준 배포 시작 ..... 17
사전 조건 ..... 17
버킷 생성 ..... 18
콘텐츠 업로드 ..... 18
배포 생성 ..... 19
콘텐츠에 액세스 ..... 20
정리 ..... 20
기본 배포 향상 ..... 20
시작(AWS CLI) ..... 21
사전 조건 ..... 21
Amazon S3 버킷 생성 ..... 22
버킷에 콘텐츠 업로드 ..... 22

오리진 액세스 제어(OAC) 생성 ... 22
표준 배포 생성 ... 23
S3 버킷 정책 업데이트 ... 24
배포 작업 확인 ... 25
CloudFront를 통해 콘텐츠에 액세스 ... 25
정리 ... 26
안전한 정적 웹 사이트 시작하기 ... 27
솔루션 개요 ... 28
솔루션 배포 ... 29
CloudFront 정액 요금제 ... 34
CloudFront 정액 요금제의 이점 ... 35
요금제 티어별 기능 ... 35
요금제 기능 ... 36
월별 사용 허용량 ... 59
과거 사용량에 따른 자격 ... 59
플랜에서 부담하는 비용 ... 59
Route 53 DNS 관리 및 플랜 ... 60
요금제로 전체 AWS 비용 절감 ... 61
정액 요금제 관리 ... 62
요금제에 대한 새 배포 구독 ... 62
기존 배포에서 요금제 구독 ... 62
요금제 업그레이드 ... 63
요금제 다운그레이드 ... 64
요금제 취소 ... 64
보류 중인 플랜 변경 취소 ... 65
요금제를 사용하는 배포 삭제 ... 65
권한 ... 66
정액 요금제 할당량 ... 66
지원되지 않는 기능 ... 66
지원되지 않는 기능 ... 67
지원되지 않는 연결 ... 68
계정 수준 제약 조건 ... 69
리소스 수준 제약 조건 ... 69
요금제에 영향을 미칠 수 있는 추가 기능 ... 69
요금제와 종량제 요금 비교 ... 70
배포 구성 ... 71

다중 테넌트 배포의 작동 방식 이해 ..... 72
작동 방식 ..... 74
용어 ..... 75
지원되지 않는 기능 ..... 77
배포 테넌트 사용자 지정 ..... 78
인증서 요청(배포 테넌트) ..... 81
사용자 지정 연결 그룹 생성(선택 사항) ..... 88
다중 테넌트 배포로 마이그레이션 ..... 89
배포 생성 ..... 91
콘솔에서 CloudFront 배포 생성 ..... 92
표시되는 값 ..... 97
추가 링크 ..... 98
CloudFront 표준 배포에 도메인 추가 ..... 98
사전 구성된 배포 설정 ..... 101
Amazon S3 오리진 ..... 101
API 게이트웨이 요금 ..... 102
사용자 지정 오리진 및 EC2 인스턴스 ..... 103
Elastic Load Balancing 오리진 ..... 105
MediaPackage v1 오리진 ..... 107
MediaPackage v2 오리진 ..... 108
MediaTailor 오리진 ..... 109
모든 배포 설정 ..... 110
오리진 설정 ..... 111
캐시 동작 설정 ..... 121
배포 설정 ..... 135
사용자 지정 오류 페이지 및 오류 캐싱 ..... 143
지리적 제한 ..... 145
배포 테스트 ..... 145
객체에 대한 링크 생성 ..... 145
배포 업데이트 ..... 146
콘솔에서 배포 업데이트 ..... 146
배포 태깅 ..... 149
태그 제한 ..... 150
배포에 태그 추가, 편집, 삭제 ..... 150
프로그래밍 방식 태깅 ..... 151
배포 삭제 ..... 151

다양한 오리진 사용 ..... 153
Amazon S3 버킷 사용 ..... 154
MediaStore 컨테이너 또는 MediaPackage 채널 사용 ..... 166
Application Load Balancer 사용 ..... 166
Network Load Balancer 사용 ..... 166
Lambda 함수 URL 사용 ..... 167
Amazon EC2(또는 기타 사용자 지정 오리진) 사용 ..... 168
CloudFront 오리진 그룹 사용 ..... 169
Amazon API Gateway 사용 ..... 170
IPv6 사용 ..... 170
IPv6 뷰어 요청 ..... 170
IPv6 오리진 요청 ..... 171
CloudFront 지속적 배포를 사용하여 변경을 안전하게 테스트 ..... 172
CloudFront 지속적 배포 워크플로 ..... 174
스테이징 배포 및 지속적 배포 정책 사용 ..... 175
스테이징 배포 모니터링 ..... 184
지속적 배포의 작동 방식 알아보기 ..... 185
지속적 배포를 위한 할당량 및 기타 고려 사항 ..... 187
사용자 지정 URL 사용 ..... 188
대체 도메인 이름 사용과 관련된 요구 사항 ..... 188
대체 도메인 이름 사용에 대한 제한 ..... 190
대체 도메인 이름 사용 ..... 193
대체 도메인 이름 이동 ..... 196
대체 도메인 이름 제거 ..... 207
대체 도메인 이름에서 와일드카드 사용 ..... 208
WebSocket 사용 ..... 209
WebSocket 프로토콜의 작동 방식 ..... 209
WebSocket 요구 사항 ..... 210
권장 WebSocket 헤더 ..... 210
허용 목록에 사용할 애니캐스트 고정 IP 요청 ..... 211
사전 조건 ..... 211
애니캐스트 고정 IP 목록 요청 ..... 211
애니캐스트 고정 IP 목록 생성 ..... 212
애니캐스트 고정 IP 목록을 기존 배포와 연결 ..... 212
애니캐스트 고정 IP 목록을 새 배포와 연결 ..... 213
애니캐스트 고정 IP 목록을 연결 그룹과 연결 ..... 214

애니캐스트 고정 IP 목록 업데이트 ..... 214
IPAM을 사용하여 CloudFront로 자체 IP 가져오기 ..... 215
gRPC 사용 ..... 219
CloudFront에서 gRPC가 작동하는 방식 ..... 219
CloudFront에서 공유 리소스 작업 ..... 222
리소스 공유를 위한 사전 조건 ..... 222
VPC 오리진 공유 ..... 223
공유 VPC 오리진 사용 ..... 225
공유 VPC 오리진 식별 ..... 226
공유 VPC 오리진 공유 해제 ..... 226
공유 VPC 오리진에 대한 책임 및 권한 ..... 227
소유자에 대한 권한 ..... 227
소비자에 대한 권한 ..... 227
AWSRAMDefaultPermissionCloudFront ..... 228
결제 및 측정 ..... 228
공유 리소스 할당량 ..... 228
캐싱 및 가용성 ..... 229
캐시 적중률 개선 ..... 229
CloudFront에서 객체를 캐싱하는 시간 지정 ..... 230
Origin Shield 사용 ..... 230
쿼리 문자열 파라미터 기반 캐싱 ..... 230
쿠키 값 기반 캐싱 ..... 231
요청 헤더 기반 캐싱 ..... 232
압축이 불필요할 때 `Accept-Encoding` 헤더 제거 ..... 233
HTTP를 통한 미디어 콘텐츠 제공 ..... 233
Origin Shield 사용 ..... 233
Origin Shield 사용 사례 ..... 234
Origin Shield의 AWS 리전 선택 ..... 239
Origin Shield 사용 ..... 241
Origin Shield 비용 추정 ..... 243
Origin Shield 고가용성 ..... 244
Origin Shield가 다른 CloudFront 기능과 상호 작용하는 방법 ..... 244
오리진 장애 조치를 통한 고가용성 향상 ..... 245
오리진 그룹 생성 ..... 247
오리진 제한 시간 및 시도 횟수 제어 ..... 248
Lambda@Edge 함수와 함께 오리진 장애 조치 사용 ..... 249

오리진 장애 조치와 함께 사용자 지정 오류 페이지 사용 ..... 250
캐시 만료 관리 ..... 251
헤더를 사용하여 개별 객체의 캐시 기간 제어 ..... 252
오래된(만료된) 콘텐츠 제공 ..... 253
CloudFront에서 객체를 캐싱하는 시간 지정 ..... 256
Amazon S3 콘솔을 사용하여 객체에 헤더 추가 ..... 261
캐싱 및 쿼리 문자열 파라미터 ..... 261
쿼리 문자열 전달 및 캐싱에 대한 콘솔 및 API 설정 ..... 263
캐싱 최적화 ..... 263
쿼리 문자열 파라미터 및 CloudFront 표준 로그(액세스 로그) ..... 265
쿠키 기반의 콘텐츠 캐싱 ..... 265
요청 헤더 기반의 콘텐츠 캐싱 ..... 268
헤더 및 배포 – 개요 ..... 269
캐싱의 기반이 되는 헤더 선택 ..... 270
CORS 설정을 준수하도록 CloudFront 구성 ..... 271
디바이스 유형을 기반으로 캐싱하도록 구성 ..... 271
뷰어 언어를 기반으로 캐싱하도록 구성 ..... 272
뷰어 언어를 기반으로 캐싱하도록 구성 ..... 272
요청의 프로토콜을 기반으로 캐싱하도록 구성 ..... 272
압축 파일에 대한 캐싱 구성 ..... 272
헤더를 기반으로 한 캐싱이 성능에 미치는 영향 ..... 272
헤더 및 헤더 값의 대소문자가 캐싱에 미치는 영향 ..... 273
CloudFront에서 최종 사용자에게 반환하는 헤더 ..... 273
정책으로 캐시 키 제어 ..... 274
캐시 정책 이해 ..... 275
정책 정보 ..... 275
TTL(Time To Live) 설정 ..... 275
캐시 키 설정 ..... 276
캐시 정책 생성 ..... 282
관리형 캐시 정책 사용 ..... 285
Amplify ..... 286
CachingDisabled ..... 287
CachingOptimized ..... 288
CachingOptimizedForUncompressedObjects ..... 289
Elemental-MediaPackage ..... 290
UseOriginCacheControlHeaders ..... 290

UseOriginCacheControlHeaders-QueryStrings ... 291
캐시 키 이해 ... 292
기본 캐시 키 ... 293
캐시 키 사용자 지정 ... 294
정책을 통한 오리진 요청 제어 ... 296
오리진 요청 정책 이해 ... 297
정책 정보 ... 297
오리진 요청 설정 ... 297
오리진 요청 정책 생성 ... 299
관리형 오리진 요청 정책 사용 ... 303
AllViewer ... 303
AllViewerAndCloudFrontHeaders-2022-06 ... 304
AllViewerExceptHostHeader ... 305
CORS-CustomOrigin ... 306
CORS-S3Origin ... 306
Elemental-MediaTailor-PersonalizedManifests ... 307
HostHeaderOnly ... 307
UserAgentRefererHeaders ... 308
CloudFront 요청 헤더 추가 ... 308
디바이스 유형 헤더 ... 309
뷰어 위치 헤더 ... 310
뷰어의 헤더 구조를 결정하기 위한 헤더 ... 311
TLS 관련 헤더 ... 311
기타 CloudFront 헤더 ... 312
오리진 요청 정책과 캐시 정책이 함께 작동하는 방식 이해 ... 313
정책을 통해 응답 헤더 추가 또는 제거 ... 316
응답 헤더 정책 이해 ... 317
정책 세부 정보(메타데이터) ... 317
CORS 헤더 ... 318
보안 헤더 ... 321
사용자 지정 헤더 ... 323
헤더 제거 ... 324
Server-Timing 헤더 ... 326
응답 헤더 정책 생성 ... 330
관리형 응답 헤더 정책 사용 ... 336
CORS-and-SecurityHeadersPolicy ... 337

CORS-With-Preflight ... 337
CORS-with-preflight-and-SecurityHeadersPolicy ... 338
SecurityHeadersPolicy ... 339
SimpleCORS ... 340
요청 및 응답 동작 ... 342
CloudFront에서 HTTP 및 HTTPS 요청을 처리하는 방법 ... 342
Amazon S3 오리진에 대한 요청 및 응답 동작 ... 343
CloudFront에서 요청을 처리하고 Amazon S3 오리진에 요청을 전달하는 방법 ... 343
CloudFront에서 Amazon S3 오리진의 요청을 처리하는 방법 ... 349
사용자 지정 오리진에 대한 요청 및 응답 동작 ... 352
CloudFront에서 요청을 처리하고 사용자 지정 오리진에 요청을 전달하는 방법 ... 352
CloudFront에서 사용자 지정 오리진 서버의 요청을 처리하는 방법 ... 368
오리진 그룹에 대한 요청 및 응답 동작 ... 373
사용자 지정 헤더를 오리진 요청에 추가 ... 374
사용 사례 ... 374
사용자 지정 헤더를 오리진 요청에 추가하도록 CloudFront 구성 ... 375
CloudFront에서 오리진 요청에 추가할 수 없는 사용자 지정 헤더 ... 375
`Authorization` 헤더를 전달하도록 CloudFront 구성 ... 376
CloudFront에서 범위 GET을 처리하는 방법 ... 377
범위 요청을 사용하여 대형 객체 캐시 ... 378
CloudFront에서 오리진의 HTTP 3xx 상태 코드를 처리하는 방법 ... 378
CloudFront에서 오리진의 HTTP 4xx 및 5xx 상태 코드를 처리하는 방법 ... 379
사용자 지정 오류 페이지를 구성했을 때 CloudFront에서 오류를 처리하는 방법 ... 380
사용자 지정 오류 페이지를 구성하지 않은 경우 CloudFront에서 오류를 처리하는 방법 ... 382
CloudFront가 캐싱하는 HTTP 4xx 및 5xx 상태 코드 ... 384
사용자 지정 오류 응답 생성 ... 386
오류 응답 동작 구성 ... 386
특정 HTTP 상태 코드에 대한 사용자 지정 오류 페이지 생성 ... 388
다른 위치에 객체 및 사용자 지정 오류 페이지 저장 ... 390
CloudFront에서 반환하는 응답 코드 변경 ... 390
CloudFront에서 오류를 캐싱하는 기간 제어 ... 391
콘텐츠 추가, 제거 또는 교체 ... 393
콘텐츠 추가 및 액세스 ... 393
파일 버전 관리를 사용하여 기존 콘텐츠 업데이트 또는 제거 ... 394
버전이 지정된 파일 이름을 사용하여 기존 파일 업데이트 ... 394
CloudFront가 콘텐츠를 배포하지 않도록 콘텐츠 제거 ... 394

파일 URL 사용자 지정 ... 395
고유의 도메인 이름 사용(example.com) ... 396
URL에서 후행 슬래시(/) 사용 ... 396
제한된 콘텐츠에 대한 서명된 URL 생성 ... 396
기본 루트 객체 지정 ... 397
기본 루트 객체를 지정하는 방법 ... 397
기본 루트 객체 작동 방식 ... 398
루트 객체를 정의하지 않을 경우 CloudFront의 작동 방식 ... 400
파일을 무효화하여 콘텐츠 제거 ... 400
파일을 무효화하는 방법과 버전이 지정된 파일 이름을 사용하는 방법 중에 선택 ... 401
무효화할 파일 결정 ... 402
파일 무효화 시 알아야 할 사항 ... 402
파일 무효화 ... 406
동시 무효화 요청 최대값 ... 409
파일 무효화에 대한 요금 결제 ... 410
압축된 파일 제공 ... 410
객체를 압축하도록 CloudFront 구성 ... 411
CloudFront 압축의 작동 원리 ... 411
압축 조건 ... 412
CloudFront가 압축하는 파일 형식 ... 414
ETag 헤더 변환 ... 416
AWS WAF 보호 사용 ... 417
배포에 AWS WAF 활성화 ... 418
새 배포에 AWS WAF 활성화 ... 418
기존 웹 ACL 사용 ... 419
Bot Control 활성화 ... 419
봇 범주별 보호 구성 ... 420
CloudFront에서 AWS WAF 보안 보호 관리 ... 421
사전 조건 ... 422
AWS WAF 로그 활성화 ... 422
속도 제한 설정 ... 423
AWS WAF 보안 보호 비활성화 ... 424
보안 액세스 구성 및 콘텐츠에 대한 액세스 제한 ... 425
CloudFront에서 HTTPS 사용 ... 426
뷰어와 CloudFront 간에 HTTPS 요구 ... 427
사용자 지정 오리진에 대해 HTTPS 요구 ... 429

Amazon S3 오리진에 대해 HTTPS 요구 ... 432
최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호 ... 434
CloudFront와 오리진 간에 지원되는 프로토콜 및 암호 ... 442
대체 도메인 이름과 HTTPS 사용 ... 444
CloudFront에서 HTTPS 요청을 제공하는 방식 선택 ... 445
CloudFront에서 SSL/TLS 인증서를 사용하기 위한 요구 사항 ... 448
CloudFront에서 SSL/TLS 인증서 사용 시의 할당량(뷰어와 CloudFront 간의 HTTPS만 해당) ... 453
대체 도메인 이름과 HTTPS 구성 ... 455
SSL/TLS RSA 인증서에서 퍼블릭 키 크기 확인 ... 458
SSL/TLS 인증서에 대한 할당량 증가 ... 459
SSL/TLS 인증서 교체 ... 460
사용자 지정 SSL/TLS 인증서에서 기본 CloudFront 인증서로 되돌리기 ... 461
사용자 지정 SSL/TLS 인증서를 전용 IP 주소 사용에서 SNI 사용으로 전환 ... 462
CloudFront에 대한 상호 TLS 인증(뷰어 mTLS) ... 463
작동 방식 ... 463
사용 사례 ... 464
트러스트 스토어 및 인증서 관리 ... 464
CloudFront 배포에 상호 TLS 활성화 ... 471
CloudFront 연결 함수 연결 ... 474
추가 설정 구성 ... 480
오리진으로 전달된 캐시 정책에 대한 뷰어 mTLS 헤더 ... 482
CloudFront 연결 함수 및 KVS를 사용한 해지 ... 485
연결 로그를 사용한 관찰성 ... 489
CloudFront에 대한 상호 TLS(오리진) ... 494
뷰어 mTLS와 오리진 mTLS 비교 ... 494
작동 방식 ... 495
사용 사례 ... 495
중요: 오리진 서버 요구 사항 ... 496
시작하기 ... 496
AWS Certificate Manager를 사용한 인증서 관리 ... 496
CloudFront 배포에 상호 TLS(오리진) 활성화 ... 500
상호 TLS(오리진)와 함께 CloudFront Functions 사용 ... 503
서명된 URL과 서명된 쿠키를 사용하여 콘텐츠 제한 ... 505
프라이빗 콘텐츠를 제공하는 방법 ... 506
파일에 대한 액세스 제한 ... 507

신뢰할 수 있는 서명자 지정 ... 509
서명된 URL 또는 서명된 쿠키 사용 결정 ... 518
서명된 URL 사용 ... 519
서명된 쿠키 사용 ... 539
base64 인코딩 및 암호화를 위한 Linux 명령 및 OpenSSL ... 565
서명된 URL 코드 예제 ... 566
AWS 오리진에 대한 액세스 제한 ... 594
AWS Elemental MediaPackage v2 오리진에 대한 액세스 제한 ... 595
AWS Elemental MediaStore 오리진에 대한 액세스 제한 ... 602
AWS Lambda 함수 URL 오리진에 대한 액세스 제한 ... 609
Amazon S3 오리진에 대한 액세스 제한 ... 619
VPC 오리진으로 액세스 제한 ... 634
Application Load Balancer에 대한 액세스 제한 ... 641
사용자 지정 HTTP 헤더를 요청에 추가하도록 CloudFront 구성 ... 642
특정 헤더가 포함된 요청만 전달하도록 Application Load Balancer 구성 ... 644
(선택 사항) 이 솔루션의 보안 강화 ... 645
(선택 사항) CloudFront의 AWS-managed 접두사 목록을 사용하여 오리진에 대한 액세스를 제한합니다. ... 646
지리적 제한 ... 647
CloudFront 지리적 제한 사용 ... 647
서드 파티 지리적 위치 서비스 사용 ... 649
필드 수준 암호화를 사용하여 민감한 데이터 보호 ... 650
필드 레벨 암호화 개요 ... 652
필드 레벨 암호화 설정 ... 653
오리진의 데이터 필드 해독 ... 658
온디맨드 비디오 및 라이브 스트리밍 비디오 ... 662
스트리밍 비디오 정보 ... 662
온디맨드 비디오 제공 ... 663
Microsoft Smooth Streaming용 온디맨드 비디오 구성 ... 664
동영상 스트리밍 제공 ... 665
AWS Elemental MediaStore를 오리진으로 사용하여 비디오를 제공합니다. ... 667
AWS Elemental MediaPackage를 사용하여 포맷된 라이브 비디오 제공 ... 668
AWS Elemental MediaPackage로 동영상 온디맨드 콘텐츠 제공 ... 673
미디어 품질 인식 복원력 ... 678
MQAR 로그 필드 ... 680
함수를 사용하여 엣지에서 사용자 지정 ... 681

CloudFront Functions와 Lambda@Edge 간 차이점 ..... 682
CloudFront Functions를 사용하여 사용자 지정 ..... 684
튜토리얼: 간단한 CloudFront 함수 생성 ..... 685
튜토리얼: 키 값을 사용하는 CloudFront 함수 생성 ..... 688
함수 코드 작성 ..... 691
함수 생성 ..... 786
함수 테스트 ..... 789
함수 업데이트 ..... 794
함수 게시 ..... 796
배포에 함수 연결 ..... 797
CloudFront KeyValueStore ..... 801
CloudFront 연결 함수를 사용하여 사용자 지정 ..... 822
개요 및 워크플로 ..... 822
구성 및 제한 사항 ..... 824
상호 TLS(뷰어) 검증을 위한 CloudFront 연결 함수 생성 ..... 825
상호 TLS(뷰어) 검증을 위한 CloudFront 연결 함수 코드 쓰기 ..... 828
배포 전에 CloudFront 연결 함수 테스트 ..... 837
배포에 연결 함수 연결 ..... 838
CloudFront Functions 및 KeyValueStore를 사용하여 상호 TLS(뷰어)에 대한 인증서 해지 구현 ..... 839
Lambda@Edge를 사용하여 사용자 지정 ..... 846
Lambda@Edge가 요청 및 응답을 처리하는 방법 ..... 847
Lambda@Edge 사용 방법 ..... 847
Lambda@Edge 시작하기 ..... 848
IAM 권한 및 역할 설정 ..... 856
Lambda@Edge 함수 작성 및 생성 ..... 862
Lambda@Edge 함수에 대한 트리거 추가 ..... 866
테스트 및 디버깅 ..... 873
함수 및 복제본 삭제 ..... 879
이벤트 구조 ..... 880
요청 및 응답 작업 수행 ..... 897
예제 함수 ..... 902
엣지 함수에 대한 제한 사항 ..... 941
모든 엣지 함수에 대한 제한 사항 ..... 941
CloudFront 함수에 대한 제한 ..... 947
Lambda@Edge에 대한 제한 사항 ..... 949

보고서, 지표 및 로그 ..... 954
CloudFront에 대한 AWS 결제 및 사용 보고서 ..... 954
CloudFront에 대한 AWS 결제 보고서 ..... 955
CloudFront에 대한 AWS 사용 보고서 보기 ..... 956
CloudFront에 대한 AWS 청구서 및 사용 보고서 해석 ..... 957
CloudFront 콘솔 보고서 보기 ..... 963
CloudFront 캐시 통계 보고서 보기 ..... 964
CloudFront 인기 객체 보고서 보기 ..... 970
CloudFront 상위 참조자 보고서 ..... 975
CloudFront 사용 보고서 보기 ..... 978
CloudFront 최종 사용자 보고서 보기 ..... 986
Amazon CloudWatch를 사용한 CloudFront 지표 모니터링 ..... 996
CloudFront 및 엣지 함수 지표 보기 ..... 997
경보 생성 ..... 1004
지표 데이터 다운로드 ..... 1005
CloudFront 지표 ..... 1008
CloudFront 및 엣지 함수 로깅 ..... 1014
요청 로깅 ..... 1014
엣지 함수 로깅 ..... 1015
서비스 활동 로깅 ..... 1015
액세스 로그(표준 로그) ..... 1016
실시간 액세스 로그 사용 ..... 1060
엣지 함수 로그 ..... 1080
AWS CloudTrail 로그 ..... 1084
AWS Config를 사용하여 구성 변경 추적 ..... 1096
CloudFront로 AWS Config 설정 ..... 1097
CloudFront 구성 기록 보기 ..... 1098
AWS Config 규칙을 사용하여 CloudFront 구성 평가 ..... 1099
보안 ..... 1100
데이터 보호 ..... 1100
전송 중 데이터 암호화 ..... 1102
저장된 암호화 ..... 1102
콘텐츠에 대한 액세스 제한 ..... 1103
자격 증명 및 액세스 관리 ..... 1104
고객 ..... 1104
ID를 통한 인증 ..... 1104

정책을 사용하여 액세스 관리 ..... 1106
Amazon CloudFront와 함께 IAM을 사용하는 방법 ..... 1107
ID 기반 정책 예시 ..... 1112
AWS 관리형 정책 ..... 1123
서비스 연결 역할 사용 ..... 1131
CloudFront 자격 증명 및 액세스 문제 해결 ..... 1135
로깅 및 모니터링 ..... 1137
규정 준수 확인 ..... 1138
CloudFront 규정 준수 모범 사례 ..... 1139
복원력 ..... 1140
CloudFront 오리진 장애 조치 ..... 1140
인프라 보안 ..... 1141
문제 해결 ..... 1142
배포 문제 해결 ..... 1142
CloudFront에서 Access Denied 오류가 반환되는 경우 ..... 1142
대체 도메인 이름을 추가하려고 하면 CloudFront에서 InvalidViewerCertificate 오류가 반환됩니다. ..... 1144
새 CNAME을 추가하려고 하면 CloudFront에서 잘못 구성된 DNS 레코드 오류가 반환됩니다. ..... 1146
배포에서 파일을 볼 수 없습니다. ..... 1146
오류 메시지: Certificate: <certificate-id> is being used by CloudFront ..... 1148
오류 응답 상태 코드 문제 해결 ..... 1148
HTTP 400 상태 코드(잘못된 요청) ..... 1149
HTTP 401 상태 코드(승인되지 않음) ..... 1150
HTTP 403 상태 코드(잘못된 메서드) ..... 1151
HTTP 403 상태 코드(권한 거부됨) ..... 1151
HTTP 404 상태 코드(찾을 수 없음) ..... 1153
HTTP 412 상태 코드(사전 조건 실패) ..... 1154
HTTP 500 상태 코드(내부 서버 오류) ..... 1154
HTTP 502 상태 코드(잘못된 게이트웨이) ..... 1155
HTTP 503 상태 코드(Service Unavailable) ..... 1159
HTTP 504 상태 코드(게이트웨이 제한 시간) ..... 1162
CloudFront 로드 테스트 ..... 1167
할당량 ..... 1168
일반 할당량 ..... 1169
배포의 일반 할당량 ..... 1169

정책에 대한 일반 할당량 ..... 1172
mTLS 및 트러스트 스토어의 할당량 ..... 1173
CloudFront 함수의 할당량 ..... 1174
연결 함수에 대한 할당량 ..... 1174
키 값 저장소의 할당량 ..... 1175
Lambda@Edge에 대한 할당량 ..... 1175
SSL 인증서의 할당량 ..... 1177
무효화에 대한 할당량 ..... 1178
키 그룹에 대한 할당량 ..... 1178
WebSocket 연결에 대한 할당량 ..... 1179
필드 레벨 암호화에 대한 할당량 ..... 1179
쿠키에 대한 할당량(레거시 캐시 설정) ..... 1180
쿼리 문자열에 대한 할당량(레거시 캐시 설정) ..... 1180
헤더에 대한 할당량 ..... 1181
다중 테넌트 배포의 할당량 ..... 1182
관련 정보 ..... 1183
코드 예제 ..... 1184
기본 사항 ..... 1185
작업 ..... 1186
시나리오 ..... 1254
다중 테넌트 배포 및 배포 테넌트 생성 ..... 1255
서명 리소스 삭제 ..... 1266
CloudFront 시작하기 ..... 1268
URL 및 쿠키에 서명 ..... 1276
CloudFront Functions 예제 ..... 1280
HTTP 보안 헤더 추가 ..... 1280
CORS 헤더 추가 ..... 1281
캐시 제어 헤더 추가 ..... 1282
실제 클라이언트 IP 헤더 추가 ..... 1283
오리진 헤더 추가 ..... 1284
요청 URL에 index.html 추가 ..... 1285
쿼리 문자열 파라미터 정규화 ..... 1286
새 URL로 리디렉션 ..... 1287
요청 URI 재작성 ..... 1288
뷰어에 더 가까운 오리진 선택 ..... 1290
키 값 페어 사용 ..... 1292

단순 토큰 검증 ..... 1293
문서 기록 ..... 1298

# Amazon CloudFront란 무엇입니까?

Amazon CloudFront는 .html, .css, .js 및 이미지 파일과 같은 정적 및 동적 웹 콘텐츠를 사용자에게 더 빨리 배포하도록 지원하는 웹 서비스입니다. CloudFront는 엣지 로케이션이라고 하는 데이터 센터의 전 세계 네트워크를 통해 콘텐츠를 제공합니다. CloudFront를 통해 서비스하는 콘텐츠를 사용자가 요청하면 지연 시간이 가장 낮은 엣지 로케이션으로 요청이 라우팅되므로 가능한 최고의 성능으로 콘텐츠가 제공됩니다.

- 콘텐츠가 이미 지연 시간이 가장 낮은 엣지 로케이션에 있는 경우 CloudFront가 콘텐츠를 즉시 제공합니다.
- 콘텐츠가 엣지 로케이션에 없는 경우 CloudFront는 콘텐츠의 최종 버전에 대한 소스로 지정된 오리진(Amazon S3 버킷, MediaPackage 채널, HTTP 서버(예: 웹 서버) 등)에서 콘텐츠를 검색합니다.

예를 들어, CloudFront가 아닌 일반적인 웹 서버에서 이미지를 제공한다고 가정합니다. 예를 들어 `https://example.com/sunsetphoto.png` URL을 사용하여 sunsetphoto.png라는 이미지를 서비스할 수 있습니다.

사용자는 이 URL로 쉽게 이동해 해당 이미지를 볼 수 있습니다. 하지만 이미지가 발견될 때까지 인터넷으로 이루어진 상호 연결된 네트워크의 복잡한 모음을 통해 네트워크에서 다른 네트워크로 요청이 라우팅되었다는 사실은 아마도 모르고 있을 것입니다.

CloudFront는 AWS 백본 네트워크를 통해 콘텐츠를 가장 효과적으로 서비스할 수 있는 엣지로 각 사용자 요청을 라우팅하여 콘텐츠 배포 속도를 높입니다. 일반적으로 CloudFront 엣지가 최종 사용자에게 가장 빨리 제공합니다. AWS 네트워크를 사용하면 사용자의 요청이 반드시 통과해야 하는 네트워크의 수가 줄어들어 성능이 향상됩니다. 파일의 첫 바이트를 로드하는 데 걸리는 지연 시간이 줄어들고 데이터 전송 속도가 빨라집니다.

또한 파일(객체라고도 함)의 사본이 전 세계 여러 엣지 로케이션에 유지(또는 캐시)되므로 안정성과 가용성이 향상됩니다.

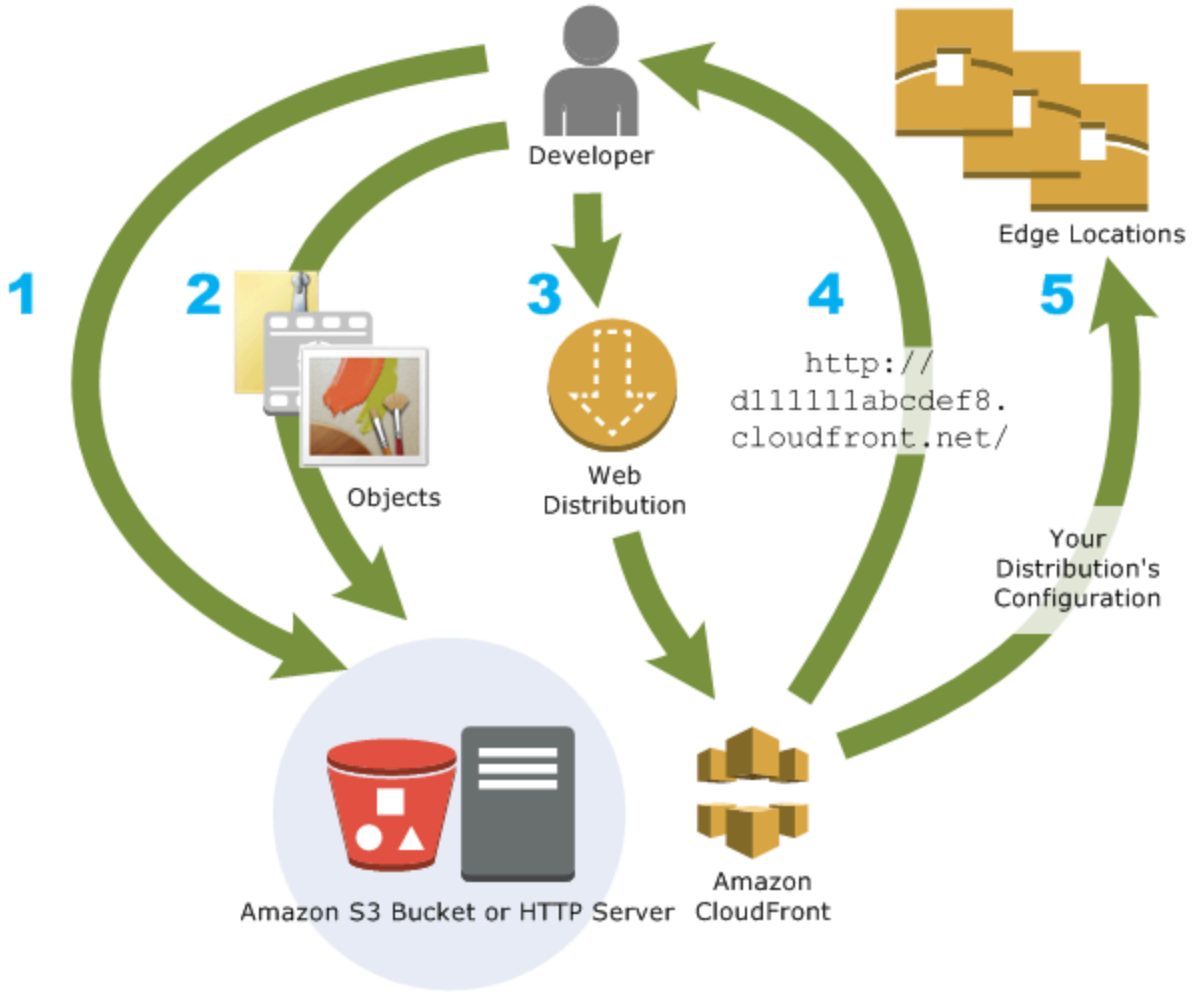
주제


- 콘텐츠를 전송하도록 CloudFront를 설정하는 방법
- 표준 배포 또는 다중 테넌트 배포 중에서 선택
- 요금
- CloudFront 사용 방법
- CloudFront에서 콘텐츠를 제공하는 방법

- CloudFront 엣지 서버의 위치 및 IP 주소 범위
- AWS SDK와 함께 CloudFront 사용
- CloudFront 기술 리소스

# 콘텐츠를 전송하도록 CloudFront를 설정하는 방법

CloudFront 배포를 생성하여 CloudFront에 어디로부터 콘텐츠를 전송하고자 하는지와 이러한 콘텐츠 전송을 추적 및 관리하는 방법에 대한 세부 정보를 알립니다. 그런 다음 CloudFront는 최종 사용자와 인접한 컴퓨터(엣지 서버)를 사용하여 사용자가 콘텐츠를 사용하거나 보고자 할 때 빠르게 전송합니다.




콘텐츠를 전송하도록 CloudFront를 구성하는 방법

1. Amazon S3 버킷 또는 고유 HTTP 서버와 같은 오리진 서버를 지정하고, CloudFront는 이로부터 파일을 가져온 다음 전 세계 CloudFront 엣지 로케이션에 배포합니다.

오리진 서버는 객체의 최종 원본 버전을 저장합니다. HTTP를 통해 콘텐츠를 서비스하는 경우 오리진 서버가 Amazon S3 버킷 또는 웹 서버 같은 HTTP 서버입니다. HTTP 서버는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스나 사용자가 관리하는 서버에서 실행할 수 있습니다. 이 서버를 사용자 지정 오리진이라고도 합니다.

2. 오리진 서버에 파일을 업로드합니다. 객체라고도 하는 파일은 일반적으로 웹 페이지, 이미지 및 미디어 파일을 포함하지만 HTTP를 통해 제공될 수 있는 모든 항목이 될 수 있습니다.

   Amazon S3 버킷을 오리진 서버로 사용할 경우 버킷에 있는 객체를 공개적으로 읽을 수 있는 상태로 만들 수 있으므로 객체의 CloudFront URL을 아는 사람이라면 누구나 액세스할 수 있습니다. 객체를 비공개로 유지하고 액세스할 수 있는 사용자를 제어할 수 있는 옵션도 있습니다. 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공을(를) 참조하세요.

3. 사용자가 웹 사이트나 애플리케이션을 통해 파일을 요청할 경우 &CF;에 어떤 오리진 서버에서 파일을 가져올지 알려 주는 CloudFront 배포를 만듭니다. 동시에 CloudFront에서 모든 요청을 기록할지, 배포를 만들자마자 활성화할지 여부와 같은 세부 사항을 지정합니다.

4. CloudFront는 새 배포에 도메인 이름을 할당하고, 이는 CloudFront 콘솔에서 볼 수 있습니다. 또는 API 요청 등과 같은 프로그램 요청에 대한 응답으로 반환됩니다. 원하는 경우 대신 사용할 대체 도메인 이름을 추가할 수 있습니다.

5. CloudFront에서는 배포의 구성(사용자의 콘텐츠가 아님)을 모든 해당 엣지 로케이션 또는 CloudFront가 파일의 사본을 캐싱하는 지리적으로 분산된 데이터 센터의 POP(Point of Presence) 서버 모음으로 보냅니다.

웹 사이트 또는 애플리케이션을 개발할 경우 CloudFront가 URL에 제공하는 도메인 이름을 사용합니다. 예를 들어, CloudFront가 `d111111abcdef8.cloudfront.net`을 배포의 도메인 이름으로 반환할 경우 Amazon S3 버킷(또는 HTTP 서버의 루트 디렉터리)에 있는 logo.jpg의 URL이 `https://d111111abcdef8.cloudfront.net/logo.jpg`가 됩니다.

또는 CloudFront를 설정하여 고유한 도메인 이름을 배포와 사용할 수도 있습니다. 이 경우 URL이 `https://www.example.com/logo.jpg`가 될 수 있습니다.

또는 파일에 머리글을 추가하도록 오리진 서버를 구성할 수 있습니다. 이는 CloudFront 엣지 로케이션의 캐시에 파일을 얼마나 오래 보관할지 나타냅니다. 기본적으로 각 파일은 만료되기 전에 24시간 동안 엣지 로케이션에 남아 있습니다. 최소 만료 시간은 0초이며, 최대 만료 시간은 없습니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 섹션을 참조하세요.

# 표준 배포 또는 다중 테넌트 배포 중에서 선택

CloudFront는 단일 웹사이트 또는 앱과 다중 테넌트 시나리오에 대한 배포 옵션을 제공합니다.

표준 배포

웹사이트 또는 애플리케이션별로 고유한 구성을 위해 설계되었습니다. 이는 다음과 같은 사용 사례에서 선택합니다.

- 독립 실행형 CloudFront 배포가 필요함
- 각 사이트 또는 애플리케이션에 자체 사용자 지정 설정이 필요함

대부분의 사람들은 표준 배포로 시작합니다.

다중 테넌트 배포 및 배포 테넌트(CloudFront SaaS Manager)

SaaS 공급자 및 다중 테넌트 시나리오를 위해 특별히 설계되었습니다. 이는 다음과 같은 사용 사례에서 선택합니다.

- 여러 고객 웹사이트 또는 애플리케이션에 서비스를 제공하기 위해 SaaS 플랫폼을 구축하고 있음
- 여러 유사한 배포를 효율적으로 관리해야 함
- 공유 구성을 중앙 집중식으로 제어하려고 함

자세한 내용은 다중 테넌트 배포의 작동 방식 이해 섹션을 참조하세요.

# 요금

CloudFront는 엣지 로케이션에서 전송되는 데이터에 대해 HTTP 또는 HTTPS 요청과 함께 요금을 부과합니다. 요금은 사용 유형, 리전 및 기능 선택에 따라 다릅니다.

Amazon Simple Storage Service (S3), Elastic Load Balancing, Amazon API Gateway와 같은 AWS 오리진을 사용하면 오리진에서 CloudFront로 데이터를 전송할 때 항상 무료입니다. AWS 오리진을 사용할 때 CloudFront에서 최종 사용자로의 아웃바운드 데이터 전송에 대한 요금만 청구됩니다.

자세한 내용은 CloudFront 요금 및 청구 및 비용 절감 번들 FAQ를 참조합니다.

# CloudFront 사용 방법

CloudFront를 사용하면 다양한 목표를 달성할 수 있습니다. 이 단원에서는 이에 대한 정보를 몇 가지 설명하고 추가 정보에 대한 링크를 제공합니다.

주제


- 정적 웹 사이트 콘텐츠 전송 속도 향상
- 온디맨드 비디오 또는 라이브 스트리밍 비디오 제공
- 시스템 처리를 통해 특정 필드 암호화
- 엣지에서의 사용자 지정
- Lambda@Edge 사용자 지정을 사용하여 프라이빗 콘텐츠 제공


## 정적 웹 사이트 콘텐츠 전송 속도 향상

CloudFront는 전 세계 최종 사용자에게 제공되는 정적 콘텐츠(이미지, 스타일 시트, JavaScript 등)의 전송 속도를 높일 수 있습니다. CloudFront를 사용하면 AWS 백본 네트워크와 CloudFront 엣지 서버의 장점을 활용하여 해당 웹 사이트를 방문하는 뷰어에게 빠르고 안전하며 신뢰할 수 있는 환경을 제공할 수 있습니다.

정적 콘텐츠를 저장하고 전송하기 위한 간단한 방법은 Amazon S3 버킷을 사용하는 것입니다. S3와 CloudFront를 함께 사용하면 오리진 액세스 제어를 통해 Amazon S3 콘텐츠에 대한 액세스를 쉽게 제한할 수 있으며 이외에도 다양한 장점이 있습니다.

빠른 시작을 도와주는 CloudFormation 템플릿을 비롯해 CloudFront와 Amazon S3를 함께 사용하는 방법에 대한 자세한 내용은 안전한 정적 웹 사이트 시작하기 섹션을 참조하세요.

## 온디맨드 비디오 또는 라이브 스트리밍 비디오 제공

CloudFront는 전 세계 최종 사용자에게 미디어(녹화 파일 및 라이브 이벤트)를 스트리밍하기 위한 몇 가지 옵션을 제공합니다.

- 온디맨드 비디오(VOD) 스트리밍의 경우 CloudFront를 사용하면 MPEG DASH, Apple HLS, Microsoft Smooth Streaming, CMAF 등과 같은 일반적인 포맷으로 디바이스에 상관없이 스트리밍할 수 있습니다.
- 라이브 스트림 방송의 경우에는, 엣지에 미디어 조각을 캐싱하여 해당 조각을 올바른 순서로 전송하는 매니페스트 파일에 대한 여러 요청을 결합함으로써 오리진 서버의 부하를 줄일 수 있습니다.

CloudFront를 사용하여 스트리밍 콘텐츠를 전송하는 방법에 대한 자세한 내용은 CloudFront를 사용한 온디맨드 비디오 및 라이브 스트리밍 비디오 단원을 참조하십시오.

## 시스템 처리를 통해 특정 필드 암호화

CloudFront에서 HTTPS를 구성할 경우 오리진 서버에 대한 종단 간 연결의 보안이 보장됩니다. 파일 수준 암호화를 추가하는 경우, HTTPS 보안뿐 아니라 시스템 처리 전체에서 특정 데이터를 보호함으로써 오리진의 특정 애플리케이션만 데이터를 보게 할 수 있습니다.

필드 레벨 암호화를 설정하려면 CloudFront에 퍼블릭 키를 추가한 후 이 키를 사용하여 암호화하려는 필드 세트를 지정합니다. 자세한 내용은 필드 수준 암호화를 사용하여 민감한 데이터 보호 단원을 참조하세요.

## 엣지에서의 사용자 지정

엣지에서 서버리스 코드를 실행하면 지연 시간을 최소화하면서 다양한 방식으로 최종 사용자에 대한 콘텐츠와 환경을 사용자 지정할 수 있습니다. 예를 들어 오리진 서버가 유지보수를 위해 다운된 경우 최종 사용자에게 일반 HTTP 오류 메시지를 표시하는 대신 사용자 지정 오류 메시지를 표시할 수 있습니다. 또는 함수를 사용하여 CloudFront가 오리진으로 요청을 전송하기 전에 사용자를 인증하고 콘텐츠에 대한 액세스를 제어할 수 있습니다.

CloudFront에서 LLambda@Edge를 사용하면 CloudFront가 제공하는 콘텐츠를 다양한 방법으로 사용자 지정할 수 있습니다. Lambda@Edge에 대한 자세한 내용과 CloudFront를 사용하여 함수를 생성하고 배포하는 방법을 알아보려면 Lambda@Edge를 사용하여 엣지에서 사용자 지정 단원을 참조하십시오. 해당 솔루션에 맞게 사용자 지정할 수 있는 다양한 코드 샘플을 보려면 Lambda@Edge 예제 함수 단원을 참조하십시오.

## Lambda@Edge 사용자 지정을 사용하여 프라이빗 콘텐츠 제공

Lambda@Edge를 사용하면 서명된 URL 또는 서명된 쿠키를 사용하는 것 이외에, 자체 사용자 지정 오리진에서 프라이빗 콘텐츠를 제공하도록 CloudFront 배포를 구성할 수 있습니다.

CloudFront를 통해 프라이빗 콘텐츠를 제공하려면 다음과 같이 합니다.

- 최종 사용자가 서명된 URL 또는 서명된 쿠키를 사용하여 콘텐츠에 액세스하도록 요구합니다.
- CloudFront의 오리진 대상 서버에서만 사용할 수 있도록 오리진에 대한 액세스를 제한합니다. 이렇게 하려면 다음 중 하나를 수행합니다.
  - Amazon S3 오리진의 경우 오리진 액세스 제어(OAC)를 사용할 수 있습니다.
  - 사용자 지정 오리진의 경우 다음과 같이 할 수 있습니다.
    - 사용자 지정 오리진이 Amazon VPC 보안 그룹 또는 AWS Firewall Manager에 의해 보호되는 경우 CloudFront 관리형 접두사 목록을 사용하여 CloudFront의 오리진 대상 IP 주소에서만 오리진으로의 인바운드 트래픽을 허용할 수 있습니다.

- 사용자 지정 HTTP 헤더를 사용하여 CloudFront의 요청에 대해서만 액세스를 제한합니다. 자세한 내용은 the section called "사용자 지정 오리진의 파일에 대한 액세스 제한" 및 the section called "사용자 지정 헤더를 오리진 요청에 추가" 섹션을 참조하세요. 사용자 지정 헤더를 사용하여 Application Load Balancer 오리진에 대한 액세스를 제한하는 예제는 the section called "Application Load Balancer에 대한 액세스 제한" 단원을 참조하세요.
- 사용자 지정 오리진에 사용자 지정 액세스 제어 로직이 필요한 경우 Lambda@Edge를 사용하여 블로그 게시물 Amazon CloudFront 및 Lambda@Edge를 사용한 프라이빗 콘텐츠 제공에 설명된 대로 해당 로직을 구현할 수 있습니다.

# CloudFront에서 콘텐츠를 제공하는 방법

초기 설정 후에 CloudFront는 해당 웹 사이트나 애플리케이션과 상호 작용하여 콘텐츠 전송 속도를 높입니다. 이 단원에서는 최종 사용자가 콘텐츠를 요청할 때 CloudFront가 콘텐츠를 어떻게 제공하는지를 설명합니다.

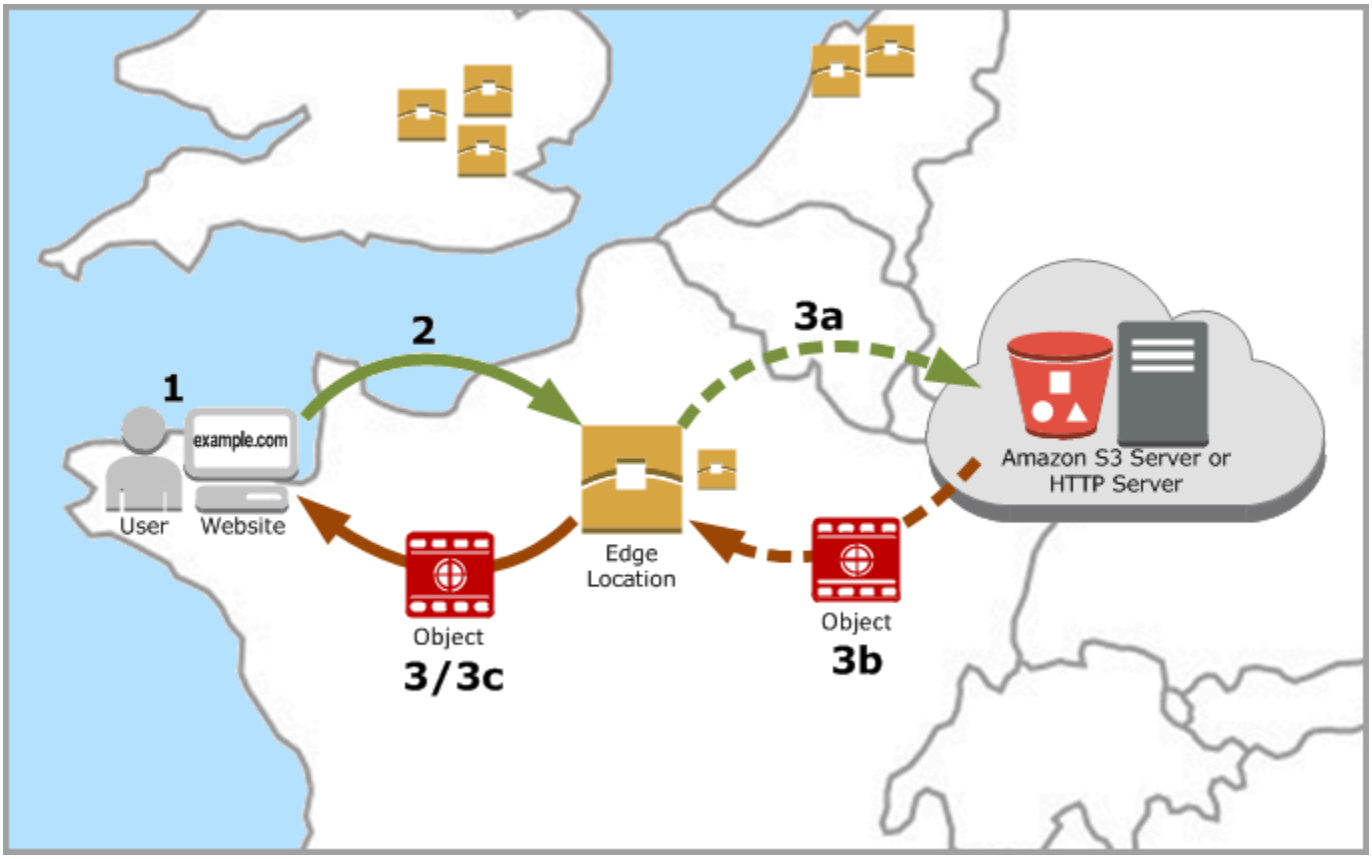
주제


- CloudFront에서 사용자에게 콘텐츠를 제공하는 방법
- CloudFront에서 리전 에지 캐시를 사용하는 방식


## CloudFront에서 사용자에게 콘텐츠를 제공하는 방법

콘텐츠를 제공하도록 CloudFront를 구성한 후에 사용자가 객체를 요청하면 다음과 같은 결과를 얻습니다.

1. 사용자가 웹 사이트 또는 애플리케이션에 액세스하고 이미지 파일 및 HTML 파일 같은 하나 이상의 객체에 대한 요청을 보냅니다.
2. DNS는 요청을 최적으로 처리할 수 있는 CloudFront POP(엣지 로케이션)로 라우팅합니다. 일반적으로는 지연 시간 관점에서 가장 가까운 CloudFront POP입니다.
3. CloudFront는 해당 캐시에 요청된 객체가 있는지 확인합니다. 객체가 캐시에 있으면 CloudFront는 객체를 사용자에게 반환합니다. 객체가 캐시에 없으면 CloudFront가 다음을 수행합니다.
   a. CloudFront는 배포의 사양과 요청을 비교하고 요청을 해당하는 객체로 사용자의 원본 서버(예: Amazon S3 버킷 또는 HTTP 서버)에 전달합니다.
   b. 원본 서버는 객체를 다시 엣지 로케이션으로 보냅니다.
   c. 오리진에서 첫 번째 바이트가 도착하면 CloudFront가 객체를 사용자에게 전달하기 시작합니다. CloudFront는 다음에 다른 사용자가 객체를 요청할 때 캐시에 해당 객체를 추가합니다.




## CloudFront에서 리전 에지 캐시를 사용하는 방식

CloudFront 상호 접속 위치(POP 또는 엣지 로케이션이라고도 함)는 많이 사용되는 콘텐츠를 최종 사용자에게 빠르게 제공할 수 있도록 합니다. CloudFront에는 또한 리전 엣지 캐시가 있어 POP에 있을 정도로 많이 사용되는 콘텐츠가 아닌 경우에도 최종 사용자가 보다 많은 콘텐츠를 가까이에서 액세스할 수 있도록 하여 콘텐츠의 성능을 개선하는 데 도움을 줍니다.

리전 엣지 캐시는 모든 유형의 콘텐츠, 특히 시간이 지나면서 점차 사용되지 않게 되는 콘텐츠에 유용합니다. 이러한 콘텐츠의 예로는 동영상, 사진 또는 아트워크와 같은 사용자 생성 콘텐츠, 제품 사진 및 동영상과 같은 전자 상거래 자산, 갑자기 사용자가 많아질 수 있는 뉴스 및 이벤트 관련 콘텐츠 등이 있습니다.

### 리전 캐시 작동 방식

리전 엣지 캐시는 최종 사용자가 쉽게 접할 수 있도록 전역으로 배포되는 CloudFront 위치입니다. 최종 사용자에게 콘텐츠를 직접 제공하는 전역 엣지 로케이션인 POP와 원본 서버 사이에 위치합니다. 객체의 사용도가 떨어지면 개별 POP가 이러한 객체를 제거하여 보다 많이 사용되는 콘텐츠가 해당 공간을 사용하도록 합니다. 리전 엣지 캐시는 개별 POP보다 캐시가 더 큽니다. 따라서 객체가 가장 가까운 리전 엣지 캐시 위치에 더 오래 캐시 상태로 유지됩니다. 따라서 더 많은 콘텐츠를 최종 사용자와 가까운

거리가 유지할 수 있으므로 CloudFront가 오리진 서버로 돌아가야 할 필요성이 줄어들고 최종 사용자를 위한 전반적인 성능도 향상됩니다.

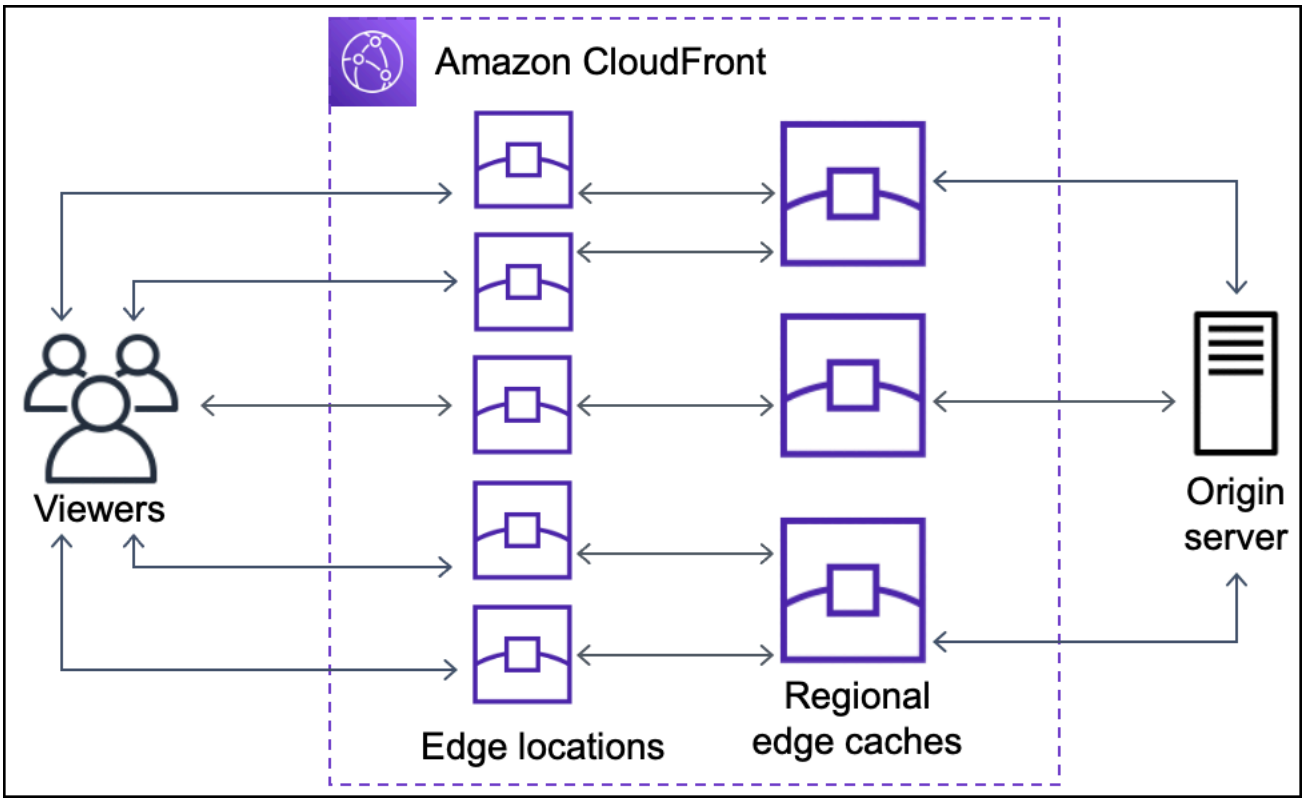
최종 사용자가 웹 사이트나 애플리케이션을 통해 요청하면, DNS는 이 요청을 사용자의 요청을 가장 잘 처리할 수 있는 POP로 라우팅합니다. 일반적으로 지연 시간을 기준으로 가장 가까운 CloudFront 엣지 로케이션이 여기에 해당합니다. POP에서 CloudFront는 해당 캐시에 요청된 객체가 있는지 확인합니다. 객체가 캐시에 있으면 CloudFront는 객체를 사용자에게 반환합니다. 객체가 캐시에 없으면, 일반적으로 POP가 가장 가까운 리전 엣지 캐시로 이동하여 객체를 가져옵니다. POP가 리전 엣지 캐시를 건너뛰고 오리진으로 직접 이동하는 경우에 대한 자세한 내용은 다음 참고 사항을 참조하세요.

리전 엣지 캐시 위치에서 CloudFront는 해당 캐시에 요청된 객체가 있는지 다시 확인합니다. 객체가 캐시에 있으면 CloudFront가 객체를 요청한 POP에 전달합니다. 리전 엣지 캐시 위치에서 첫 번째 바이트가 도착하면 CloudFront가 객체를 사용자에게 전달하기 시작합니다. CloudFront는 다음에 다른 사용자가 객체를 요청할 때 POP에서 캐시에 해당 객체를 추가합니다.

객체가 POP 또는 리전 엣지 캐시 위치에서 캐시되지 않는 경우, CloudFront는 배포의 사양과 요청을 비교하고 요청을 원본 서버로 전달합니다. 원본 서버가 객체를 리전 엣지 캐시 위치로 다시 보낸 이후, 해당 객체가 POP에 전달되고, CloudFront가 파일을 사용자에게 전달합니다. 이 경우 CloudFront는 다음 번에 최종 사용자가 해당 객체를 요청할 때 POP 뿐만 아니라 리전 엣지 캐시 위치의 캐시에도 객체를 추가합니다. 이를 통해 리전의 모든 POP에서 로컬 캐시를 공유하도록 하고, 오리진 서버에 요청이 여러 번 제출되지 않습니다. CloudFront는 또한 원본 서버와의 연결을 계속 유지하므로 최대한 빨리 오리진으로부터 해당 객체를 가져올 수 있습니다.

> **Note**
>
> - 리전 엣지 캐시는 POP에 기능 패리티를 둡니다. 예를 들어, 캐시 무효화 요청이 들어오면 객체가 만료되기 전에 POP 캐시 및 리전 엣지 캐시 모두에서 객체가 제거됩니다. 다음에 최종 사용자가 객체를 요청할 때, CloudFront에서는 최신 버전의 객체를 가져오도록 오리진에 반환합니다.
> - 프록시 HTTP 메서드(`PUT`, `POST`, `PATCH`, `OPTIONS`, `DELETE`)는 POP에서 오리진으로 바로 이동하며 리전 엣지 캐시를 통해 프록시되지 않습니다.
> - 요청 시 결정된 동적 요청은 리전 엣지 캐시를 통과하지 않고 오리진으로 직접 이동합니다.
> - 오리진이 Amazon S3 버킷이고 요청의 최적 리전 엣지 캐시가 S3 버킷과 동일한 AWS 리전에 있는 경우, POP가 리전 엣지 캐시를 건너뛰고 S3 버킷으로 직접 이동합니다.

다음 다이어그램에서는 CloudFront 엣지 로케이션 및 리전 엣지 캐시를 통해 요청 및 응답이 어떻게 전달되는지 보여줍니다.




## CloudFront 엣지 서버의 위치 및 IP 주소 범위

CloudFront 엣지 서버의 위치 목록은 Amazon CloudFront 글로벌 엣지 네트워크 페이지를 참조하십시오.

Amazon Web Services(AWS)는 현재 IP 주소 범위를 JSON 형식으로 게시합니다. 현재 범위를 보려면 ip-ranges.json을 다운로드합니다. 자세한 내용은 Amazon Web Services 일반 참조의 AWS IP 주소 범위 단원을 참조하세요.

CloudFront 엣지 서버와 연결된 IP 주소 범위를 찾으려면 ip-ranges.json에서 다음 문자열을 검색합니다.

```
"region": "GLOBAL",
"service": "CLOUDFRONT"
```

또는 https://d7uri8nf7uskq.cloudfront.net/tools/list-cloudfront-ips에서 CloudFront IP 범위만 볼 수 있습니다.

## CloudFront 관리형 접두사 목록 사용

CloudFront 관리형 접두사 목록은 전 세계적으로 배포된 CloudFront의 모든 원본 서버의 IP 주소 범위를 포함합니다. 원본이 AWS에 호스트되고 Amazon VPC 보안 그룹에서 보호되는 경우 CloudFront 관리형 접두사 목록을 사용하여 CloudFront의 원본 서버에서 원본으로의 인바운드 트래픽만 허용하여 CloudFront 이외의 트래픽이 원본에 도달하지 못하도록 차단할 수 있습니다. CloudFront가 관리형 접두사 목록을 유지 관리하기 때문에 CloudFront의 모든 글로벌 원본 서버의 IP 주소를 언제나 최신 상태로 유지합니다. CloudFront 관리형 접두사 목록을 사용할 경우, IP 주소 범위 목록을 직접 읽거나 유지 관리할 필요가 없습니다.

예를 들어 오리진이 유럽(런던) 리전(`eu-west-2`)의 Amazon EC2 인스턴스라고 가정합니다. 인스턴스가 VPC에 있는 경우 CloudFront 관리형 접두사 목록에서 인바운드 HTTPS 액세스를 허용하는 보안 그룹 규칙을 만들 수 있습니다. 그 결과 CloudFront의 모든 글로벌 원본 서버가 인스턴스에 도달할 수 있습니다. 보안 그룹에서 나머지 인바운드 규칙을 모두 제거할 경우 CloudFront가 아닌 트래픽이 인스턴스에 도달하는 것을 차단할 수 있습니다.

CloudFront 관리형 접두사 목록은 다음과 같습니다.

- `com.amazonaws.global.cloudfront.origin-facing`(IPv4)
- `com.amazonaws.global.ipv6.cloudfront.origin-facing`(IPv6)

자세한 내용은 https://docs.aws.amazon.com/vpc/latest/userguide/working-with-aws-managed-prefix-lists.html#use-aws-managed-prefix-listAmazon VPC 사용 설명서의 AWS 관리형 접두사 목록 사용을 참조하십시오.

> **Important**
>
> CloudFront 관리형 접두사 목록은 고유한 방식으로 Amazon VPC 할당량에 적용됩니다. 자세한 내용은 Amazon VPC 사용 설명서의 AWS 관리형 접두사 목록 가중치를 참조하십시오.

## AWS SDK와 함께 CloudFront 사용

다양한 프로그래밍 언어에 대해 AWS 소프트웨어 개발 키트(SDK)을 사용할 수 있습니다. 각 SDK는 개발자가 선호하는 언어로 애플리케이션을 쉽게 구축할 수 있도록 하는 API, 코드 예제 및 설명서를 제공합니다.

| SDK 설명서 | 코드 예시 |
| --- | --- |
| AWS SDK for C++ | AWS SDK for C++ 코드 예시 |
| AWS CLI | AWS CLI 코드 예시 |
| AWS SDK for Go | AWS SDK for Go 코드 예시 |
| AWS SDK for Java | AWS SDK for Java 코드 예시 |
| AWS SDK for JavaScript | AWS SDK for JavaScript 코드 예시 |
| AWS SDK for Kotlin | AWS SDK for Kotlin 코드 예시 |
| AWS SDK for .NET | AWS SDK for .NET 코드 예시 |
| AWS SDK for PHP | AWS SDK for PHP 코드 예시 |
| AWS Tools for PowerShell | AWS Tools for PowerShell 코드 예시 |
| AWS SDK for Python (Boto3) | AWS SDK for Python (Boto3) 코드 예시 |
| AWS SDK for Ruby | AWS SDK for Ruby 코드 예시 |
| AWS SDK for Rust | AWS SDK for Rust 코드 예시 |
| AWS SDK for SAP ABAP | AWS SDK for SAP ABAP 코드 예시 |
| AWS SDK for Swift | AWS SDK for Swift 코드 예시 |

가용성 예제

필요한 예제를 찾을 수 없습니까? 이 페이지 하단의 피드백 제공 링크를 사용하여 코드 예시를 요청하세요.

## CloudFront 기술 리소스

다음 리소스를 사용하여 CloudFront에 대한 기술적인 질문에 대한 답변을 얻으세요.

- AWS re:Post – 개발자가 CloudFront와 관련된 기술 질문에 대해 토론을 할 수 있는 커뮤니티 기반의 질문 및 답변 사이트입니다.
- 지원 센터 - 이 사이트에는 최근 지원 사례와 AWS Trusted Advisor 및 상태 확인 결과에 대한 정보가 포함되어 있습니다. 또한 토론 포럼, 기술 FAQ, 서비스 상태 대시보드에 대한 링크와 지원 플랜에 대한 정보를 제공합니다.
- AWS Premium Support – 1:1 신속 대응 지원 채널을 통해 AWS에서 애플리케이션을 구축하고 실행할 수 있도록 지원하는 AWS Premium Support에 대해 알아봅니다.
- AWS IQ - AWS 공인 전문가의 도움을 받습니다.

# CloudFront 시작하기

이 섹션의 주제들은 Amazon CloudFront에서 콘텐츠 전송을 시작하는 방법을 보여 줍니다.

AWS 계정 설정 주제에서는 다음 튜토리얼의 사전 요구 사항(예: AWS 계정 생성 및 관리 액세스 권한이 있는 사용자 생성)을 설명합니다.

기본 배포 튜토리얼에서는 Amazon S3 오리진에 인증된 요청을 전송하기 위해 오리진 액세스 제어(OAC)를 설정하는 방법을 보여 줍니다.

보안 정적 웹 사이트 튜토리얼에서는 Amazon S3 오리진에서 OAC를 사용하여 도메인 이름에 대한 안전한 정적 웹 사이트를 생성하는 방법을 보여 줍니다. 이 튜토리얼에서는 구성 및 배포에 Amazon CloudFront(CloudFront) 템플릿을 사용합니다.

주제


- AWS 계정 설정
- CloudFront 표준 배포 시작
- 표준 배포 시작(AWS CLI)
- 안전한 정적 웹 사이트 시작하기


## AWS 계정 설정

이 주제에서는 AWS 계정 생성과 같이 Amazon CloudFront 사용을 준비하기 위한 예비 단계를 설명합니다.

주제


- AWS 계정에 가입
- 관리자 액세스 권한이 있는 사용자 생성
- CloudFront에 액세스하는 방법 선택


### AWS 계정에 가입

AWS 계정이 없는 경우 다음 절차에 따라 계정을 생성합니다.

AWS 계정에 가입하려면

1. https://portal.aws.amazon.com/billing/signup을 엽니다.

2. 온라인 지침을 따릅니다.

   등록 절차 중 전화 또는 텍스트 메시지를 받고 전화 키패드로 확인 코드를 입력하는 과정이 있습니다.

   AWS 계정에 가입하면 AWS 계정 루트 사용자이 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스에 액세스할 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

가입 프로세스가 완료되면 AWS은 사용자에게 확인 이메일을 전송합니다. 언제든지 https://aws.amazon.com/으로 이동하고 내 계정을 선택하여 현재 계정 활동을 확인하고 계정을 관리할 수 있습니다.

## 관리자 액세스 권한이 있는 사용자 생성

AWS 계정에 가입하고 AWS 계정 루트 사용자에 보안 조치를 한 다음, AWS IAM Identity Center을 활성화하고 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 생성합니다.

AWS 계정 루트 사용자 보호

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 AWS Management Console에 계정 소유자로 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면 AWS Sign-In 사용 설명서의 루트 사용자로 로그인을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

   지침은 IAM 사용자 안내서의 AWS 계정 루트 사용자용 가상 MFA 디바이스 활성화(콘솔)를 참조하세요.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용자 안내서의 AWS IAM Identity Center 설정을 참조하세요.
2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

IAM Identity Center 디렉터리를 ID 소스로 사용하는 방법에 대한 자습서는 AWS IAM Identity Center 사용자 안내서의 기본 IAM Identity Center 디렉터리로 사용자 액세스 구성을 참조하세요.

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM Identity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자로 로그인하는 데 도움이 필요한 경우 AWS Sign-In 사용 설명서의 Signing in to the AWS access portal을 참조하세요.

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

   지침은 Create a permission set에 대한 지침은 AWS IAM Identity Center사용 설명서에서 참조하세요.
2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 Add groups를 참조하세요.

## CloudFront에 액세스하는 방법 선택

Amazon CloudFront에 액세스하는 방법은 다음과 같습니다.

- AWS Management Console – 이 가이드의 절차는 AWS Management Console을 사용하여 태스크를 수행하는 방법을 설명합니다.
- AWS SDK – AWS에서 SDK를 제공하는 프로그래밍 언어를 사용하는 경우 SDK를 사용하여 CloudFront에 액세스할 수 있습니다. SDK는 인증을 단순화하고, 개발 환경에 쉽게 통합되며, CloudFront 명령에 액세스할 수 있도록 합니다. 자세한 내용은 AWS SDK와 함께 CloudFront 사용 섹션을 참조하세요.
- CloudFront API – SDK를 사용할 수 없는 프로그래밍 언어를 사용하는 경우 API 작업 및 API 요청 방법에 대한 자세한 내용을 Amazon CloudFront API 참조에서 확인하십시오.
- AWS CLI - AWS Command Line Interface(AWS CLI)는 AWS 서비스를 관리하는 통합 도구입니다. AWS CLI 설치 및 구성 방법에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서에서 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

- Tools for Windows PowerShell - Windows PowerShell을 사용한 경험이 있다면 AWS Tools for Windows PowerShell을 사용하는 것이 좋습니다. 자세한 내용은 AWS Tools for PowerShell 사용자 가이드에서 AWS Tools for Windows PowerShell 설치를 참조하세요.

# CloudFront 표준 배포 시작

이 섹션에 나온 절차에서는 CloudFront를 사용하여 다음을 수행하는 표준 배포 설정 방법을 보여줍니다.

- 배포 오리진으로 사용할 S3 버킷을 생성합니다.
- 객체의 원래 버전을 Amazon Simple Storage Service(S3) 버킷에 저장합니다.
- 오리진 액세스 제어(OAC)를 사용하여 Amazon S3 오리진에 인증된 요청을 전송합니다. OAC는 CloudFront를 통해 요청을 전송하여 최종 사용자가 S3 버킷에 직접 액세스하는 것을 방지합니다. OAC에 대한 자세한 내용은 Amazon S3 오리진에 대한 액세스 제한을 참조합니다.
- 객체에 대한 URL에 CloudFront 도메인 이름을 사용합니다(예: `https://d111111abcdef8.cloudfront.net/index.html`).
- 기본 24시간(최소 시간 0초) 동안 CloudFront 엣지 로케이션에 객체를 보관합니다.

이 중 대부분은 CloudFront 배포를 생성할 때 자동으로 구성됩니다.

주제


- 사전 조건
- Amazon S3 버킷 생성
- 버킷에 콘텐츠 업로드
- OAC와 함께 Amazon S3 오리진을 사용하는 CloudFront 배포 생성
- CloudFront를 통해 콘텐츠에 액세스
- 정리
- 기본 배포 향상


## 사전 조건

시작하기 전에 먼저 AWS 계정 설정의 단계를 완료해야 합니다.

# Amazon S3 버킷 생성

Amazon S3 버킷은 파일(객체) 또는 폴더를 위한 컨테이너입니다. CloudFront는 S3 버킷을 소스로 사용하여 거의 모든 유형의 파일을 배포할 수 있습니다. 예를 들어 CloudFront에서는 텍스트, 이미지 및 비디오를 배포할 수 있습니다. Amazon S3에 저장할 수 있는 데이터의 양에는 최대값이 없습니다.

이 자습서에서는 사용자는 기본 웹페이지를 생성하는 데 사용할 샘플 `hello world` 파일이 포함된 S3 버킷을 생성합니다.

버킷을 생성하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 이 시작하기에서는 hello world 샘플을 사용하는 것이 좋습니다. hello world 웹 페이지 hello-world-html.zip을 다운로드합니다. 압축을 풀고 브라우저를 실행 중인 데스크톱과 같은 편리한 위치에 `css` 폴더와 `index` 파일을 저장합니다.
3. 버킷 생성을 선택합니다.
4. Amazon Simple Storage Service 사용 설명서의 범용 버킷 이름 지정 규칙을 준수하는 고유한 버킷 이름을 입력합니다.
5. 리전의 경우 지리적으로 가까운 AWS 리전을 선택하는 것이 좋습니다. 이렇게 하면 지연 시간이 줄어들고 비용이 줄어듭니다.
   - 다른 리전을 선택하는 것도 효과가 있습니다. 예를 들어 규제 요구 사항을 해결하기 위해 이 작업을 수행할 수 있습니다.
6. 다른 모든 설정은 기본값으로 두고 버킷 생성(Create bucket)을 선택합니다.

# 버킷에 콘텐츠 업로드

Amazon S3 버킷을 생성한 후에는 압축이 풀린 `hello world` 파일의 콘텐츠를 버킷에 업로드합니다. (이 파일을 Amazon S3 버킷 생성에 다운로드하고 압축을 풀었습니다.)

Amazon S3에 콘텐츠를 업로드하려면

1. 범용 버킷 섹션에서 새 버킷의 이름을 선택합니다.
2. 업로드를 선택합니다.
3. 업로드 페이지에서 `css` 폴더와 `index` 파일을 드롭 영역에 끌어다 놓습니다.

4. 다른 모든 설정은 기본값으로 두고 업로드를 선택합니다.

## OAC와 함께 Amazon S3 오리진을 사용하는 CloudFront 배포 생성

이 자습서에서, 사용자는 오리진 액세스 제어(OAC)가 있는 Amazon S3 오리진을 사용하는 CloudFront 배포를 생성합니다. OAC를 사용하면 Amazon S3 오리진에 인증된 요청을 안전하게 전송할 수 있습니다. OAC에 대한 자세한 내용은 Amazon S3 오리진에 대한 액세스 제한을 참조합니다.

OAC를 사용하는 Amazon S3 오리진으로 CloudFront 배포를 생성하려는 경우

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다
2. 배포 생성을 선택합니다.
3. 표준 배포에 배포 이름을 입력합니다. 이름은 태그로서 `Name` 키의 값으로 표시됩니다. 이 값은 나중에 변경할 수 있습니다. 표준 배포에 최대 50개의 태그를 추가할 수 있습니다. 자세한 내용은 배포 태깅 섹션을 참조하세요.
4. 단일 웹사이트 또는 앱, 다음을 선택합니다.
5. 다음을 선택합니다.
6. 오리진 유형 페이지에서 Amazon S3를 선택합니다.
7. S3 오리진에서 S3 찾아보기를 선택하고 이 자습서용으로 생성한 S3 버킷을 선택합니다.
8. 설정에서 권장 오리진 설정 사용을 선택합니다. CloudFront는 오리진 액세스 제어(OAC) 설정을 포함하여 Amazon S3 오리진에 대한 기본 권장 캐시 및 오리진 설정을 사용합니다. 권장 설정에 대한 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하시기 바랍니다.
9. 다음을 선택합니다.
10. 보안 보호 활성화 페이지에서 AWS WAF 보안 보호 기능을 사용 설정할지 여부를 선택합니다.
11. 다음을 선택합니다.
12. 배포 생성을 선택합니다. CloudFront는 S3 버킷 정책을 자동으로 업데이트합니다.
13. 새 배포에 대한 세부 정보 섹션을 검토합니다. 배포가 완료되면 마지막 수정 필드가 배포 중에서 날짜 및 시간으로 변경됩니다.
14. CloudFront가 배포에 할당하는 도메인 이름을 기록해 둡니다. `d111111abcdef8.cloudfront.net`와 유사하게 표시됩니다.

이 자습서의 배포 및 S3 버킷을 프로덕션 환경에서 사용하기 전에 먼저 특정 요구 사항에 맞게 구성해야 합니다. 프로덕션 환경에서 액세스를 구성하는 방법에 대한 자세한 내용은 보안 액세스 구성 및 콘텐츠에 대한 액세스 제한 섹션을 참조합니다.

## CloudFront를 통해 콘텐츠에 액세스

CloudFront를 통해 콘텐츠에 액세스하려면 CloudFront 배포의 도메인 이름과 콘텐츠의 기본 페이지를 결합합니다. (배포 도메인 이름을 OAC와 함께 Amazon S3 오리진을 사용하는 CloudFront 배포 생성에 기록했습니다.)

- 배포 도메인 이름은 `d111111abcdef8.cloudfront.net`와 같을 수 있습니다.
- 웹 사이트의 기본 페이지 경로는 일반적으로 `/index.html`입니다.

따라서 CloudFront를 통해 콘텐츠에 액세스하기 위한 URL은 다음과 같을 수 있습니다.

`https://d111111abcdef8.cloudfront.net/index.html`.

이전 단계를 수행했고 hello world 웹 페이지를 사용했다면 Hello world!라고 써있는 웹 페이지가 표시됩니다.

이 S3 버킷에 더 많은 콘텐츠를 업로드하는 경우 CloudFront 배포 도메인 이름과 S3 버킷의 객체 경로를 결합하여 CloudFront를 통해 콘텐츠에 액세스할 수 있습니다. 예를 들어 `new-page.html`이라는 새 파일을 S3 버킷의 루트에 업로드하는 경우 URL은 다음과 같습니다.

`https://d111111abcdef8.cloudfront.net/new-page.html`.

## 정리

실습용으로만 배포 및 S3 버킷을 생성한 경우에는 요금이 발생하지 않도록 삭제합니다. 먼저 배포를 삭제합니다. 자세한 내용은 다음 링크를 참조하세요.

- 배포 삭제
- 버킷 삭제

## 기본 배포 향상

이 시작하기 튜토리얼에서는 배포 생성에 필요한 최소한의 프레임워크를 제공합니다. 다음과 같은 개선 사항을 살펴보는 것이 좋습니다.

- CloudFront 프라이빗 콘텐츠 기능을 사용하여 Amazon S3 버킷의 콘텐츠에 대한 액세스를 제한할 수 있습니다. 비공개 콘텐츠를 배포하는 방법에 대한 자세한 내용은 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공를 참조하십시오.

- 사용자 지정 도메인 이름(예: `d111111abcdef8.cloudfront.net` 대신 `www.example.com`)을 사용하도록 CloudFront 배포를 구성할 수 있습니다. 자세한 내용은 사용자 지정 URL 사용 섹션을 참조하세요.
- 이 자습서에서는 오리진 액세스 제어(OAC)가 있는 Amazon S3 오리진을 사용합니다. 하지만 오리진이 웹 사이트 엔드포인트로 구성된 S3 버킷인 경우에는 OAC를 사용할 수 없습니다. 이 경우에는 CloudFront를 사용자 지정 오리진으로 사용하여 버킷을 설정해야 합니다. 자세한 내용은 웹 사이트 엔드포인트로 구성된 Amazon S3 버킷 사용 섹션을 참조하세요. OAC에 대한 자세한 내용은 Amazon S3 오리진에 대한 액세스 제한을 참조합니다.

# 표준 배포 시작(AWS CLI)

이 섹션에 나온 절차에서는 CloudFront에서 AWS CLI를 사용하여 다음을 수행하는 기본 구성 설정 방법을 보여줍니다.

- 배포 오리진으로 사용할 Amazon S3 버킷 생성.
- 객체의 원래 버전을 S3 버킷에 저장.
- 오리진 액세스 제어(OAC)를 사용하여 Amazon S3 오리진에 인증된 요청 전송. OAC는 CloudFront를 통해 요청을 전송하여 최종 사용자가 S3 버킷에 직접 액세스하는 것을 방지합니다. OAC에 대한 자세한 내용은 Amazon S3 오리진에 대한 액세스 제한을 참조합니다.
- 객체에 대한 URL에 CloudFront 도메인 이름(예: `https://d111111abcdef8.cloudfront.net/index.html`) 사용.
- 기본 24시간(최소 시간 0초) 동안 CloudFront 엣지 로케이션에 객체 보관.

이러한 옵션은 대부분 사용자 지정이 가능합니다. CloudFront 배포 옵션을 사용자 지정하는 방법은 배포 생성 단원을 참조하세요.

## 사전 조건

시작하기 전에 먼저 AWS 계정 설정의 단계를 완료해야 합니다.

AWS CLI를 설치하고 자격 증명을 사용하여 구성. 자세한 내용은 AWS CLI 사용 설명서에서 AWS CLI 시작하기를 참조하세요.

## Amazon S3 버킷 생성

Amazon S3 버킷은 파일(객체) 또는 폴더를 위한 컨테이너입니다. CloudFront는 S3 버킷을 소스로 사용하여 거의 모든 유형의 파일을 배포할 수 있습니다. 예를 들어 CloudFront에서는 텍스트, 이미지 및 비디오를 배포할 수 있습니다. Amazon S3에 저장할 수 있는 데이터의 양에는 최대값이 없습니다.

이 자습서에서는 S3 버킷을 생성하고 기본 웹페이지를 생성하는 데 사용할 HTML 파일을 업로드합니다.

```
aws s3 mb s3://amzn-s3-demo-bucket/ --region us-east-1
```

*amzn-s3-demo-bucket*을 전역적으로 고유한 버킷 이름으로 바꿉니다. AWS 리전의 경우 지리적으로 가까운 리전을 선택하는 것이 좋습니다. 이렇게 하면 지연 시간과 비용이 줄어듭니다. 물론 다른 리전을 선택하는 것도 가능합니다. 예를 들어 규제 요구 사항을 해결하기 위해 이렇게 할 수 있습니다.

## 버킷에 콘텐츠 업로드

이 자습서에서는 기본적인 "Hello World" 웹페이지의 샘플 콘텐츠 파일을 다운로드하고 추출합니다.

```
# Create a temporary directory
mkdir -p ~/cloudfront-demo

# Download the sample Hello World files
curl -o ~/cloudfront-demo/hello-world-html.zip https://docs.aws.amazon.com/
AmazonCloudFront/latest/DeveloperGuide/samples/hello-world-html.zip

# Extract the zip file
unzip ~/cloudfront-demo/hello-world-html.zip -d ~/cloudfront-demo/hello-world
```

이렇게 하면 `index.html` 파일과 `css` 폴더가 있는 디렉터리가 생성됩니다. 이러한 파일을 S3 버킷에 업로드합니다.

```
aws s3 cp ~/cloudfront-demo/hello-world/ s3://amzn-s3-demo-bucket/ --recursive
```

## 오리진 액세스 제어(OAC) 생성

이 자습서에서는 오리진 액세스 제어(OAC)를 생성합니다. OAC를 사용하면 Amazon S3 오리진에 인증된 요청을 안전하게 전송할 수 있습니다. OAC에 대한 자세한 내용은 Amazon S3 오리진에 대한 액세스 제한을 참조합니다.

```
aws cloudfront create-origin-access-control \
    --origin-access-control-config Name="oac-for-
s3",SigningProtocol=sigv4,SigningBehavior=always,OriginAccessControlOriginType=s3
```

출력의 OAC ID를 환경 변수로 저장합니다. 예시 값을 실제 OAC ID로 바꿉니다. 다음 단계에서 이 값을 사용합니다.

```
OAC_ID="E1ABCD2EFGHIJ"
```

## 표준 배포 생성

`distribution-config.json`이라는 이름으로 배포 구성 파일을 만듭니다. `Id`, `DomainName`, `TargetOriginId` 값을 포함해 예시 버킷 이름을 실제 버킷 이름으로 바꿉니다.

```
cat > distribution-config.json << EOF
{
    "CallerReference": "cli-example-$(date +%s)",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "S3-amzn-s3-demo-bucket",
                "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                },
                "OriginAccessControlId": "$OAC_ID"
            }
        ]
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "S3-amzn-s3-demo-bucket",
        "ViewerProtocolPolicy": "redirect-to-https",
        "AllowedMethods": {
            "Quantity": 2,
            "Items": ["GET", "HEAD"],
            "CachedMethods": {
                "Quantity": 2,
                "Items": ["GET", "HEAD"]
            }
        },
        "DefaultTTL": 86400,
```

```
        "MinTTL": 0,
        "MaxTTL": 31536000,
        "Compress": true,
        "ForwardedValues": {
            "QueryString": false,
            "Cookies": {
                "Forward": "none"
            }
        }
    },
    "Comment": "CloudFront distribution for S3 bucket",
    "Enabled": true
}
EOF
```

표준 배포를 생성합니다.

```
aws cloudfront create-distribution --distribution-config file://distribution-
config.json
```

출력의 배포 ID와 도메인 이름을 환경 변수로 저장합니다. 예제 값을 사용자의 값으로 바꿉니다. 이 자습서 뒷부분에서 이 정보가 필요합니다.

```
DISTRIBUTION_ID="EABCD1234XMPL"
DOMAIN_NAME="d111111abcdef8.cloudfront.net"
```

이 자습서의 배포 및 S3 버킷을 프로덕션 환경에서 사용하기 전에 먼저 특정 요구 사항에 맞게 구성해야 합니다. 프로덕션 환경에서 액세스를 구성하는 방법에 대한 자세한 내용은 보안 액세스 구성 및 콘텐츠에 대한 액세스 제한 섹션을 참조합니다.

## S3 버킷 정책 업데이트

CloudFront가 객체에 액세스할 수 있도록 S3 버킷 정책을 업데이트합니다. 예시 버킷 이름을 실제 버킷 이름으로 바꿉니다.

```
# Get your AWS account ID
ACCOUNT_ID=$(aws sts get-caller-identity --query 'Account' --output text)

# Create the bucket policy
cat > bucket-policy.json << EOF
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCloudFrontServicePrincipal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudfront.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceArn": "arn:aws:cloudfront::$ACCOUNT_ID:distribution/
$DISTRIBUTION_ID"
                }
            }
        }
    ]
}
EOF

# Apply the bucket policy
aws s3api put-bucket-policy \
    --bucket amzn-s3-demo-bucket \
    --policy file://bucket-policy.json
```

## 배포 작업 확인

배포를 생성한 후 배포를 완료하는 데 다소 시간이 걸립니다. 배포 상태가 `InProgress`에서 `Deployed`로 변경되면 다음 단계로 진행합니다.

```
aws cloudfront get-distribution --id $DISTRIBUTION_ID --query 'Distribution.Status'
```

또는 `wait` 명령을 사용하여 배포 작업을 기다릴 수 있습니다.

```
aws cloudfront wait distribution-deployed --id $DISTRIBUTION_ID
```

## CloudFront를 통해 콘텐츠에 액세스

CloudFront를 통해 콘텐츠에 액세스하려면 CloudFront 배포의 도메인 이름과 콘텐츠의 기본 페이지를 결합합니다. 예시 CloudFront 도메인 이름을 실제 도메인 이름으로 바꿉니다.

```
https://d111111abcdef8.cloudfront.net/index.html
```

이전 단계를 수행했고 HTML 파일을 생성했다면 Hello world!라고 써있는 웹 페이지가 표시됩니다.

이 S3 버킷에 더 많은 콘텐츠를 업로드하는 경우 CloudFront 배포 도메인 이름과 S3 버킷의 객체 경로를 결합하여 CloudFront를 통해 콘텐츠에 액세스할 수 있습니다. 예를 들어 `new-page.html`이라는 새 파일을 S3 버킷의 루트에 업로드하는 경우 URL은 다음과 같습니다.

`https://d111111abcdef8.cloudfront.net/new-page.html`.

## 정리

실습용으로만 배포 및 S3 버킷을 생성한 경우에는 요금이 발생하지 않도록 삭제합니다. 먼저 배포를 비활성화하고 삭제합니다.

표준 배포를 비활성화 및 삭제하려면(AWS CLI)

1. 먼저 배포를 비활성화합니다.

```
# Get the current configuration and ETag
ETAG=$(aws cloudfront get-distribution-config --id $DISTRIBUTION_ID --query 'ETag'
 --output text)

# Create a modified configuration with Enabled=false
aws cloudfront get-distribution-config --id $DISTRIBUTION_ID | \
jq '.DistributionConfig.Enabled = false' > temp_disabled_config.json

# Update the distribution to disable it
aws cloudfront update-distribution \
    --id $DISTRIBUTION_ID \
    --distribution-config file://<(jq '.DistributionConfig'
 temp_disabled_config.json) \
    --if-match $ETAG
```

2. 배포가 비활성화될 때까지 기다립니다.

```
aws cloudfront wait distribution-deployed --id $DISTRIBUTION_ID
```

3. 배포를 삭제합니다.

```
# Get the current ETag
```

```
ETAG=$(aws cloudfront get-distribution-config --id $DISTRIBUTION_ID --query 'ETag'
 --output text)

# Delete the distribution
aws cloudfront delete-distribution --id $DISTRIBUTION_ID --if-match $ETAG
```

S3 버킷을 삭제하려면(AWS CLI)

- S3 버킷과 해당 콘텐츠를 삭제합니다. 예시 버킷 이름을 실제 버킷 이름으로 바꿉니다.

```
# Delete the bucket contents
aws s3 rm s3://amzn-s3-demo-bucket --recursive

# Delete the bucket
aws s3 rb s3://amzn-s3-demo-bucket
```

이 자습서용으로 생성된 로컬 파일을 정리하려면 다음 명령을 실행합니다.

```
# Clean up local files
rm -f distribution-config.json bucket-policy.json temp_disabled_config.json
rm -rf ~/cloudfront-demo
```

선택적으로 이 자습서용으로 생성한 OAC를 삭제할 수 있습니다.

```
# Get the OAC ETag
OAC_ETAG=$(aws cloudfront get-origin-access-control --id $OAC_ID --query 'ETag' --
output text)

# Delete the OAC
aws cloudfront delete-origin-access-control --id $OAC_ID --if-match $OAC_ETAG
```

# 안전한 정적 웹 사이트 시작하기

이 주제에서 설명하는 솔루션을 통해 도메인 이름에 대한 안전한 정적 웹 사이트를 생성하여 Amazon CloudFront를 시작할 수 있습니다. 정적 웹 사이트는 HTML, CSS, JavaScript, 이미지 및 비디오와 같은 정적 파일만 사용하며, 서버나 서버 측 처리가 필요하지 않습니다. 이 솔루션을 사용하면 웹 사이트에서 다음과 같은 이점을 얻을 수 있습니다.

- Amazon Simple Storage Service(Amazon S3)의 내구성이 뛰어난 스토리지 사용 - 이 솔루션은 정적 웹 사이트의 콘텐츠를 호스팅하는 Amazon S3 버킷을 생성합니다. 웹 사이트를 업데이트하려면 새 파일을 S3 버킷에 업로드하면 됩니다.
- Amazon CloudFront 콘텐츠 전송 네트워크를 통해 속도 향상 - 이 솔루션은 짧은 지연 시간으로 최종 사용자에게 웹 사이트를 제공하는 CloudFront 배포를 생성합니다. 배포는 웹 사이트가 S3에서 직접 액세스하지 않고 CloudFront를 통해서만 액세스할 수 있도록 오리진 액세스 제어(OAC)로 구성됩니다.
- HTTPS 및 보안 헤더에 의해 보호 – 이 솔루션은 AWS Certificate Manager(ACM)에 SSL/TLS 인증서를 생성하고 이를 CloudFront 배포에 연결합니다. 이 인증서를 사용하면 배포가 HTTPS를 사용하여 도메인의 웹 사이트를 안전하게 제공할 수 있습니다.
- AWS CloudFormation으로 구성 및 배포 – 이 솔루션은 CloudFormation 템플릿을 사용하여 모든 구성 요소를 설정하므로 구성 요소 구성보다 웹 사이트의 콘텐츠에 더 집중할 수 있습니다.

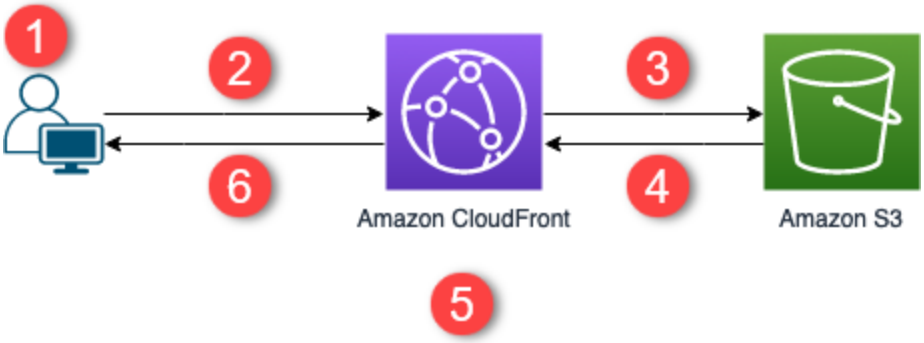
이 솔루션은 GitHub의 오픈 소스입니다. 코드를 보거나 끌어오기 요청을 제출하거나 문제를 열려면 로 이동합니다https://github.com/aws-samples/amazon-cloudfront-secure-static-site

주제

- 솔루션 개요
- 솔루션 배포

## 솔루션 개요

다음 다이어그램은 이 정적 웹 사이트 솔루션의 작동 방식에 대한 개요를 보여줍니다.




1. 최종 사용자는 www.example.com에서 웹 사이트를 요청합니다.
2. 요청된 객체가 캐싱된 경우 CloudFront가 캐시의 객체를 최종 사용자에게 반환합니다.
3. 객체가 CloudFront 캐시에 없는 경우 CloudFront는 오리진(S3 버킷)에서 객체를 요청합니다.

4. S3가 객체를 CloudFront로 반환합니다.
5. CloudFront가 객체를 캐시합니다.
6. 객체가 최종 사용자에게 반환됩니다. 동일한 CloudFront 엣지 로케이션에 들어오는 객체에 대한 후속 요청은 CloudFront 캐시에서 제공됩니다.

## 솔루션 배포

이 안전한 정적 웹 사이트 솔루션을 배포하기 위해 다음 옵션 중 하나를 선택할 수 있습니다.

- CloudFormation 콘솔을 사용하여 기본 콘텐츠가 포함된 솔루션을 배포한 다음, 웹 사이트 콘텐츠를 Amazon S3에 업로드합니다.
- 솔루션을 컴퓨터에 복제하여 웹 사이트 콘텐츠를 추가합니다. 그런 다음 AWS Command Line Interface(AWS CLI)를 사용하여 솔루션을 배포합니다.

> **Note**
>
> CloudFormation 템플릿을 배포하려면 미국 동부(버지니아 북부) 리전을 사용해야 합니다.

주제


- 사전 조건
- CloudFormation 콘솔을 사용합니다.
- 로컬로 솔루션 복제
- 액세스 로그 찾기


### 사전 조건

이 솔루션을 사용하려면 다음과 같은 사전 요구 사항이 있어야 합니다.

- example.com과 같이 Amazon Route 53 호스팅 영역을 가리키는 등록된 도메인 이름. 호스팅 영역은 이 솔루션을 배포하는 동일한 AWS 계정에 있어야 합니다. 등록된 도메인 이름이 없는 경우 Route 53에 등록할 수 있습니다. 등록된 도메인 이름이 있지만 Route 53 호스팅 영역을 가리키지 않는 경우 Route 53를 DNS 서비스로 구성합니다.
- IAM 역할을 생성하는 CloudFormation 템플릿을 시작할 수 있는 AWS Identity and Access Management(IAM) 권한 및 솔루션의 모든 AWS 리소스를 생성할 수 있는 권한. 자세한 내용은 AWS

CloudFormation 사용 설명서에서 AWS Identity and Access Management를 사용한 액세스 제어 섹션을 참조하세요.

이 솔루션을 사용하는 동안 발생하는 비용에 대한 책임은 귀하에게 있습니다. 비용에 대한 자세한 내용은 각 AWS 서비스의 요금 페이지를 참조하세요.

## CloudFormation 콘솔을 사용합니다.

CloudFormation 콘솔을 사용하여 배포하려면

1. CloudFormation 콘솔에서 이 솔루션을 시작합니다. 필요한 경우 AWS 계정에 로그인합니다.
2. CloudFormation 콘솔에서 이 솔루션의 CloudFormation 템플릿을 지정하는 필드가 미리 채워져 있는 스택 생성 마법사가 열립니다.

   페이지 하단에서 [Next]를 선택합니다.
3. 스택 세부 정보 지정 페이지에서 다음 필드에 값을 입력합니다.
   - SubDomain - 웹 사이트에 사용할 하위 도메인을 입력합니다. 예를 들어, 하위 도메인이 www인 경우 웹 사이트는 *www*.example.com에서 사용할 수 있습니다. 다음 글머리 기호에 설명된 대로 example.com을 도메인 이름으로 바꿉니다.
   - DomainName - 도메인 이름(예: *example.com*)을 입력합니다. 이 도메인은 Route 53 호스팅 영역을 가리켜야 합니다.
   - HostedZoneId — 도메인 이름을 위한 Route 53 호스팅 영역 ID.
   - CreateApex - (선택 사항) CloudFront 구성에서 도메인 apex(example.com)에 대한 별칭을 생성합니다.
4. 마친 후에는 다음을 선택합니다.
5. (선택 사항) 스택 옵션 구성(Configure stack options) 페이지에서 태그 및 기타 스택 옵션을 추가합니다.
6. 마친 후에는 다음을 선택합니다.
7. 검토 페이지에서 페이지 하단으로 스크롤한 다음 기능 섹션에서 두 상자를 선택합니다. 이러한 기능을 통해 CloudFormation은 스택의 리소스에 대한 액세스를 허용하는 IAM 역할을 생성하고 리소스의 이름을 동적으로 지정할 수 있습니다.
8. [Create stack]을 선택합니다.
9. 스택 생성이 완료될 때까지 기다립니다. 스택은 일부 중첩된 스택을 생성하고 완료하는 데 몇 분 정도 걸릴 수 있습니다. 완료되면 상태가 CREATE_COMPLETE로 변경됩니다.

상태가 CREATE_COMPLETE이면 https://*www.example.com*으로 이동하여 웹 사이트를 봅니다 (www.example.com을 3단계에서 지정한 하위 도메인 및 도메인 이름으로 바꿉니다). 웹 사이트의 기본 콘텐츠가 표시됩니다.

웹 사이트의 기본 콘텐츠를 사용자 고유의 콘텐츠로 바꾸려면

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 이름이 amazon-cloudfront-secure-static-site-s3bucketroot-로 시작하는 버킷을 선택합니다.

> **Note**
> 이름에 s3bucketlogs가 아닌 s3bucketroot가 있는 버킷을 선택해야 합니다. 이름에 s3bucketroot가 있는 버킷에는 웹 사이트 콘텐츠가 포함되어 있습니다. s3bucketlogs가 있는 버킷에는 로그 파일만 포함되어 있습니다.

3. 웹 사이트의 기본 콘텐츠를 삭제한 다음, 직접 업로드합니다.

> **Note**
> 이 솔루션의 기본 콘텐츠가 있는 웹 사이트를 본 경우, 일부 기본 콘텐츠가 CloudFront 엣지 로케이션에 캐싱될 가능성이 높습니다.. 최종 사용자가 업데이트된 웹 사이트 콘텐츠를 볼 수 있도록 하려면 파일을 무효화하여 CloudFront 엣지 로케이션에서 캐싱된 복사본을 제거합니다. 자세한 내용은 파일을 무효화하여 콘텐츠 제거 단원을 참조하십시오.

## 로컬로 솔루션 복제

### 사전 조건

이 솔루션을 배포하기 전에 웹 사이트 콘텐츠를 추가하려면 Node.js 및 npm이 필요한 솔루션의 아티팩트를 로컬로 패키징해야 합니다. 자세한 내용은 https://www.npmjs.com/get-npm 단원을 참조하세요.

웹 사이트 콘텐츠를 추가하고 솔루션을 배포하려면

1. 에서 솔루션을 복제하거나 다운로드합니다https://github.com/aws-samples/amazon-cloudfront-secure-static-site 복제하거나 다운로드한 후 명령 프롬프트 또는 터미널을 열고 해당 `amazon-cloudfront-secure-static-site` 폴더로 이동합니다.
2. 다음 명령을 실행하여 솔루션의 아티팩트를 설치하고 패키징합니다.

```
make package-static
```

3. 웹 사이트의 콘텐츠를 `www` 폴더에 복사하여 기본 웹 사이트 콘텐츠를 덮어씁니다.
4. 다음 AWS CLI 명령을 실행하여 솔루션의 아티팩트를 저장할 Amazon S3 버킷을 생성합니다. *`amzn-s3-demo-bucket-for-artifacts`*를 사용자 고유의 버킷 이름으로 바꿉니다.

```
aws s3 mb s3://amzn-s3-demo-bucket-for-artifacts --region us-east-1
```

5. 솔루션의 아티팩트를 CloudFormation 템플릿으로 패키징하려면 다음 AWS CLI 명령을 실행합니다. *`amzn-s3-demo-bucket-for-artifacts`*를 이전 단계에서 생성한 버킷의 이름으로 바꿉니다.

```
aws cloudformation package \
    --region us-east-1 \
    --template-file templates/main.yaml \
    --s3-bucket amzn-s3-demo-bucket-for-artifacts \
    --output-template-file packaged.template
```

6. 다음 명령을 실행하여 CloudFormation으로 솔루션을 배포하고 다음 값을 바꿉니다.
   - *`your-CloudFormation-stack-name`* - CloudFormation 스택의 이름으로 바꿉니다.
   - *`example.com`* - 도메인 이름으로 바꿉니다. 이 도메인은 동일한 AWS 계정의 Route 53 호스팅 영역을 가리켜야 합니다.
   - *`www`* - 웹 사이트에 사용할 하위 도메인으로 바꿉니다. 예를 들어, 하위 도메인이 www인 경우 웹 사이트는 www.example.com에서 사용할 수 있습니다.
   - *`hosted-zone-ID`* – 도메인 이름의 Route 53 호스팅 영역 ID로 바꿉니다.

```
aws cloudformation deploy \
    --region us-east-1 \
    --stack-name your-CloudFormation-stack-name \
    --template-file packaged.template \
```

```
    --capabilities CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND \
    --parameter-overrides DomainName=example.com SubDomain=www HostedZoneId=hosted-
zone-ID
```

- (선택 사항) 도메인 apex를 사용하여 스택을 배포하려면 다음 명령을 대신 실행하세요.

```
aws --region us-east-1 cloudformation deploy \
    --stack-name your-CloudFormation-stack-name \
    --template-file packaged.template \
    --capabilities CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND \
    --parameter-overrides  DomainName=example.com SubDomain=www
 HostedZoneId=hosted-zone-ID CreateApex=yes
```

7. CloudFormation 스택의 생성이 완료될 때까지 기다립니다. 스택은 일부 중첩된 스택을 생성하고 완료하는 데 몇 분 정도 걸릴 수 있습니다. 완료되면 상태가 CREATE_COMPLETE로 변경됩니다.

   상태가 CREATE_COMPLETE이면 https://www.example.com으로 이동하여 웹 사이트를 봅니다 (www.example.com을 이전 단계에서 지정한 하위 도메인 및 도메인 이름으로 바꿉니다). 웹 사이트의 내용이 보일 것입니다.

## 액세스 로그 찾기

이 솔루션은 CloudFront 배포에 대한 액세스 로그를 활성화합니다. 배포의 액세스 로그를 찾으려면 다음 단계를 완료하십시오.

배포의 액세스 로그를 찾으려면

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 이름이 amazon-cloudfront-secure-static-site-s3bucketlogs-로 시작하는 버킷을 선택합니다.

   > **Note**
   >
   > 이름에 s3bucketroot가 아닌 s3bucketlogs가 있는 버킷을 선택해야 합니다. 이름에 s3bucketlogs가 있는 버킷에는 로그 파일이 포함되어 있습니다. s3bucketroot가 있는 버킷에는 웹 사이트 콘텐츠가 포함되어 있습니다.

3. cdn 폴더에는 CloudFront 액세스 로그가 포함되어 있습니다.

# CloudFront 정액 요금제

CloudFront 정액 요금제는 Amazon CloudFront 글로벌 콘텐츠 전송 네트워크(CDN)와 여러 AWS 서비스 및 기능을 트래픽 급증 또는 공격에 관계없이 월별 가격으로 결합하며 초과 요금이 없습니다.

정액 요금제는 간단한 월별 가격으로 다음과 같은 기능을 제공합니다.

- CloudFront CDN
- AWS WAF 및 DDoS 보호
- 봇 관리 및 분석
- Amazon Route 53 DNS
- Amazon CloudWatch Logs 수집
- TLS 인증서
- 서버리스 엣지 컴퓨팅
- 매월 Amazon S3 스토리지 크레딧

플랜은 애플리케이션의 요구 사항에 맞게 Free, Pro, Business 및 Premium 티어에서 사용할 수 있습니다. 플랜은 이용 가능한 최상의 요금을 얻기 위한 연간 약정이 필요하지 않습니다. Free 플랜으로 시작하고 업그레이드하여 더 많은 기능과 더 많은 사용 허용량에 액세스하세요.

주제


- CloudFront 정액 요금제의 이점
- 요금제 티어별 기능
- 월별 사용 허용량
- 플랜에서 부담하는 비용
- 요금제로 전체 AWS 비용 절감
- 정액 요금제 관리
- 권한
- 정액 요금제 할당량
- 지원되지 않는 기능

# CloudFront 정액 요금제의 이점

CloudFront 요금제는 다음과 같은 몇 가지 주요 이점을 제공합니다.

- 통합 서비스 및 요금

  단일 고정 요금으로 여러 AWS 서비스 및 기능을 단일 플랜으로 결합합니다. 별도의 서비스 구매 및 선결제 요금 계산을 제거하도록 설계되었습니다.
- 초과분 없음

  트래픽 급증이나 공격에 관계없이 초과 요금이 없습니다.
- 사용 허용량 지우기

  각 플랜에는 해당 티어에서 최적의 성능을 제공하도록 설계된 게시된 사용 허용량이 포함되어 있습니다. 장기 약정 없이 사용량을 모니터링하고, 사전 알림을 받고, 애플리케이션의 필요에 따라 업그레이드하세요.
- DDoS 공격으로부터 보호

  CloudFront 및 AWS WAF는 인프라에 도달하기 전에 공격을 흡수하고 차단합니다. 합법적인 트래픽에 대해서만 컴퓨팅, 데이터베이스 및 인프라 사용률을 예약합니다. AWS WAF로 차단된 DDoS 공격 및 요청은 사용 허용량에 포함되지 않습니다.
- 전체 AWS 비용 절감

  Amazon S3, AWS Application Load Balancer(ALB) 또는 Amazon API Gateway와 같은 서비스에서 실행되는 AWS 애플리케이션에서 CloudFront로의 데이터 전송은 계속 무료입니다. 인터넷에 직접 접속하는 대신 CloudFront를 통해 AWS 애플리케이션을 제공하는 경우 고정 요금제에서는 초과 요금 걱정 없이 간단한 월별 요금으로 애플리케이션과 뷰어 간의 데이터 전송 비용을 충당합니다. 오리진에 도달하는 요청이 적을수록 사용량에 따라 요금이 부과되는 서비스에 대한 비용도 절감됩니다.

# 요금제 티어별 기능

각 요금제에는 필수 기능과 서비스를 월 가격으로 결합하는 최대 하나의 정점(루트) 도메인이 있는 CloudFront 배포 하나가 포함됩니다. 각 플랜에는 추가 S3 스토리지 크레딧도 포함됩니다.

상위 티어에 대한 플랜에는 하위 티어 플랜의 모든 기능과 추가 기능이 포함됩니다.

- Free - 애호가, 학습자 및 개발자가 시작할 수 있습니다.
- Pro - 소규모 웹 사이트, 블로그 및 애플리케이션을 시작하고 확장합니다.

- Business - 비즈니스 애플리케이션을 보호하고 가속화합니다.
- Premium - 비즈니스 및 미션 크리티컬 애플리케이션을 확장하고 보호합니다.

애플리케이션에 필요한 기능 및 구성이 포함된 플랜 티어를 선택합니다. 요금제당 다음 기능을 참조하세요.

## 요금제 기능

다음 표에는 각 요금제 계층에 포함된 CloudFront AWS WAF 및 DDoS, Amazon Route 53, Amazon CloudWatch 및 Amazon S3 기능이 나와 있습니다.

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
|---|---|---|---|---|
| 글로벌 CDN | 예 | 예 | 예 | 예 |
| CloudFront의 750개 이상의 글로벌 엣지 로케이션을 웹 애플리케이션의 대규모 분산 단일 진입점으로 사용합니다. 정적, 동적 및 캐시 불가능한 애플리케이션을 가속화합니다. | ✓ | ✓ | ✓ | ✓ |
| 콘텐츠 캐싱 | 예 | 예 | 예 | 예 |
| 콘텐츠 사본을 CloudFront의 전 세계 750개 이상의 엣지 로케이션에 저장하여 가장 가까운 위치에서 사용자에게 제공합니다. 애플리케이션 서버 대신 | ✓ | ✓ | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 로컬에서 반복 요청을 처리하여 로드 시간을 줄이고, 트래픽 급증으로부터 애플리케이션을 보호하고, 비용을 절감합니다. | | | | |
| 인스턴스 캐시 무효화 | 예 | 예 | 예 | 예 |
| 모든 엣지 로케이션에서 캐시된 콘텐츠를 몇 초 이내에 제거하거나 업데이트합니다. | ✓ | ✓ | ✓ | ✓ |
| 스마트 라우팅 | 예 | 예 | 예 | 예 |
| 실시간 네트워크 데이터를 사용하여 사용자를 최적의 엣지 로케이션으로 지능적으로 라우팅하고 성능을 높이기 위해 AWS 프라이빗 네트워크를 통해 AWS 오리진에 연결합니다. | ✓ | ✓ | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
|---|---|---|---|---|
| 계층형 캐싱 | 예 | 예 | 예 | 예 |
| 리전 엣지 캐시는 엣지 로케이션과 애플리케이션 사이에 위치하며 콘텐츠를 더 오래 저장하여 애플리케이션의 부하를 줄이고 빠른 전송을 유지합니다. | ✓ | ✓ | ✓ | ✓ |
| 기본 캐싱 규칙 | 예 | 예 | 예 | 예 |
| 사용자 지정 구성 없이 대부분의 웹 애플리케이션을 캐싱하기 위한 효과적인 캐싱 결정을 내립니다. | ✓ | ✓ | ✓ | ✓ |
| 사용자 지정 캐싱 규칙 | | | 예 | 예 |
| CloudFront에서 사용할 요청 값을 지정하고 캐시 정책을 사용해 애플리케이션의 성능, 개인화 및 최신 요구 사항에 맞게 최적화하여 콘텐츠를 캐싱하는 방법을 제어합니다. | | | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 고속 오리진 라우팅 | | | | 예 |
| Origin Shield를 사용하면 오리진에 대한 고성능 경로를 위해 CloudFront의 프라이빗 네트워크를 사용하여 엣지 로케이션에서 오리진으로 동적 요청이 라우팅됩니다. | | | | ✓ |
| 오리진 로드 감소 | | | | 예 |
| Origin Shield를 사용하여 웹 애플리케이션 근처에 캐싱 계층을 추가합니다. Origin Shield는 모든 엣지 로케이션의 요청을 통합하여 특히 트래픽이 급증하는 동안 애플리케이션의 부하를 줄입니다. | | | | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 자동 오리진 장애 조치<br><br>기본 오리진에 장애가 발생할 경우 트래픽을 백업 오리진으로 자동 라우팅하여 사용자를 방해하지 않고 고가용성을 유지합니다. | | | | 예 |
| 기본 오리진 요청 규칙<br><br>일반적인 시나리오에 최적화된 AWS 관리형 오리진 요청 정책을 사용하여 오리진에 대한 요청에 자동으로 포함되는 뷰어 요청의 정보를 제어합니다. | 예 | 예 | 예 | 예 |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 기본 응답 헤더 규칙 | 예 | 예 | 예 | 예 |
| AWS 관리형 응답 헤더 정책을 사용하여 일반 보안 헤더, CORS 설정 및 기타 표준 사용 사례에 맞게 미리 구성된 뷰어에 대한 응답에서 HTTP 헤더를 추가하거나 제거합니다. | ✓ | ✓ | ✓ | ✓ |
| 사용자 지정 오리진 요청 규칙 | | | 예 | 예 |
| 고유한 오리진 요청 정책을 생성하여 오리진으로 전달되는 URL 쿼리 문자열, 헤더 및 쿠키를 정확히 지정하여 사용자 지정 분석 및 요청 처리를 활성화합니다. | | | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 사용자 지정 응답 헤더 규칙 | | | 예 | 예 |
| 자체 응답 헤더 정책을 생성하여 CloudFront가 보안 헤더, 콘텐츠 보안 정책(CSP), CORS 설정 및 사용자 지정 애플리케이션 헤더와 같이 뷰어에게 응답으로 추가하거나 제거하는 HTTP 헤더를 정확하게 제어합니다. | | | ✓ | ✓ |
| 캐시 동작 수 | 5 | 10 | 50 | 100 |
| CloudFront가 콘텐츠를 제공하는 오리진, 콘텐츠 캐싱 방법, HTTPS 또는 서명된 URL이 필요한지 여부 등 특정 URLs 패턴에 대한 요청을 처리하는 방법을 제어하도록 캐시 동작을 구성합니다. | | | | |
| 보안 및 보호 | | | | |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 상시 작동 DDoS 보호 | 예 | 예 | 예 | 예 |
| 웹 사이트 또는 애플리케이션을 대상으로 하는 DDoS 공격으로부터 보호합니다. | ✓ | ✓ | ✓ | ✓ |
| 고급 DDoS 보호 | | | 예 | 예 |
| AntiDDoS AMR을 사용하여 DDoS 공격을 순식간에 식별 및 차단합니다. AWS는 고유한 애플리케이션 패턴을 학습하여 합법적인 사용자의 공격과 자연적인 급증을 구분합니다. | | | ✓ | ✓ |
| 웹 애플리케이션 방화벽(WAF) | 예 | 예 | 예 | 예 |
| Amazon 내부 위협 인텔리전스를 기반으로 일반적인 애플리케이션 취약성 및 잠재적 위협으로부터 보호합니다. 서버에 도달하기 전에 요청이 차단됩니다. | ✓ | ✓ | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| WAF 규칙 수 | 5 | 25 | 50 | 75 |
| 사용자 지정 규칙과 AWS 관리형 규칙을 모두 포함하여 WAF 구성에서 생성하고 활성화할 수 있는 총 보안 규칙 수입니다. | | | | |
| WordPress, PHP 및 SQL 데이터베이스에 대한 보호 | | 예 | 예 | 예 |
| WordPress, PHP, SQL 데이터베이스, Linux 및 Windows와 같은 일반적인 애플리케이션 및 운영 체제를 보호하는 사용 사례 기반 보안 규칙입니다. | | | | |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| IP 기반 속도 제한<br><br>5분 동안 구성 가능한 요청 수를 초과하는 IP 주소를 자동으로 차단하여 HTTP 플러드 공격 및 서비스 거부(DoS) 시도로부터 보호합니다. | 예 | 예 | 예 | 예 |
| 지리적 트래픽 차단<br><br>선택한 국가 또는 리전의 요청을 차단합니다. | 예 | 예 | 예 | 예 |
| 헤더 기반 위협 필터링<br><br>HTTP 요청 헤더를 기반으로 위협을 필터링하는 WAF 보안 규칙을 생성합니다. | | 예 | 예 | 예 |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 정규식 기반 위협 필터링 | | | 예 | 예 |
| 정규식을 사용하여 URI 경로 및 HTTP 요청 속성과 일치하는 WAF 보안 규칙을 생성합니다. | | | ✓ | ✓ |
| JavaScript 챌린지 | | | 예 | 예 |
| 브라우저가 합법적인 사용자를 확인하는 JavaScript 챌린지를 완료하도록 요구하여 자동화된 위협을 차단합니다. | | | ✓ | ✓ |
| 봇 관리 및 분석 | | | 예 | 예 |
| 일반적인 봇에 대해 AWS WAF 봇 컨트롤을 사용하여 봇 트래픽을 감지하고 분석합니다. 검색 엔진과 같이 확인된 봇을 식별하고 구분하면서 확인되지 않은 봇을 차단, 챌린지 또는 허용하는 제어를 제공합니다. | | | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 사용자 지정 WAF 응답 | | 예 | 예 | 예 |
| 규칙에 의해 요청이 차단될 때 특정 HTTP 상태 코드와 선택적 사용자 지정 HTML, 일반 텍스트 또는 JSON 응답을 설정합니다. | | ✓ | ✓ | ✓ |
| 헤더 삽입 | 예 | 예 | 예 | 예 |
| WAF 검사를 통과하는 요청에 사용자 지정 HTTP 헤더를 추가하여 다운스트림 애플리케이션이 요청을 다르게 처리하거나 분석을 위해 플래그를 지정할 수 있도록 합니다. | ✓ | ✓ | ✓ | ✓ |
| 본문 검사 요청 | 16 KB | 16 KB | 64 KB | 64 KB |
| AWS WAF에서 위협 및 악성 패턴을 검사할 수 있는 HTTP 요청 본문 콘텐츠의 최대 크기입니다. | | | | |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| VPC 내 프라이빗 오리진<br><br>VPC 오리진을 사용하여 애플리케이션을 VPC 프라이빗 서브넷에 유지하고, CloudFront 배포를 통해서만 액세스할 수 있도록 하고, 퍼블릭 인터넷에서 숨겨 보안을 강화합니다. | | | 예 | 예 |
| 상호 TLS(오리진)<br><br>TLS 기반 클라이언트 인증서를 사용하여 애플리케이션(오리진)에 대한 무단 액세스를 제한합니다. 그러면 승인된 CloudFront 배포만 애플리케이션에 대한 연결을 설정할 수 있습니다. | | | 예 | 예 |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 오리진 액세스 제어(OAC) | 예 | 예 | 예 | 예 |
| 프라이빗 S3 버킷을 유지하고 지정된 CloudFront 배포를 통해서만 액세스를 허용하여 콘텐츠가 CloudFront 배포에 구성된 WAF 규칙, 속도 제한 및 기타 보안 제어로 보호되도록 합니다. | ✓ | ✓ | ✓ | ✓ |
| 무료 TLS 인증서 | 예 | 예 | 예 | 예 |
| AWS Certificate Manager를 통한 자동 갱신이 포함된 도메인의 무료 TLS 인증서입니다. | ✓ | ✓ | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 서명된 URL | 예 | 예 | 예 | 예 |
| 특정 사용자에게 프라이빗 콘텐츠에 대한 임시 액세스를 제공하는 보안 URL을 생성합니다. 일반적으로 승인된 사용자와 프라이빗 문서를 공유하거나 결제 확인 후 보호된 콘텐츠에 대한 보안 액세스 권한을 부여하는 데 사용됩니다. | ✓ | ✓ | ✓ | ✓ |
| 상호 TLS(mTLS) | | | | 예 |
| mTLS 인증을 사용하여 애플리케이션에 대한 액세스를 제한하여 유효한 인증서가 있는 신뢰할 수 있는 클라이언트만 연결할 수 있도록 합니다. | | | | ✓ |
| 엣지 컴퓨팅 | | | | |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 서버리스 엣지 컴퓨팅 | 예 | 예 | 예 | 예 |
| CloudFront Functions를 사용하면 엣지에서 경량 JavaScript를 실행하여 URL, HTTP 헤더 및 요청/응답 요소를 밀리초 만에 수정할 수 있습니다. | ✓ | ✓ | ✓ | ✓ |
| 엣지 키-값 저장소 | | 예 | 예 | 예 |
| CloudFront Functions를 사용하여 엣지에 데이터를 저장하면 KeyValueStore를 사용하여 콘텐츠를 빠르고 동적으로 사용자 지정할 수 있습니다. | | ✓ | ✓ | ✓ |
| 네트워크 및 프로토콜 지원 | | | | |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| IPv6 | 예 | 예 | 예 | 예 |
| CloudFront에서 뷰어 및 오리진으로의 최신 IPv6 및 기존 IPv4 연결을 통해 콘텐츠를 제공합니다. 애플리케이션에 대한 엔드 투 엔드 IPv6 지원을 활성화합니다. | ✓ | ✓ | ✓ | ✓ |
| HTTP/2 | 예 | 예 | 예 | 예 |
| 멀티플렉싱, 헤더 압축 및 스트림 우선 순위 지정과 같은 최신 프로토콜 기능을 통해 페이지를 더 빠르게 로드합니다. 브라우저 및 클라이언트에서 지원하는 경우 자동으로 사용됩니다. | ✓ | ✓ | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| HTTP/3 | 예 | 예 | 예 | 예 |
| 콘텐츠를 지원하는 브라우저 및 클라이언트에 QUIC를 통해 콘텐츠를 제공하여 더 빠른 연결과 향상된 성능을 지원합니다. 이는 특히 모바일 사용자에게 도움이 되며 네트워크 조건이 변경되어도 연결을 유지합니다. | ✓ | ✓ | ✓ | ✓ |
| TLS 1.3 | 예 | 예 | 예 | 예 |
| TLS 1.2에서 두 번의 왕복이 필요한 핸드셰이크 프로세스와 달리, 한 번의 왕복만으로 더 빠른 HTTPS 연결을 제공합니다. 이전 TLS 버전에 비해 첫 번째 바이트 지연 시간을 최대 33% 줄입니다. 애플리케이션의 엔드투엔드 기능을 지원합니다. | ✓ | ✓ | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| WebSocket | 예 | 예 | 예 | 예 |
| 브라우저와 서버 간에 지속적인 실시간 양방향 통신을 활성화합니다. AI 채팅 애플리케이션, 멀티플레이어 게임, 협업 워크스페이스, 금융 거래 플랫폼과 같은 실시간 데이터 피드에 적합합니다. | ✓ | ✓ | ✓ | ✓ |
| 로깅 및 모니터링 | | | | |
| 액세스 로그 | | 예 | 예 | 예 |
| 자세한 CloudFront 요청 로그에 액세스하여 보안 및 전송 트래픽 패턴을 이해할 수 있으며 Amazon CloudWatch Logs 수집이 추가 비용 없이 포함됩니다. | | ✓ | ✓ | ✓ |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| WAF 요청 로그 | | 예 | 예 | 예 |
| 세부 AWS WAF 요청 로그에 액세스하여 보안 및 전송 트래픽 패턴을 파악합니다. Amazon CloudWatch Logs 수집은 추가 비용 없이 포함됩니다. | | ✓ | ✓ | ✓ |
| 보안 대시보드 | 예 | 예 | 예 | 예 |
| 보안 규칙을 작성하지 않고 시각적 분석을 사용하여 보안 이벤트를 모니터링하고, 위협을 조사하고, 즉각적인 차단 조치를 취합니다. 프로 이상에는 로그를 쿼리하지 않고도 트래픽 패턴을 빠르게 이해할 수 있는 시각적 로그 분석기가 포함되어 있습니다. | ✓ | ✓ | ✓ | ✓ |
| DNS | | | | |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| Amazon Route 53 DNS<br><br>Route 53을 사용하는 빠르고 안정적이며 신뢰할 수 있는 퍼블릭 DNS 서비스입니다. | 예 | 예 | 예 | 예 |
| 호스팅 영역당 레코드 수<br><br>호스팅 영역에 허용되는 최대 DNS 레코드 수입니다. | 50 | 100 | 1000 | 5000 |
| DNSSEC<br><br>공격자가 DNS 쿼리를 가로채고 방문자를 가짜 웹사이트로 리디렉션하는 DNS 스푸핑 및 중간자 공격으로부터 도메인을 보호합니다. DNS 레코드에 암호화 방식으로 서명하여 DNS 트래픽을 보호합니다. | 예 | 예 | 예 | 예 |
| 스토리지(%) | | | | |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| Amazon S3 스토리지<br><br>AWS 계정에서 S3 Standard 스토리지 비용을 상쇄하는 Amazon S3 스토리지 크레딧입니다. CloudFront 콘텐츠 또는 플랜 사용 허용량에 국한되지 않습니다. | 5 GB | 50 GB | 1 TB | 5 TB |
| 지원 및 신뢰성 | | | | |
| 연중무휴 계정 및 결제 지원<br><br>계정 및 결제 관련 질문에 일대일 대응합니다. | 예 | 예 | 예 | 예 |
| 유료 지원 플랜이 있는 경우 모든 고정 요금제에 대한 지원을 받을 수 있습니다. | | | | |

| 성능 및 제공 | 무료 | Pro | 업무 | Premium |
| --- | --- | --- | --- | --- |
| 설명서 및 AWS Support 포럼 | 예 | 예 | 예 | 예 |
| 제품 설명서, 기술 문서, 모범 사례 가이드, AWS re:Post 커뮤니티 포럼 및 서비스 상태 정보에 액세스하여 계획을 세우고 문제를 해결하는 데 도움이 됩니다. | ✓ | ✓ | ✓ | ✓ |
| 가동 시간 SLA | | | 예 | 예 |
| Amazon CloudFront, AWS WAF, Amazon Route 53 및 Amazon CloudWatch에 대한 서비스 수준 계약(SLA)은 서비스 가용성 약정을 제공합니다. AWS가 관련 SLA의 약정을 충족하지 않는 경우 서비스 크레딧을 받을 수 있습니다. | | | ✓ | ✓ |

# 월별 사용 허용량

각 고정 요금제에는 해당 티어에서 최적의 성능을 제공하도록 설계된 월별 사용량 허용량이 포함됩니다. CloudFront 콘솔에서 언제든지 사용 허용량을 추적할 수 있습니다. 허용량의 50%, 80%, 100%에 도달하면 자동 이메일 알림도 받게 됩니다(알림이 지연될 수 있음).

월별 사용 허용량이 요청 및 데이터 전송 모두에서 기준 트래픽을 수용할 수 있는 플랜을 선택합니다. 허용량을 초과하는 경우 초과 요금이 발생하지 않습니다. 이를 통해 예상치 못한 트래픽 급증 또는 공격으로 인한 비용을 걱정하지 않고도 애플리케이션을 운영할 수 있습니다. 플랜의 기능을 확장하거나 기준 트래픽이 변경된 경우 다음 티어로 업그레이드하여 더 많은 기능에 액세스하고 월별 사용량 허용량을 늘리세요. 사용량이 CloudFront 정액 요금제의 허용량을 초과하는 경우 AWS는 성능 저하(예: 더 적거나 더 먼 엣지 로케이션에서 트래픽 처리, 처리량 감소 또는 제한) 또는 요금 구조 변경 요구 등 적절한 조치를 취할 수 있습니다.

애플리케이션의 기준 사용량이 500M 요청 또는 월별 50TB를 초과하는 경우 사용자 지정 요금을 문의하세요.

플랜 티어당 월별 사용 허용량

| | 무료 | Pro | 업무 | Premium |
|---|---|---|---|---|
| 요청 | 1M | 10M | 125M | 500M |
| 데이터 전송 | 100GB | 50TB | 50TB | 50TB |

> **Note**
> AWS WAF로 차단된 DDoS 공격 및 요청은 사용 허용량에 포함되지 않습니다.

## 과거 사용량에 따른 자격

과거 CloudFront 사용량은 특정 플랜 티어에 가입하거나 다운그레이드할 수 있는 자격에 영향을 미칠 수 있습니다. 최근 사용량이 플랜 티어의 사용 허용량을 초과하는 경우 워크로드에 더 잘 맞는 상위 티어를 선택해야 할 수 있습니다.

# 플랜에서 부담하는 비용

플랜은 다음에 대한 비용을 부담합니다.

- CloudFront 배포
- AWS WAF 웹 ACL의 웹 배포 연결
- 배포의 CloudFront 액세스 로그 및 관련 WAF 로그에 대한 CloudWatch Logs 수집
- 배포 계획에 연결된 Route 53 호스팅 영역, DNS 레코드 및 DNS 쿼리

또한 S3 버킷이 CloudFront 배포의 오리진으로 사용되는지 여부에 관계없이 지급인 계정에서 S3 Standard 스토리지 사용량을 상쇄하는 S3 크레딧을 받게 됩니다.

## Route 53 DNS 관리 및 플랜

DNS에 Route 53를 사용하고 해당 영역을 플랜에 연결하는 경우 고정 요금제에 Route 53 호스팅 영역 비용이 포함될 수 있습니다. CloudFront 배포의 플랜 관리 섹션에서 해당 영역을 플랜에 연결할 수 있습니다. 영역이 플랜에 연결되면 플랜은 월별 호스팅 영역 요금, DNS 레코드 및 아래에 제공된 티어당 각 허용량이 적용되는 DNS 쿼리 요금을 포함하여 호스팅 영역의 표준 비용을 부담합니다. 호스팅 영역은 다음 요구 사항을 충족해야 합니다.

- CloudFront 배포와 동일한 AWS 계정에 존재
- 플랜 티어의 호스팅 영역당 허용되는 레코드 수 유지
- CloudFront 배포에서 사용하는 도메인 포함

호스팅 영역이 플랜에 연결되지 않은 경우 모든 표준 Route 53 비용을 사용자가 부담하는 종량제 요금이 적용됩니다.

### 월별 DNS 쿼리 허용량 이해

호스팅 영역이 플랜에 연결되면 다음을 얻을 수 있습니다.

1. CloudFront 배포 및 기타 지원되는 AWS 서비스를 가리키는 ALIAS 레코드에 대한 DNS 쿼리
2. 다른 DNS 레코드 유형에 대한 추가 월별 허용량

| | 무료 | Pro | 업무 | Premium |
|---|---|---|---|---|
| 월별 ALIAS 레코드(CloudFront 및 기타 지원되는 | 제한 없음 | 제한 없음 | 제한 없음 | 제한 없음 |

| | 무료 | Pro | 업무 | Premium |
|---|---|---|---|---|
| AWS 서비스)에 대한 DNS 쿼리 | | | | |
| 월별 추가 DNS 쿼리 허용량 | 1M | 5M | 20M | 100M |

> **Note**
> 플랜 혜택을 극대화하려면 ALIAS 레코드를 사용하여 CloudFront 배포를 지정합니다. CloudFront 및 기타 지원되는 AWS 서비스를 지정하는 ALIAS 레코드는 월별 DNS 쿼리 허용량에 포함되지 않습니다. CloudFront에 대한 CNAME 레코드를 포함한 다른 모든 DNS 쿼리는 DNS 쿼리 허용량에 포함됩니다.

### DNS 쿼리 허용량 초과

DNS 쿼리 사용량이 플랜의 월별 허용량을 초과하는 경우 AWS에서 알림을 보낼 수 있습니다. 이때 CloudFront 배포의 플랜 관리 섹션에 있는 플랜에서 호스팅 영역을 분리하여 호스팅 영역을 종량제 요금으로 되돌릴 수 있습니다. 이 알림을 받은 후 호스팅 영역을 분리하지 않으면 AWS는 호스팅 영역을 종량제 요금으로 자동 전환할 수 있습니다. 호스팅 영역이 종량제 요금으로 전환되면 모든 표준 Route 53 비용은 사용자가 부담합니다. CloudFront 배포 및 기타 모든 플랜 혜택은 변경되지 않습니다.

## 요금제로 전체 AWS 비용 절감

CloudFront 정액 요금제는 다음과 같은 세 가지 방법으로 전체 AWS 비용을 절감할 수 있습니다.

먼저 CloudFront와 Amazon S3, AWS Application Load Balancer(ALB) 또는 Amazon API Gateway와 같은 서비스에서 실행되는 AWS 애플리케이션 간의 데이터 전송 비용이 자동으로 면제됩니다. 인터넷에 직접 접속하는 대신 CloudFront를 통해 AWS 애플리케이션을 제공하는 경우 고정 요금제에서는 초과 요금 걱정 없이 간단한 월별 요금으로 애플리케이션과 뷰어 간의 데이터 전송 비용을 충당합니다.

둘째, CloudFront는 애플리케이션 인프라를 보호하고 오리진에 도달하는 요청 수를 줄여 컴퓨팅 및 데이터베이스 비용을 절감합니다. 엣지 로케이션 또는 리전 엣지 캐시에서 캐시된 콘텐츠를 제공하고, 중복 요청을 축소하고, 백엔드 서비스에 도달하기 전에 악성 및 원치 않는 트래픽을 차단합니다. 즉, 사용

량에 따라 요금이 부과되는 애플리케이션 서버, 데이터베이스 및 기타 AWS 서비스에 도달하는 요청이 줄어들어 비용이 절감됩니다.

마지막으로 각 플랜에는 Amazon S3 Standard 스토리지 크레딧이 포함되어 AWS 계정의 스토리지 사용량을 상쇄합니다.

이러한 절감 효과를 극대화하려면 CloudFront의 트래픽만 허용하도록 AWS 오리진을 구성합니다. S3의 경우 프라이빗 버킷과 함께 오리진 액세스 제어(OAC)를 사용하여 지정된 CloudFront 배포에 대한 액세스 권한을 부여합니다. 프라이빗 서브넷의 Application Load Balancer, Network Load Balancer 및 Amazon EC2 인스턴스의 경우 VPC 오리진을 사용하여 지정된 CloudFront 배포에 대한 액세스를 제한합니다.

# 정액 요금제 관리

CloudFront 콘솔에서 다음 절차에 따라 배포에 대한 요금제를 구독, 업그레이드, 다운그레이드 또는 취소합니다.

## 요금제에 대한 새 배포 구독

새 배포를 생성할 때 요금제를 구독할 수 있습니다.

요금제에 대한 새 배포를 구독하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 배포를 선택한 다음 단계에 따라 배포를 생성합니다.
3. 배포의 요금제를 선택합니다. 요금제 티어에 따라 일부 기능은 사용할 수 없습니다. 플랜당 기능을 검토하고 애플리케이션에 필요한 요금제를 선택합니다.
4. 배포 생성 단계를 완료합니다.

## 기존 배포에서 요금제 구독

배포를 업데이트할 때 요금제를 구독할 수 있습니다. 요금제를 선택하기 전에 배포 구성이 원하는 요금제와 호환되는지 확인합니다.

> **Tip**
> 현재 배포에서 지원되지 않는 기능을 사용하는 경우 요금제를 구독하려면 먼저 해당 기능을 비활성화해야 합니다. 여기에는 Lambda@Edge 또는 실시간 액세스 로그와 같은 기능 비활성화가 포함됩니다.

배포 구성이 호환되면 배포를 업데이트하는 동안 원하는 요금제를 선택할 수 있습니다.

기존 배포에서 요금제를 구독하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 배포를 선택한 다음 단계에 따라 기존 배포를 업데이트합니다.
3. 배포의 요금제를 선택합니다. 요금제 티어에 따라 일부 기능은 사용할 수 없습니다. 플랜당 기능을 검토하고 애플리케이션에 필요한 요금제를 선택합니다.
4. 배포 업데이트 단계를 완료합니다.

## 요금제 업그레이드

월별 사용량 허용량에 가까워지거나 초과했거나 다음 티어에서 사용할 수 있는 기능을 활성화하려는 경우 플랜을 업그레이드하는 것이 좋습니다.

더 높은 플랜 티어로 업그레이드하면 변경 사항이 즉시 적용됩니다. 가격 및 사용 허용량은 비례 배분됩니다. 배포 및 관련 리소스는 새 플랜의 사용 가능한 기능과 더 높은 사용 허용량에 액세스할 수 있습니다.

요금제를 업그레이드하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 기존 요금제를 구독하는 배포를 선택합니다.
4. 안내에 따라 배포의 요금제를 업그레이드합니다.
5. 기존 배포 업데이트 단계를 완료합니다.

# 요금제 다운그레이드

기존 티어에 추가 기능이 필요하지 않은 경우 더 낮은 플랜 티어로 다운그레이드하는 것이 좋습니다. 예를 들어 애플리케이션에서 트래픽이 낮아질 것으로 예상되는 경우 다운그레이드할 수 있습니다.

하위 티어로 다운그레이드하면 다음 결제 주기가 시작될 때 결제 변경 사항이 적용됩니다.

배포가 현재 플랜의 사용 허용량을 초과하는 경우 사용량이 원하는 티어의 사용 허용량 내에 있으면 다운그레이드할 수 있습니다. 다음 결제 주기에서 기존 플랜 티어에 대한 요금이 청구되지 않도록 하려면 월말 이전에 다운그레이드하세요.

요금제를 다운그레이드하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 기존 요금제를 구독하는 배포를 선택합니다.
4. 안내에 따라 배포의 요금제를 다운그레이드합니다. 지원되지 않는 기능이 있는 경우 배포에서 해당 기능 또는 리소스를 제거해야 합니다.
5. 기존 배포 업데이트 단계를 완료합니다.

# 요금제 취소

요금제를 취소하면 현재 결제 주기가 끝날 때까지 고정 요금제를 유지할 수 있습니다. 그러면 배포 및 모든 관련 플랜 리소스가 다음 결제 주기가 시작될 때 종량제 요금으로 전환됩니다.

요금제를 취소하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 기존 요금제를 구독하는 배포를 선택합니다.
4. 안내에 따라 배포의 요금제를 취소합니다. 지원되지 않는 기능이 있는 경우 배포에서 해당 기능 또는 리소스를 제거해야 합니다.
5. 기존 배포 업데이트 단계를 완료합니다.

## 보류 중인 플랜 변경 취소

정액 요금제를 다운그레이드하거나 취소한 경우 현재 결제 주기가 끝날 때까지 기다려야 변경 사항이 적용됩니다. 기존 정액 요금제를 유지하거나 요금제를 업그레이드 또는 다운그레이드하려면 먼저 보류 중인 요금제 변경을 취소해야 합니다.

보류 중인 요금제 변경을 취소하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 기존 요금제를 구독하는 배포를 선택합니다.
4. 안내에 따라 배포의 보류 중인 플랜 변경을 취소합니다.
5. 배포에 사용할 요금제를 선택합니다.
6. 기존 배포 업데이트 단계를 완료합니다.

## 요금제를 사용하는 배포 삭제

요금제를 구독하는 배포는 삭제할 수 없습니다. 먼저 요금제를 취소한 다음 현재 결제 주기 후에 배포를 삭제해야 합니다.

요금제를 사용하는 배포를 삭제하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 이전 단계에 따라 배포의 요금제를 취소합니다.
4. 단계에 따라 배포를 삭제합니다.

> **Note**
>
> 요금제를 구독하는 배포를 비활성화할 수 있지만 해당 요금제에 대한 요금은 계속 발생합니다. 요금제에 대한 요금 발생을 중지하려면 먼저 요금제를 취소해야 합니다.

## 권한

CloudFront 배포에 대한 요금제 구독을 보거나 관리하려면 필요한 권한이 있어야 합니다. 자세한 내용은 AWS 관리형 정책: CloudFrontFullAccess 및 AWS 관리형 정책: CloudFrontReadOnlyAccess(을)를 참조하세요.

## 정액 요금제 할당량

다음 표에는 CloudFront 정액 요금제에 대한 할당량 및 제한 사항이 나와 있습니다.

> **Note**
> AWS 계정의 이러한 할당량은 늘릴 수 없습니다.

| 계정 수준 할당량 | 할당량 |
| --- | --- |
| AWS 계정당 요금제 | 100 |
| AWS 계정당 무료 요금제 | 3 |
| 계획당 Apex 수준 도메인 | 1 |

## 지원되지 않는 기능

배포를 요금제와 연결하려면 먼저 특정 기능을 비활성화하고 연결을 제거해야 합니다.

> **참고**
>
> - 배포 또는 계정에 이러한 제한 사항이 있는 경우 요금제를 사용하려면 먼저 해결해야 합니다. 배포를 변경한 후 변경 사항이 모든 엣지 로케이션에 전파될 때까지 기다립니다.
> - 요금제를 사용하는 경우 배포와 연결된 AWS WAF 웹 ACL이 있어야 합니다. 이 리소스는 해당 배포의 종량제 요금으로 전환하지 않는 한 배포에서 제거하거나 연결 해제할 수 없습니다.

# 지원되지 않는 기능

구성에 다음과 같은 미지원 기능이 포함된 경우 요금제를 구독할 수 없습니다. 지원되지 않는 기능을 비활성화하고 대체 옵션을 사용하거나 배포에 종량제 요금을 유지할 수 있습니다.

| 지원되지 않는 기능 | 대체 옵션 | AWS 서비스 |
| --- | --- | --- |
| 다중 테넌트 배포 | 표준 배포 또는 종량제 요금 사용 | CloudFront |
| 지속적 배포 및 스테이징 배포 | 종량제 요금 사용 | CloudFront |
| 애니캐스트 IP 목록 구성 | 종량제 요금 사용 | CloudFront |
| 실시간 액세스 로그 | 표준 액세스 로그 또는 종량제 요금 사용 | CloudFront |
| Lambda@Edge 함수 | CloudFront Functions 또는 종량제 요금 사용 | CloudFront |
| 대상 봇 | 일반 봇 또는 종량제 요금 사용 | AWS WAF |
| CAPTCHA | 챌린지 또는 종량제 요금 사용 | AWS WAF |
| 파트너 관리형 규칙 | 종량제 요금 사용 | AWS WAF |
| 계정 생성 사기 방지 | 종량제 요금 사용 | AWS WAF |
| 계정 탈취 방지 | 종량제 요금 사용 | AWS WAF |
| 규칙 그룹 | 개별 규칙 생성(규칙 그룹은 CloudFront의 정책과 마찬가지로 웹 ACL에 적용할 수 있는 공유 AWS WAF 규칙임) | AWS WAF |

| 지원되지 않는 기능 | 대체 옵션 | AWS 서비스 |
| --- | --- | --- |
| 레거시 기능 | | |
| ForwardedValues 구성 | 오리진 요청 정책 사용 | CloudFront |
| 전용 IP/SSL | 종량제 요금 사용 | CloudFront |
| 필드 레벨 암호화 | 종량제 요금 사용 | CloudFront |
| AWS Identity and Access Management(IAM) 서버 인증서 | AWS Certificate Manager(ACM) 인증서 사용 | CloudFront |
| 원본 액세스 ID(OAI) | 오리진 액세스 제어(OAC) 사용 | CloudFront |
| 레거시 캐시 설정 | 캐시 정책 및 오리진 요청 정책을 사용합니다. | CloudFront |

## 지원되지 않는 연결

배포가 다른 배포에 이미 연결된 다음 리소스와 연결되어 있는 경우 요금제를 구독할 수 없습니다. 요금제를 구독하는 배포와 연결된 리소스는 해당 배포에만 사용할 수 있습니다. 예를 들어 키 값 저장소를 사용하는 CloudFront Functions가 있는 경우, 요금제에 있는 배포에 대해 함수와 키 값 저장소를 공유할 수 없습니다.

- CloudFront 함수
- 키 값 저장소를 CloudFront Functions와 연결할 수 있습니다.
- AWS WAF 웹 ACL

요금제를 구독하려면 관련 리소스를 제거하거나 다른 리소스로 대체합니다.

## 계정 수준 제약 조건

다음 조건 중 하나라도 충족하는 경우 AWS 계정이 요금제를 이용할 수 없습니다.

- 최대 이벤트 구독 수에 도달했습니다. 정액 요금제 할당량을(를) 참조하세요.
- 계정이 AWS 프리 티어를 사용하고 있습니다.

## 리소스 수준 제약 조건

다음 조건 중 하나라도 충족하는 경우 배포가 요금제를 이용할 수 없습니다.

- 배포에 AWS Shield Advanced가 활성화되었습니다.
- 배포에서 웹 ACL에 대해 Firewall Manager 서비스를 활성화했습니다. Firewall Manager는 요금제에서 CloudFront 배포의 WebACL을 관리하지 않습니다.

## 요금제에 영향을 미칠 수 있는 추가 기능

정액 요금제를 사용하면 CloudFront 배포 및 요금제에 포함되고 CloudFront 배포와 연결된 위에 나열된 기능에 대해 고정 요금을 지불할 수 있습니다. 다른 모든 기능에는 다음을 포함하되 이에 국한되지 않는 추가 요금이 발생할 수 있습니다.

### Route 53

- Route 53 DNSSEC에는 AWS KMS 비용이 발생합니다.
- Route 53 IP(CIDR) 블록(처음 1,000개는 AWS 계정당 무료)
- Route 53 상태 확인(처음 50개는 AWS 계정당 무료)

### 로깅 기능

- Route 53 DNS 쿼리 로그, CloudFront Functions 로그 및 CloudFront 연결 함수 로그
- Amazon S3로 AWS WAF 로그 전송
- CloudFront 또는 AWS WAF 로그를 Amazon Data Firehose로 전송
- CloudFront에 대한 추가 CloudWatch 지표
- Parquet 형식의 CloudFront 액세스 로그

> **Note**
>
> 플랜에는 추가 비용 없이 CloudFront 표준 로그(액세스 로그) 및 WAF 로그에 대한 Amazon CloudWatch Logs 수집이 포함됩니다. 스토리지 및 쿼리와 같은 다른 모든 CloudWatch 비용은 플랜에서 부담하지 않습니다. 다른 모든 로그도 별도로 청구됩니다.

> **Note**
>
> 요금제에는 Route 53의 퍼블릭 권한 DNS가 포함되어 있습니다. Route 53 호스팅 영역이 요금제에 연결되면 플랜은 월별 호스팅 영역 요금, DNS 레코드 및 티어당 각 허용량이 적용되는 DNS 쿼리 요금을 포함하여 호스팅 영역의 표준 비용을 부담합니다. 플랜에 포함된 위에 나열되지 않은 Route 53 사용 및 기능으로 인한 다른 모든 비용은 플랜에서 부담하지 않습니다.

## 요금제와 종량제 요금 비교

정액제 플랜과 종량제 요금은 필요에 따라 다양한 이점을 제공합니다. 고정 요금제를 사용하면 CloudFront, AWS WAF, Route 53, CloudWatch Logs 수집과 같은 여러 AWS 서비스를 포함하는 단일 요금을 지불하고 트래픽 급증 또는 공격 중에도 초과 요금이 발생하지 않습니다.

종량제 요금을 사용하면 실제 사용량에 따라 각 서비스 및 기능에 대해 별도로 요금이 청구됩니다. 이를 통해 서비스 선택 및 구성에 대해 완전한 유연성을 얻을 수 있지만 트래픽 패턴에 따라 비용이 매월 다를 수 있으므로 비용을 관리하기 위해 여러 서비스의 사용량을 모니터링해야 합니다.

정액제 플랜은 초과 요금에 대한 걱정 없이 월별 결제, 간소화된 서비스 구성 및 기본 보안 기능을 결합하려는 경우에 적합합니다. 개별 서비스 기능, 사용자 지정 구성, 고정 요금제에서 사용할 수 없는 기능에 대한 액세스를 완전히 제어해야 하거나 대규모의 예측 가능한 트래픽 급증을 처리할 것으로 예상되는 경우 종량제 요금을 선택하는 것이 좋습니다. Amazon CloudFront 정액 요금제는 다른 오퍼, 프로모션 또는 할인과 결합할 수 없습니다.

# 배포 구성

Amazon CloudFront 배포를 생성하여 CloudFront에 어디로부터 콘텐츠를 전송하고자 하는지와 이러한 콘텐츠 전송을 추적 및 관리하는 방법에 대한 세부 정보를 알립니다.

CloudFront 배포를 만들면 CloudFront가 콘텐츠 오리진 유형에 따라 대부분의 배포 설정을 자동으로 구성합니다. 재구성 설정에 대한 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하시기 바랍니다. 선택적으로 배포 설정을 수동으로 편집하도록 선택할 수 있습니다. 자세한 내용은 모든 배포 설정 참조 섹션을 참조하세요.

다음과 같은 설정을 구성할 수 있습니다.

- 콘텐츠 오리진 - CloudFront가 배포할 파일을 가져오는 Amazon S3 버킷, AWS Elemental MediaPackage 채널, AWS Elemental MediaStore 컨테이너, Elastic Load Balancing 로드 밸런서 또는 HTTP 서버. 배포당 최대 25개 오리진의 조합을 지정할 수 있습니다.
- 액세스 - 파일을 모든 사람이 사용할 수 있도록 할지 아니면 일부 사용자로만 액세스를 제한할지 여부
- 보안 - AWS WAF 보호를 활성화하고 사용자가 HTTPS를 사용하여 콘텐츠에 액세스하도록 지정할지 여부 다중 테넌트 배포의 경우 AWS WAF V2 웹 액세스 제어 목록(ACL)만 지원됩니다.
- 캐시 키 - 캐시 키에 포함할 값(있는 경우) 캐시 키는 지정된 배포에 대한 캐시의 각 파일을 고유하게 식별합니다.
- 오리진 요청 설정 - CloudFront가 오리진에 전송하는 요청에 HTTP 헤더, 쿠키 또는 쿼리 문자열을 포함할지 여부
- 지리적 제한 - CloudFront에서 특정 국가의 사용자가 콘텐츠에 액세스하는 것을 차단할지 여부
- 로그 - CloudFront가 뷰어 활동을 표시하는 표준 로그 또는 실시간 액세스 로그를 생성할지 여부입니다.

자세한 내용은 모든 배포 설정 참조 섹션을 참조하세요.

현재 각 AWS 계정에 대해 생성할 수 있는 배포의 최대 수는 배포의 일반 할당량 섹션을 참조하세요. 배포당 제공할 수 있는 최대 파일 수는 없습니다.

배포를 사용하여 HTTP 또는 HTTPS를 통해 다음 콘텐츠를 제공할 수 있습니다.

- HTTP 또는 HTTPS를 사용하는 정적 및 동적 다운로드 콘텐츠입니다(예: HTML, CSS, JavaScript 및 이미지 파일).

- Apple HTTP Live Streaming(HLS) 및 Microsoft Smooth Streaming 등과 같은 다양한 형식의 온디맨드 비디오. (다중 테넌트 배포의 경우 스무스 스트리밍이 지원되지 않음) 자세한 내용은 CloudFront를 사용한 온디맨드 비디오 제공 섹션을 참조하세요.
- 실시간으로 발생하는 모임, 회의 또는 콘서트 같은 라이브 이벤트 제공. 라이브 스트리밍의 경우 CloudFormation 스택을 사용하여 배포를 자동으로 생성할 수 있습니다. 자세한 내용은 CloudFront 및 AWS Media Services를 사용하여 동영상 스트리밍 제공 섹션을 참조하세요.

다음 주제에서는 CloudFront 배포에 대한 자세한 내용과 해당 비즈니스 요건에 맞게 배포를 구성하는 방법을 제공합니다. 배포 생성에 대한 자세한 내용은 배포 생성 단원을 참조하세요.

주제


- 다중 테넌트 배포의 작동 방식 이해
- 배포 생성
- 사전 구성된 배포 설정 참조
- 모든 배포 설정 참조
- 배포 테스트
- 배포 업데이트
- 배포 태깅
- 배포 삭제
- CloudFront 배포에 다양한 오리진 사용
- CloudFront 배포에 IPv6 활성화
- CloudFront 지속적 배포를 사용하여 CDN 구성 변경을 안전하게 테스트
- 대체 도메인 이름(CNAME)을 추가하여 사용자 지정 URL 사용
- CloudFront 배포에 WebSocket 사용
- 허용 목록에 사용할 애니캐스트 고정 IP 요청
- CloudFront 배포에 gRPC 사용


# 다중 테넌트 배포의 작동 방식 이해

여러 배포 테넌트에서 재사용할 수 있는 설정을 사용하여 CloudFront 다중 테넌트 배포를 만들 수 있습니다. 다중 테넌트 배포를 사용하면 CloudFront가 콘텐츠 오리진 유형에 따라 배포 설정을 구성하도록 할 수 있습니다. 재구성 설정에 대한 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하시기 바랍니다.

표준 배포 대신 다중 테넌트 배포를 사용하면 다음과 같은 이점이 있습니다.

- 운영 부담을 줄입니다.
- 웹 관리자 및 소프트웨어 공급자가 최종 사용자에게 콘텐츠를 제공하는 여러 웹 애플리케이션에서 CloudFront 배포를 관리하기 위한 재사용 가능한 구성입니다.
- 자동화된 인증서 관리, 통합 보안 제어 및 규모에 맞는 간편한 구성 제어를 제공하기 위해 다른 AWS 서비스와의 통합을 개선했습니다.
- 구현 전반에 걸쳐 일관된 리소스 패턴을 유지합니다. 공유해야 하는 설정을 정의한 다음, 재정의할 설정을 사용자 지정합니다.
- 배포 테넌트 수준에서 특정 요구 사항을 충족하도록 사용자 지정 가능한 오리진 및 보안 설정입니다.
- 배포 테넌트를 서로 다른 계층에 정리합니다. 예를 들어 일부 배포 테넌트에 Origin Shield가 필요하고 일부는 필요하지 않은 경우 배포 테넌트를 서로 다른 다중 테넌트 배포로 그룹화할 수 있습니다.
- 여러 도메인에서 공통 DNS 구성을 공유합니다.

표준 배포와 달리 다중 테넌트 배포에는 CloudFront 라우팅 엔드포인트가 없으므로 직접 액세스할 수 없습니다. 따라서 연결 그룹 및 하나 이상의 배포 테넌트와 함께 사용해야 합니다. 표준 배포에는 자체 CloudFront 엔드포인트가 있으며 최종 사용자가 직접 액세스할 수 있지만 템플릿으로 다른 배포에 사용할 수 없습니다.

다중 테넌트 배포 할당량에 대한 자세한 내용은 다중 테넌트 배포의 할당량 섹션을 참조하시기 바랍니다.

주제


- 작동 방식
- 용어
- 지원되지 않는 기능
- 배포 테넌트 사용자 지정
- CloudFront 배포 테넌트에 대한 인증서 요청
- 사용자 지정 연결 그룹 생성(선택 사항)
- 다중 테넌트 배포로 마이그레이션

# 작동 방식

표준 배포의 배포에는 오리진 구성, 캐시 동작, 보안 설정 등 웹 사이트 또는 애플리케이션에서 사용 설정하려는 모든 설정이 포함됩니다. 별도의 웹 사이트를 만들고 동일한 설정을 많이 사용하려면 매번 새 배포를 만들어야 합니다.

CloudFront 다중 테넌트 배포는 초기 다중 테넌트 배포를 만들 수 있다는 점에서 다릅니다. 각 새 웹 사이트에서 소스 배포의 정의된 값을 자동으로 상속하는 배포 테넌트를 만듭니다. 그런 다음 배포 테넌트에 대한 특정 설정을 사용자 지정합니다.

개요

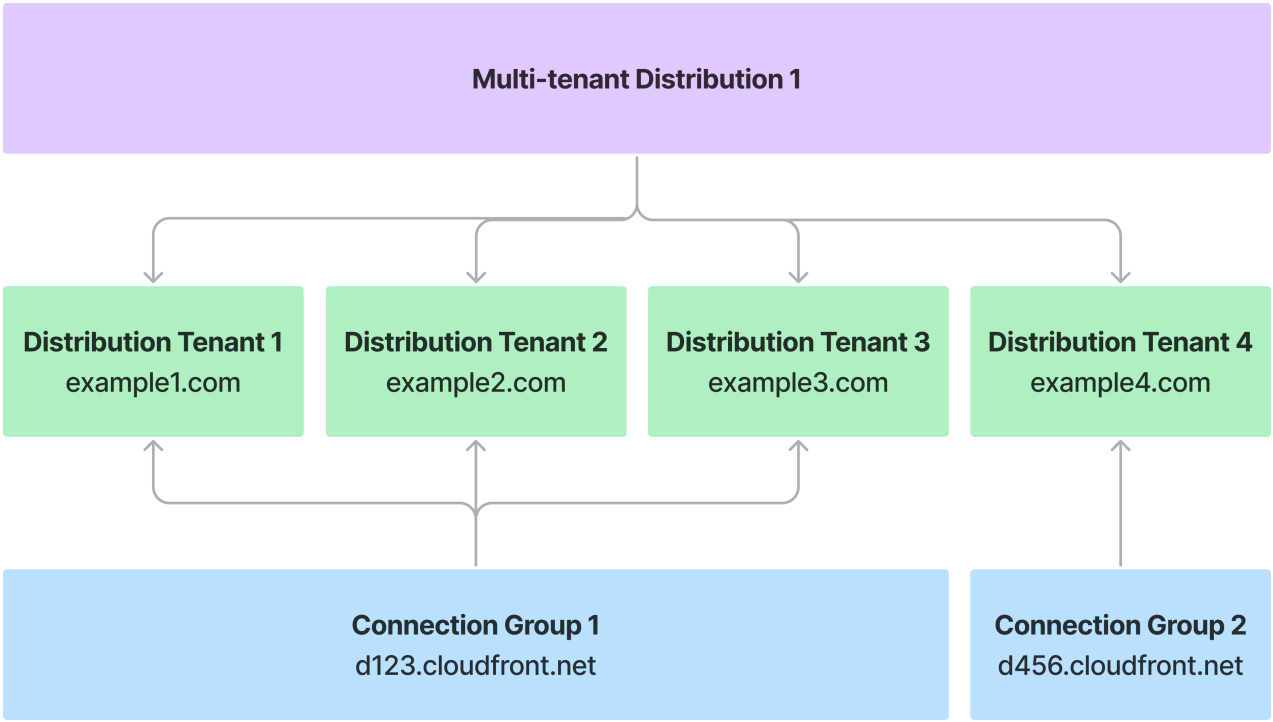
1. 시작하려면 먼저 다중 테넌트 배포를 만듭니다. CloudFront는 콘텐츠 오리진 유형에 따라 배포 설정을 구성합니다. VPC 오리진을 제외한 모든 오리진에서 설정을 사용자 지정할 수 있습니다. VPC 오리진 설정은 VPC 오리진 리소스 자체에서 사용자 지정됩니다. 사용자 지정할 수 있는 다중 테넌트 배포 설정에 대한 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하시기 바랍니다.
   - 다중 테넌트 배포에 사용하는 TLS 인증서는 배포 테넌트에서 상속될 수 있습니다. 다중 테넌트 배포 자체는 라우팅할 수 없으므로 연결된 도메인 이름이 없습니다.
2. 기본적으로 CloudFront에서는 연결 그룹을 만듭니다. 연결 그룹은 콘텐츠에 대한 뷰어 요청이 CloudFront에 연결되는 방식을 제어합니다. 연결 그룹에서 일부 라우팅 설정을 사용자 지정할 수 있습니다.

   연결 그룹을 직접 수동으로 만들어 이를 변경할 수 있습니다. 자세한 내용은 사용자 지정 연결 그룹 생성(선택 사항) 섹션을 참조하세요.
3. 그런 다음 하나 이상의 배포 테넌트를 만듭니다. 배포 테넌트는 뷰어가 콘텐츠에 액세스할 수 있는 '출입구'입니다. 각 배포 테넌트는 다중 테넌트 배포를 참조하며 CloudFront에서 만든 연결 그룹과 자동으로 연결됩니다. 배포 테넌트는 개별 도메인 또는 하위 도메인을 지원합니다.
4. 그런 다음 가상 도메인 및 오리진 경로 등 일부 배포 테넌트 설정을 사용자 지정할 수 있습니다. 자세한 내용은 배포 테넌트 사용자 지정 섹션을 참조하세요.
5. 마지막으로 DNS 호스트의 DNS 레코드를 업데이트하여 트래픽을 배포 테넌트로 라우팅해야 합니다. 이렇게 하려면 연결 그룹에서 CloudFront 엔드포인트 값을 가져오고 CloudFront 엔드포인트를 가리키는 CNAME 레코드를 만듭니다.

Example예제

다음 그래픽은 다중 테넌트 배포, 배포 테넌트, 연결 그룹이 함께 작동하여 여러 도메인의 뷰어에게 콘텐츠를 제공하는 방법을 보여줍니다.

1. 다중 테넌트 배포는 각 배포 테넌트에서 상속된 설정을 정의합니다. 다중 테넌트 배포를 템플릿으로 사용합니다.
2. 다중 테넌트 배포에서 만든 각 배포 테넌트에는 자체 도메인이 있습니다.
3. 배포 테넌트는 다중 테넌트 배포를 만들 때 CloudFront에서 만든 연결 그룹에 자동으로 추가됩니다. 연결 그룹은 뷰어 요청이 CloudFront 네트워크에 연결되는 방식을 제어합니다.




자세한 다중 테넌트 배포 만들기 설명은 콘솔에서 CloudFront 배포 생성 섹션을 참조하시기 바랍니다.

## 용어

다음 개념은 다중 테넌트 배포의 구성 요소를 설명합니다.

다중 테넌트 배포

캐시 동작, 보안 보호 및 오리진을 포함하여 모든 배포 테넌트에 대한 모든 공유 구성 설정을 지정하는 블루프린트 멀티 테넌트 배포입니다. 다중 테넌트 배포는 트래픽을 직접 제공할 수 없습니다. 연결 그룹 및 배포 테넌트와 함께 사용해야 합니다.

표준 배포

다중 테넌트 기능이 없는 배포입니다. 이러한 배포는 단일 웹 사이트 또는 앱을 지원하는 데 가장 적합합니다.

배포 테넌트

배포 테넌트는 다중 테넌트 배포 구성을 상속합니다. 일부 구성 설정은 배포 테넌트 수준에서 사용자 지정할 수 있습니다. 배포 테넌트 도메인 또는 하위 도메인을 포함하면 배포 테넌트에는 다중 테넌트 배포에서 상속할 수 있는 유효한 TLS 인증서가 있어야 합니다.

배포 테넌트는 연결 그룹과 연결되어야 합니다. CloudFront는 배포 테넌트를 만들 때 연결 그룹을 만들고 해당 연결 그룹에 배포 테넌트를 자동으로 할당합니다.

멀티테넌시

구성과 인프라를 공유하면서 다중 테넌트 배포를 사용하여 여러 도메인의 콘텐츠를 제공할 수 있습니다. 이 접근 방식을 사용하면 여러 도메인(테넌트라고 함)이 자체 사용자 지정을 유지하면서 다중 테넌트 배포의 공통 설정을 공유할 수 있습니다.

연결 그룹

뷰어에게 콘텐츠를 제공하는 CloudFront 라우팅 엔드포인트를 제공합니다. 배포 테넌트 도메인 또는 하위 도메인에 대해 만든 CNAME 레코드에 해당하는 CloudFront 라우팅 엔드포인트를 가져오려면 각 배포 테넌트를 연결 그룹에 연결해야 합니다. 연결 그룹은 여러 배포 테넌트 간에 공유할 수 있습니다. 연결 그룹은 IPv6 및 애니캐스트 IP 목록 설정 등의 배포 테넌트의 라우팅 설정을 관리합니다.

파라미터

오리진 경로 및 도메인 이름과 같은 자리 표시자 값의 키-값 페어 목록입니다. 다중 테넌트 배포에서 파라미터를 정의하고 배포 테넌트 수준에서 해당 파라미터의 값을 제공할 수 있습니다. 배포 테넌트에 파라미터 값을 입력해야 하는지 여부를 선택합니다.

배포 테넌트에서 선택 사항 파라미터 값을 제공하지 않으면 다중 테넌트 배포의 기본값이 값으로 사용됩니다.

CloudFront 라우팅 엔드포인트

연결 그룹의 표준 DNS입니다(예: `d123.cloudfront.net`). 배포 테넌트 도메인 또는 하위 도메인의 CNAME 레코드에 사용됩니다.

사용자 지정

배포 테넌트가 다중 테넌트 배포와 다른 설정을 사용하도록 사용자 지정할 수 있습니다. 각 배포 테넌트에서 다른 AWS WAF 웹 액세스 제어 목록(ACL), TLS 인증서 및 지리적 제한 사항을 지정할 수 있습니다.

## 지원되지 않는 기능

다음 기능은 다중 테넌트 배포에 사용할 수 없습니다. 표준 배포와 동일한 설정을 사용하여 새 멀티 테넌트 배포를 만들면 일부 설정을 사용할 수 없으니 참고하시기 바랍니다.

> **참고**
>
> - 현재 AWS Firewall Manager 정책은 표준 배포에만 적용됩니다. Firewall Manager는 향후 릴리스에서 다중 테넌트 배포에 대한 지원을 추가할 예정입니다.
> - 표준 배포와 달리 배포 테넌트 수준에서 도메인 이름(별칭)을 지정합니다. 자세한 내용은 CloudFront 배포 테넌트에 대한 인증서 요청 및 CreateDistributionTenant API 작업을 확인하시기 바랍니다.

- 지속적 배포
- 오리진 액세스 ID(OAI) - 오리진 액세스 제어(OAC)를 대신 사용합니다.
- 전용 IP 사용자 지정 SSL 지원 - `sni-only` 메서드만 지원됩니다.
- AWS WAF 클래식(V1) 웹 ACL - AWS WAF V2 웹 ACL만 지원됩니다.
- 표준 로깅(레거시)
- 최소 TTL
- 기본 TTL
- 최대 TTL
- ForwardedValues
- PriceClass
- 신뢰할 수 있는 서명자
- 스무스 스트리밍
- AWS Identity and Access Management(IAM) 서버 인증서

- 전용 IP 주소
- 최소 프로토콜 버전 SSLv3

다음 설정은 다중 테넌트 배포 또는 배포 테넌트에서 구성할 수 없습니다. 대신 연결 그룹에서 원하는 값을 설정합니다. 연결 그룹에 연결된 모든 배포 테넌트는 이러한 설정을 사용합니다. 자세한 내용은 사용자 지정 연결 그룹 생성(선택 사항) 섹션을 참조하세요.

- IPv6 활성화(뷰어 요청)
- 애니캐스트 고정 IP 목록

## 배포 테넌트 사용자 지정

다중 테넌트 배포를 사용하는 경우 배포 테넌트는 다중 테넌트 배포 구성을 상속합니다. 하지만 배포 테넌트 수준에서 일부 설정을 사용자 지정할 수 있습니다.

다음을 사용자 지정할 수 있습니다.

- 파라미터 - 파라미터는 오리진 도메인 또는 원본 경로에 사용할 수 있는 키-값 페어입니다. 파라미터가 배포 테넌트와 작동하는 방식을(를) 참조하세요.
- AWS WAF 웹 ACL(V2) - 배포 테넌트에 대해 별도의 웹 ACL을 지정할 수 있습니다. 그러면 다중 테넌트 배포에 사용되는 웹 ACL이 재정의됩니다. 특정 배포 테넌트에 대해 이 설정을 사용 해제할 수도 있습니다. 즉, 배포 테넌트는 다중 테넌트 배포에서 웹 ACL 보호를 상속하지 않습니다. 자세한 내용은 AWS WAF 웹 ACL 섹션을 참조하세요.
- 지리적 제한 - 배포 테넌트에 지정하는 지리적 제한은 다중 테넌트 배포에 대한 모든 지리적 제한을 재정의합니다. 예를 들어 다중 테넌트 배포에서 독일(DE)을 차단하면 모든 연결된 배포 테넌트도 DE를 차단합니다. 그러나 특정 배포 테넌트에 대해 DE를 허용하면 해당 배포 테넌트 설정이 다중 테넌트 배포에 대한 설정을 재정의합니다. 자세한 내용은 콘텐츠의 지리적 배포 제한 섹션을 참조하세요.
- 무효화 경로 - 배포 테넌트에 대해 무효화하려는 콘텐츠의 파일 경로를 지정합니다. 자세한 내용은 파일 무효화 섹션을 참조하세요.
- 사용자 지정 TLS 인증서 - 배포 테넌트에 지정하는 AWS Certificate Manager(ACM) 인증서는 다중 테넌트 배포에 제공된 인증서를 보완합니다. 하지만 동일한 도메인이 다중 테넌트 배포 및 배포 테넌트 인증서 모두에 포함되는 경우 테넌트 인증서가 사용됩니다. 자세한 내용은 CloudFront 배포 테넌트에 대한 인증서 요청 섹션을 참조하세요.
- 도메인 이름 - 배포 테넌트당 하나 이상의 도메인 이름을 지정해야 합니다.

## 파라미터가 배포 테넌트와 작동하는 방식

파라미터는 자리 표시자 값에 사용할 수 있는 키-값 페어입니다. 다중 테넌트 배포에 사용할 파라미터를 정의하고 필요 여부를 지정합니다.

다중 테넌트 배포에서 파라미터를 정의할 때 해당 파라미터를 배포 테넌트 수준에서 입력해야 하는지 여부를 선택합니다.

- 다중 테넌트 배포에서 필요에 따라 파라미터를 정의하는 경우 배포 테넌트 수준에서 입력해야 합니다. (상속되지 않음)
- 파라미터가 필요하지 않은 경우 배포 테넌트에서 상속된 다중 테넌트 배포에 기본값을 제공할 수 있습니다.

다음 속성에 파라미터를 사용할 수 있습니다.

- 오리진 도메인 이름
- 오리진 경로

다중 테넌트 배포에서는 위의 각 속성에 대해 최대 2개의 파라미터를 정의할 수 있습니다.

## 파라미터 예시

도메인 이름 및 오리진 경로에 파라미터를 사용하려면 다음 예시를 참조하세요.

### 도메인 이름 파라미터

다중 테넌트 배포 구성에서 다음 예시와 같이 오리진 도메인 이름에 대한 파라미터를 정의할 수 있습니다.

#### Amazon S3

- `{{parameter1}}.amzn-s3-demo-logging-bucket.s3.us-east-1.amazonaws.com`
- `{{parameter1}}-amzn-s3-demo-logging-bucket.s3.us-east-1.amazonaws.com`

#### 사용자 지정 오리진

- `{{parameter1}}.lambda-url.us-east-1.on.aws`
- `{{parameter1}}.mediapackagev2.ap-south-1.amazonaws.com`

배포 테넌트를 생성할 때 *parameter1*에 사용할 값을 지정합니다.

```
"Parameters": [
  {
    "Name": "parameter1",
    "Value": "mycompany-website"
  }
]
```

다중 테넌트 배포에 지정된 이전 예시를 사용하면 배포 테넌트의 오리진 도메인 이름이 다음으로 확인됩니다.

- *mycompany-website*.amzn-s3-demo-bucket3.s3.us-east-1.amazonaws.com
- *mycompany-website*-amzn-s3-demo-bucket3.s3.us-east-1.amazonaws.com
- *mycompany-website*.lambda-url.us-east-1.on.aws
- *mycompany-website*.mediapackagev2.ap-south-1.amazonaws.com

### 오리진 경로 파라미터

마찬가지로 다음 예시와 같이 다중 테넌트 배포에서 오리진 경로에 대한 파라미터를 정의할 수 있습니다.

- /{{*parameter2*}}
- /{{*parameter2*}}/test
- /public/{{*parameter2*}}/test
- /search?name={{*parameter2*}}

배포 테넌트를 생성할 때 *parameter2*에 사용할 값을 지정합니다.

```
"Parameters": [
  {
    "Name": "parameter2",
    "Value": "myBrand"
  }
]
```

다중 테넌트 배포에 지정된 이전 예시를 사용하면 배포 테넌트의 오리진 경로가 다음으로 확인됩니다.

- /*myBrand*
- /*myBrand*/test
- /public/*myBrand*/test
- /search?name=*myBrand*

Example예제

고객을 위해 여러 웹 사이트(테넌트)를 만들려는 경우 각 배포 테넌트 리소스가 올바른 값을 사용하는지 확인해야 합니다.

1. 다중 테넌트 배포를 만들고 배포 테넌트 구성에 2개의 파라미터를 포함합니다.
2. 오리진 도메인 이름에서 *customer-name*이라는 파라미터를 만들고 필수로 지정합니다. S3 버킷 앞에 파라미터를 입력하면 다음과 같이 표시됩니다.

   {{*customer-name*}}.amzn-s3-demo-bucket3.s3.us-east-1.amazonaws.com.
3. 오리진 경로에서 *my-theme*이라는 두 번째 파라미터를 만들고 기본값이 *##*인 선택 사항을 지정합니다. 오리진 경로는 /{{*my-theme*}}으로 표시됩니다.
4. 배포 테넌트를 만들 때는 다음과 같이 합니다.
   - 도메인 이름의 경우 다중 테넌트 배포에서 필수로 표시되는 *customer-name*의 값을 지정해야 합니다.
   - 오리진 경로의 경우 선택적으로 *my-theme*의 값을 지정하거나 기본값을 사용할 수 있습니다.

## CloudFront 배포 테넌트에 대한 인증서 요청

배포 테넌트를 만들면 테넌트는 다중 테넌트 배포에서 공유 AWS Certificate Manager(ACM) 인증서를 상속합니다. 이 공유 인증서는 다중 테넌트 배포와 연결된 모든 테넌트에 HTTPS를 제공합니다.

CloudFront 배포 테넌트를 만들거나 업데이트하여 도메인을 추가할 때 ACM에서 관리형 CloudFront 인증서를 추가할 수 있습니다. 그런 다음 CloudFront는 사용자를 대신하여 ACM에서 HTTP 검증 인증서를 가져옵니다. 이 테넌트 수준 ACM 인증서를 사용자 지정 도메인 구성에 사용할 수 있습니다. CloudFront는 갱신 워크플로를 간소화하여 인증서를 최신 상태로 유지하고 콘텐츠 전송을 중단 없이 보호합니다.

> **Note**
> 인증서를 소유하고 있지만 CloudFront 리소스에만 사용할 수 있으며 프라이빗 키는 내보낼 수 없습니다.

배포 테넌트를 만들거나 업데이트할 때 인증서를 요청할 수 있습니다.

주제

- 도메인 및 인증서 추가(분산 테넌트)
- 도메인 설정 완료
- 도메인을 CloudFront로 가리키기
- 도메인 고려 사항(배포 테넌트)
- 와일드카드 도메인(배포 테넌트)

## 도메인 및 인증서 추가(분산 테넌트)

다음 절차에서는 도메인을 추가하고 배포 테넌트의 인증서를 업데이트하는 방법을 보여줍니다.

도메인 및 인증서를 추가하려면(배포 테넌트) 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. SaaS에서 배포 테넌트를 선택합니다.
3. 배포 테넌트를 검색합니다. 검색 창의 드롭다운 메뉴를 사용하여 도메인, 이름, 배포 ID, 인증서 ID, 연결 그룹 ID 또는 웹 ACL ID를 기준으로 필터링합니다.
4. 배포 테넌트 이름을 선택합니다.
5. 도메인에서 도메인 관리를 선택합니다.
6. 인증서에서 배포 테넌트에 대한 사용자 지정 TLS 인증서를 원하는지 선택합니다. 인증서는 도메인 이름을 사용할 권한이 있는지 확인합니다. 미국 동부(버지니아 북부) 리전에서 인증서를 가져와야 합니다.
7. 도메인에서 도메인 추가를 선택하고 도메인 이름을 입력합니다. 도메인에 따라 입력한 도메인 이름 아래에 다음 메시지가 표시됩니다.

   - 이 도메인은 인증서의 적용을 받습니다.

- 이 도메인은 검증 보류 중인 인증서의 적용을 받습니다.
- 이 도메인은 인증서의 적용을 받지 않습니다. (즉, 도메인 소유권을 확인해야 합니다.)

8. 배포 테넌트 업데이트를 선택합니다.

   테넌트 세부 정보 페이지의 도메인에서 다음 필드를 볼 수 있습니다.

   - 도메인 소유권 - 도메인 소유권의 상태입니다. CloudFront가 콘텐츠를 제공할 수 있으려면 먼저 TLS 인증서 검증을 사용하여 도메인 소유권을 확인해야 합니다.
   - DNS 상태 - 트래픽을 올바르게 라우팅하려면 도메인의 DNS 레코드가 CloudFront를 가리켜야 합니다.

9. 도메인 소유권이 확인되지 않은 경우 테넌트 세부 정보 페이지의 도메인에서 도메인 설정 완료를 선택한 다음, 다음 절차를 완료하여 DNS 레코드가 CloudFront 도메인 이름을 가리키도록 합니다.

## 도메인 설정 완료

다음 절차에 따라 배포 테넌트의 도메인을 소유하고 있는지 확인합니다. 도메인에 따라 다음 절차 중 하나를 선택합니다.

> **Note**
>
> 도메인이 이미 Amazon Route 53 별칭 레코드가 있는 CloudFront를 가리키는 경우 도메인 이름 앞에 `_cf-challenge.`를 사용하여 DNS TXT 레코드를 추가해야 합니다. 이 TXT 레코드는 도메인 이름이 CloudFront에 연결되어 있는지 확인합니다. 각 도메인에 대해 이 단계를 반복합니다. 다음은 TXT 레코드를 업데이트하는 방법을 보여줍니다.
>
> - 레코드 이름: `_cf-challenge.`*`DomainName`*
> - 레코드 유형: TXT
> - 레코드 값: *`CloudFrontRoutingEndpoint`*
>
> 예를 들어 TXT 레코드는 `_cf-challenge.example.com TXT d111111abcdef8.cloudfront.net`과 같을 수 있습니다.
> CloudFront 라우팅 엔드포인트는 콘솔의 배포 테넌트 세부 정보 페이지에서 찾거나 Amazon CloudFront API 참조의 ListConnectionGroups API 작업을 사용하여 찾을 수 있습니다.

> **Tip**
>
> SaaS 공급자이고 고객(테넌트)이 DNS에 TXT 레코드를 직접 추가할 필요 없이 인증서를 발급하도록 허용하려면 다음을 수행합니다.
>
> 1. 도메인 `example-saas-provider.com`을 소유한 경우 `customer-123.example-saas-provider.com`과 같이 테넌트에 하위 도메인을 할당합니다.
> 2. DNS에서 DNS 구성에 `_cf-challenge.customer-123.example-saas-provider.com TXT d111111abcdef8.cloudfront.net` TXT 레코드를 추가합니다.
> 3. 그런 다음 고객(테넌트)은 자체 DNS 레코드를 업데이트하여 도메인 이름을 제공된 하위 도메인에 매핑할 수 있습니다.
>
> `www.customer-domain.com CNAME customer-123.example-saas-provider.com`

I have existing traffic

도메인이 가동 중지 시간을 허용할 수 없는 경우 이 옵션을 선택합니다. 오리진/웹 서버에 액세스할 수 있어야 합니다. 다음 절차에 따라 도메인 소유권을 검증합니다.

기존 트래픽이 있을 때 도메인 설정을 완료하려면 다음과 같이 합니다.

1. 웹 트래픽 지정에서 기존 트래픽이 있음을 선택한 후 다음을 선택합니다.
2. 도메인 소유권 확인에서 다음 옵션 중 하나를 선택합니다.
   - 기존 인증서 사용 - 기존 ACM 인증서를 검색하거나 나열된 도메인을 포함하는 인증서 ARN을 입력합니다.
   - 수동 파일 업로드 - 웹 서버에 파일을 업로드할 수 있는 직접 액세스 권한이 있는 경우 선택합니다.

     각 도메인에 대한 토큰 위치에서 검증 토큰이 포함된 일반 텍스트 파일을 만들고 기존 서버의 지정된 파일 경로에서 오리진에 업로드합니다. 예를 들어 이 파일에 대한 경로는 다음(`/.well-known/pki-validation/acm_9c2a7b2ec0524d09fa6013efb73ad123.txt`)과 같을 수 있습니다. 이 단계를 완료하면 ACM은 토큰을 확인한 다음, 도메인에 대한 TLS 인증서를 발급합니다.
   - HTTP 리디렉션 - 웹 서버에 파일을 업로드할 수 있는 직접 액세스 권한이 없거나 CDN 또는 프록시 서비스를 사용하는 경우 선택합니다.

각 도메인에서 기존 서버에 301 리디렉션을 만듭니다. 다음에서 리디렉션에서 잘 알려진 경로를 복사하고 다음으로 리디렉션에서 지정된 인증서 엔드포인트를 가리킵니다. 리디렉션은 다음 예제와 비슷할 수 있습니다.

```
If the URL matches: example.com/.well-known/pki-validation/
leabe938a4fe077b31e1ff62b781c123.txt
Then the settings are:Forwarding URL
Then 301 Permanent Redirect:To validation.us-east-1.acm-
validations.aws/123456789012/.well-known/pki-validation/
leabe938a4fe077b31e1ff62b781c123.txt
```

> **Note**
> 인증서 상태 확인을 선택하여 ACM이 도메인에 대한 인증서를 발급하는 경우를 확인할 수 있습니다.

3. 다음을 선택합니다.
4. 도메인을 CloudFront로 가리키기의 단계를 완료합니다.

I don't have traffic

새 도메인을 추가하는 경우 이 옵션을 선택합니다. CloudFront에서 인증서 검증을 관리합니다.

트래픽이 없는 경우 도메인 설정을 완료하려면 다음과 같이 합니다.

1. 웹 트래픽 지정에서 아직 트래픽이 없음을 선택합니다.
2. 각 도메인 이름에서 도메인을 CloudFront로 가리키기의 단계를 완료합니다.
3. 각 도메인 이름에서 DNS 레코드를 업데이트한 후 다음을 선택합니다.
4. 인증서가 발급될 때까지 기다립니다.

> **Note**
> 인증서 상태 확인을 선택하여 ACM이 도메인에 대한 인증서를 발급하는 경우를 확인할 수 있습니다.

5. 제출을 선택합니다.

## 도메인을 CloudFront로 가리키기

DNS 레코드를 업데이트하여 각 도메인의 트래픽을 CloudFront 라우팅 엔드포인트로 라우팅합니다. 여러 도메인 이름을 가질 수 있지만 모두 이 엔드포인트로 확인되어야 합니다.

도메인을 CloudFront로 가리키려면 다음과 같이 합니다.

1. d111111abcdef8.cloudfront.net 등의 CloudFront 라우팅 엔드포인트 값을 복사합니다.
2. DNS 레코드를 업데이트하여 각 도메인의 트래픽을 CloudFront 라우팅 엔드포인트로 라우팅합니다.
   1. 도메인 등록 또는 DNS 공급자의 관리 콘솔에 로그인합니다.
   2. 도메인의 DNS 관리 섹션으로 이동합니다.
      - 하위 도메인 - CNAME 레코드를 만듭니다. 예제:
        - 이름 - 하위 도메인(예: `www` 또는 `app`)
        - 값/대상 - CloudFront 라우팅 엔드포인트
        - 레코드 유형 - CNAME
        - TTL – 3600(또는 사용 사례에 적합한 것)
      - apex/루트 도메인의 경우 - 루트 또는 apex 도메인 수준에서는 표준 CNAME 레코드를 사용할 수 없으므로 고유한 DNS 구성이 필요합니다. 대부분의 DNS 공급자는 ALIAS 레코드를 지원하지 않기 때문에 Route 53에서 ALIAS 레코드를 생성하는 것이 좋습니다. 예제:
        - 이름 - apex 도메인(예: `example.com`)
        - 레코드 유형 - A
        - 별칭 - 예
        - 별칭 대상 - CloudFront 라우팅 엔드포인트
        - 라우팅 정책 - 단순(또는 사용 사례에 적합한 것)
   3. DNS 변경 사항이 채워졌는지 확인합니다. (이는 일반적으로 TTL이 만료될 때 발생합니다. 경우에 따라 24~48시간이 걸릴 수 있습니다.) `dig` 또는 `nslookup` 등의 도구를 사용합니다.

      ```
      dig www.example.com
      # Should eventually return a CNAME pointing to your CloudFront routing endpoint
      ```

3. CloudFront 콘솔로 이동하여 제출을 선택합니다. 도메인이 활성 상태이면 CloudFront는 도메인이 트래픽을 처리할 준비가 되었음을 나타내도록 도메인 상태를 업데이트합니다.

자세한 내용은 DNS 공급자의 설명서를 참조하세요.

- Cloudflare
- ClouDNS
- DNSimple
- Gandi.net
- GoDaddy
- Google Cloud DNS
- Namecheap

## 도메인 고려 사항(배포 테넌트)

도메인이 활성 상태이면 도메인 제어가 설정되고 CloudFront는 이 도메인에 대한 모든 뷰어 요청에 응답합니다. 활성화되면 도메인을 비활성화하거나 비활성 상태로 변경할 수 없습니다. 이미 사용 중인 동안에는 도메인을 다른 CloudFront 리소스와 연결할 수 없습니다. 도메인을 다른 배포와 연결하려면 UpdateDomainAssociation 요청을 사용하여 도메인을 한 CloudFront 리소스에서 다른 CloudFront 리소스로 이전합니다.

도메인이 비활성 상태이면 CloudFront는 도메인에 대한 뷰어 요청에 응답하지 않습니다. 도메인이 비활성 상태일 때 다음 사항에 유의합니다.

- 보류 중인 인증서 요청이 있는 경우 CloudFront는 잘 알려진 경로에 대한 요청에 응답합니다. 요청이 보류 중인 동안에는 도메인을 다른 CloudFront 리소스와 연결할 수 없습니다.
- 보류 중인 인증서 요청이 없는 경우 CloudFront는 도메인에 대한 요청에 응답하지 않습니다. 도메인을 다른 CloudFront 리소스와 연결할 수 있습니다.
- 배포 테넌트당 보류 중인 인증서 요청은 하나만 가질 수 있습니다. 추가 도메인에 대해 다른 인증서를 요청하려면 먼저 보류 중인 기존 요청을 취소해야 합니다. 기존 인증서 요청을 취소해도 연결된 ACM 인증서는 삭제되지 않습니다. ACM API를 사용하여 인증서를 삭제할 수 있습니다.
- 배포 테넌트에 새 인증서를 적용하면 이전 인증서의 연결이 해제됩니다. 인증서를 재사용하여 다른 배포 테넌트의 도메인에 적용할 수 있습니다.

DNS 검증 인증서 갱신과 마찬가지로 인증서 갱신이 성공하면 알림이 전송됩니다. 하지만 사용자는 조치를 취할 필요가 없습니다. CloudFront는 도메인의 인증서 갱신을 자동으로 관리합니다.

> **Note**
>
> 인증서 리소스를 만들거나 업데이트하기 위해 ACM API 작업을 호출할 필요가 없습니다. CreateDistributionTenant 및 UpdateDistributionTenant API 작업을 사용하여 관리형 인증서 요청에 대한 세부 정보를 지정하여 인증서를 관리할 수 있습니다.

### 와일드카드 도메인(배포 테넌트)

다음과 같은 경우 배포 테넌트에 와일드카드 도메인이 지원됩니다.

- 상위 다중 테넌트 배포에서 상속된 공유 인증서에 와일드카드가 포함된 경우
- 배포 테넌트에 유효한 기존 사용자 지정 TLS 인증서를 사용하는 경우

## 사용자 지정 연결 그룹 생성(선택 사항)

기본적으로 CloudFront는 다중 테넌트 배포를 만들 때 연결 그룹을 만듭니다. 연결 그룹은 콘텐츠에 대한 뷰어 요청이 CloudFront에 연결되는 방식을 제어합니다.

기본값 연결 그룹을 그대로 사용하는 것이 좋습니다. 그러나 엔터프라이즈 애플리케이션을 격리하거나 배포 테넌트 그룹을 별도로 관리해야 하는 경우 사용자 지정 연결 그룹을 만들도록 선택할 수 있습니다. 예를 들어 DDoS 공격이 발생하는 경우 배포 테넌트를 별도의 연결 그룹으로 이전해야 할 수 있습니다. 이렇게 하면 다른 배포 테넌트가 영향을 받지 않도록 보호할 수 있습니다.

### 사용자 지정 연결 그룹 생성(선택 사항)

필요에 따라 배포 테넌트에 대한 사용자 지정 연결 그룹을 만들도록 선택할 수 있습니다.

사용자 지정 연결 그룹을 만들려면 다음과 같이 합니다(선택 사항).

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 설정을 선택합니다.
3. 연결 그룹 설정을 켭니다.
4. 탐색 창에서 연결 그룹을 선택한 다음 연결 그룹 생성을 선택합니다.
5. 연결 그룹 이름에 연결 그룹의 이름을 입력합니다. 연결 그룹을 만든 후에는 이 이름을 업데이트할 수 없습니다.

6. IPv6에서 이 IP 프로토콜을 사용 설정할지 여부를 지정합니다. 자세한 내용은 IPv6 활성화(뷰어 요청) 섹션을 참조하세요.
7. 애니캐스트 고정 IP 목록에서, IP 주소 집합에서 배포 테넌트로 트래픽을 전송할지 여부를 지정합니다. 자세한 내용은 애니캐스트 고정 IP 목록을 참조하시기 바랍니다.
8. (선택 사항) 연결 그룹에 태그를 추가합니다.
9. 연결 그룹 생성을 선택합니다.

연결 그룹이 만들어지면 지정한 설정과 ARN 및 엔드포인트를 찾을 수 있습니다.

- ARN은 다음 예제(`arn:aws:cloudfront::123456789012:connection-group/cg_2uVbA9KeWaADTbKzhj9lcKDoM25`)와 비슷하게 표시됩니다.
- 엔드포인트는 다음 예제(d111111abcdef8.cloudfront.net)와 비슷하게 표시됩니다.

사용자 지정 연결 그룹을 만든 후 편집하거나 삭제할 수 있습니다. 연결 그룹을 삭제하려면 먼저 모든 연결된 배포 테넌트를 삭제해야 합니다. 다중 테넌트 배포를 만들 경우 CloudFront에서 만든 기본 연결 그룹은 삭제할 수 없습니다.

> **Important**
>
> 배포 테넌트의 연결 그룹을 변경하면 CloudFront는 배포 테넌트에 대한 트래픽을 계속 전달하지만 지연 시간이 증가합니다. 새 연결 그룹에서 CloudFront 라우팅 엔드포인트를 사용하도록 배포 테넌트의 DNS 레코드를 업데이트하는 것이 좋습니다.
> DNS 레코드를 업데이트할 때까지 CloudFront는 웹 사이트가 현재 DNS를 가리키고 있는 라우팅 엔드포인트에 대해 정의된 설정을 기반으로 라우팅합니다. 예를 들어 기본값 연결 그룹이 애니캐스트 고정 IP를 사용하지 않지만 새로운 사용자 지정 연결 그룹은 사용한다고 가정합니다. CloudFront가 사용자 지정 연결 그룹의 배포 테넌트에 애니캐스트 고정 IP를 사용하기 전에 DNS 레코드를 업데이트해야 합니다.

## 다중 테넌트 배포로 마이그레이션

CloudFront 표준 배포가 있으며 다중 테넌트 배포로 마이그레이션하려는 경우 다음 단계를 따르세요.

표준 배포에서 다중 테넌트 배포로 마이그레이션하려면

1. 지원되지 않는 기능을(를) 검토합니다.

2. 지원되지 않는 기능을 제외하고 표준 배포와 동일한 구성으로 다중 테넌트 배포를 생성합니다. 자세한 내용은 콘솔에서 CloudFront 배포 생성 섹션을 참조하세요.
3. 배포 테넌트를 생성하고 소유한 대체 도메인 이름을 추가합니다.

> **Warning**
>
> 표준 배포와 연결된 현재 도메인 이름은 사용하지 마세요. 대신 자리 표시자 도메인을 추가합니다. 나중에 도메인을 이동합니다. 배포 테넌트 생성에 대한 자세한 내용은 콘솔에서 CloudFront 배포 생성 섹션을 참조하세요.

4. 배포 테넌트 도메인에 대한 기존 인증서를 제공합니다. 이 인증서는 자리 표시자 도메인과 이동하려는 도메인에 적용됩니다.
5. 콘솔의 배포 테넌트 세부 정보 페이지에서 CloudFront 라우팅 엔드포인트를 복사합니다. 또는 Amazon CloudFront API 참조의 ListConnectionGroups API 작업을 사용하여 이를 찾을 수 있습니다.
6. 도메인 소유권을 확인하려면 배포 테넌트의 CloudFront 라우팅 엔드포인트를 가리키는 DCV TXT 레코드를 밑줄( _ ) 접두사를 사용하여 생성합니다. 자세한 내용은 도메인을 CloudFront로 가리키기 섹션을 참조하세요.
7. 변경 사항이 전파되면 이전에 표준 배포에 사용한 도메인을 사용하도록 배포 테넌트를 업데이트합니다.
   - 콘솔 - 자세한 지침은 도메인 및 인증서 추가(분산 테넌트) 섹션을 참조하세요.
   - API - Amazon CloudFront API 참조의 UpdateDomainAssociation API 작업을 사용합니다.

> **Important**
>
> 이렇게 하면 콘텐츠의 캐시 키가 재설정됩니다. 그런 다음 CloudFront는 새 캐시 키를 사용하여 콘텐츠 캐싱을 시작합니다. 자세한 내용은 캐시 키 이해 섹션을 참조하세요.

8. 도메인이 배포 테넌트의 CloudFront 라우팅 엔드포인트를 가리키도록 DNS 레코드를 업데이트합니다. 이 단계를 완료하면 도메인이 배포 테넌트에 트래픽을 제공할 준비가 됩니다. 자세한 내용은 도메인을 CloudFront로 가리키기 섹션을 참조하세요.
9. (선택 사항) 도메인을 배포 테넌트에 성공적으로 마이그레이션한 후에는 배포 테넌트의 도메인 이름에 적용되는 다른 CloudFront 관리형 인증서를 사용할 수 있습니다. 관리형 인증서를 요청하려면 별도의 TXT 레코드를 생성하여 인증서를 발급하고 도메인 설정 완료의 단계를 따릅니다.

# 배포 생성

이 항목에서는 CloudFront 콘솔을 사용하여 배포를 생성하는 방법을 설명합니다.

개요

1. 하나 이상의 Amazon S3 버킷을 만들거나 HTTP 서버를 오리진 서버로 구성합니다. 오리진은 콘텐츠의 원본 버전을 저장하는 장소입니다. CloudFront에서 파일에 대한 요청을 가져오면, 이 요청은 엣지 로케이션에서 배포할 파일을 가져오는 오리진이 됩니다. Amazon S3 버킷 및 HTTP 서버를 결합하여 오리진 서버로 사용할 수 있습니다.

   - Amazon S3를 사용 중인 경우 버킷의 이름은 전부 소문자로만 구성되어야 하며 공백을 포함할 수 없습니다.
   - Amazon EC2 서버나 다른 사용자 지정 서버를 사용 중인 경우, Amazon EC2(또는 기타 사용자 지정 오리진) 사용 섹션을 검토하세요.
   - 현재 하나의 배포에 대해 생성할 수 있는 오리진의 최대 수를 살펴보고 더 높은 할당량을 요청하려면 배포의 일반 할당량 섹션을 참조하세요.
2. 오리진 서버에 콘텐츠를 업로드합니다. 객체를 공개적으로 읽기 가능하도록 설정할 수도 있고, CloudFront 서명된 URL을 사용하여 콘텐츠에 대한 액세스를 제한할 수도 있습니다.

   > **Important**
   >
   > 이 경우, 오리진 서버의 보안을 보장하는 것은 사용자의 책임입니다. CloudFront에서 서버에 대한 액세스 권한을 보유해야 하며 콘텐츠를 안전하게 보호할 수 있도록 보안 설정이 적용되어 있어야 합니다.

3. CloudFront 배포 생성:

   - CloudFront 콘솔에서 배포를 생성하는 자세한 절차는 콘솔에서 CloudFront 배포 생성 섹션을 참조하세요.
   - CloudFront API를 사용하여 배포를 생성하는 방법에 대한 자세한 내용은 Amazon CloudFront API 참조의 CreateDistribution을 참조하세요.
4. (선택 사항) CloudFront 콘솔을 사용하여 배포를 생성한 경우, 캐시 동작 또는 배포에 대한 오리진을 추가로 생성합니다. 동작과 오리진에 대한 자세한 내용은 다중 테넌트 배포를 업데이트하려면 다음과 같이 합니다. 섹션을 참조하세요.
5. 배포를 테스트합니다. 테스트에 대한 자세한 내용은 배포 테스트 섹션을 참조하세요.

6. 3단계에서 배포를 만든 뒤 CloudFront에서 반환된 도메인 이름을 사용하여 콘텐츠에 액세스하는 웹 사이트 또는 애플리케이션을 작성합니다. 예를 들어 CloudFront에서 d111111abcdef8.cloudfront.net을 배포의 도메인 이름으로 반환할 경우, Amazon S3 버킷 또는 HTTP 서버의 루트 디렉터리에 있는 `image.jpg` 파일의 URL은 `https://d111111abcdef8.cloudfront.net/image.jpg`가 됩니다.

   배포를 만들 때 하나 이상의 대체 도메인 이름(CNAME)을 지정한 경우, 자체 도메인 이름을 사용할 수 있습니다. 이 경우 `image.jpg`에 대한 URL은 `https://www.example.com/image.jpg`가 될 수 있습니다.

   다음을 참조하십시오.

   - 서명된 URL을 사용하여 콘텐츠에 대한 액세스를 제한하려는 경우 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공 단원을 참조하십시오.
   - 압축된 콘텐츠를 제공하려는 경우 압축된 파일 제공 단원을 참조하십시오.
   - Amazon S3 및 사용자 지정 오리진에 대한 CloudFront 요청 및 응답 동작과 관련한 자세한 내용은 요청 및 응답 동작 단원을 참조하세요.

주제


- 콘솔에서 CloudFront 배포 생성
- CloudFront가 콘솔에 표시하는 값
- 추가 링크
- CloudFront 표준 배포에 도메인 추가


## 콘솔에서 CloudFront 배포 생성

배포를 만들면 CloudFront가 콘텐츠 오리진 유형에 따라 배포 설정을 구성합니다. 재구성 설정에 대한 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하시기 바랍니다. 여러 배포 테넌트에서 재사용할 수 있는 설정을 사용하여 다중 테넌트 배포를 만들 수도 있습니다. 자세한 내용은 다중 테넌트 배포의 작동 방식 이해 섹션을 참조하세요. 또는 자체 배포 설정을 수동으로 구성할 수 있습니다.

Multi-tenant

다중 테넌트 배포를 만들려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.

2. 탐색 창에서 배포를 선택한 후 배포 생성을 선택합니다.
3. 다중 테넌트 아키텍처, 다음을 선택합니다.
4. 다중 테넌트 배포에 배포 이름을 입력합니다. 이름은 Name 키 값으로 표시됩니다. 이 값은 나중에 변경할 수 있습니다. 다중 테넌트 배포에 최대 50개의 태그를 추가할 수 있습니다. 자세한 내용은 섹션을 참조하세요배포 태깅
5. (선택 사항) 와일드카드 인증서에서 루트 도메인 아래의 모든 하위 도메인을 포함하는 AWS Certificate Manager(ACM) 인증서(예: *.example.com)를 선택합니다. 미국 동부(버지니아 북부) 리전에서 인증서를 가져와야 합니다.
6. 다음을 선택합니다.
7. 오리진 지정 페이지에서 CloudFront에서 가져올 콘텐츠의 오리진 유형을 선택합니다. CloudFront는 다중 테넌트 배포의 해당 오리진 유형에 대한 권장 설정을 사용합니다. 권장 설정에 대한 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하시기 바랍니다.
8. 오리진의 선택한 오리진 유형에서 사용할 오리진을 선택하거나 입력합니다.
9. 오리진 경로에 슬래시(/) 문자를 입력한 다음 오리진 경로를 입력합니다.
10. (선택 사항) 파라미터를 추가하려면 오리진 도메인 이름 또는 오리진 경로 중 하나에서 파라미터 삽입을 선택합니다. 각 필드에 최대 2개의 파라미터를 입력할 수 있습니다.
    a. 새 파라미터 생성을 선택합니다.
    b. 새 파라미터 생성 대화 상자의 파라미터 이름에 파라미터의 고유한 이름과 설명(선택 사항)을 입력합니다.
    c. 필수 파라미터에서 확인란을 선택하여 배포 테넌트 수준에서 이 파라미터 값을 필수로 설정합니다. 필요하지 않은 경우 배포 테넌트가 상속할 기본값을 입력합니다.
    d. 파라미터 생성(Create parameter)을 선택합니다. 해당 필드에 이 파라미터가 나타납니다.
11. 옵션에서 다음 옵션 중 하나를 선택합니다.
    - 권장 오리진 설정 사용 - 선택한 오리진 유형의 기본 권장 캐시 및 오리진 설정을 사용합니다.
    - 오리진 설정 사용자 지정 - 캐시 및 오리진 설정을 사용자 지정합니다. 이 옵션을 선택하는 경우 표시되는 고유한 값을 지정합니다.
12. 다음을 선택합니다.
13. 보안 보호 활성화 페이지에서 AWS WAF 보안 보호 기능을 사용 설정할지 여부를 선택합니다. 나중에 특정 배포 테넌트에 대해 웹 ACL을 사용자 지정할 수 있습니다. 자세한 내용은 새 배포에 AWS WAF 활성화 섹션을 참조하세요.
14. 다음, 배포 생성을 차례로 선택합니다.

15. 배포 페이지의 리소스 목록에 다중 테넌트 배포가 나타납니다. 모든 배포 드롭다운을 선택하여 표준 배포 또는 다중 테넌트 배포별로 필터링할 수 있습니다. 또한 유형 열을 선택하여 표준 배포 또는 다중 테넌트 배포별로 필터링할 수 있습니다.

기본적으로 CloudFront에서는 연결 그룹을 만듭니다. 연결 그룹은 콘텐츠에 대한 뷰어 요청이 CloudFront에 연결되는 방식을 제어합니다. 연결 그룹에서 일부 라우팅 설정을 사용자 지정할 수 있습니다. 자세한 내용은 다중 테넌트 배포의 작동 방식 이해 섹션을 참조하세요.

다중 테넌트 배포를 템플릿으로 사용하여 추가 배포 테넌트를 만들 수 있습니다.

테넌트 배포를 만들려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 다음 중 하나를 수행합니다.
   - 배포를 선택하고 다중 테넌트 배포를 선택한 다음 테넌트 생성을 선택합니다.
   - 배포 테넌트를 선택한 다음 테넌트 생성을 선택합니다.
3. 배포 테넌트 이름에 이름을 입력합니다. AWS 계정에서 중복되지 않는 이름이어야 하며, 만들고 나면 변경할 수 없습니다.
4. 템플릿 배포의 목록에서 다중 테넌트 배포 ID를 선택합니다.
5. 태그 관리에서 배포 테넌트에 대해 최대 50개의 키-값 페어를 추가합니다. 자세한 내용은 배포 태깅 섹션을 참조하세요.
6. 다음을 선택합니다.
7. 도메인 추가 페이지의 인증서에서 배포 테넌트에 대한 사용자 지정 TLS 인증서를 원하는지 선택합니다. 인증서는 도메인 이름을 사용할 권한이 있는지 확인합니다. 미국 동부(버지니아 북부) 리전에서 인증서를 가져와야 합니다.
8. 도메인에 도메인의 이름을 입력합니다.

   > **Note**
   >
   > 인증서에 포함되지 않는 도메인 이름을 입력한 경우 도메인을 소유하고 있는지 확인해야 합니다. 지금은 배포 테넌트를 만들고 나중에 도메인 소유권을 확인할 수 있습니다. 자세한 내용은 CloudFront 배포 테넌트에 대한 인증서 요청 섹션을 참조하세요.

9. 다음을 선택합니다.

10. 파라미터 정의 페이지에 다중 테넌트 배포에 지정한 파라미터가 나타납니다. 필수 파라미터에서 파라미터 이름 옆에 값을 입력하고 변경 사항을 저장합니다.
11. 다른 파라미터를 추가하려면 파라미터 추가를 선택하고 이름과 값을 입력합니다.
12. 다음을 선택합니다.
13. (선택 사항) 보안 사용자 지정에서 배포 설정 재정의를 선택한 경우 사용 사례에 맞는 옵션을 선택합니다.
14. (선택 사항) 지리적 제한 사용자 지정에서 배포 설정 재정의를 선택한 경우 배포 테넌트에 적합한 제한 유형 및 국가를 선택합니다. 자세한 내용은 콘텐츠의 지리적 배포 제한 섹션을 참조하세요.
15. 다음을 선택합니다.
16. 배포 테넌트 생성을 선택합니다.

배포 테넌트 페이지에서 모든 배포 테넌트를 확인할 수 있습니다. 다음 항목을 기준으로 필터링할 수 있습니다.

연결

- 배포 ID
- 인증서 ID
- 연결 그룹 ID
- 웹 ACL ID

속성

- 이름
- 도메인

배포 테넌트를 편집하여 특정 설정을 사용자 지정할 수 있습니다. 자세한 내용은 배포 테넌트 사용자 지정 섹션을 참조하세요.

Standard

표준 배포를 만들려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.

2. 탐색 창에서 배포를 선택한 후 배포 생성을 선택합니다.
3. 표준 배포에 배포 이름을 입력합니다. 이름은 태그로서 Name 키의 값으로 표시됩니다. 이 값은 나중에 변경할 수 있습니다. 표준 배포에 최대 50개의 태그를 추가할 수 있습니다. 자세한 내용은 배포 태깅 섹션을 참조하세요.
4. 단일 웹사이트 또는 앱, 다음을 선택합니다.
5. (선택 사항) 도메인 설정에 AWS 계정에서 Route 53에 이미 등록된 도메인을 입력하거나 새 도메인을 등록합니다. 설정 단계를 완료하세요.

   - 도메인이 Route 53 이외의 DNS 공급자를 사용하는 경우에도 도메인을 추가할 수 있지만 배포를 생성한 후 추가해야 합니다. 배포 생성을 계속하려면 지금 도메인 설정을 건너뜁니다. 나중에 도메인 및 TLS 인증서를 수동으로 구성해야 합니다. 자세한 내용은 CloudFront 표준 배포에 도메인 추가 섹션을 참조하세요.
6. 다음을 선택합니다.
7. 오리진 지정 페이지에서 CloudFront에서 가져올 콘텐츠의 오리진 유형을 선택합니다. CloudFront는 배포의 해당 오리진 유형에 대한 권장 설정을 사용합니다. 권장 설정에 대한 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하시기 바랍니다.
8. 오리진에서 오리진을 선택하거나 입력합니다.
9. 설정에서 다음 옵션 중 하나를 선택합니다.

   - 권장 오리진 설정 사용 - 선택한 오리진 유형의 기본 권장 캐시 및 오리진 설정을 사용합니다.
   - 오리진 설정 사용자 지정 - 캐시 및 오리진 설정을 사용자 지정합니다. 이 옵션을 선택하는 경우 고유한 값을 지정합니다.
10. 다음을 선택합니다.
11. 보안 보호 활성화 페이지에서 AWS WAF 보안 보호 기능을 사용 설정할지 여부를 선택합니다.
12. 다음을 선택합니다.
13. (선택 사항) 도메인에 Route 53을 사용하는 경우 TLS 인증서 페이지가 표시됩니다. CloudFront가 AWS 계정의 `us-east-1` AWS 리전에서 도메인에 대한 기존 AWS Certificate Manager(ACM) 인증서를 찾을 수 없는 경우 인증서를 자동으로 생성하거나 수동으로 생성하도록 선택할 수 있습니다. 인증서가 생성되면 다음을 선택합니다.
14. 배포 세부 정보를 검토하고 배포 생성을 선택합니다.
15. CloudFront에서 배포를 생성하면 배포에 대한 상태 열의 값이 진행 중에서 배포된 날짜와 시간으로 변경됩니다.

    CloudFront가 배포에 할당하는 도메인 이름이 배포 목록에 나타납니다. 선택한 배포에 대한 [General] 탭에도 나타납니다.

> **Tip**
> 대체 도메인 이름(CNAME)을 추가하여 사용자 지정 URL 사용의 단계를 따르면 CloudFront에서 할당한 이름 대신 대체 도메인 이름을 사용할 수 있습니다.

16. 배포가 완료되면 새 CloudFront URL(d111111abcdef8.cloudfront.net) 또는 CNAME를 사용하여 콘텐츠에 액세스할 수 있는지 확인합니다. 자세한 내용은 배포 테스트 섹션을 참조하세요.
17. 트래픽을 배포에 전송할 준비가 되면 CloudFront를 가리키도록 DNS 레코드를 업데이트해야 합니다. 자세한 내용은 도메인을 CloudFront로 가리키기(표준 배포) 섹션을 참조하세요.

## CloudFront가 콘솔에 표시하는 값

새 배포를 생성하거나 기존 배포를 업데이트할 때, CloudFront에서는 다음 정보를 CloudFront 콘솔에 표시합니다.

> **Note**
> 활성화된 신뢰할 수 있는 서명자, 활성 CloudFront 키 페어를 보유하고 있으며 유효한 서명된 URL을 만드는 데 사용할 수 있는 AWS 계정은 현재 CloudFront 콘솔에 표시되지 않습니다.

### 배포 ID

CloudFront API를 사용하여 배포에 대한 작업을 수행할 경우, 배포 ID를 사용하여 대상 배포를 지정합니다(예: `EDFDVBD6EXAMPLE`). 배포의 배포 ID는 변경할 수 없습니다.

### 배포 및 상태

배포를 배포할 때 마지막 수정일 열 아래에 배포 상태가 표시됩니다. 배포가 완료될 때까지 기다린 다음 상태 열에 활성화됨으로 표시되는지 확인합니다. 자세한 내용은 배포 상태 섹션을 참조하세요.

### 마지막 수정

배포가 마지막으로 수정된 날짜 및 시간을 ISO 8601 형식을 사용하여 나타냅니다(예: 2012-05-19T19:37:58Z). 자세한 내용은 https://www.w3.org/TR/NOTE-datetime 섹션을 참조하세요.

## 도메인 이름

객체에 대한 링크에 배포의 도메인 이름을 사용합니다. 예를 들어, 배포의 도메인 이름이 `d111111abcdef8.cloudfront.net`인 경우, `/images/image.jpg`에 대한 링크는 `https://d111111abcdef8.cloudfront.net/images/image.jpg`가 됩니다. 배포에 대한 CloudFront 도메인 이름은 변경할 수 없습니다. 객체에 대한 링크의 CloudFront URL에 대한 자세한 내용은 CloudFront에서 파일에 대한 URL 형식 사용자 지정 단원을 참조하세요.

하나 이상의 대체 도메인 이름(CNAME)을 지정한 경우, 객체에 대한 링크에 CloudFront 도메인 이름을 사용하는 대신 자체 도메인 이름을 사용할 수 있습니다. CNAME에 대한 자세한 내용은 대체 도메인 이름(CNAME)을 참조합니다.

> **Note**
>
> CloudFront 도메인 이름은 고유합니다. 즉, 배포의 도메인 이름은 이전 배포에 사용된 적이 없고 이후 다른 배포에 재사용되지도 않습니다.

## 추가 링크

배포 생성에 대한 자세한 내용은 다음 링크를 참조하세요.

- Amazon Simple Storage Service(S3) 버킷 오리진, 오리진 액세스 제어(OAC)를 사용하는 배포를 생성하는 방법을 알아보려면 CloudFront 표준 배포 시작 섹션을 참조하세요.
- CloudFront API를 사용하여 배포를 생성하는 방법에 대한 자세한 내용은 Amazon CloudFront API 참조의 CreateDistribution을 참조하세요.
- 배포 업데이트(예: 표준 배포에 캐시 동작 추가 또는 배포 테넌트 사용자 지정)에 대한 자세한 내용은 배포 업데이트 섹션을 참조하시기 바랍니다.
- 현재 각 AWS 계정에 대해 생성할 수 있는 배포의 최대 수를 살펴보고 더 높은 할당량(이전에는 제한이라고 함)을 요청하려면, 배포의 일반 할당량 단원을 참조하십시오.

## CloudFront 표준 배포에 도메인 추가

새 CloudFront 표준 배포를 생성한 후 도메인을 추가할 수 있습니다. 선택적으로, 표준 배포를 생성할 때 표준 배포에 대한 Amazon Route 53 도메인을 설정할 수 있습니다. 자세한 내용은 콘솔에서 CloudFront 배포 생성 섹션을 참조하세요.

## 기존 표준 배포에 도메인 추가

표준 배포에 도메인을 추가하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 배포를 선택한 후 배포 ID를 선택합니다.
3. 설정, 대체 도메인 이름에서 도메인 추가를 선택합니다.
4. 서비스를 제공할 도메인을 최대 5개까지 입력합니다.
5. 다음을 선택합니다.
6. TLS 인증서의 경우 CloudFront가 AWS 계정의 `us-east-1` AWS 리전에서 도메인에 대한 기존 AWS Certificate Manager(ACM) 인증서를 찾을 수 없는 경우 인증서를 생성할 수 있습니다.
   - Amazon Route 53(Route 53)을 사용하는 경우 CloudFront가 자동으로 인증서를 생성합니다.
7. 인증서가 프로비저닝되면 DNS 공급자로 DNS 레코드를 업데이트하여 도메인 소유권을 입증해야 합니다. 그런 다음 인증서 검증을 선택합니다. 자세한 내용은 도메인을 CloudFront로 가리키기(표준 배포) 섹션을 참조하세요.
   - Route 53을 사용하는 경우 CloudFront가 DNS 레코드를 자동으로 업데이트합니다.
8. 다음을 선택합니다.
9. 변경 사항을 검토하고 도메인 추가를 선택합니다.
10. 트래픽을 배포에 전송하기 전에 CloudFront를 가리키도록 DNS 레코드를 업데이트해야 합니다. 자세한 내용을 보려면 배포 세부 정보 페이지의 설정 섹션에서 CloudFront로 도메인 라우팅을 선택합니다.
    - Route 53을 사용하는 경우 CloudFront에서 자동으로 DNS 라우팅을 설정하도록 할 수 있습니다.

## 도메인을 CloudFront로 가리키기(표준 배포)

DNS 레코드를 업데이트하여 각 도메인의 트래픽을 CloudFront 호스트 이름으로 라우팅합니다. 여러 도메인 이름을 가질 수 있지만 모두 이 호스트 이름으로 확인되어야 합니다.

도메인을 CloudFront로 가리키려면 다음과 같이 합니다.

1. d111111abcdef8.cloudfront.net 같은 CloudFront 호스트 이름 값을 복사합니다.

2. DNS 레코드를 업데이트하여 각 도메인의 트래픽을 CloudFront 호스트 이름으로 라우팅합니다.

   1. 도메인 등록 또는 DNS 공급자의 관리 콘솔에 로그인합니다.
   2. 도메인의 DNS 관리 섹션으로 이동합니다.
      - 하위 도메인 - CNAME 레코드를 만듭니다. 예제:
        - 이름 - 하위 도메인(예: `www` 또는 `app`)
        - 값/대상 - CloudFront 호스트 이름
        - 레코드 유형 - CNAME
        - TTL – 3600(또는 사용 사례에 적합한 것)
      - apex/루트 도메인의 경우 - 루트 또는 apex 도메인 수준에서는 표준 CNAME 레코드를 사용할 수 없으므로 고유한 DNS 구성이 필요합니다. 대부분의 DNS 공급자는 ALIAS 레코드를 지원하지 않기 때문에 Route 53에서 ALIAS 레코드를 생성하는 것이 좋습니다. 예제:
        - 이름 - apex 도메인(예: `example.com`)
        - 레코드 유형 - A
        - 별칭 - 예
        - 별칭 대상 - CloudFront 호스트 이름
        - 라우팅 정책 - 단순(또는 사용 사례에 적합한 것)
   3. DNS 변경 사항이 채워졌는지 확인합니다. (이는 일반적으로 TTL이 만료될 때 발생합니다. 경우에 따라 24~48시간이 걸릴 수 있습니다.) `dig` 또는 `nslookup` 등의 도구를 사용합니다.

      ```
      dig www.example.com
      # Should eventually return a CNAME pointing to your CloudFront hostname
      ```

3. CloudFront 콘솔로 이동하여 제출을 선택합니다. 도메인이 활성 상태이면 CloudFront는 도메인이 트래픽을 처리할 준비가 되었음을 나타내도록 도메인 상태를 업데이트합니다.

자세한 내용은 DNS 공급자의 설명서를 참조하세요.

- Cloudflare
- ClouDNS
- DNSimple
- Gandi.net
- GoDaddy
- Google Cloud DNS

- Namecheap

# 사전 구성된 배포 설정 참조

CloudFront 배포를 만들면 CloudFront가 콘텐츠 오리진 유형에 따라 대부분의 배포 설정을 자동으로 구성합니다. 선택적으로 배포 설정을 수동으로 편집하도록 선택할 수 있습니다. 자세한 내용은 모든 배포 설정 참조 섹션을 참조하세요.

다음 섹션에서는 배포의 기본 사전 구성 설정과 사용자 지정할 수 있는 설정에 대해 설명합니다.

## Amazon S3 오리진

다음은 CloudFront가 다중 테넌트 배포에서 Amazon S3 오리진에 대해 사전 구성하는 오리진 설정입니다.

오리진 설정(사전 구성)

- 오리진 액세스 제어(콘솔만 해당) - CloudFront에서 이를 설정합니다. 표준 배포와 오리진 도메인에서 파라미터를 사용하지 않는 다중 테넌트 배포의 경우 CloudFront는 S3 버킷 정책을 추가하려고 시도합니다.
- 사용자 지정 헤더 추가 - 없음
- Origin Shield 활성화 - 아니요
- 연결 시도 - 3

다음은 CloudFront가 다중 테넌트 배포에서 Amazon S3 오리진에 대해 사전 구성하는 캐시 설정입니다.

캐시 설정(사전 구성)

- 자동으로 객체 압축 - 예
- 뷰어 프로토콜 정책 - HTTPS로 리디렉션
- 허용된 HTTP 메서드 - `GET, HEAD`
- 뷰어 액세스 제한 - 아니요
- 캐시 정책 - `CachingOptimized`
- 원본 요청 정책 - 없음
- 응답 헤더 정책 - 없음

- 스무스 스트리밍 - 아니요
- 필드 레벨 암호화 - 아니요
- 실시간 액세스 로그 활성화 - 아니요
- 함수 - 아니요

다음은 다중 테넌트 배포에서 Amazon S3 오리진에 대해 사용자 지정할 수 있는 설정입니다.

사용자 지정 가능한 설정

- S3 액세스 - CloudFront는 S3 버킷 설정에 따라 이를 설정합니다.
  - 버킷이 퍼블릭인 경우 - 오리진 액세스 제어(OAC) 정책이 필요하지 않습니다.
  - 버킷이 프라이빗인 경우 - 사용할 OAC 정책을 선택하거나 만들 수 있습니다.
- Origin Shield 활성화 - 아니요
- 자동으로 객체 압축 - 예
  - 예를 선택하면 `CachingOptimized` 캐싱 정책이 사용됩니다.
  - 아니요를 선택하면 `CachingOptimizedForUncompressedObjects` 캐싱 정책이 사용됩니다.

## API 게이트웨이 요금

다음은 CloudFront가 다중 테넌트 배포에서 API 게이트웨이 오리진에 대해 사전 구성하는 오리진 설정입니다.

오리진 설정(사전 구성)

- 프로토콜 - HTTPS 전용
- HTTPS 포트 - 443
- 최소 원본 SSL 프로토콜 – TLSv1.2
- 오리진 경로 - 없음
- 오리진 액세스 제어(콘솔만 해당) - CloudFront에서 설정
- 사용자 지정 헤더 추가 - 없음
- Origin Shield 활성화 - 아니요
- 연결 시도 - 3
- 응답 시간 초과 – 30

- 연결 유지 시간 초과 – 5

다음은 CloudFront가 다중 테넌트 배포에서 API 게이트웨이 오리진에 대해 사전 구성하는 캐시 설정입니다.

캐시 설정(사전 구성)

- 자동으로 객체 압축 - 예
- 뷰어 프로토콜 정책 - HTTPS로 리디렉션
- 허용된 HTTP 메서드 - `GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE`
- 캐시 HTTP 메서드 - 아니요
- HTTP/2를 통한 gRPC 요청 허용 - 아니요
- 뷰어 액세스 제한 - 아니요
- 캐시 정책 - `CachingDisabled`(가능한 값: `UseOriginCacheControlHeaders`, `UseOriginCacheControlHeaders-QueryStrings`)
- 원본 요청 정책 - `AllViewerExceptHostHeader`(가능한 값: `AllViewer`, `AllViewerandCloudFrontHeaders-2022-06`)
- 응답 헤더 정책 - 없음
- 스무스 스트리밍 - 아니요
- 필드 레벨 암호화 - 아니요
- 실시간 액세스 로그 활성화 - 아니요
- 함수 - 아니요

다음은 다중 테넌트 배포에서 API 게이트웨이 오리진에 대해 사용자 지정할 수 있는 설정입니다.

사용자 지정 가능한 설정

- Origin Shield 활성화 – 기본값(아니요)
- 자동으로 객체 압축 - 기본값(예)

## 사용자 지정 오리진 및 EC2 인스턴스

다음은 CloudFront가 다중 테넌트 배포에서 사용자 지정 오리진에 대해 사전 구성하는 오리진 설정입니다.

오리진 설정(사전 구성)

- 프로토콜 - 뷰어 일치
- HTTP 포트 - 80
- HTTPS 포트 - 443
- 최소 원본 SSL 프로토콜 – TLSv1.2
- 오리진 경로 - 없음
- 사용자 지정 헤더 추가 - 없음
- Origin Shield 활성화 - 아니요
- 연결 시도 - 3
- 응답 시간 초과 – 30
- 연결 유지 시간 초과 – 5

다음은 CloudFront가 다중 테넌트 배포의 사용자 지정 오리진 및 EC2 인스턴스에 대해 사전 구성하는 캐시 설정입니다.

캐시 설정(사전 구성)

- 자동으로 객체 압축 - 예
- 뷰어 프로토콜 정책 - HTTPS로 리디렉션
- 허용된 HTTP 메서드 - `GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE`
- 캐시 HTTP 메서드 - 아니요
- HTTP/2를 통한 gRPC 요청 허용 - 아니요
- 뷰어 액세스 제한 - 아니요
- 캐시 정책 - `UseOriginCacheControlHeaders`(가능한 값: `UseOriginCacheControlHeaders-QueryStrings`, `CachingDisabled`, `CacheOptimized`, `CachingOptimizedForUncompressedObjects`)
- 원본 요청 정책 - `AllViewer`(가능한 값: `AllViewerExceptHostHeader`, `AllViewerandCloudFrontHeaders-2022-06`)
- 응답 헤더 정책 - 없음
- 스무스 스트리밍 - 아니요
- 필드 레벨 암호화 - 아니요

- 실시간 액세스 로그 활성화 - 아니요
- 함수 - 아니요

다음은 다중 테넌트 배포에서 사용자 지정 오리진 및 EC2 인스턴스에 대해 사용자 지정할 수 있는 설정입니다.

사용자 지정 가능한 설정

- Origin Shield 활성화 – 기본값(아니요)
- 자동으로 객체 압축 - 기본값(예)
- 캐싱 - 기본값(`Cache by Default`)
  - `Cache by Default`를 선택하면 `UseOriginCacheControlHeaders` 캐시 정책이 사용됩니다.
  - `Do Not Cache by Default`를 선택하면 `CachingDisabled` 캐시 정책이 사용됩니다.
- 캐시에 쿼리 문자열 포함 - 기본값(`Cache by Default`가 이미 선택된 경우 '예')
  - `Do Not Cache by Default`가 이미 선택되어 있고 캐시에 쿼리 문자열을 포함하도록 선택하면 `UseOriginCacheControlHeaders-QueryStrings` 캐시 정책이 사용됩니다.

## Elastic Load Balancing 오리진

다음은 CloudFront가 다중 테넌트 배포에서 Elastic Load Balancing 오리진에 대해 사전 구성하는 오리진 설정입니다.

오리진 설정(사전 구성)

- 프로토콜 - HTTPS 전용
- HTTPS 포트 - 443
- 최소 원본 SSL 프로토콜 – TLSv1.2
- 오리진 경로 - 없음
- 사용자 지정 헤더 추가 - 없음
- Origin Shield 활성화 - 아니요
- 연결 시도 - 3
- 응답 시간 초과 – 30

- 연결 유지 시간 초과 – 5

다음은 CloudFront가 다중 테넌트 배포에서 Elastic Load Balancing 오리진에 대해 사전 구성하는 캐시 설정입니다.

캐시 설정(사전 구성)

- 자동으로 객체 압축 - 예
- 뷰어 프로토콜 정책 - HTTPS로 리디렉션
- 허용된 HTTP 메서드 - `GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE`
- 캐시 HTTP 메서드 - 아니요
- HTTP/2를 통한 gRPC 요청 허용 - 아니요
- 뷰어 액세스 제한 - 아니요
- 캐싱 - 기본값(`Cache by Default`)
  - `Cache by Default`를 선택하면 `UseOriginCacheControlHeaders` 캐시 정책이 사용됩니다.
  - `Do Not Cache by Default`를 선택하면 `CachingDisabled` 캐시 정책이 사용됩니다.
- 캐시에 쿼리 문자열 포함 - 기본값(`Cache by Default`가 이미 선택된 경우 '예')
  - `Do Not Cache by Default`가 이미 선택되어 있고 캐시에 쿼리 문자열을 포함하도록 선택하면 `UseOriginCacheControlHeaders-QueryStrings` 캐시 정책이 사용됩니다.
- 원본 요청 정책 - `All Viewer`(가능한 값: `AllViewerExceptHostHeader`, `AllViewerandCloudFrontHeaders-2022-06`)
- 응답 헤더 정책 - 없음
- 스무스 스트리밍 - 아니요
- 필드 레벨 암호화 - 아니요
- 실시간 액세스 로그 활성화 - 아니요
- 함수 - 아니요

다음은 다중 테넌트 배포에서 Elastic Load Balancing 오리진에 대해 사용자 지정할 수 있는 설정입니다.

사용자 지정 가능한 설정

- Origin Shield 활성화 – 기본값(아니요)

- 자동으로 객체 압축 - 기본값(예)
- 캐싱 - 기본값(`Cache by Default`)
  - `Cache by Default`를 선택하면 `UseOriginCacheControlHeaders` 캐시 정책이 사용됩니다.
  - `Do Not Cache by Default`를 선택하면 `CachingDisabled` 캐시 정책이 사용됩니다.
- 캐시에 쿼리 문자열 포함 - 기본값(`Cache by Default`가 이미 선택된 경우 '예')
  - `Do Not Cache by Default`가 이미 선택되어 있고 캐시에 쿼리 문자열을 포함하도록 선택하면 `UseOriginCacheControlHeaders-QueryStrings` 캐시 정책이 사용됩니다.

## MediaPackage v1 오리진

다음은 CloudFront가 다중 테넌트 배포에서 MediaPackage v1 오리진에 대해 사전 구성하는 오리진 설정입니다.

오리진 설정(사전 구성)

- 프로토콜 - HTTPS 전용
- HTTPS 포트 - 443
- 최소 원본 SSL 프로토콜 – TLSv1.2
- 원본 경로 - MediaPackage URL을 입력하여 이를 제공
- 사용자 지정 헤더 추가 - 없음
- Origin Shield 활성화 - 아니요
- 연결 시도 - 3
- 응답 시간 초과 – 30
- 연결 유지 시간 초과 – 5

다음은 CloudFront가 다중 테넌트 배포에서 MediaPackage v1 오리진에 대해 사전 구성하는 캐시 설정입니다.

캐시 설정(사전 구성)

- 자동으로 객체 압축 - 예
- 뷰어 프로토콜 정책 - HTTPS로 리디렉션
- 허용된 HTTP 메서드 - `GET, HEAD`

- 캐시 HTTP 메서드 - 아니요
- HTTP/2를 통한 gRPC 요청 허용 - 아니요
- 뷰어 액세스 제한 - 아니요
- 캐시 정책 - `Elemental-MediaPackage`
- 원본 요청 정책 - 없음
- 응답 헤더 정책 - 없음
- 스무스 스트리밍 - 아니요
- 필드 레벨 암호화 - 아니요
- 실시간 액세스 로그 활성화 - 아니요
- 함수 - 아니요

## MediaPackage v2 오리진

다음은 CloudFront가 다중 테넌트 배포에서 MediaPackage v2 오리진에 대해 사전 구성하는 오리진 설정입니다.

오리진 설정(사전 구성)

- 원본 액세스 제어 - CloudFront에서 이를 설정하고 정책 추가
- 프로토콜 - HTTPS 전용
- HTTPS 포트 - 443
- 최소 원본 SSL 프로토콜 – TLSv1.2
- 오리진 경로 - 없음
- 사용자 지정 헤더 추가 - 없음
- Origin Shield 활성화 - 아니요
- 연결 시도 - 3
- 응답 시간 초과 – 30
- 연결 유지 시간 초과 – 5

다음은 CloudFront가 다중 테넌트 배포에서 MediaPackage v2 오리진에 대해 사전 구성하는 캐시 설정입니다.

캐시 설정(사전 구성)

- 자동으로 객체 압축 - 예
- 뷰어 프로토콜 정책 - HTTPS로 리디렉션
- 허용된 HTTP 메서드 - `GET, HEAD`
- 캐시 HTTP 메서드 - 아니요
- HTTP/2를 통한 gRPC 요청 허용 - 아니요
- 뷰어 액세스 제한 - 아니요
- 캐시 정책 - `Elemental-MediaPackage`
- 원본 요청 정책 - 없음
- 응답 헤더 정책 - 없음
- 스무스 스트리밍 - 아니요
- 필드 레벨 암호화 - 아니요
- 실시간 액세스 로그 활성화 - 아니요
- 함수 - 아니요

## MediaTailor 오리진

다음은 CloudFront가 다중 테넌트 배포에서 MediaTailor 오리진에 대해 사전 구성하는 오리진 설정입니다.

오리진 설정(사전 구성)

- 프로토콜 - HTTPS 전용
- HTTPS 포트 - 443
- 최소 원본 SSL 프로토콜 – TLSv1.2
- 원본 경로 - MediaPackage URL을 입력하여 이를 제공
- 사용자 지정 헤더 추가 - 없음
- Origin Shield 활성화 - 아니요
- 연결 시도 - 3
- 응답 시간 초과 – 30
- 연결 유지 시간 초과 – 5

다음은 CloudFront가 다중 테넌트 배포에서 MediaTailor 오리진에 대해 사전 구성하는 캐시 설정입니다.

캐시 설정(사전 구성)

- 자동으로 객체 압축 - 예
- 뷰어 프로토콜 정책 - HTTPS로 리디렉션
- 허용된 HTTP 메서드 - `GET, HEAD`
- 캐시 HTTP 메서드 - 아니요
- HTTP/2를 통한 gRPC 요청 허용 - 아니요
- 뷰어 액세스 제한 - 아니요
- 캐시 정책 - 없음
- 원본 요청 정책 - `Elemental-MediaTailor-PersonalizedManifests`
- 응답 헤더 정책 - 없음
- 스무스 스트리밍 - 아니요
- 필드 레벨 암호화 - 아니요
- 실시간 액세스 로그 활성화 - 아니요
- 함수 - 아니요

# 모든 배포 설정 참조

배포를 생성하거나 업데이트할 때 CloudFront 배포 설정을 수동으로 편집하도록 선택할 수 있습니다. 다음은 편집할 수 있는 설정입니다.

하지만 CloudFront는 콘텐츠 오리진 유형에 따라 대부분의 배포 설정을 자동으로 구성합니다. 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하세요.

CloudFront 콘솔을 사용하여 배포를 만들거나 업데이트하는 자세한 내용은 the section called “배포 생성” 또는 the section called “배포 업데이트” 단원을 참조하세요.

주제


- 오리진 설정
- 캐시 동작 설정
- 배포 설정
- 사용자 지정 오류 페이지 및 오류 캐싱

- 지리적 제한

## 오리진 설정

CloudFront 콘솔을 사용하여 배포를 생성 또는 업데이트할 경우, 원래 버전의 웹 콘텐츠를 저장하는 하나 이상의 위치(오리진이라고도 함)에 대한 정보를 제공합니다. CloudFront는 오리진에서 웹 콘텐츠를 가져와 이를 엣지 서버의 전 세계 네트워크를 통해 최종 사용자에게 제공합니다.

현재 하나의 배포에 대해 생성할 수 있는 오리진의 최대 수를 살펴보고 더 높은 할당량을 요청하려면, the section called “배포의 일반 할당량” 섹션을 참조하세요.

오리진을 삭제하려는 경우 먼저 해당 오리진에 연결된 캐시 동작을 편집하거나 삭제해야 합니다.

> **Important**
>
> 오리진을 삭제할 경우 이전에 해당 오리진에서 제공된 파일이 다른 오리진에서 사용 가능하고 캐시 동작이 현재 그러한 파일에 대한 요청을 새 오리진에 라우팅하는지 확인합니다.

배포를 생성 또는 업데이트할 경우 각 오리진에 대해 다음 값을 지정합니다.

주제

- 오리진 도메인
- 프로토콜(사용자 지정 오리진만 해당)
- 오리진 경로
- 이름
- 오리진 액세스(Amazon S3 오리진에만 해당)
- 사용자 지정 헤더 추가
- Origin Shield 사용
- 연결 시도
- 연결 제한 시간
- 응답 제한 시간
- 응답 완료 제한 시간
- 연결 유지 제한 시간(사용자 지정 및 VPC 오리진만 해당)
- 응답 및 연결 유지 제한 시간 할당량

## 오리진 도메인

오리진 도메인은 CloudFront가 오리진(예: Amazon S3 버킷 또는 HTTP 서버)에 대한 객체를 가져올 리소스의 DNS 도메인 이름입니다. 예제:

- Amazon S3 버킷 – `amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com`

> **Note**
>
> 최근에 S3 버킷을 만든 경우 CloudFront 배포에서 최대 24시간 동안 `HTTP 307 Temporary Redirect` 응답을 반환할 수 있습니다. S3 버킷 이름이 모든 AWS 리전에 전파되는 데 최대 24시간이 걸릴 수 있습니다. 전파가 완료되면, 배포에서는 이러한 리디렉션 응답 전송을 자동으로 중지하므로 별도로 조치를 취할 필요가 없습니다. 자세한 내용은 Amazon S3에서 HTTP 307 임시 리디렉션 응답을 받는 이유는 무엇입니까? 및 임시 요청 리디렉션을 참조하십시오.

- 웹 사이트로 구성된 Amazon S3 버킷 - `amzn-s3-demo-bucket.s3-website.us-west-2.amazonaws.com`
- MediaStore 컨테이너 - `examplemediastore.data.mediastore.us-west-1.amazonaws.com`
- MediaPackage 엔드포인트 – `examplemediapackage.mediapackage.us-west-1.amazonaws.com`
- Amazon EC2 인스턴스 – `ec2-203-0-113-25.compute-1.amazonaws.com`
- Elastic Load Balancing 로드 밸런서 – `example-load-balancer-1234567890.us-west-2.elb.amazonaws.com`
- 자체 웹 서버 - `www.example.com`

오리진 도메인(Origin domain) 필드에서 도메인 이름을 선택하거나, 이름을 입력합니다. 옵트인 리전의 리소스는 수동으로 입력해야 합니다. 도메인 이름은 대소문자를 구분하지 않습니다. 오리진 도메인은 인터넷을 통해 클라이언트의 요청을 대상으로 라우팅하는 공개적으로 확인 가능한 DNS 이름을 포함해야 합니다.

CloudFront가 HTTPS를 통해 오리진에 연결하도록 구성하는 경우, 인증서의 도메인 이름 중 하나는 오리진 도메인 이름으로 지정하는 도메인 이름과 일치해야 합니다. 일치하는 도메인 이름이 없는 경우 CloudFront는 최종 사용자에게 HTTP 상태 코드 502(잘못된 게이트웨이)를 반환합니다. 자세한 내용은 CloudFront 배포 및 인증서의 도메인 이름 및 CloudFront와 사용자 지정 원본 서버 간의 SSL/TLS 협상 실패(을)를 참조하세요.

> **Note**
>
> 최종 사용자 호스트 헤더를 오리진으로 전달하는 오리진 요청 정책을 사용하는 경우, 오리진은 최종 사용자 호스트 헤더와 일치하는 인증서로 응답해야 합니다. 자세한 내용은 CloudFront 요청 헤더 추가 섹션을 참조하세요.

오리진이 Amazon S3 버킷인 경우 다음에 유의하세요.

- 버킷이 웹 사이트로 구성된 경우, 버킷에 대한 Amazon S3 정적 웹 사이트 호스팅 엔드포인트를 입력합니다. 오리진 도메인(Origin domain) 필드의 목록에서 버킷 이름을 선택하지 마세요. 정적 웹 사이트 호스팅 엔드포인트는 Amazon S3 콘솔에서 속성(Properties) 페이지의 정적 웹 사이트 호스팅(Static website hosting) 아래에 나타납니다. 자세한 내용은 the section called "웹 사이트 엔드포인트로 구성된 Amazon S3 버킷 사용" 섹션을 참조하세요.
- 버킷에 Amazon S3 Transfer Acceleration을 구성한 경우, 오리진 도메인(Origin domain)에 `s3-accelerate` 엔드포인트를 지정하지 마세요.
- 다른 AWS 계정에서 버킷을 사용하려는 경우 버킷이 웹 사이트로 구성되어 있지 않으면 다음 형식을 사용하여 이름을 입력합니다.

  ```
  bucket-name.s3.region.amazonaws.com
  ```

  버킷이 미국 리전에 있고 Amazon S3에서 요청을 버지니아 북부의 시설로 라우팅하려는 경우, 다음 형식을 사용합니다.

  ```
  bucket-name.s3.us-east-1.amazonaws.com
  ```
- CloudFront 오리진 액세스 제어를 사용하여 Amazon S3의 콘텐츠에 보안을 적용하지 않는 한, 파일은 공개적으로 읽기 가능해야 합니다. 액세스 제어에 대한 자세한 내용은 the section called "Amazon S3 오리진에 대한 액세스 제한" 섹션을 참조하세요.

> **Important**
>
> 오리진이 Amazon S3 버킷인 경우, DNS 명명 요구 사항을 준수하는 버킷 이름을 사용해야 합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 규제 및 제한을 참조하세요.

오리진에 대한 오리진 도메인(Origin domain)의 값을 변경하면 CloudFront에서는 CloudFront 엣지 로케이션에 대한 변경 사항 복제를 즉시 시작합니다. 배포 구성이 지정된 엣지 로케이션에서 업데이트될 때까지 CloudFront는 계속해서 요청을 이전 오리진으로 전달합니다. 배포 구성이 해당 엣지 로케이션에서 업데이트되는 즉시 CloudFront에서는 요청을 새 오리진으로 전달하기 시작합니다.

버킷을 변경하더라도 CloudFront에서는 엣지 캐시를 새 오리진의 객체로 다시 채울 필요가 없습니다. 애플리케이션의 최종 사용자 요청이 변경되지 않는 한, CloudFront에서는 각 객체의 TTL이 만료되거나 자주 요청되지 않는 객체가 제거될 때까지는 계속해서 엣지 캐시에 이미 존재하는 객체를 제공합니다.

## 프로토콜(사용자 지정 오리진만 해당)

> **Note**
>
> 이는 사용자 지정 오리진에만 적용됩니다.

오리진에서 객체를 가져올 때 CloudFront에서 사용하려는 프로토콜 정책입니다.

다음 값 중 하나를 선택합니다.

- HTTP만(HTTP only): CloudFront에서 HTTPS만 사용하여 오리진에 액세스합니다.

  > **Important**
  >
  > Amazon S3가 정적 웹 사이트 호스팅 엔드포인트에 대한 HTTPS 연결을 지원하지 않으므로 오리진이 Amazon S3 정적 웹 사이트 호스팅 엔드포인트인 경우 HTTP만(HTTP only)이 기본 설정입니다. CloudFront 콘솔에서는 Amazon S3 정적 웹 사이트 호스팅 엔드포인트에 대해 이 설정을 변경할 수 없습니다.

- HTTPS만(HTTPS only): CloudFront에서 HTTPS만 사용하여 오리진에 액세스합니다.
- 최종 사용자와 일치(Match viewer) - CloudFront에서 최종 사용자 요청의 프로토콜에 따라 HTTP 또는 HTTPS를 사용하여 오리진과 통신합니다. 최종 사용자가 HTTP 및 HTTPS 프로토콜 모두를 사용하여 요청하더라도 CloudFront에서는 한 번만 객체를 캐싱합니다.

  > **Important**
  >
  > CloudFront에서 이 오리진에 전달하는 HTTPS 최종 사용자 요청의 경우, 원본 서버의 SSL/TLS 인증서에 있는 도메인 이름 중 하나가 오리진 도메인(Origin domain)으로 지정한 도메

인 이름과 일치해야 합니다. 그렇지 않은 경우 CloudFront는 최종 사용자 요청에 대해 요청된 객체를 반환하는 대신 HTTP 상태 코드 502(잘못된 게이트웨이)로 응답합니다. 자세한 내용은 the section called "CloudFront에서 SSL/TLS 인증서를 사용하기 위한 요구 사항" 단원을 참조합니다.

주제

- HTTP 포트
- HTTPS 포트
- 최소 오리진 SSL 프로토콜

## HTTP 포트

> **Note**
> 이는 사용자 지정 오리진에만 적용됩니다.

(선택 사항) 사용자 지정 오리진이 수신 대기하는 HTTP 포트를 지정할 수 있습니다. 유효한 값에는 포트 80, 443 및 1024~65535가 포함됩니다. 기본값은 포트 80입니다.

> **Important**
> Amazon S3가 정적 웹 사이트 호스팅 엔드포인트에 대해 포트 80만 지원하므로 오리진이 Amazon S3 정적 웹 사이트 호스팅 엔드포인트인 경우 포트 80이 기본 설정입니다. CloudFront 콘솔에서는 Amazon S3 정적 웹 사이트 호스팅 엔드포인트에 대해 이 설정을 변경할 수 없습니다.

## HTTPS 포트

> **Note**
> 이는 사용자 지정 오리진에만 적용됩니다.

(선택 사항) 사용자 지정 오리진이 수신 대기하는 HTTPS 포트를 지정할 수 있습니다. 유효한 값에는 포트 80, 443 및 1024~65535가 포함됩니다. 기본값은 포트 443입니다. 프로토콜(Protocol)이 HTTP만(HTTP only)으로 설정된 경우 HTTPS 포트(HTTPS port)에 값을 지정할 수 없습니다.

최소 오리진 SSL 프로토콜

> **Note**
> 이는 사용자 지정 오리진에만 적용됩니다.

오리진에 대한 HTTPS 연결을 설정할 때 CloudFront가 사용할 수 있는 최소 TLS/SSL 프로토콜을 선택합니다. TLS 프로토콜이 낮을수록 보안이 낮으므로, 오리진이 지원하는 최신 TLS 프로토콜을 선택하는 것이 좋습니다. 프로토콜(Protocol)이 HTTP만(HTTP only)으로 설정된 경우 최소 오리진 SSL 프로토콜(Minimum origin SSL protocol)에 값을 지정할 수 없습니다.

CloudFront API를 사용하여 CloudFront에 사용할 TLS/SSL 프로토콜을 설정하는 경우 최소 프로토콜을 설정할 수 없습니다. 대신, CloudFront에서 오리진과 함께 사용할 수 있는 모든 TLS/SSL 프로토콜을 지정합니다. 자세한 내용은 Amazon CloudFront API 참조의 OriginSslProtocols를 참조하세요.

## 오리진 경로

CloudFront가 해당 오리진에 있는 디렉터리에서 콘텐츠를 요청하게 하려면 슬래시(/) 뒤에 디렉터리 경로를 입력합니다. CloudFront는 오리진 도메인(Origin domain)의 값에 디렉터리 이름을 추가합니다(예: **`cf-origin.example.com/production/images`**). 경로 끝에는 슬래시(/)를 추가하지 마십시오.

예를 들어, 배포에 대해 다음 값을 지정했다고 가정해 보겠습니다.

- 오리진 도메인(Origin domain) - 이름이 **`amzn-s3-demo-bucket`**인 Amazon S3 버킷
- 오리진 경로(Origin path)**`/production`** -
- 대체 도메인 이름(CNAME)(Alternate domain names(CNAME))**`example.com`** –

한편, 사용자가 브라우저에 `example.com/index.html`을 입력하면 CloudFront는 Amazon S3에 `amzn-s3-demo-bucket/production/index.html`에 대한 요청을 보냅니다.

한편, 사용자가 브라우저에 `example.com/acme/index.html`을 입력하면 CloudFront는 Amazon S3에 `amzn-s3-demo-bucket/production/acme/index.html`에 대한 요청을 보냅니다.

## 이름

이름은 이 배포 내의 이 오리진을 고유하게 식별하는 문자열입니다. 기본 캐시 동작에 추가로 캐시 동작을 만들 경우, 여기에서 지정한 이름을 사용하여 요청이 해당 캐시 동작의 경로 패턴과 일치할 때 CloudFront에서 요청을 라우팅하려는 오리진을 식별합니다.

## 오리진 액세스(Amazon S3 오리진에만 해당)

> **Note**
>
> 이는 Amazon S3 버킷 오리진(S3 정적 웹 사이트 엔드포인트를 사용하지 않는 오리진)에만 적용됩니다.

Amazon S3 버킷 오리진에 대한 액세스를 특정 CloudFront 배포로만 제한할 수 있도록 하려면 Origin access control settings(recommended)(원본 액세스 제어 설정(권장))를 선택합니다.

Amazon S3 버킷 오리진에 공개적으로 액세스할 수 있는 경우 Public(공개)을 선택합니다.

자세한 내용은 the section called “Amazon S3 오리진에 대한 액세스 제한” 단원을 참조합니다.

사용자가 CloudFront URL만 사용하여 사용자 지정 오리진의 객체에 액세스할 수 있도록 요구하는 방법에 대한 자세한 내용은 the section called “사용자 지정 오리진의 파일에 대한 액세스 제한” 단원을 참조하세요.

## 사용자 지정 헤더 추가

CloudFront가 오리진에 요청을 보낼 때마다 사용자 지정 헤더를 추가하도록 하려면 헤더 이름과 값을 지정합니다. 자세한 내용은 the section called “사용자 지정 헤더를 오리진 요청에 추가” 단원을 참조합니다.

현재 추가할 수 있는 사용자 지정 헤더의 최대 수, 사용자 지정 헤더 이름과 값의 최대 길이 및 모든 헤더 이름 및 값의 최대 총 길이는 할당량 단원을 참조하세요.

## Origin Shield 사용

CloudFront Origin Shield를 사용하려면 예(Yes)를 선택합니다. Origin Shield에 대한 자세한 내용은 the section called “Origin Shield 사용” 단원을 참조하세요.

## 연결 시도

CloudFront가 오리진에 연결을 시도하는 횟수를 설정할 수 있습니다. 시도 횟수로 1, 2 또는 3을 지정할 수 있습니다. 기본 숫자는 3입니다(달리 지정하지 않은 경우).

이 설정을 연결 제한 시간(Connection timeout)과 함께 사용하여 CloudFront가 보조 오리진에 연결하려고 시도하거나 최종 사용자에게 오류 응답을 반환하기 전에 대기하는 시간을 지정합니다. 기본적으로 CloudFront는 보조 오리진에 연결하거나 오류 응답을 반환하기 전에 30초(각각 10초 동안 3회 시도)까지 기다립니다. 더 적은 시도 횟수, 더 짧은 연결 제한 시간 또는 둘 다를 지정하여 이 시간을 줄일 수 있습니다.

지정된 수의 연결 시도가 실패하면 CloudFront에서 다음 중 하나를 수행합니다.

- 오리진이 오리진 그룹의 일부인 경우 CloudFront는 보조 오리진에 연결을 시도합니다. 보조 오리진에 대한 지정된 수의 연결 시도가 실패하면 CloudFront는 최종 사용자에게 오류 응답을 반환합니다.
- 오리진이 오리진 그룹의 일부가 아닌 경우 CloudFront가 최종 사용자에게 오류 응답을 반환합니다.

사용자 지정 오리진(정적 웹 사이트 호스팅으로 구성된 Amazon S3 버킷 포함)의 경우 이 설정은 CloudFront가 오리진에서 응답을 받으려고 시도하는 횟수도 지정합니다. 자세한 내용은 the section called “응답 제한 시간” 단원을 참조합니다.

## 연결 제한 시간

오리진 연결 제한 시간은 CloudFront가 오리진에 대한 연결을 설정하려고 할 때 기다리는 시간(초)입니다. 1초에서 10초까지 시간을 지정할 수 있습니다. 기본 제한 시간은 10초입니다(달리 지정하지 않은 경우).

이 설정을 연결 시도 횟수(Connection attempts)와 함께 사용하여 CloudFront가 보조 오리진에 연결하려고 시도하거나 최종 사용자에게 오류 응답을 반환하기 전에 대기하는 시간을 지정합니다. 기본적으로 CloudFront는 보조 오리진에 연결하거나 오류 응답을 반환하기 전에 30초(각각 10초 동안 3회 시도)까지 기다립니다. 더 적은 시도 횟수, 더 짧은 연결 제한 시간 또는 둘 다를 지정하여 이 시간을 줄일 수 있습니다.

CloudFront에서 지정된 시간(초) 내에 오리진에 대한 연결을 설정하지 않으면 CloudFront에서 다음 중 하나를 수행합니다.

- 지정된 연결 시도 횟수(Connection attempts)가 1보다 많으면 CloudFront가 다시 연결을 설정하려고 시도합니다. CloudFront는 연결 시도 횟수(Connection attempts) 값에 따라 최대 3회까지 시도합니다.

- 연결 시도가 실패하고 오리진이 오리진 그룹의 일부인 경우 CloudFront는 보조 오리진에 연결을 시도합니다. 보조 오리진에 대한 지정된 수의 연결 시도가 실패하면 CloudFront는 최종 사용자에게 오류 응답을 반환합니다.
- 모든 연결 시도가 실패하고 오리진이 오리진 그룹의 일부가 아닌 경우 CloudFront가 최종 사용자에게 오류 응답을 반환합니다.

## 응답 제한 시간

오리진 응답 제한 시간(오리진 읽기 제한 시간 또는 오리진 요청 제한 시간이라고도 함)은 다음 두 값에 모두 적용됩니다.

- CloudFront가 오리진에 요청을 전달한 후 응답을 기다리는 시간(초).
- CloudFront가 오리진으로부터 응답 패킷을 수신한 후 다음 패킷을 수신할 때까지 대기하는 시간(초).

> **Tip**
>
> 최종 사용자가 HTTP 504 상태 코드 오류를 경험하고 있기 때문에 오리진 응답 제한 시간 값을 늘리고 싶은 경우에는 제한 시간 값을 변경하기 앞서 이러한 오류를 제거할 수 있는 다른 방법들을 시도해 봅니다. the section called “HTTP 504 상태 코드(게이트웨이 제한 시간)”에서 문제 해결 주제를 참조합니다.

CloudFront 동작은 최종 사용자 요청 내 HTTP 메서드에 따라 달라집니다.

- `GET` 및 `HEAD` 요청 - 오리진에서 응답 제한 시간 내에 응답하지 않거나 응답이 중지된 경우, CloudFront에서는 연결을 끊습니다. CloudFront는 the section called “연결 시도”의 값에 따라 연결을 다시 시도합니다.
- `DELETE`, `OPTIONS`, `PATCH`, `PUT` 및 `POST` 요청 - 오리진이 읽기 제한 시간 동안 응답하지 않는 경우, CloudFront는 연결을 끊고 오리진에 다시 연결을 시도하지 않습니다. 필요한 경우 클라이언트는 요청을 다시 제출할 수 있습니다.

## 응답 완료 제한 시간

> **Note**
>
> 응답 완료 제한 시간은 지속적 배포 기능을 지원하지 않습니다.

CloudFront에서 오리진으로의 요청이 열려 있고 응답을 기다릴 수 있는 시간(초). 이 시간까지 오리진에서 완전한 응답을 받지 못하면 CloudFront는 연결을 종료합니다.

개별 응답 패킷의 대기 시간인 응답 제한 시간과 달리 응답 완료 제한 시간은 CloudFront가 응답이 완료될 때까지 대기하는 최대 허용 시간입니다. 이 설정을 사용하면 다른 제한 시간 설정에서 더 긴 대기 시간을 허용하는 경우에도 CloudFront가 느리거나 응답하지 않는 오리진을 무기한 기다리지 않도록 할 수 있습니다.

이 최대 제한 시간에는 다른 제한 시간 설정에 지정한 사항과 각 재시도에 대한 연결 시도 횟수가 포함됩니다. 이러한 설정을 함께 사용하여 CloudFront가 전체 요청을 기다리는 시간과 요청이 완료되었는지 여부에 관계없이 요청을 종료할 시기를 지정할 수 있습니다.

예를 들어 다음 설정을 지정했다고 해보겠습니다.

- 연결 시도 3회
- 연결 제한 시간 10초
- 응답 제한 시간 30초
- 응답 완료 제한 시간 60초

이 경우 CloudFront는 오리진에 연결을 시도하며(최대 총 3회 시도), 각 연결 시도는 10초 후에 시간 초과됩니다. 연결되면 CloudFront는 오리진이 요청에 응답할 때까지 최대 30초 동안 대기한 후 마지막 응답 패킷을 수신합니다.

연결 시도 횟수 또는 응답 제한 시간에 관계없이 오리진의 전체 응답이 완료되는 데 60초 이상 걸리는 경우 CloudFront는 연결을 종료합니다. 그런 다음 CloudFront는 the section called “HTTP 504 상태 코드(게이트웨이 제한 시간)” 오류 응답 또는 사용자 지정 오류 응답(지정한 경우)을 뷰어에 반환합니다.

> **참고**
>
> - 응답 완료 제한 시간 값을 설정하는 경우, 해당 값은 응답 제한 시간(오리진 읽기 제한 시간) 값보다 크거나 같아야 합니다.
> - 응답 완료 제한 시간 값을 설정하지 않으면 CloudFront에서 최대값을 적용하지 않습니다.

## 연결 유지 제한 시간(사용자 지정 및 VPC 오리진만 해당)

연결 유지 제한 시간은 CloudFront가 응답의 마지막 패킷을 가져온 후 연결을 유지하기 위해 시도하는 시간(초)입니다. 지속적인 연결을 유지하면 TCP 연결 재설정 및 이후 요청에 대한 별도의 TLS 핸드쉐

이크 수행에 필요한 시간이 절약됩니다. 연결 유지 제한 시간을 늘리면 배포에서 연결당 요청 지표를 개선할 수 있습니다.

> **Note**
>
> 연결 유지 제한 시간(Keep-alive timeout) 값이 유효하려면 오리진이 지속적 연결을 허용하도록 구성되어야 합니다.

### 응답 및 연결 유지 제한 시간 할당량

- 응답 시간 제한의 경우 기본값은 30초입니다.
- 연결 유지 제한 시간의 경우 기본값은 5초입니다.

AWS 계정에 대한 제한 시간 증가를 요청하는 경우 배포 오리진을 업데이트하여 원하는 응답 제한 시간 및 연결 유지 제한 시간 값을 갖도록 합니다. 계정 할당량을 늘려도 오리진이 자동으로 업데이트되지는 않습니다. 예를 들어 Lambda@Edge 함수를 사용하여 연결 유지 제한 시간을 90초로 설정하는 경우 오리진의 연결 유지 제한 시간이 이미 90초 이상으로 설정되어 있어야 합니다. 그렇지 않으면 Lambda@Edge 함수가 실행되지 않을 수 있습니다.

증가 요청 방법을 비롯한 배포 할당량에 대한 자세한 내용은 배포의 일반 할당량 섹션을 참조하세요.

## 캐시 동작 설정

캐시 동작을 설정하면 웹사이트에 있는 파일의 지정된 URL 경로 패턴에 대해 다양한 CloudFront 기능을 구성할 수 있습니다. 예를 들어, 하나의 캐시 동작이 CloudFront용 오리진 서버로 사용 중인 웹 서버의 `.jpg` 디렉터리에 있는 전체 `images` 파일에 적용될 수 있습니다. 각 캐시 동작에 대해 구성 가능한 기능에는 다음이 포함됩니다.

- 경로 패턴
- CloudFront 배포에 대해 다수의 오리진을 구성한 경우, CloudFront에서 요청을 전달하려는 대상 오리진
- 쿼리 문자열을 오리진에 전달할지 여부
- 지정된 파일에 액세스하기 위해 서명된 URL이 필요한지 여부
- 사용자가 HTTPS를 사용하여 그러한 파일에 액세스하도록 지정할지 여부
- 오리진에서 파일에 추가한 `Cache-Control` 헤더의 값과 관계없이 그러한 파일이 CloudFront 캐시에 머무르는 최소 시간

새 배포를 만든 경우, 배포를 만들 때 지정한 오리진에 모든 요청을 자동으로 전달하는 기본 캐시 동작에 대해 설정을 지정합니다. 배포를 만든 후에는 경로 패턴과 일치하는 객체(예: `*.jpg`)에 대한 요청을 받았을 때 CloudFront에서 응답하는 방식을 정의하는 추가 캐시 동작을 만들 수 있습니다. 추가 캐시 동작을 만든 경우 기본 캐시 동작은 항상 마지막으로 처리됩니다. 나머지 캐시 동작은 CloudFront 콘솔에 나열한 순서에 따라 처리됩니다. 또는 CloudFront API를 사용하는 경우 배포에 대한 `DistributionConfig` 요소에 나열한 순서대로 처리됩니다. 자세한 내용은 경로 패턴 단원을 참조합니다.

캐시 동작을 만들 때 CloudFront에서 객체를 가져오려는 대상 오리진을 하나만 지정합니다. CloudFront에서 오리진 전체로부터 객체를 배포하려는 경우, 기본 캐시 동작을 포함해 캐시 동작은 최소한 보유한 오리진의 수만큼 있어야 합니다. 예를 들어 두 개의 오리진이 있는 경우 기본 캐시 동작이 하나뿐이라면, 이 기본 캐시 동작으로 인해 CloudFront는 두 개의 오리진 중 하나에서 객체를 가져오지만, 나머지 오리진은 사용되지 않습니다.

현재 하나의 배포에 추가할 수 있는 캐시 동작의 최대 수를 살펴보고 더 높은 할당량(이전에는 제한이라고 함)을 요청하려면, 배포의 일반 할당량 단원을 참조하세요.

주제


- 경로 패턴
- 오리진 또는 오리진 그룹
- 뷰어 프로토콜 정책
- 허용되는 HTTP 메서드
- 필드 레벨 암호화 구성
- 캐싱된 HTTP 메서드
- HTTP/2를 통한 gRPC 요청 허용
- 선택한 요청 헤더 기반의 캐시
- 허용 목록 헤더
- 객체 캐싱
- Minimum TTL
- Maximum TTL
- 기본 TTL
- 쿠키 전달
- 허용 목록 쿠키

- 쿼리 문자열 전달 및 캐싱
- 쿼리 문자열 허용 목록
- Smooth Streaming
- 최종 사용자 액세스 제한(서명된 URL 또는 서명된 쿠키 사용)
- 신뢰할 수 있는 서명자
- AWS 계정 번호
- 자동으로 객체를 압축
- CloudFront 이벤트
- Lambda 함수 ARN
- 본문 포함

## 경로 패턴

경로 패턴(예: `images/*.jpg`)은 이 캐시 동작을 적용하려는 요청을 지정합니다. CloudFront가 최종 사용자 요청을 받으면, 요청된 경로는 배포에 나열된 캐시 동작의 순서대로 경로 패턴과 비교됩니다. 첫 번째 일치 결과로 해당 요청에 적용되는 캐시 동작이 결정됩니다. 예를 들어, 다음과 같은 3개의 경로 패턴과 순서로 3개의 캐시 동작이 있다고 가정합니다.

- `images/*.jpg`
- `images/*`
- `*.gif`

> **Note**
>
> `/images/*.jpg`와 같이 경로 패턴 시작 부분에 슬래시(/)를 포함하도록 선택할 수 있습니다. 선행 / 포함 여부와 상관 없이 CloudFront의 동작은 동일합니다. 경로 시작 부분에 /를 지정하지 않으면 이 문자가 자동으로 암시되며, CloudFront는 선행 /가 있든 없든 경로를 동일하게 처리합니다. 예를 들어, CloudFront는 `/*product.jpg`와 `*product.jpg`를 동일하게 취급합니다.

`images/sample.gif` 파일에 대한 요청은 첫 번째 경로 패턴을 충족하지 않으므로 연결된 캐시 동작은 이 요청에 적용되지 않습니다. 이 파일은 두 번째 경로 패턴을 만족하므로 세 번째 경로 패턴과도 일치하더라도 두 번째 경로 패턴과 연결된 캐시 동작이 이 요청에 적용됩니다.

> **Note**
>
> 새 배포를 만들 때 기본 캐시 동작에 대한 경로 패턴의 값은 *(모든 파일)로 설정되고 이는 변경할 수 없습니다. 이 값에 따라 CloudFront에서는 객체에 대한 모든 요청을 오리진 도메인 필드에서 지정한 오리진으로 전달합니다. 객체에 대한 요청이 나머지 캐시 동작 중 어느 것의 경로 패턴과도 일치하지 않는 경우, CloudFront에서는 기본 캐시 동작에서 지정한 동작을 적용합니다.

> **Important**
>
> 경로 패턴과 그 순서를 주의 깊게 정의하지 않으면 사용자에게 콘텐츠에 대한 원치 않는 액세스 권한을 주게 될 수 있습니다. 예를 들어, 하나의 요청이 두 캐시 동작의 경로 패턴과 일치한다고 가정합니다. 첫 번째 캐시 동작은 서명된 URL이 필요하고 두 번째 캐시 동작은 서명된 URL이 필요하지 않습니다. CloudFront에서는 첫 번째 일치 결과와 연결된 캐시 동작을 처리하므로 사용자는 서명된 URL을 사용하지 않고도 객체에 액세스할 수 있습니다.

MediaPackage 채널을 사용하는 경우, 오리진의 엔드포인트 유형에 대해 정의하는 캐시 동작을 위한 특정 경로 패턴을 포함시켜야 합니다. 예를 들어 DASH 엔드포인트의 경우 경로 패턴에 `*.mpd`를 입력합니다. 자세한 정보와 특정 지침은 AWS Elemental MediaPackage를 사용하여 포맷된 라이브 비디오 제공 단원을 참조하십시오.

사용자가 지정한 경로는 지정된 디렉터리와 이 디렉터리의 하위 디렉터리에 있는 모든 파일에 대한 요청에 적용됩니다. CloudFront에서는 경로 패턴을 평가할 때 쿼리 문자열 또는 쿠키를 고려하지 않습니다. 예를 들어, `images` 디렉터리에 `product1` 및 `product2` 하위 디렉터리가 포함되어 있으면, 경로 패턴 `images/*.jpg`는 모든 `images`, `images/product1`, `images/product2` 디렉터리의 모든 .jpg 파일에 대한 요청에 적용됩니다. 다른 캐시 동작을 `images/product1` 및 `images` 디렉터리의 파일이 아닌 `images/product2` 디렉터리의 파일에 적용하려는 경우, `images/product1`에 대한 별도의 캐시 동작을 만들어 `images` 디렉터리에 대한 캐시 동작보다 위로(이전으로) 해당 캐시 동작을 이동합니다.

다음 와일드카드 문자를 경로 패턴에 사용할 수 있습니다.

- *는 0개 이상의 문자에 해당합니다.
- ?는 정확히 1문자에 해당합니다.

다음 예는 와일드카드 문자가 어떻게 작동하는지 보여줍니다.

| 경로 패턴 | 경로 패턴과 일치하는 파일 |
| --- | --- |
| `*.jpg` | 모든 .jpg 파일 |
| `images/*.jpg` | `images` 디렉터리와 `images` 디렉터리의 하위 디렉터리에 있는 모든 .jpg 파일 |
| `a*.jpg` | • a로 시작하는 파일 이름을 지닌 모든 .jpg 파일(예: `apple.jpg` , `appalachian_trail_2012_05_21.jpg` )<br>• a로 시작하는 파일 경로에 있는 모든 .jpg 파일(예: `abra/cadabra/magic.jpg` ) |
| `a??.jpg` | a로 시작하여 나머지는 정확히 두 문자가 오는 파일 이름을 지닌 모든 .jpg 파일(예: `ant.jpg`, `abe.jpg`) |
| `*.doc*` | 파일 이름 확장명이 `.doc`로 시작하는 모든 .jpg 파일(예: `.doc`, `.docx`, `.docm` 파일). 이 경우, `*.doc?` 경로 패턴을 사용할 수 없습니다. 이 경로 패턴은 `.doc` 파일에 대한 요청에 적용할 수 없기 때문입니다. ? 와일드카드 문자는 정확히 한 문자를 대체합니다. |

경로 패턴의 최대 길이는 255자입니다. 이 값에는 다음 문자 중 어느 것이든 포함할 수 있습니다.

- A~Z, a~z

  경로 패턴은 대소문자를 구분하므로 경로 패턴 `*.jpg`는 `LOGO.JPG` 파일에 적용되지 않습니다.
- 0~9
- _ - . * $ / ~ " ' @ : +
- &, `&`로 처리되고 반환됨

### 경로 정규화

CloudFront는 RFC 3986과 일치하는 URI 경로를 정규화한 다음 경로를 올바른 캐시 동작과 일치시킵니다. 캐시 동작이 일치하면 CloudFront는 원본 URI 경로를 오리진에 보냅니다. 일치하지 않으면 요청이 기본 캐시 동작과 일치하도록 선택할 수 있습니다.

다중 슬래시 (`//`) 또는 마침표 (`..`) 와 같은 일부 문자는 정규화되어 경로에서 제거됩니다. 이렇게 하면 CloudFront가 의도한 캐시 동작과 일치시키기 위해 사용하는 URL을 변경할 수 있습니다.

Example예제

캐시 동작의 `/a/b*` 및 `/a*` 경로를 지정합니다.

- `/a/b?c=1` 경로를 보내는 뷰어는 `/a/b*` 캐시 동작과 일치할 것입니다.
- `/a/b/..?c=1` 경로를 보내는 뷰어는 `/a*` 캐시 동작과 일치합니다.

경로가 정규화되는 문제를 해결하려면 요청 경로나 캐시 동작의 경로 패턴을 업데이트하면 됩니다.

## 오리진 또는 오리진 그룹

이 설정은 기존 배포에 대한 캐시 동작을 생성하거나 업데이트하는 경우에만 적용됩니다.

기존 오리진 또는 오리진 그룹의 값을 입력합니다. 이를 통해 요청(예: https://example.com/logo.jpg)이 캐시 동작의 경로 패턴(예: *.jpg)이나 기본 캐시 동작의 경로 패턴(*)과 일치할 때 CloudFront에서 요청을 라우팅하려는 오리진 또는 오리진 그룹을 식별합니다.

## 뷰어 프로토콜 정책

최종 사용자가 CloudFront 엣지 로케이션의 콘텐츠를 액세스하는 데 사용하게 하려는 프로토콜 정책을 선택합니다.

- HTTP 및 HTTPS: 최종 사용자는 두 프로토콜 모두를 사용할 수 있습니다.
- Redirect HTTP to HTTPS(HTTP를 HTTPS로 재지정): 최종 사용자는 두 프로토콜 모두 사용할 수 있지만, HTTP 요청은 자동으로 HTTPS 요청으로 리디렉션됩니다.
- HTTPS Only(HTTPS만 해당): 최종 사용자는 HTTPS를 사용하는 경우에만 콘텐츠에 액세스할 수 있습니다.

자세한 내용은 뷰어와 CloudFront 간의 통신에 HTTPS 요구 단원을 참조합니다.

## 허용되는 HTTP 메서드

CloudFront에서 오리진을 처리하고 전달하기 위한 HTTP 메서드를 지정합니다.

- GET, HEAD: 오리진에서 객체를 가져오거나 객체 헤더를 가져오기 위해서만 CloudFront를 사용할 수 있습니다.
- GET, HEAD, OPTIONS: 오리진에서 객체를 가져오거나 객체 헤더를 가져오기 위해, 또는 오리진 서버에서 지원되는 옵션 목록을 가져오기 위해서만 CloudFront를 사용할 수 있습니다.
- GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE: CloudFront를 사용하여 객체를 추가, 업데이트, 삭제하고 가져올 수 있으며 객체 헤더를 가져올 수 있습니다. 또한 웹 양식에서 데이터를 제출하는 등의 기타 POST 작업을 수행할 수 있습니다.

> **Note**
>
> 워크로드에서 gRPC를 사용하는 경우 GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE를 선택해야 합니다. gRPC 워크로드에는 `POST` 메서드가 필요합니다. 자세한 내용은 CloudFront 배포에 gRPC 사용 섹션을 참조하세요.
> CloudFront에서는 `GET` 및 `HEAD` 요청과 선택적으로 `OPTIONS` 요청에 대한 응답을 캐싱합니다. `OPTIONS` 요청에 대한 응답은 `GET` 및 `HEAD` 요청에 대한 응답과 별도로 캐시됩니다 (`OPTIONS` 메서드는 `OPTIONS` 요청에 대한 캐시 키에 포함됨). CloudFront에서는 다른 메서드를 사용하는 요청에 대한 응답을 캐시하지 않습니다.

> **Important**
>
> GET, HEAD, OPTIONS 또는 GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE를 선택하는 경우 사용자가 수행하길 원치 않는 작업을 수행하지 못하도록 Amazon S3 버킷 또는 사용자 지정 오리진에 대한 액세스를 제한해야 할 수 있습니다. 다음 예에서는 액세스를 제한하는 방법을 설명합니다.
>
> - 배포에 대한 오리진으로 Amazon S3를 사용하려는 경우: CloudFront 오리진 액세스 제어를 만들어 Amazon S3 콘텐츠에 대한 액세스를 제한하고 이 오리진 액세스 제어에 적절한 사용 권한을 부여해야 합니다. 예를 들어, PUT을 사용할 의도로 이러한 메서드만 허용 및 전달하도록 CloudFront를 구성한 경우, `DELETE` 요청을 적절히 처리하도록 Amazon S3 버킷 정책을 구성해야 합니다. 자세한 내용은 Amazon S3 오리진에 대한 액세스 제한 단원을 참조합니다.

- 사용자 지정 오리진을 사용하는 경우: 모든 메서드를 처리하도록 오리진 서버를 구성합니다. 예를 들어, POST를 사용할 의도로 이러한 메서드만 허용 및 전달하도록 CloudFront를 구성한 경우, DELETE 요청을 적절히 처리하도록 오리진 서버를 구성해야 합니다.

## 필드 레벨 암호화 구성

특정 데이터 필드에 필드 레벨 암호화를 적용하고자 하는 경우 드롭다운 목록에서 필드 레벨 암호화 구성을 선택합니다.

자세한 내용은 필드 수준 암호화를 사용하여 민감한 데이터 보호 섹션을 참조하세요.

## 캐싱된 HTTP 메서드

최종 사용자가 OPTIONS 요청을 제출할 때 CloudFront에서 오리진의 응답을 캐싱할지 여부를 지정합니다. CloudFront에서는 GET 및 HEAD 요청에 대한 응답을 항상 캐싱합니다.

## HTTP/2를 통한 gRPC 요청 허용

배포에서 gRPC 요청을 허용할지 여부를 지정합니다. gRPC를 사용하려면 다음 설정을 선택합니다.

- 허용된 HTTP 메서드에서 GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE 메서드를 선택합니다. gRPC에는 POST 메서드가 필요합니다.
- POST 메서드를 선택한 후 나타나는 gRPC 확인란을 선택합니다.
- 지원되는 HTTP 버전에서 HTTP/2를 선택합니다.

자세한 내용은 CloudFront 배포에 gRPC 사용 섹션을 참조하세요.

## 선택한 요청 헤더 기반의 캐시

CloudFront에서 지정된 헤더의 값을 기반으로 객체를 캐싱할지 여부를 지정합니다.

- 없음(캐싱 향상)(None (improves caching)) - CloudFront가 헤더 값을 기반으로 객체를 캐싱하지 않습니다.
- 허용 목록 - CloudFront가 지정된 헤더의 값만을 기반으로 객체를 캐싱합니다. 허용 목록 헤더를 사용하여 CloudFront에서 캐싱의 기준으로 설정할 헤더를 선택합니다.
- 모두(All) - CloudFront가 이 캐시 동작과 연결된 객체를 캐싱하지 않습니다. 대신 CloudFront는 각 요청을 오리진에 전송합니다. (Amazon S3 오리진에는 권장하지 않음)

어떤 옵션을 선택하든 상관없이, CloudFront는 오리진에 특정 헤더를 전달하고 전달한 헤더에 따라 작업을 수행합니다. CloudFront가 헤더 전달을 처리하는 방법에 대한 자세한 내용은 HTTP 요청 헤더 및 CloudFront 동작(사용자 지정 및 Amazon S3 오리진) 단원을 참조하세요.

요청 헤더를 사용하여 CloudFront에서 캐싱을 구성하는 방법에 대한 자세한 내용은 요청 헤더 기반의 콘텐츠 캐싱 단원을 참조하세요.

## 허용 목록 헤더

이 설정은 선택한 요청 헤더 기반의 캐시에 대해 허용 목록을 선택한 경우에만 적용됩니다.

객체를 캐싱할 때 CloudFront에서 고려할 헤더를 지정합니다. 사용 가능한 헤더 목록에서 헤더를 선택하고 추가를 선택합니다. 사용자 지정 헤더를 전달하려면 필드에 헤더 이름을 입력하고 Add Custom(사용자 지정 항목 추가)을 선택합니다.

현재 각 캐시 동작에 대해 허용 목록을 작성할 수 있는 헤더의 최대 수를 살펴보고 더 높은 할당량(이전에는 제한이라고 함)을 요청하려면, 헤더에 대한 할당량 단원을 참조하세요.

## 객체 캐싱

오리진 서버에서 객체에 `Cache-Control` 헤더를 추가하여 CloudFront 캐시에 객체가 머무르는 시간을 제어하는 경우와, `Cache-Control` 값을 변경하지 않으려는 경우 오리진 캐시 헤더 사용(Use Origin Cache Headers)을 선택합니다.

`Cache-Control` 헤더와 관계없이 CloudFront 캐시에 객체가 머무르는 최소 및 최대 시간과 객체에서 `Cache-Control` 헤더가 누락된 경우 CloudFront 캐시에 객체가 머무르는 기본 시간을 지정하려면 사용자 지정(Customize)을 선택합니다. 그런 다음 최소 TTL, 기본 TTL 및 최대 TTL 필드에서 값을 지정합니다.

자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 단원을 참조합니다.

## Minimum TTL

객체가 업데이트되었는지 여부를 확인하기 위해 CloudFront에서 오리진으로 다른 요청을 전송하기 전에 객체를 CloudFront 캐시에 유지할 최소 시간(초)을 지정합니다.

> **Warning**
>
> 최소 TTL이 0보다 큰 경우 오리진 헤더에 `Cache-Control: no-cache`, `no-store` 또는 `private` 지시문이 있더라도 CloudFront는 적어도 캐시 정책의 최소 TTL에 지정된 기간 동안 콘텐츠를 캐싱합니다.

자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 섹션을 참조하세요.

## Maximum TTL

객체가 업데이트되었는지 여부를 확인하도록 CloudFront에서 오리진에 쿼리하기 전에 CloudFront 캐시에서 객체를 머무르게 하려는 최대 시간을 초 단위로 지정합니다. 최대 TTL에 지정하는 값은 오리진이 객체에 `Cache-Control max-age`, `Cache-Control s-maxage` 또는 `Expires` 등의 HTTP 헤더를 추가할 경우에만 적용됩니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 단원을 참조합니다.

최대 TTL의 값을 지정하려면 Object Caching(객체 캐싱) 설정의 Customize(사용자 지정) 옵션을 선택해야 합니다.

최대 TTL의 기본값은 31536000초(1년)입니다. 최소 TTL 또는 기본 TTL의 값을 31536000초 이상으로 변경할 경우 최대 TTL의 기본값은 기본 TTL의 값으로 변경됩니다.

## 기본 TTL

객체가 업데이트되었는지 여부를 결정하도록 CloudFront가 오리진에 다른 요청을 전달하기 전에 객체를 CloudFront 캐시에 유지하려는 기본 시간을 초 단위로 지정합니다. 기본 TTL에 지정하는 값은 오리진이 객체에 `Cache-Control max-age`, `Cache-Control s-maxage`, `Expires` 등의 HTTP 헤더를 추가하지 않는 경우에만 적용됩니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 단원을 참조합니다.

기본 TTL에 값을 지정하려면 Object Caching(객체 캐싱) 설정에 대해 Customize(사용자 지정) 옵션을 선택해야 합니다.

기본 TTL의 기본값은 86400초(1일)입니다. Minimum TTL(최소 TTL)의 값을 86,400초 이상으로 변경할 경우 Default TTL(기본 TTL)의 기본값은 Minimum TTL(최소 TTL)의 값으로 변경됩니다.

## 쿠키 전달

> **Note**
>
> Amazon S3 오리진의 경우 이 옵션은 웹 사이트 엔드포인트로 구성된 버킷에만 적용됩니다.

CloudFront에서 쿠키를 오리진 서버로 전달할지 여부와 전달할 쿠키(전달하는 경우)를 지정합니다. 선택한 쿠키(쿠키의 허용 목록)만 전달하도록 선택한 경우, 쿠키 이름을 허용 목록 쿠키 필드에 입력합니다. 모두(All)를 선택하는 경우 CloudFront에서는 애플리케이션에서 사용하는 쿠키 수와 관계없이 모든 쿠키를 전달합니다.

Amazon S3에서는 쿠키를 처리하지 않으며 오리진에 쿠키를 전달함으로써 캐싱 가능성이 줄어듭니다. Amazon S3 오리진에 요청을 전달하는 캐시 동작의 경우 쿠키 전달(Forward Cookies)에 대해 없음(None)을 선택합니다.

오리진으로 쿠키를 전달하는 것에 대한 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조합니다.

## 허용 목록 쿠키

> **Note**
>
> Amazon S3 오리진의 경우 이 옵션은 웹 사이트 엔드포인트로 구성된 버킷에만 적용됩니다.

쿠키 전달 목록의 허용 목록을 선택하는 경우, 허용 목록 쿠키 필드에 이 캐시 동작에 대해 CloudFront에서 오리진 서버에 전달하려는 쿠키 이름을 입력합니다. 각 쿠키 이름을 새 줄에 입력합니다.

다음 와일드카드를 지정하여 쿠키 이름을 지정할 수 있습니다.

- *는 쿠키 이름의 0개 이상의 문자에 해당합니다.
- ?는 쿠키 이름의 정확히 1문자에 해당합니다.

예를 들어, 다음과 같은 쿠키 이름이 포함된 객체에 대한 최종 사용자 요청을 가정합니다.

`userid_`*`member-number`*

각각의 사용자는 *`member-number`*에 대한 고유한 값을 갖습니다. 이 경우에는 CloudFront에서 각 멤버별로 개별 버전의 객체를 캐싱하려 합니다. 모든 쿠키를 오리진에 전달함으로써 이를 달성할 수는 있지만, 최종 사용자 요청에 CloudFront에서 캐싱하길 원치 않는 쿠키가 일부 포함됩니다. 또는, 다음 값을 쿠키 이름으로 지정할 수 있습니다. 이렇게 하면 CloudFront에서 해당 오리진에 `userid_`로 시작하는 모든 쿠키를 전달합니다.

`userid_*`

현재 각 캐시 동작에 대해 허용 목록을 작성할 수 있는 쿠키 이름의 최대 수를 살펴보고 더 높은 할당량(이전에는 제한이라고 함)을 요청하려면, 쿠키에 대한 할당량(레거시 캐시 설정) 단원을 참조하세요.

## 쿼리 문자열 전달 및 캐싱

CloudFront는 쿼리 문자열 파라미터 값을 기준으로 콘텐츠의 다른 버전을 캐싱할 수 있습니다. 다음 옵션 중 하나를 선택합니다.

None (Improves Caching)

오리진에서 쿼리 문자열 파라미터 값과 상관 없이 객체의 동일한 버전을 반환하는 경우 이 옵션을 선택합니다. 이렇게 하면 CloudFront에서 캐시로부터 요청을 제공할 수 있는 가능성이 높아지고, 그에 따라 성능이 향상되며 오리진에 걸리는 부하가 줄어듭니다.

모두 전달, 허용 목록 기반으로 캐싱

오리진 서버에서 하나 이상의 쿼리 문자열 파라미터를 기준으로 다른 버전의 객체를 반환할 경우 이 옵션을 선택합니다. 그런 다음 쿼리 문자열 허용 목록 필드에서 CloudFront가 이 캐싱의 기준으로 사용하도록 하려는 파라미터를 지정합니다.

Forward all, cache based on all

오리진 서버에서 모든 쿼리 문자열 파라미터를 기준으로 다른 버전의 객체를 반환할 경우 이 옵션을 선택합니다.

성능 향상을 포함하여 쿼리 문자열 파라미터를 기준으로 하는 캐싱에 대한 자세한 내용은 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱 단원을 참조하십시오.

## 쿼리 문자열 허용 목록

이 설정은 쿼리 문자열 전달 및 캐싱에 모두 전달, 허용 목록 기반으로 캐싱을 선택한 경우에만 적용됩니다. CloudFront가 캐싱 기반으로 사용할 쿼리 문자열 파라미터를 지정할 수 있습니다.

## Smooth Streaming

Microsoft Smooth Streaming 포맷의 미디어 파일을 배포하려는데 IIS 서버가 없으면 예를 선택합니다.

Microsoft Smooth Streaming 포맷의 미디어 파일을 배포하기 위한 오리진으로 사용하려는 Microsoft IIS 서버가 있거나, Smooth Streaming 미디어 파일을 배포하지 않으려는 경우 아니요를 선택합니다.

> **Note**
>
> 예를 지정한 경우에도 콘텐츠가 경로 패턴 값과 일치하는 경우 이 캐시 동작을 사용하여 다른 콘텐츠를 배포할 수 있습니다.

자세한 내용은 Microsoft Smooth Streaming용 온디맨드 비디오 구성 단원을 참조합니다.

## 최종 사용자 액세스 제한(서명된 URL 또는 서명된 쿠키 사용)

이 캐시 동작의 `PathPattern`과 일치하는 객체에 대한 요청에 퍼블릭 URL을 사용하려는 경우 아니요를 선택합니다.

이 캐시 동작의 `PathPattern`과 일치하는 객체에 대한 요청에 서명된 URL을 사용하려는 경우 예를 선택합니다. 그런 다음, 서명된 URL 생성에 사용하려는 AWS 계정을 지정합니다. 이러한 계정은 신뢰할 수 있는 서명자라고도 합니다.

신뢰할 수 있는 서명자에 대한 자세한 내용은 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정 단원을 참조합니다.

## 신뢰할 수 있는 서명자

이 설정은 최종 사용자 액세스 제한(서명된 URL 또는 서명된 쿠키 사용)에 대해 예를 선택한 경우에만 적용됩니다.

이 캐시 동작에 대해 신뢰할 수 있는 서명자로 사용하려는 AWS 계정을 선택합니다.

- 셀프: 현재 AWS Management Console에 신뢰할 수 있는 서명자로 로그인된 계정을 사용합니다. 현재 IAM 사용자로 로그인되어 있는 경우, 연결된 AWS 계정이 신뢰할 수 있는 서명자로 추가됩니다.
- 계정 지정(Specify Accounts): 신뢰할 수 있는 서명자에 대한 계정 번호를 [AWS 계정 번호(Account Numbers)] 필드에 입력합니다.

서명된 URL을 생성하려면 AWS 계정에 하나 이상의 활성 CloudFront 키 페어가 있어야 합니다.

> **Important**
>
> 콘텐츠 배포에 이미 사용 중인 배포를 업데이트하려는 경우, 서명된 URL 생성을 시작할 준비가 되었을 때만 신뢰할 수 있는 서명자를 추가합니다. 배포에 신뢰할 수 있는 서명자를 추가한 후에, 사용자는 서명된 URL을 사용하여 이 캐시 동작의 `PathPattern`과 일치하는 객체에 액세스해야 합니다.

## AWS 계정 번호

이 설정은 신뢰할 수 있는 서명자에 대해 계정 지정을 선택한 경우에만 적용됩니다.

현재 계정에 추가로 또는 현재 계정 대신에 AWS 계정을 사용하여 서명된 URL을 생성하려는 경우, 이 필드에 한 줄당 AWS 계정 번호 하나씩을 입력합니다. 다음 사항에 유의하세요.

- 지정하는 계정에는 하나 이상의 활성 CloudFront 키 페어가 있어야 합니다. 자세한 내용은 서명자에 대해 키 페어 생성 단원을 참조합니다.
- IAM 사용자에 대해 CloudFront 키 페어를 생성할 수 없으므로 IAM 사용자는 신뢰할 수 있는 서명자로 사용할 수 없습니다.
- 계정의 AWS 계정 번호를 가져오는 방법에 대한 자세한 내용은 AWS 계정 Management 참조 안내서의 AWS 계정 식별자 보기를 참조하세요.
- 현재 계정에 대한 계정 번호를 입력하면 CloudFront에서는 셀프 확인란을 자동으로 선택하고 AWS 계정 번호 목록에서 계정 번호를 삭제합니다.

## 자동으로 객체를 압축

최종 사용자가 압축된 콘텐츠를 지원하는 경우 CloudFront가 특정 유형의 파일을 자동 압축하도록 하려면 예(Yes)를 선택합니다. CloudFront가 콘텐츠를 압축하면 파일 크기가 더 작아지므로 다운로드 속도가 빨라지고 웹 페이지는 사용자에게 더 빨리 렌더링됩니다. 자세한 내용은 압축된 파일 제공 단원을 참조합니다.

## CloudFront 이벤트

이 설정은 Lambda 함수 연결에 적용됩니다.

다음 CloudFront 이벤트가 하나 이상 발생할 때 Lambda 함수를 실행하도록 선택할 수 있습니다.

- CloudFront가 최종 사용자의 요청을 수신할 때(최종 사용자 요청)
- CloudFront가 오리진에 요청을 전달하기 전(오리진 요청)
- CloudFront가 오리진의 응답을 수신할 때(오리진 응답)
- CloudFront가 최종 사용자에게 응답을 반환하기 전(최종 사용자 응답)

자세한 내용은 함수를 트리거할 이벤트를 선택합니다. 단원을 참조합니다.

## Lambda 함수 ARN

이 설정은 Lambda 함수 연결에 적용됩니다.

트리거를 추가할 Lambda 함수의 Amazon 리소스 이름(ARN)을 지정합니다. 함수의 ARN을 가져오는 방법에 대한 자세한 내용은 CloudFront 콘솔을 사용한 트리거 추가 절차의 1단계를 참조하세요.

## 본문 포함

이 설정은 Lambda 함수 연결에 적용됩니다.

자세한 내용은 본문 포함을 참조하세요.

# 배포 설정

다음 값들이 전체 배포에 적용됩니다.

주제


- 요금 계층
- AWS WAF 웹 ACL
- 대체 도메인 이름(CNAME)
- SSL 인증서
- 사용자 지정 SSL 클라이언트 지원
- 보안 정책(최소 SSL/TLS 버전)
- 지원되는 HTTP 버전
- 기본 루트 객체
- 표준 로깅
- 연결 로그
- 로그 접두사
- 쿠키 로깅
- IPv6 활성화(뷰어 요청)
- 상호 인증
- 사용자 지정 오리진에 IPv6 활성화(오리진 요청)
- 설명
- 배포 상태


## 요금 계층

CloudFront 서비스를 위해 지급하려는 최고가에 해당하는 요금 계층을 선택합니다. 기본적으로 CloudFront는 모든 CloudFront 리전의 엣지 로케이션에서 객체를 제공합니다.

가격 등급과 가격 등급 선택이 배포를 위한 CloudFront 성능에 미치는 영향에 대한 자세한 내용은 CloudFront 요금제를 참조합니다.

## AWS WAF 웹 ACL

웹 애플리케이션 및 API를 보호하여 요청이 서버에 도달하기 전에 요청을 차단할 수 있는 웹 애플리케이션 방화벽인 AWS WAF를 사용하여 CloudFront 배포를 보호할 수 있습니다. CloudFront 배포를 생성하거나 편집할 때 배포에 AWS WAF 활성화할 수 있습니다.

선택적으로 나중에 AWS WAF 콘솔(https://console.aws.amazon.com/wafv2/)에서 애플리케이션과 관련된 다른 위협에 대한 추가 보안 보호를 구성할 수 있습니다.

AWS WAF에 대한 자세한 내용은 AWS WAF개발자 안내서 단원을 참조하십시오.

## 대체 도메인 이름(CNAME)

선택. 사용자가 배포를 만들 때 CloudFront에서 할당하는 도메인 이름 대신 객체의 URL에 사용할 하나 이상의 도메인 이름을 지정합니다. 도메인 이름을 소유하거나, 도메인 이름을 사용할 권한이 있어야 합니다. 사용 권한은 SSL/TLS 인증서를 추가하여 확인합니다.

예를 들어 객체의 URL이 아래와 같은 경우

```
/images/image.jpg
```

아래와 같이 변경하고

```
https://www.example.com/images/image.jpg
```

아래와 같이 변경하지 않으려면

```
https://d111111abcdef8.cloudfront.net/images/image.jpg
```

`www.example.com`에 대한 CNAME을 추가합니다.

> **Important**
>
> 배포에 `www.example.com`에 대한 CNAME을 추가하는 경우, 다음도 수행해야 합니다.
>
> - DNS 서비스로 CNAME 레코드를 만들거나 업데이트해 `www.example.com`에 대한 쿼리를 `d111111abcdef8.cloudfront.net`로 라우팅합니다.

- 배포에 추가하는 도메인 이름(CNAME)이 포함된 신뢰할 수 있는 인증 기관(CA)의 인증서를 CloudFront에 추가하여 도메인 이름을 사용할 권한을 검증합니다.

도메인에 대한 DNS 서비스 공급자를 통해 CNAME 레코드를 만들려면 권한이 있어야 합니다. 일반적으로 이 경우 도메인을 소유하고 있거나, 도메인 소유자를 위해 애플리케이션을 개발하는 중일 수도 있습니다.

현재 하나의 배포에 추가할 수 있는 대체 도메인 이름의 최대 수를 살펴보고 더 높은 할당량(이전에는 제한이라고 함)을 요청하려면, 배포의 일반 할당량 단원을 참조하세요.

대체 도메인 이름에 대한 자세한 내용은 대체 도메인 이름(CNAME)을 추가하여 사용자 지정 URL 사용 단원을 참조하십시오. CloudFront URL에 대한 자세한 내용은 CloudFront에서 파일에 대한 URL 형식 사용자 지정 단원을 참조하세요.

## SSL 인증서

배포에 사용할 대체 도메인 이름을 지정한 경우, Custom SSL Certificate(사용자 지정 SSL 인증서)를 선택한 후 대체 도메인 이름을 사용할 권한을 검증하고, 대체 도메인 이름이 포함된 인증서를 선택합니다. 최종 사용자가 HTTPS를 사용하여 객체에 액세스하게 하려는 경우, 이를 지원하는 설정을 선택합니다.

- 기본 CloudFront 인증서(*.cloudfront.net)(Default CloudFront Certificate (*.cloudfront.net)) - CloudFront 도메인 이름을 객체(예: `https://d111111abcdef8.cloudfront.net/image1.jpg`)의 URL에 사용하려는 경우, 이 옵션을 선택합니다.
- 사용자 정의 SSL 인증서(Custom SSL Certificate) - 자체 도메인 이름을 객체의 URL에 대체 도메인 이름(예: `https://example.com/image1.jpg`)으로 사용하려는 경우, 이 옵션을 선택합니다. 그런 다음 대체 도메인 이름이 포함된 사용할 인증서를 선택합니다. 인증서 목록에는 다음이 포함될 수 있습니다.
  - 에서 제공하는 인증서AWS Certificate Manager
  - 서드 파티 인증 기관으로부터 구매했고 ACM에 업로드한 인증서
  - 서드 파티 인증 기관으로부터 구매했고 IAM 인증서 스토어에 업로드한 인증서

  이 설정을 선택한 경우 대체 도메인 이름만 객체 URL에 사용하는 것이 좋습니다(https://example.com/logo.jpg). CloudFront 배포 도메인 이름(https://d111111abcdef8.cloudfront.net/logo.jpg)을 사용하고 클라이언트가 SNI를 지원하지 않는 이전 최종 사용자를 사용하는 경우, 최종

사용자가 응답하는 방식은 지원되는 클라이언트(Clients Supported)에서 선택하는 값에 따라 달라집니다.

- 모든 클라이언트(All Clients): CloudFront 도메인 이름이 SSL/TLS 인증서의 도메인 이름과 일치하지 않으므로 최종 사용자에게 경고가 표시됩니다.
- SNI(서버 이름 표시)를 지원하는 클라이언트만(Only Clients that Support Server Name Indication (SNI)): CloudFront에서 객체를 반환하지 않고 해당 최종 사용자와의 연결을 끊습니다.

## 사용자 지정 SSL 클라이언트 지원

SSL 인증서에 대해 사용자 정의 SSL 인증서(example.com)를 선택한 경우에만 적용됩니다. 배포에 대해 하나 이상의 대체 도메인 이름과 사용자 정의 SSL 인증서를 지정한 경우 CloudFront에서 HTTPS 요청을 제공할 방법을 선택합니다.

- SNI(서버 이름 표시)를 지원하는 클라이언트 - (권장) - 이 설정을 사용하면 거의 모든 최신 웹 브라우저와 클라이언트가 SNI를 지원하므로 배포에 연결할 수 있습니다. 그러나 일부 최종 사용자는 SNI를 지원하지 않는 이전 웹 브라우저나 클라이언트를 사용할 수 있으므로 배포에 연결할 수 없습니다.

  CloudFront API를 사용하여 이 설정을 적용하려면 `sni-only` 필드에서 `SSLSupportMethod`를 지정합니다. CloudFormation에서 필드의 이름은 `SslSupportMethod`입니다(다른 대소문자에 유의).
- 레거시 클라이언트 지원 - 이 설정을 사용하면 SNI를 지원하지 않는 이전 웹 브라우저와 클라이언트가 배포에 연결할 수 있습니다. 그러나 이 설정에는 추가 월별 요금이 발생합니다. 정확한 가격을 보려면 Amazon CloudFront 요금 페이지로 이동하여 전용 IP 사용자 정의 SSL을 검색하세요.

  CloudFront API를 사용하여 이 설정을 적용하려면 `vip` 필드에서 `SSLSupportMethod`를 지정합니다. CloudFormation에서 필드의 이름은 `SslSupportMethod`입니다(다른 대소문자에 유의).

자세한 내용은 CloudFront에서 HTTPS 요청을 제공하는 방식 선택 섹션을 참조하세요.

## 보안 정책(최소 SSL/TLS 버전)

CloudFront가 최종 사용자(클라이언트)와의 HTTPS 연결에 사용할 보안 정책을 지정합니다. 보안 정책은 다음 두 가지 설정을 결정합니다.

- CloudFront가 최종 사용자와 통신하는 데 사용하는 최소 SSL/TLS 프로토콜.
- CloudFront가 최종 사용자로 반환하는 콘텐츠를 암호화할 때 사용할 수 있는 암호입니다.

각 보안 정책에 포함되는 프로토콜과 암호를 포함해 보안 정책에 관한 자세한 내용은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호 단원을 참조하세요.

사용 가능한 보안 정책은 SSL 인증서 및 사용자 지정 SSL 클라이언트 지원(CloudFront API의 `CloudFrontDefaultCertificate` 및 `SSLSupportMethod`라고도 함)에 지정한 값에 따라 다릅니다.

- SSL 인증서(SSL Certificate)가 기본 CloudFront 인증서(*.cloudfront.net)(Default CloudFront Certificate(*.cloudfront.net))인 경우(`CloudFrontDefaultCertificate`가 API에서 `true`일 때) CloudFront가 보안 정책을 자동으로 TLSv1로 설정합니다.
- SSL 인증서가 사용자 정의 SSL 인증서(example.com)이고 사용자 지정 SSL 클라이언트 지원이 서버 이름 표시(SNI)를 지원하는 클라이언트(권장)일 때(API에서 `CloudFrontDefaultCertificate`가 `false`이고 `SSLSupportMethod`가 `sni-only`일 때) 다음 보안 정책 중에서 선택할 수 있습니다.
  - TLSv1.3_2025
  - TLSv1.2_2025
  - TLSv1.2_2021
  - TLSv1.2_2019
  - TLSv1.2_2018
  - TLSv1.1_2016
  - TLSv1_2016
  - TLSv1
- SSL 인증서가 사용자 정의 SSL 인증서(example.com)이고 사용자 지정 SSL 클라이언트 지원이 레거시 클라이언트 지원일 때(API에서 `CloudFrontDefaultCertificate`가 `false`이고 `SSLSupportMethod`가 `vip`일 때) 다음 보안 정책 중에서 선택할 수 있습니다.
  - TLSv1
  - SSLv3

  이 구성에서는 TLSv1.3_2025, TLSv1.2_2025, TLSv1.2_2021, TLSv1.2_2019, TLSv1.2_2018, TLSv1.1_2016 및 TLSv1_2016 보안 정책을 CloudFront 콘솔 또는 API에서 사용할 수 없습니다. 이러한 보안 정책 중 하나를 사용할 경우 다음과 같은 옵션이 있습니다.

  - 배포에 전용 IP 주소로 레거시 클라이언트 지원이 필요한지 여부를 평가합니다. 최종 사용자 측에서 서버 이름 표시(SNI)를 지원하는 경우에는 배포의 사용자 지정 SSL 클라이언트 지원 설정을 서버 이름 표시(SNI)를 지원하는 클라이언트로 설정하는 것이 좋습니다(API에서는

SSLSupportMethod를 sni-only로 설정). 이를 통해 사용 가능한 TLS 보안 정책을 사용할 수 있으며 CloudFront 요금도 줄일 수 있습니다.

- 레거시 클라이언트 지원을 전용 IP 주소로 유지해야 하는 경우에는 AWS Support Center에서 사례를 생성하여 다른 TLS 보안 정책(TLSv1.3_2025, TLSv1.2_2025, TLSv1.2_2021, TLSv1.2_2019, TLSv1.2_2018, TLSv1.1_2016 또는 TLSv1_2016) 중 하나를 요청할 수 있습니다.

> **Note**
>
> AWS Support로 연락해 이러한 변경을 요청하기 전에 다음 사항을 고려합니다.
>
> - 이러한 보안 정책 중 하나(TLSv1.3_2025, TLSv1.2_2025, TLSv1.2_2021, TLSv1.2_2019, TLSv1.2_2018, TLSv1.1_2016 또는 TLSv1_2016)를 레거시 클라이언트 지원 배포에 추가하면 AWS 계정의 모든 레거시 클라이언트 지원 배포에 대한 SNI 이외의 모든 뷰어 요청에 보안 정책이 적용됩니다. 그러나 최종 사용자가 레거시 클라이언트 지원을 사용하여 배포에 SNI 요청을 보낼 경우 해당 배포의 보안 정책이 적용됩니다. 원하는 보안 정책이 AWS 계정의 모든 레거시 클라이언트 지원 배포에 전송된 모든 뷰어 요청에 적용되도록 하려면 원하는 보안 정책을 각 배포에 개별적으로 추가합니다.
> - 당연히 새 보안 정책에서는 이전 보안 정책과 동일한 암호 및 프로토콜을 지원하지 않습니다. 예를 들어, 배포의 보안 정책을 TLSv1에서 TLSv1.1_2016으로 업그레이드하도록 선택한 경우 해당 배포는 DES-CBC3-SHA 암호를 더 이상 지원하지 않습니다. 각 보안 정책이 지원하는 암호 및 프로토콜에 대한 자세한 내용은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호 단원을 참조하세요.

## 지원되는 HTTP 버전

최종 사용자가 CloudFront와 통신할 때 배포에서 지원할 HTTP 버전을 선택합니다.

뷰어와 CloudFront가 HTTP/2를 사용하는 경우 뷰어는 TLSv1.2 이상 및 SNI(Server Name Indication)를 지원해야 합니다.

뷰어와 CloudFront가 HTTP/3를 사용하는 경우 뷰어는 TLSv1.3 및 SNI(Server Name Indication)를 지원해야 합니다. CloudFront는 HTTP/3 연결 마이그레이션을 지원하므로 뷰어가 연결 손실 없이 네트워크를 전환할 수 있습니다. 연결 마이그레이션에 대한 자세한 내용은 RFC 9000의 연결 마이그레이션을 참조하세요.

> **Note**
>
> 지원되는 TLSv1.3 암호에 대한 자세한 내용은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호 섹션을 참조하세요.

> **Note**
>
> Amazon Route 53을 사용하는 경우, 클라이언트가 지원한다면 HTTPS 레코드를 사용하여 DNS 조회의 일부로 프로토콜 협상을 허용할 수 있습니다. 자세한 내용은 Create alias resource record set 섹션을 참조하세요.

## 기본 루트 객체

선택. 최종 사용자가 배포의 객체(`index.html`)가 아닌 배포의 루트 URL(`https://www.example.com/`)을 요청할 때 CloudFront가 소스(예: `https://www.example.com/product-description.html`)으로부터 요청할 객체입니다. 기본 루트 객체를 지정하면 배포 콘텐츠가 노출되지 않습니다.

이름의 최대 길이는 255자입니다. 이름에는 다음 문자 중 어느 것이든 포함할 수 있습니다.

- A~Z, a~z
- 0~9
- _ - . * $ / ~ " '
- &, `&`로 처리되고 반환됨

기본 루트 객체를 지정하는 경우 객체 이름만 입력합니다(예: `index.html`). 객체 이름 앞에 `/`를 추가하지 마십시오.

자세한 내용은 기본 루트 객체 지정 섹션을 참조하세요.

## 표준 로깅

CloudFront에서 객체에 대한 각 요청의 정보를 로그하고 이 로그 파일을 저장할지 지정합니다. 언제든지 로깅을 활성화 또는 비활성화할 수 있습니다. 로그를 사용해도 추가 비용이 들지 않지만 파일을 저장하고 이에 액세스하는 데 요금이 발생합니다. 로그는 언제든지 삭제할 수 있습니다.

CloudFront는 다음의 표준 로깅 옵션을 지원합니다.

- 표준 로깅(v2) - Amazon CloudWatch Logs, Amazon Data Firehose, Amazon Simple Storage Service(Amazon S3)를 포함한 전송 대상으로 로그를 전송할 수 있습니다.
- 표준 로깅(레거시) - Amazon S3 버킷에만 로그를 보낼 수 있습니다.

## 연결 로그

배포에 대한 상호 인증을 켜면 CloudFront는 배포로 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 연결 로그를 사용한 관찰성 섹션을 참조하세요.

## 로그 접두사

(선택 사항) 표준 로깅(레거시)을 사용하면 CloudFront에서 이 배포에 대한 액세스 로그 파일 이름에 접두사로 지정하려는 문자열(해당하는 경우)을 지정합니다(예: `exampleprefix/`). 후행 슬래시(/)는 선택 사항이지만 로그 파일을 쉽게 검색할 수 있도록 넣는 것이 좋습니다. 자세한 내용은 표준 로깅(레거시) 구성 섹션을 참조하세요.

## 쿠키 로깅

CloudFront에서 쿠키를 액세스 로그에 포함하려는 경우 설정(On)을 선택합니다. 로그에 쿠키를 포함하도록 선택한 경우, CloudFront에서는 이 배포에 대한 캐시 동작을 어떻게 구성했는지(전체 쿠키 전달, 쿠키 전달 안 함 또는 오리진에 지정한 쿠키 목록 전달)와 관계없이 모든 쿠키를 로깅합니다.

Amazon S3에서는 쿠키를 처리하지 않습니다. 따라서 배포에 Amazon EC2 또는 다른 사용자 지정 오리진 역시 포함하지 않는 한 쿠키 로깅(Cookie Logging) 값으로 해제(Off)를 선택하는 것이 좋습니다.

쿠키에 대한 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하십시오.

## IPv6 활성화(뷰어 요청)

CloudFront가 IPv4 및 IPv6 IP 주소의 뷰어 요청에 응답하도록 하려면 IPv6 활성화를 선택합니다. 자세한 내용은 CloudFront 배포에 IPv6 활성화 섹션을 참조하세요.

## 상호 인증

선택 사항입니다. CloudFront 배포에 대해 상호 인증을 켜도록 선택할 수 있습니다. 자세한 내용은 CloudFront에 대한 상호 TLS 인증(뷰어 mTLS) 섹션을 참조하세요.

### 사용자 지정 오리진에 IPv6 활성화(오리진 요청)

사용자 지정 오리진(Amazon S3 및 VPC 오리진 제외)을 사용하는 경우 배포의 오리진 설정을 사용자 지정하여 CloudFront가 IPv4 또는 IPv6 주소를 사용하여 오리진에 연결하는 방법을 선택할 수 있습니다. 자세한 내용은 CloudFront 배포에 IPv6 활성화 섹션을 참조하세요.

### 설명

선택 사항입니다. 배포를 만들 때 최대 128자의 설명을 포함할 수 있습니다. 설명은 언제든지 업데이트할 수 있습니다.

### 배포 상태

배포가 완료되었을 때 활성화 또는 비활성화할지 여부를 나타냅니다.

- Enabled(활성화됨)는 배포가 완료되는 대로 이 배포의 도메인 이름을 사용하는 링크를 배포할 수 있고 사용자가 콘텐츠를 가져올 수 있음을 의미합니다. 배포가 활성화되어 있으면 CloudFront에서는 해당 배포와 연결된 도메인 이름을 사용하는 콘텐츠에 대해 최종 사용자 요청을 수락하고 처리합니다.

  CloudFront 배포를 생성, 수정 또는 삭제할 경우 CloudFront 데이터베이스에 변경 사항이 전파되는 데에는 다소 시간이 소요됩니다. 배포 정보에 대한 즉각적인 요청에는 변경 사항이 반영되지 않을 수 있습니다. 전파는 대개 몇 분 내에 완료되지만, 이때 시스템에 과도한 로드가 걸리거나 네트워크 파티션이 증가할 수 있습니다.
- Disabled(비활성화됨)는 배포가 완료되어 사용할 준비가 되더라도 사용자가 이를 사용할 수 없음을 의미합니다. 배포가 비활성화되어 있으면 CloudFront에서는 해당 배포와 연결된 도메인 이름을 사용하는 최종 사용자 요청을 수락하지 않습니다. 배포의 구성을 업데이트하여 비활성 상태에서 활성 상태로 전환할 때까지는 누구도 이 배포를 사용할 수 없습니다.

원하는 만큼 배포를 비활성 상태와 활성 상태 간에 전환할 수 있습니다. 배포 구성 업데이트에 대한 프로세스를 따르세요. 자세한 내용은 배포 업데이트 단원을 참조합니다.

## 사용자 지정 오류 페이지 및 오류 캐싱

Amazon S3 또는 사용자 지정 오리진에서 HTTP 4xx 또는 5xx 상태 코드를 CloudFront에 반환할 때 CloudFront에서 최종 사용자에게 객체(예: HTML 파일)를 반환하도록 할 수 있습니다. 또한 CloudFront 엣지 캐시에서 오리진의 오류 응답 또는 사용자 지정 오류 페이지가 캐싱되는 시간을 지정할 수도 있습니다. 자세한 내용은 특정 HTTP 상태 코드에 대한 사용자 지정 오류 페이지 생성 단원을 참조합니다.

> **Note**
>
> 다음 값은 Create Distribution 마법사에 포함되어 있지 않으므로 배포를 업데이트할 때만 사용자 지정 오류 페이지를 구성할 수 있습니다.

주제


- HTTP 오류 코드
- 응답 페이지 경로
- HTTP 응답 코드
- 오류 캐싱 최소 TTL(초)


## HTTP 오류 코드

CloudFront에서 사용자 지정 오류 페이지를 반환하려는 HTTP 상태 코드입니다. CloudFront에서 캐싱하는 HTTP 상태 코드의 일부 또는 전부에 대한(또는 상태 코드가 없는) 사용자 지정 오류 페이지를 반환하도록 CloudFront를 구성할 수 있습니다.

## 응답 페이지 경로

403 등의 오류 코드(Error Code)로 지정한 HTTP 상태 코드가 오리진에서 반환될 때 CloudFront에서 최종 사용자에게 반환하는 사용자 지정 오류 페이지의 경로(예: `/4xx-errors/403-forbidden.html`)입니다. 객체와 사용자 지정 오류 페이지를 서로 다른 위치에 저장하려는 경우, 배포에는 다음을 충족하는 캐시 동작이 포함되어야 합니다.

- 경로 패턴의 값이 사용자 지정 오류 메시지와 일치합니다. `/4xx-errors`라는 이름의 디렉터리에 있는 Amazon S3 버킷에 4xx 오류에 대한 사용자 지정 오류 페이지를 저장한 경우를 예로 들어 보겠습니다. 배포에는 사용자 지정 오류 페이지에 대한 요청을 해당 위치로 라우팅하는 경로 패턴(예: /4xx-errors/*)의 캐시 동작이 포함되어야 합니다.
- Origin(오리진)의 값은 사용자 지정 오류 페이지를 포함한 오리진에 대한 Origin ID(오리진 ID)의 값을 지정합니다.

## HTTP 응답 코드

CloudFront에서 사용자 지정 오류 페이지와 함께 최종 사용자에게 반환하려는 HTTP 상태 코드입니다.

### 오류 캐싱 최소 TTL(초)

CloudFront에서 오리진 서버로부터 오류 응답을 캐싱하려는 최소 시간입니다.

## 지리적 제한

선택한 국가의 사용자가 콘텐츠에 액세스하지 못하게 차단하려면 허용 목록 또는 차단 목록으로 CloudFront 배포를 구성할 수 있습니다. 지리적 제한을 구성하는 데 드는 추가 요금은 없습니다. 자세한 내용은 콘텐츠의 지리적 배포 제한 섹션을 참조하세요.

# 배포 테스트

배포를 생성하면 CloudFront에서는 오리진 서버의 위치를 알 수 있으며, 사용자는 배포와 연결된 도메인 이름을 알 수 있습니다. 배포를 테스트하려면 다음과 같이 합니다.

1. 배포가 완료될 때까지 기다립니다.
   - 콘솔에서 배포 세부 정보를 확인합니다. 배포가 완료되면 마지막 수정 필드가 배포 중에서 날짜 및 시간으로 변경됩니다.
2. 다음 절차에 따라 CloudFront 도메인 이름을 사용하여 객체에 대한 링크를 생성합니다.
3. 링크를 테스트합니다. CloudFront가 웹 페이지 또는 애플리케이션에 객체를 제공합니다.

## 객체에 대한 링크 생성

다음 절차에 따라 CloudFront 웹 배포의 객체에 대한 테스트 링크를 생성합니다.

웹 배포에 객체에 대한 링크를 생성하려면

1. 다음 HTML 코드를 새 파일에 복사하고 *domain-name*과 *object-name*을 각각 배포의 도메인 이름과 객체의 이름으로 대체합니다.

```
<html>
<head>
 <title>My CloudFront Test</title>
</head>
<body>
 <p>My text content goes here.</p>
 <p><img src="https://domain-name/object-name" alt="my test image"></p>
</body>
</html>
```

예를 들어, 도메인 이름이 `d111111abcdef8.cloudfront.net`이고 객체가 `image.jpg`인 경우 링크의 URL은 다음과 같습니다.

`https://d111111abcdef8.cloudfront.net/image.jpg`.

객체가 오리진 서버의 폴더에 있는 경우, URL에 이 폴더도 포함해야 합니다. 예를 들어, image.jpg가 오리진 서버의 이미지 폴더에 있으면 URL은 다음과 같습니다.

`https://d111111abcdef8.cloudfront.net/images/image.jpg`

2. 파일 이름 확장명이 .html인 파일에 HTML 코드를 저장합니다.
3. 브라우저에서 웹 페이지를 열어 객체를 볼 수 있는지 확인합니다.

브라우저가 엣지 로케이션에서 제공한 이미지 파일이 포함된 페이지를 반환합니다. 이 엣지는 CloudFront에서 객체를 서비스하기에 적합한 것으로 확인된 위치입니다.

# 배포 업데이트

CloudFront 콘솔에서 AWS 계정과 연결된 CloudFront 배포를 확인하고, 배포에 대한 설정을 보며, 대부분의 설정을 업데이트할 수 있습니다. 변경한 설정은 배포가 AWS 엣지 로케이션에 전파되어야 적용됩니다.

## 콘솔에서 배포 업데이트

다음 절차에서는 콘솔에서 CloudFront 배포를 업데이트하는 방법을 보여줍니다.

Multi-tenant

다중 테넌트 배포를 업데이트하려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 다중 테넌트 배포의 ID를 검색하여 선택합니다.
3. 업데이트하려는 설정의 탭을 선택합니다.
4. 업데이트한 후 변경한 내용을 저장하려면 변경 사항 저장을 선택합니다. 업데이트할 수 있는 설정에 대한 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하시기 바랍니다.

CloudFront API를 사용하여 배포를 업데이트할 수도 있습니다.

- 배포를 업데이트하려면 Amazon CloudFront API 참조의 UpdateDistribution을 참조하세요.

> **Important**
>
> 배포 업데이트 시 배포를 처음 만들 때는 필요하지 않은 여러 개의 추가 필드가 필요합니다. CloudFront API를 사용하여 배포를 업데이트하는 경우 필수 필드를 모두 포함하려면 Amazon CloudFront API 참조의 UpdateDistribution에 설명된 단계를 따르세요.

배포 테넌트의 다중 테넌트 배포를 변경하려면 배포 테넌트를 업데이트합니다. 또한 배포 테넌트를 업데이트하여 도메인, 인증서, 사용자 지정 또는 파라미터 값을 업데이트합니다. 배포 테넌트 인증서 업데이트에 대한 자세한 내용은 도메인 및 인증서 추가(분산 테넌트) 섹션을 참조하시기 바랍니다.

배포 테넌트를 업데이트하려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. SaaS에서 배포 테넌트를 선택합니다.
3. 배포 테넌트를 검색합니다. 검색 창의 드롭다운 메뉴를 사용하여 도메인, 이름, 배포 ID, 인증서 ID, 연결 그룹 ID 또는 웹 ACL ID를 기준으로 필터링합니다.
4. 배포 테넌트 이름을 선택합니다.
5. 일반 세부 정보를 업데이트하려면 편집을 선택하고 업데이트한 다음 배포 테넌트 업데이트를 선택합니다.
6. 업데이트할 다른 설정에 적합한 탭을 선택하여 업데이트하고 이를 저장합니다. 사용자 지정할 수 있는 배포 테넌트 설정에 대한 자세한 내용은 배포 테넌트 사용자 지정 섹션을 참조하시기 바랍니다.

### Standard

표준 배포를 업데이트하려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포의 ID를 선택합니다. 이 목록에는 CloudFront 콘솔에 로그인하는 데 사용한 AWS 계정과 연결된 배포가 모두 포함되어 있습니다.

3. 일반 설정을 업데이트하려면 편집을 선택합니다. 또는 업데이트하려는 설정의 탭을 선택합니다.
4. 업데이트한 다음, 변경 사항 저장을 선택합니다. 필드에 대한 자세한 내용은 다음 주제를 참조하십시오.
   - 일반 설정: 배포 설정
   - 오리진 설정: 오리진 설정
   - 캐시 동작 설정: 캐시 동작 설정
5. 배포에서 오리진을 삭제하려면 다음과 같이 합니다.
   a. 동작을 선택하고 해당 오리진과 연결된 기본 캐시 동작을 다른 오리진으로 옮겼는지 확인합니다.
   b. 오리진을 선택한 후 오리진을 선택합니다.
   c. 삭제를 선택합니다.

CloudFront API를 사용하여 배포를 업데이트할 수도 있습니다.

- 배포를 업데이트하려면 Amazon CloudFront API 참조의 UpdateDistribution을 참조하세요.

> **Important**
> 배포를 업데이트하는 경우에는 배포를 만들 때는 필요 없었던 여러 개의 추가 필드가 필요합니다. CloudFront API를 사용하여 배포를 업데이트하는 경우 필수 필드를 모두 포함하려면 Amazon CloudFront API 참조의 UpdateDistribution에 설명된 단계를 따르세요.

배포 구성에 대한 변경 내용을 저장하면 CloudFront에서 이러한 변경 내용을 모든 엣지 로케이션으로 전파하기 시작합니다. 연속적인 구성 변경 사항은 해당 순서로 전파됩니다. 구성이 엣지 로케이션에서 업데이트될 때까지 CloudFront는 이전 구성에 따라 해당 위치에서 콘텐츠를 계속 제공합니다. 구성이 엣지 로케이션에서 업데이트된 후에는 CloudFront가 즉시 새 구성에 따라 해당 위치에서 콘텐츠를 제공하기 시작합니다.

변경 사항이 모든 엣지 로케이션으로 동시에 전파되지는 않습니다. CloudFront가 변경 내용을 전파하고 있는 동안은 지정된 엣지 로케이션이 이전 구성과 새 구성 중 어느 것에 기초하여 콘텐츠를 제공하는지 확인할 수 없습니다.

> **Note**
>
> 드문 경우지만 호스트 또는 네트워크 링크가 중단되는 경우 일부 배포 테넌트 트래픽은 변경 사항이 네트워크를 통과할 때까지 짧은 기간 동안 이전 구성을 사용하여 제공될 수 있습니다.

변경 사항이 전파되는 시기를 확인하려면 콘솔에서 배포 세부 정보를 확인합니다. 배포가 완료되면 마지막 수정 필드가 배포 중에서 날짜 및 시간으로 변경됩니다.

# 배포 태깅

태그란 AWS 리소스를 식별하고 정리할 때 사용할 수 있는 단어 또는 문구입니다. 각 리소스에 태그를 여러 개 추가할 수 있고, 각 태그는 사용자가 정의한 키와 값을 포함할 수 있습니다. 예를 들어, 키는 "도메인"이고 값은 "example.com"일 수 있습니다. 추가하는 태그에 따라 리소스를 검색하고 필터링할 수 있습니다.

다음 예와 같이 CloudFront에서 태그를 사용할 수 있습니다.

- CloudFront 배포에 태그 기반 권한을 적용합니다. 자세한 내용은 CloudFront와 ABAC 섹션을 참조하세요.
- 다양한 범주에서 청구 정보를 추적합니다. CloudFront 배포나 다른 AWS 리소스(Amazon EC2 인스턴스 또는 Amazon S3 버킷 등)에 태그를 적용하고 그 태그를 활성화하면 AWS는 사용 내역 및 비용을 활성 태그 기준으로 집계한 비용 할당 보고서를 CSV 파일 형식으로 작성합니다.

  비즈니스 범주를 나타내는 태그(예: 비용 센터, 애플리케이션 이름 또는 소유자)를 적용하여 여러 서비스에 대한 비용을 정리할 수 있습니다. 비용 할당 태그 사용에 대한 자세한 내용은 AWS Billing 사용 설명서의 비용 할당 태그 사용을 참조하십시오.

> **참고**
>
> - 배포에는 태그를 지정할 수 있지만 원본 액세스 ID 또는 배포에는 태그 지정이 불가능합니다.
> - 현재 CloudFront에는 Tag Editor나 리소스 그룹을 사용할 수 없습니다.
> - 현재 배포에 추가할 수 있는 태그의 최대 수는 일반 할당량 단원을 참조하세요.

목차

- 태그 제한
- 배포에 태그 추가, 편집, 삭제
- 프로그래밍 방식 태깅

## 태그 제한

태그에 적용되는 기본 제한은 다음과 같습니다.

- 배포당 최대 태그 수는 일반 할당량 섹션을 참조하세요.
- 최대 키 길이 - 유니코드 128자
- 최대 값 길이 - 유니코드 256자
- 키 및 값의 유효값 - a-z, A-Z, 0-9, 공백 및 특수 문자 _ . : / = + - 및 @
- 태그 키와 값은 대/소문자를 구분합니다
- 키 접두사로 `aws:`를 사용하지 마세요. 이 접두사는 AWS용으로 예약되어 있습니다.

## 배포에 태그 추가, 편집, 삭제

CloudFront 콘솔을 사용하여 배포의 태그를 관리할 수 있습니다.

배포에 태그를 추가, 편집 또는 삭제하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 업데이트하려는 배포의 ID를 선택합니다.
3. [Tags] 탭을 선택합니다.
4. 태그 관리를 선택합니다.
5. 태그 관리 페이지에서 다음 작업을 수행할 수 있습니다.
   - 태그를 추가하려면 키를 입력하고, 필요한 경우 태그 값도 입력합니다. 태그를 더 추가하려면 새 태그 추가를 선택합니다.
   - 태그를 편집하려면 태그의 키나 해당 값 또는 둘 다를 변경합니다. 태그의 값은 삭제할 수 있지만 키는 필수입니다.
   - 태그를 삭제하려면 제거를 선택합니다.
6. 변경 사항 저장을 선택합니다.

## 프로그래밍 방식 태깅

CloudFront API, AWS Command Line Interface(AWS CLI), AWS SDK 및 AWS Tools for Windows PowerShell을 사용하여 태그를 적용할 수도 있습니다. 자세한 내용은 다음 항목을 참조하세요.

- CloudFront API 작업:
  - ListTagsForResource
  - TagResource
  - UntagResource
- AWS CLI - AWS CLI 명령 참조에서 cloudfront 참조
- AWS SDK - AWS 설명서 페이지에서 해당 SDK 설명서 참조
- Windows PowerShell용 도구 – AWS Tools for PowerShell Cmdlet Reference(Cmdlet 참조)에서 Amazon CloudFront를 참조하십시오.

## 배포 삭제

다음 절차에서는 CloudFront 콘솔을 사용하여 배포를 삭제합니다. CloudFront API를 사용한 삭제에 대한 자세한 내용은 Amazon CloudFront API 참조의 DeleteDistribution을 참조하세요.

S3 버킷에 OAC가 연결된 배포를 삭제해야 하는 경우 중요한 세부 사항은 S3 버킷에 OAC가 연결된 배포 삭제를 참조합니다.

> **Warning**
>
> - 배포를 삭제하려면 먼저 비활성화해야 하며, 이를 위해서는 배포 업데이트 권한이 필요합니다. 삭제된 배포는 복구할 수 없습니다.
> - 대체 도메인 이름이 연결된 배포를 비활성화하면, 다른 배포에 동일한 도메인과 일치하는 와일드카드(*)가 있는 대체 도메인 이름(예: *.example.com)이 있더라도 CloudFront가 해당 도메인 이름(예: www.example.com)에 대한 트래픽 수신을 중지합니다.
> - CloudFront 정액 요금제를 구독하는 배포는 삭제할 수 없습니다. "You can't delete this distribution while it's subscribed to a pricing plan."이라는 오류가 표시됩니다. 먼저 요금제를 취소한 다음 현재 결제 주기 후에 배포를 삭제해야 합니다.

### Multi-tenant

다중 테넌트 배포를 삭제하려면 먼저 모든 연결된 배포 테넌트를 삭제해야 합니다.

다중 테넌트 배포를 삭제하려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. CloudFront 콘솔의 오른쪽 창에서 삭제하려는 다중 테넌트 배포의 이름을 선택합니다.
3. 테넌트에서 모든 연결된 배포 테넌트를 선택하고 삭제합니다.
4. 배포를 사용 해제하려면 비활성화를 선택하고 배포 비활성화를 선택하여 확인합니다.
5. 마지막 수정 날짜 열 아래에 새 타임스탬프가 나타날 때까지 기다리십시오.
   - CloudFront가 변경 사항을 모든 엣지 로케이션에 전파하는 데 몇 분 정도 걸릴 수 있습니다.
6. 삭제, 배포 삭제를 차례로 선택합니다.

배포 테넌트를 삭제하려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. SaaS에서 배포 테넌트를 선택합니다.
3. 배포 테넌트를 검색합니다. 검색 창의 드롭다운 메뉴를 사용하여 도메인, 이름, 배포 ID, 인증서 ID, 연결 그룹 ID 또는 웹 ACL ID를 기준으로 필터링합니다.
4. 삭제할 배포 테넌트를 선택합니다.
5. 테넌트 삭제, 배포 테넌트 삭제를 선택합니다.

### Standard

표준 배포를 삭제하려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. CloudFront 콘솔의 오른쪽 창에서 삭제할 배포를 찾습니다.
   - 상태 열에 배포가 이미 비활성화됨으로 표시되면 6단계로 건너뜁니다.

- 상태가 활성화됨으로 표시되지만 배포의 마지막 수정 날짜 열에 여전히 배포 중으로 표시되는 경우 배포가 완료될 때까지 기다린 다음 3단계로 진행합니다.

3. CloudFront 콘솔의 오른쪽 창에서 삭제할 배포에 해당하는 확인란을 선택합니다.
4. 비활성화를 선택하여 배포를 비활성화하고 예, 비활성화를 선택하여 확인합니다. 그런 다음 닫기를 선택합니다.

   - 상태 열의 값이 비활성화됨으로 즉시 변경됩니다.

5. 마지막 수정 날짜 열 아래에 새 타임스탬프가 나타날 때까지 기다리십시오.

   - CloudFront가 변경 사항을 모든 엣지 로케이션에 전파하는 데 몇 분 정도 걸릴 수 있습니다.

6. 삭제할 배포에 해당하는 확인란을 선택합니다.
7. 삭제와 삭제를 차례로 선택합니다.

   - 삭제 옵션을 사용할 수 없는 경우 CloudFront가 여전히 변경 사항을 엣지 로케이션에 전파하고 있음을 의미합니다. 마지막 수정 날짜 열 아래에 새 타임스탬프가 나타날 때까지 기다린 후 6~7단계를 반복합니다.

# CloudFront 배포에 다양한 오리진 사용

배포를 만들 때 CloudFront가 파일에 대한 요청을 보내는 원본을 지정합니다. CloudFront에서 여러 원본을 사용할 수 있습니다. 예를 들어 Amazon S3 버킷, MediaStore 컨테이너, MediaPackage 채널, Application Load Balancer 또는 AWS Lambda 함수 URL을 사용할 수 있습니다. CloudFront 배포를 만들면 CloudFront가 콘텐츠 오리진 유형에 따라 대부분의 배포 설정을 자동으로 구성합니다. 자세한 내용은 사전 구성된 배포 설정 참조 섹션을 참조하세요.

프라이빗 서브넷에 Application Load Balancer, Network Load Balancer, EC2 인스턴스가 있는 경우 VPC 오리진으로 사용할 수 있습니다. VPC 오리진을 사용하면 CloudFront 배포가 있는 프라이빗 서브넷에서만 애플리케이션에 액세스할 수 있으므로 퍼블릭 인터넷에서는 애플리케이션에 액세스할 수 없습니다. 자세한 내용은 the section called “VPC 오리진으로 액세스 제한” 섹션을 참조하세요.

> **Note**
>
> 엣지 함수를 사용하여 각 요청에 적합한 오리진을 동적으로 선택할 수 있습니다. CloudFront Functions 또는 Lambda@Edge를 사용하여 뷰어의 지리적 위치, 요청 헤더 또는 쿼리 문자열 파라미터와 같은 요인에 따라 요청을 다른 오리진으로 라우팅할 수 있습니다. 자세한 내용은 함수를 사용하여 엣지에서 사용자 지정 섹션을 참조하세요.

주제


- Amazon S3 버킷 사용
- MediaStore 컨테이너 또는 MediaPackage 채널 사용
- Application Load Balancer 사용
- Network Load Balancer 사용
- Lambda 함수 URL 사용
- Amazon EC2(또는 기타 사용자 지정 오리진) 사용
- CloudFront 오리진 그룹 사용
- Amazon API Gateway 사용


# Amazon S3 버킷 사용

다음 주제에서는 Amazon S3 버킷을 CloudFront 배포의 원본으로 사용하는 여러 가지 방법을 설명합니다.

주제


- 표준 Amazon S3 버킷 사용
- Amazon S3 객체 Lambda 사용
- Amazon S3 액세스 포인트 사용
- 웹 사이트 엔드포인트로 구성된 Amazon S3 버킷 사용
- 기존 Amazon S3 버킷에 CloudFront 추가
- Amazon S3 버킷을 다른 AWS 리전으로 이동


## 표준 Amazon S3 버킷 사용

Amazon S3를 배포의 원본으로 사용하는 경우 CloudFront에서 제공하려는 모든 객체를 Amazon S3 버킷에 배치합니다. Amazon S3에서 지원되는 모든 방식을 사용하여 객체를 Amazon S3 넣을 수 있습니다. 예를 들어 Amazon S3 콘솔이나 API 또는 서드 파티 도구를 사용할 수 있습니다. 다른 표준 Amazon S3 버킷과 마찬가지로 버킷에 계층을 만들어 객체를 저장할 수 있습니다.

기존 Amazon S3 버킷을 CloudFront 오리진 서버로 사용하더라도 버킷은 어떠한 방식으로도 변경되지 않습니다. 평상시처럼 Amazon S3 객체를 저장 및 액세스할 용도로 Standard Amazon S3 가격에 버킷을 계속해서 사용할 수 있습니다. 버킷에 객체를 저장하기 위해서는 정규 Amazon S3 요금이 발생합니다. CloudFront 사용을 위한 요금에 대한 자세한 내용은 Amazon CloudFront 요금을 참조하세요.

CloudFront에서 기존 S3 버킷을 사용하는 자세한 방법은 the section called "기존 Amazon S3 버킷에 CloudFront 추가" 단원을 참조하세요.

> **Important**
>
> 버킷을 CloudFront와 함께 사용하려면 DNS 명명 요구 사항을 준수하는 이름을 사용해야 합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 이름 지정 규칙을 참조하세요.

Amazon S3 버킷을 CloudFront의 원본으로 지정하는 경우 다음 형식을 사용하는 것이 좋습니다.

`bucket-name.s3.region.amazonaws.com`

이 형식으로 버킷 이름을 지정할 경우 다음 CloudFront 기능을 사용할 수 있습니다.

- SSL/TLS를 사용하여 Amazon S3 버킷과 통신하도록 CloudFront를 구성합니다. 자세한 내용은 the section called "CloudFront에서 HTTPS 사용" 섹션을 참조하세요.
- 오리진 액세스 제어를 사용하여 뷰어에게 Amazon S3 URL이 아닌 CloudFront URL을 사용하여 콘텐츠에 액세스하도록 요구합니다. 자세한 내용은 the section called "Amazon S3 오리진에 대한 액세스 제한" 섹션을 참조하세요.
- `POST` 및 `PUT` 요청을 CloudFront에 제출하여 버킷의 콘텐츠를 업데이트합니다. 자세한 내용은 the section called "HTTP 메소드" 주제에서 the section called "CloudFront에서 요청을 처리하고 Amazon S3 오리진에 요청을 전달하는 방법" 단원을 참조하십시오.

다음 형식을 사용하여 버킷을 지정하지 마세요.

- Amazon S3 경로 형식: `s3.amazonaws.com/bucket-name`
- Amazon S3 CNAME

> **Note**
>
> CloudFront는 S3 Intelligent-Tiering을 포함한 모든 스토리지 클래스를 사용하여 S3 오리진을 지원합니다. CloudFront가 S3 오리진에서 객체를 요청하면 현재 있는 스토리지 계층에 관계없이 객체가 검색됩니다. CloudFront에서 S3 Intelligent-Tiering을 사용하더라도 배포의 성능이나 기능에는 영향을 주지 않습니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 Amazon S3 Intelligent-Tiering을 사용한 스토리지 비용 관리를 참조하세요.

## Amazon S3 객체 Lambda 사용

객체 Lambda 액세스 포인트를 생성할 때 Amazon S3는 객체 Lambda 액세스 포인트에 대한 고유한 별칭을 자동으로 생성합니다. Amazon S3 버킷 이름 대신 이 별칭을 CloudFront 배포의 오리진으로 사용할 수 있습니다.

객체 Lambda 액세스 포인트 별칭을 CloudFront의 오리진으로 사용할 때는 다음 형식을 사용하는 것이 좋습니다.

```
alias.s3.region.amazonaws.com
```

*alias*을 찾는 방법에 대한 자세한 내용은 Amazon S3 사용 설명서의 S3 버킷 객체 Lambda 액세스 포인트에 버킷 스타일 별칭을 사용하는 방법을 참조하세요.

> **Important**
>
> 객체 Lambda 액세스 포인트를 CloudFront의 오리진으로 사용하는 경우 오리진 액세스 제어를 사용해야 합니다.

사용 사례의 예는 Amazon CloudFront와 함께 Amazon S3 객체 Lambda를 사용하여 최종 사용자에게 콘텐츠 맞춤화를 참조하세요.

CloudFront는 객체 Lambda 액세스 포인트 오리진을 표준 Amazon S3 버킷 오리진과 동일하게 취급합니다.

Amazon S3 객체 Lambda를 배포용 오리진으로 사용하는 경우 다음 네 가지 권한을 구성해야 합니다.

Object Lambda Access Point

객체 Lambda 액세스 포인트에 대한 사용 권한을 추가하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 객체 Lambda 액세스 포인트를 선택합니다.
3. 사용할 객체 Lambda 액세스 포인트를 선택합니다.
4. 권한 탭을 선택합니다.
5. 객체 Lambda 액세스 포인트 정책 섹션에서 편집을 선택합니다.

6. 정책 필드에 다음 정책을 붙여 넣습니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudfront.amazonaws.com"
            },
            "Action": "s3-object-lambda:Get*",
            "Resource": "arn:aws:s3-object-lambda:us-
east-1:123456789012:accesspoint/Object-Lambda-Access-Point-name",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn":
 "arn:aws:cloudfront::123456789012:distribution/CloudFront-distribution-
ID"
                }
            }
        }
    ]
}
```

7. 변경 사항 저장을 선택합니다.

## Amazon S3 Access Point

Amazon S3 액세스 포인트에 대한 사용 권한을 추가하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 탐색 창에서 액세스 포인트를 선택합니다.
3. 사용할 Amazon S3 액세스 포인트를 선택합니다.
4. 권한 탭을 선택합니다.
5. 액세스 포인트 정책 섹션에서 편집을 선택합니다.
6. 정책 필드에 다음 정책을 붙여 넣습니다.

JSON

```
{
    "Version":"2012-10-17",
    "Id": "default",
    "Statement": [
        {
            "Sid": "s3objlambda",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudfront.amazonaws.com"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:us-east-1:123456789012:accesspoint/Access-
Point-name",
                "arn:aws:s3:us-east-1:123456789012:accesspoint/Access-
Point-name/object/*"
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": "s3-object-lambda.amazonaws.com"
                }
            }
        }
    ]
}
```

7. 저장을 선택합니다.

## Amazon S3 bucket

Amazon S3 버킷에 권한을 부여합니다.

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 버킷을 선택합니다.
3. 사용할 Amazon S3 버킷을 선택합니다.
4. 권한 탭을 선택합니다.

5. 버킷 정책 섹션에서 편집을 선택합니다.
6. 정책 필드에 다음 정책을 붙여 넣습니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:DataAccessPointAccount": "AWS-account-ID"
                }
            }
        }
    ]
}
```

7. 변경 사항 저장을 선택합니다.

## AWS Lambda function

Lambda 함수에 대한 권한 추가하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/lambda/에서 AWS Lambda 콘솔을 엽니다.
2. 탐색 창에서 함수를 선택합니다.
3. 사용할 AWS Lambda 함수를 선택합니다.
4. 구성 탭을 선택한 다음, 권한을 선택합니다.
5. 리소스 기반 정책 명령문 섹션에서 권한 추가를 선택합니다.

6. 을 선택합니다..AWS 계정
7. 명령문 ID의 이름을 입력합니다.
8. 보안 주체에 `cloudfront.amazonaws.com`을 입력합니다.
9. 작업 드롭다운 메뉴에서 `lambda:InvokeFunction`을 선택합니다.
10. 저장을 선택합니다.

## Amazon S3 액세스 포인트 사용

S3 액세스 포인트를 사용하면 Amazon S3는 고유한 별칭을 자동으로 생성합니다. Amazon S3 버킷 이름 대신 이 별칭을 CloudFront 배포의 오리진으로 사용할 수 있습니다.

객체 Amazon S3 액세스 포인트 별칭을 CloudFront의 오리진으로 사용할 때는 다음 형식을 사용하는 것이 좋습니다.

```
alias.s3.region.amazonaws.com
```

*alias*를 찾는 방법에 대한 자세한 내용은 Amazon S3 사용 설명서의 S3 버킷 액세스 포인트에 버킷 스타일 별칭을 사용을 참조합니다.

> **Important**
>
> Amazon S3 액세스 포인트를 CloudFront의 오리진으로 사용하는 경우 오리진 액세스 제어를 사용해야 합니다.

CloudFront는 Amazon S3 액세스 포인트 오리진을 표준 Amazon S3 버킷 오리진과 동일하게 취급합니다.

Amazon S3 객체 Lambda를 배포용 오리진으로 사용하는 경우 다음 두 가지 권한을 구성해야 합니다.

Amazon S3 Access Point

Amazon S3 액세스 포인트에 대한 사용 권한을 추가하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 탐색 창에서 액세스 포인트를 선택합니다.
3. 사용할 Amazon S3 액세스 포인트를 선택합니다.

4. 권한 탭을 선택합니다.
5. 액세스 포인트 정책 섹션에서 편집을 선택합니다.
6. 정책 필드에 다음 정책을 붙여 넣습니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Id": "default",
    "Statement": [
        {
            "Sid": "s3objlambda",
            "Effect": "Allow",
            "Principal": {"Service": "cloudfront.amazonaws.com"},
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:us-east-1:123456789012:accesspoint/Access-
Point-name",
                "arn:aws:s3:us-east-1:123456789012:accesspoint/Access-
Point-name/object/*"
            ],
            "Condition": {
                "StringEquals": {"aws:SourceArn":
 "arn:aws:cloudfront::123456789012:distribution/CloudFront-distribution-
ID"}
            }
        }
    ]
}
```

7. 저장을 선택합니다.

## Amazon S3 bucket

Amazon S3 버킷에 권한을 부여합니다.

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 버킷을 선택합니다.
3. 사용할 Amazon S3 버킷을 선택합니다.

4. 권한 탭을 선택합니다.
5. 버킷 정책 섹션에서 편집을 선택합니다.
6. 정책 필드에 다음 정책을 붙여 넣습니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:DataAccessPointAccount": "AWS-account-ID"
                }
            }
        }
    ]
}
```

7. 변경 사항 저장을 선택합니다.

## 웹 사이트 엔드포인트로 구성된 Amazon S3 버킷 사용

CloudFront를 사용하여 사용자 지정 원본으로서 웹 사이트 엔드포인트로 구성되는 Amazon S3 버킷을 사용할 수 있습니다. CloudFront 배포를 구성할 때 오리진의 경우 버킷에 대한 Amazon S3 정적 웹 사이트 호스팅 엔드포인트를 입력합니다. 이 값은 Amazon S3 콘솔에서 속성(Properties) 페이지의 정적 웹 사이트 호스팅(Static website hosting) 창에 나타납니다. 예:

`http://bucket-name.s3-website-region.amazonaws.com`

Amazon S3 정적 웹 사이트 엔드포인트 지정에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 웹 사이트 엔드포인트를 참조하세요.

오리진으로서 이 형식으로 된 버킷 이름을 지정할 경우 Amazon S3 리디렉션과 Amazon S3 사용자 지정 오류 문서를 사용할 수 있습니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 사용자 지정 오류 문서 구성 및 리디렉션 구성을 참조하세요. (CloudFront에서도 사용자 지정 오류 페이지를 제공합니다. 자세한 내용은 the section called "특정 HTTP 상태 코드에 대한 사용자 지정 오류 페이지 생성" 단원을 참조하세요.)

Amazon S3 버킷을 CloudFront 원본 서버로 사용해도 버킷은 변경되지 않습니다. 평소처럼 계속해서 사용할 수 있으며 정규 Amazon S3 요금이 발생합니다. CloudFront 사용을 위한 요금에 대한 자세한 내용은 Amazon CloudFront 요금을 참조하세요.

> **Note**
>
> CloudFront API를 사용하여 웹 사이트 엔드포인트로 구성되는 Amazon S3 버킷 포함 배포를 사용하는 경우 웹 사이트가 Amazon S3 버킷에서 호스팅된 경우에도 `CustomOriginConfig`를 사용하여 구성해야 합니다. CloudFront API를 사용하여 배포를 만드는 방법에 대한 자세한 내용은 Amazon CloudFront API 참조의 CreateDistribution을 참조하세요.

## 기존 Amazon S3 버킷에 CloudFront 추가

Amazon S3 버킷에 객체를 저장하는 경우 사용자에게 S3에서 객체를 직접 가져오게 할 수 있고, 아니면 S3에서 객체를 가져와 사용자들에게 배포하도록 CloudFront를 구성할 수 있습니다. 사용자가 객체에 자주 액세스하는 경우 CloudFront를 사용하는 것이 더 비용 효율적일 수 있습니다. 왜냐하면 사용량이 더 많은 경우에는 CloudFront 데이터 전송 가격이 Amazon S3 데이터 전송 가격보다 저렴하기 때문입니다. 또한 CloudFront를 함께 사용할 경우 객체가 사용자 측에 더 가깝게 저장되므로 다운로드 속도도 Amazon S3만 사용할 때보다 더 빠릅니다.

> **Note**
>
> CloudFront에서 Amazon S3 cross-origin 리소스 공유 설정을 지키도록 하려는 경우 `Origin` 헤더를 Amazon S3로 전달하도록 CloudFront를 구성합니다. 자세한 내용은 the section called "요청 헤더 기반의 콘텐츠 캐싱" 섹션을 참조하세요.

현재 Amazon S3 버킷의 도메인 이름(예: amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com) 대신 고유 도메인 이름(예: example.com)을 사용하여 Amazon S3 버킷에서 콘텐츠를 직접 배포하는 경우, 다음 절차를 이용해 중단 없이 CloudFront를 추가할 수 있습니다.

Amazon S3에서 콘텐츠를 이미 배포하고 있는 경우 CloudFront를 추가하려면

1. CloudFront 배포를 만듭니다. 자세한 내용은 the section called "배포 생성" 섹션을 참조하세요.

   배포를 생성할 때 Amazon S3 버킷의 이름을 오리진 서버로 지정합니다.

   > **Important**
   >
   > 버킷을 CloudFront와 함께 사용하려면 DNS 명명 요구 사항을 준수하는 이름을 사용해야 합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 이름 지정 규칙을 참조하세요.

   Amazon S3에서 CNAME을 사용하려는 경우 배포에 대한 CNAME도 지정합니다.

2. Amazon S3 버킷 안에 공개 읽기 가능 객체의 링크를 포함하는 테스트 웹 페이지를 생성하고 해당 링크를 테스트합니다. 이 초기 테스트에서는 객체 URL에서 배포의 CloudFront 도메인 이름을 사용합니다(예: `https://d111111abcdef8.cloudfront.net/images/image.jpg`).

   CloudFront URL의 형식에 대한 자세한 내용은 the section called "파일 URL 사용자 지정" 단원을 참조하세요.

3. Amazon S3 CNAME을 사용할 경우 애플리케이션은 Amazon S3 버킷에서 객체를 참조할 때 버킷 이름(예: amzn-s3-demo-bucket.s3.amazonaws.com) 대신에 사용자의 도메인 이름(예: example.com)을 사용합니다. 객체를 참조할 때 배포의 CloudFront 도메인 이름(예: d111111abcdef8.cloudfront.net) 대신에 고유 도메인 이름을 계속 사용하려면 설정을 해당 DNS 서비스 공급자로 업데이트해야 합니다.

   Amazon S3 CNAME이 작동하려면 DNS 서비스 공급자에게 고유한 도메인에 대한 CNAME 리소스 레코드 세트가 반드시 있어야 합니다. 이 리소스 레코드 세트는 현재 도메인에 대한 쿼리를 Amazon S3 버킷으로 라우팅합니다. 예를 들어 사용자가 이 객체를 요청하는 경우:

   ```
   https://example.com/images/image.jpg
   ```

   요청이 자동으로 재라우팅되어 사용자에게 표시되는 객체는 다음과 같습니다.

   ```
   https://amzn-s3-demo-bucket.s3.amazonaws.com/images/image.jpg
   ```

쿼리를 Amazon S3 버킷 대신에 CloudFront 배포로 라우팅하려면 DNS 서비스 공급자가 제공하는 방법을 사용하여 도메인의 CNAME 리소스 레코드 세트를 업데이트해야 합니다. 이 업데이트된 CNAME 레코드가 DNS 쿼리를 고유 도메인에서 배포의 CloudFront 도메인 이름으로 리디렉션합니다. 자세한 내용은 DNS 서비스 공급자가 제공하는 설명서를 참조하십시오.

> **Note**
>
> Route 53을 DNS 서비스로 사용하는 경우 CNAME 리소스 레코드 세트 또는 별칭 리소스 레코드 세트를 사용할 수 있습니다. 리소스 레코드 세트 편집에 대한 자세한 내용은 레코드 편집을 참조하세요. 별칭 리소스 레코드 세트에 대한 자세한 내용은 별칭 및 비-별칭 레코드 간의 선택을 참조하세요. 두 주제는 모두 Amazon Route 53 개발자 안내서에 나와 있습니다.

CloudFront에 CNAME 사용에 대한 자세한 내용은 the section called “사용자 지정 URL 사용” 단원을 참조하세요.

일반적으로 더 빠를 수도 있지만 CNAME 리소스 레코드 세트를 업데이트한 후 DNS 시스템 전체에 변경 사항이 전파되는 데는 최대 72시간이 걸릴 수 있습니다. 이 시간 동안은 일부 콘텐츠 요청이 Amazon S3 버킷으로 계속 라우팅되며 그 밖의 요청은 CloudFront로 라우팅됩니다.

## Amazon S3 버킷을 다른 AWS 리전으로 이동

Amazon S3를 CloudFront 배포에 대한 원본으로 사용하고 있고 버킷을 다른 AWS 리전으로 옮기는 경우, 다음 두 조건이 참일 때 CloudFront가 그 레코드를 업데이트하여 새 리전을 사용하는 데 최대 1시간이 걸릴 수 있습니다.

- CloudFront 원본 액세스 ID(OAI)를 사용하여 버킷에 대한 액세스를 제한하고 있습니다.
- 인증을 위해 서명 버전 4가 필요한 Amazon S3 리전으로 버킷을 옮깁니다.

OAI를 사용할 때 CloudFront는 그 리전(다른 값들 중에서)을 사용해 버킷에서 객체를 요청할 때 사용하는 서명을 계산합니다. OAI에 대한 자세한 내용은 the section called “오리진 액세스 ID 사용(레거시, 권장하지 않음)” 단원을 참조하십시오. 서명 버전 2를 지원하는 AWS 리전의 목록은 Amazon Web Services 일반 참조의 서명 버전 2 서명 과정을 참조하세요.

CloudFront의 레코드에 대한 더 빠른 업데이트를 강제하려면, CloudFront 콘솔에서 일반(General) 탭의 설명(Comment) 필드를 업데이트하는 등의 작업을 통해 CloudFront 배포를 업데이트할 수 있습니

다. 배포를 업데이트하는 경우 CloudFront는 버킷이 있는 리전을 즉시 확인합니다. 모든 엣지 로케이션에 변경 사항을 전파하는 데 몇 분 밖에 걸리지 않습니다.

## MediaStore 컨테이너 또는 MediaPackage 채널 사용

CloudFront를 사용하여 비디오를 스트리밍하려면 MediaStore 컨테이너로 구성된 Amazon S3 버킷을 설정하거나 MediaPackage로 채널 및 엔드포인트를 생성할 수 있습니다. 그런 다음 CloudFront에서 배포를 만들고 구성하여 비디오를 스트리밍합니다.

자세한 내용 및 단계별 지침은 다음 단원을 참조하십시오.

- the section called “AWS Elemental MediaStore를 오리진으로 사용하여 비디오를 제공합니다.”
- the section called “AWS Elemental MediaPackage를 사용하여 포맷된 라이브 비디오 제공”

## Application Load Balancer 사용

CloudFront를 사용하여 트래픽을 내부 및 인터넷 연결 Application Load Balancer로 라우팅할 수 있습니다.

오리진이 하나 이상의 Amazon EC2 인스턴스에서 호스팅되는 하나 이상의 HTTP(S) 서버(웹 서버)인 경우 인터넷 연결 Application Load Balancer를 사용하여 인스턴스에 트래픽을 배포할 수 있습니다. 인터넷 경계 로드 밸런서는 공개적으로 확인 가능한 DNS 이름이 있으며 인터넷을 통해 클라이언트의 요청을 대상으로 라우팅합니다.

뷰어가 로드 밸런서에 직접 액세스하지 않고 CloudFront를 통해서만 웹 서버에 액세스할 수 있도록 하는 방법을 비롯하여 인터넷 연결 Application Load Balancer를 CloudFront의 오리진으로 사용하는 방법에 대한 자세한 내용은 the section called “Application Load Balancer에 대한 액세스 제한” 섹션을 참조하세요.

또는 VPC 오리진을 사용하여 가상 프라이빗 클라우드(VPC) 프라이빗 서브넷의 내부 Application Load Balancer를 통해 호스팅되는 애플리케이션에서 콘텐츠를 제공할 수 있습니다. VPC 오리진은 퍼블릭 인터넷에서 애플리케이션에 액세스할 수 없도록 합니다. 자세한 내용은 VPC 오리진으로 액세스 제한 섹션을 참조하세요.

## Network Load Balancer 사용

Amazon CloudFront에서는 내부 및 인터넷 연결 Network Load Balancer를 모두 사용할 수 있습니다. VPC 오리진을 사용하여 CloudFront의 프라이빗 서브넷 내에서 내부 Network Load Balancer를 사용할

수 있습니다. CloudFront VPC 오리진을 사용하면 퍼블릭 인터넷에 노출시키지 않고 프라이빗 VPC 서브넷에 호스팅된 애플리케이션에서 콘텐츠를 제공할 수 있습니다. 자세한 내용은 VPC 오리진으로 액세스 제한 섹션을 참조하세요.

또는 CloudFront를 사용하여 인터넷 연결 Network Load Balancer에서 트래픽을 전송할 수도 있습니다. 인터넷 연결 로드 밸런서는 공개적으로 확인 가능한 DNS 이름을 가지며 인터넷의 클라이언트와 CloudFront 배포 모두에서 요청을 수신할 수 있습니다.

## Lambda 함수 URL 사용

Lambda 함수 URL은 Lambda 함수를 위한 전용 HTTPS 엔드포인트입니다. Lambda 함수 URL을 사용하여 서버리스 웹 애플리케이션을 Lambda 내에서 완전히 구축할 수 있습니다. API 게이트웨이 또는 Application Load Balancer와 통합할 필요 없이 함수 URL을 통해 Lambda 웹 애플리케이션을 직접 호출할 수 있습니다.

함수 URL이 포함된 Lambda 함수를 사용하여 서버리스 웹 애플리케이션을 구축하는 경우 CloudFront를 추가하여 다음과 같은 이점을 얻을 수 있습니다.

- 콘텐츠를 뷰어에 더 가깝게 캐시하여 애플리케이션 속도 향상
- 웹 애플리케이션에 대한 사용자 지정 도메인 이름 사용
- CloudFront 캐시 동작을 사용하여 서로 다른 URL 경로를 서로 다른 Lambda 함수로 라우팅
- CloudFront 지리적 제한 또는 AWS WAF(또는 둘 다)를 사용하여 특정 요청 차단
- CloudFront에 AWS WAF를 사용하면 악성 봇으로부터 애플리케이션을 보호하고, 일반적인 애플리케이션 악용을 방지하며, DDoS 공격으로부터 보호를 강화할 수 있습니다.

Lambda 함수 URL을 CloudFront 배포의 원본으로 사용하려면 Lambda 함수 URL의 전체 도메인 이름을 원본 도메인으로 지정합니다. Lambda 함수 URL 도메인 이름은 다음 형식을 사용합니다.

```
function-URL-ID.lambda-url.AWS-Region.on.aws
```

CloudFront 배포의 오리진으로 Lambda 함수 URL을 사용하는 경우 함수 URL에 공개적으로 액세스할 수 있어야 합니다. 이렇게 하려면 다음 방법 중 하나를 사용하세요.

- 오리진 액세스 제어(OAC)를 사용하는 경우 Lambda 함수 URL의 `AuthType` 파라미터는 `AWS_IAM` 값을 사용하고 리소스 기반 정책에서 `lambda:InvokeFunctionUrl` 및 `lambda:InvokeFunction` 권한을 허용해야 합니다. OAC에 Lambda 함수 URL 사용에 대한 자세한 내용은 AWS Lambda 함수 URL 오리진에 대한 액세스 제한 섹션을 참조하세요.

- OAC를 사용하지 않는 경우 함수 URL의 `AuthType` 파라미터를 `NONE`으로 설정하고 리소스 기반 정책에서 `lambda:InvokeFunctionUrl` 권한을 허용합니다.

또한 CloudFront가 오리진으로 보내는 요청에 사용자 지정 오리진 헤더를 추가하고 헤더가 요청에 없는 경우 오류 응답을 반환하는 함수 코드를 작성할 수도 있습니다. 이렇게 하면 Lambda 함수 URL을 직접 사용하지 않고 CloudFront를 통해서만 웹 애플리케이션에 액세스할 수 있습니다.

Lambda 함수 URL에 대한 자세한 내용은 AWS Lambda 개발자 가이드에서 다음 주제를 참조하세요.

- Lambda 함수 URL - Lambda 함수 URL 기능에 대한 일반적인 개요
- Lambda 함수 URL 호출 - 서버리스 웹 애플리케이션 코딩에 사용할 요청 및 응답 페이로드에 대한 세부 정보 포함
- Lambda 함수 URL에 대한 보안 및 인증 모델 – Lambda 인증 유형에 대한 세부 정보 포함

## Amazon EC2(또는 기타 사용자 지정 오리진) 사용

Amazon CloudFront에서는 내부 및 인터넷 연결 EC2 인스턴스를 모두 사용할 수 있습니다. VPC 오리진을 사용하여 CloudFront에서 프라이빗 서브넷 내의 내부 EC2 인스턴스를 사용할 수 있습니다. CloudFront VPC 오리진을 사용하면 퍼블릭 인터넷에 노출시키지 않고 프라이빗 VPC 서브넷에 호스팅된 애플리케이션에서 콘텐츠를 제공할 수 있습니다. 자세한 내용은 VPC 오리진으로 액세스 제한 섹션을 참조하세요.

사용자 지정 오리진은 공개적으로 확인 가능한 DNS 이름을 가진 HTTP(S) 웹 서버로서 클라이언트의 요청을 인터넷을 통해 대상으로 라우팅합니다. HTTP(S) 서버는 AWS(예: Amazon EC2 인스턴스)에 호스팅하거나 다른 곳에서 호스팅할 수 있습니다. 웹 사이트 엔드포인트로서 구성된 Amazon S3 오리진은 또한 사용자 지정 오리진으로 간주됩니다. 자세한 내용은 the section called “웹 사이트 엔드포인트로 구성된 Amazon S3 버킷 사용” 섹션을 참조하세요.

고유 HTTP 서버를 사용자 지정 원본으로 사용하는 경우 서버의 DNS 이름, HTTP 및 HTTPS 포트, 원본에서 객체를 가져올 때 CloudFront에서 사용하려는 프로토콜을 지정합니다.

사용자 지정 원본을 사용할 경우 비공개 콘텐츠를 제외하고 대부분의 CloudFront 기능이 지원됩니다. 서명된 URL을 사용하여 사용자 지정 원본으로부터 콘텐츠를 배포할 수는 있지만, CloudFront에서 사용자 지정 원본을 액세스하려면 원본이 공개적으로 액세스 가능한 상태여야 합니다. 자세한 내용은 the section called “서명된 URL과 서명된 쿠키를 사용하여 콘텐츠 제한” 섹션을 참조하세요.

CloudFront를 통한 Amazon EC2 인스턴스 및 기타 사용자 지정 오리진 사용에 대한 이러한 지침을 따르세요.

- 동일한 CloudFront 오리진에 대한 콘텐츠를 서비스하는 모든 서버의 동일한 콘텐츠를 호스팅 및 서비스합니다. 자세한 내용은 the section called "오리진 설정" 주제에서 the section called "모든 배포 설정" 단원을 참조하십시오.
- 지원 또는 CloudFront에서 이 값을 디버깅에 사용해야 할 경우 모든 서버에 `X-Amz-Cf-Id` 헤더 항목을 로그합니다.
- 사용자 지정 원본이 수신 대기하는 HTTP 및 HTTPS 포트에 대한 요청을 제한합니다.
- 구현 시 모든 서버의 클록을 동기화합니다. CloudFront는 로그 및 보고용으로 서명된 URL 및 서명된 쿠키에 대해 UTC(협정 세계시)를 사용합니다. 또한 CloudWatch 지표를 사용하여 CloudFront 활동을 모니터링하는 경우 CloudWatch에서도 UTC를 사용한다는 점에 유의하세요.
- 이중화 서버를 사용하여 오류를 처리합니다.
- 프라이빗 콘텐츠를 제공하기 위한 사용자 지정 오리진 사용에 대한 자세한 내용은 the section called "사용자 지정 오리진의 파일에 대한 액세스 제한"을 참조하십시오.
- 요청 및 응답 동작과 지원되는 HTTP 상태 코드에 대한 자세한 내용은 요청 및 응답 동작 단원을 참조하십시오.

Amazon EC2를 사용자 지정 원본으로 사용하는 경우 다음을 수행하는 것이 좋습니다.

- 웹 서버용 소프트웨어를 자동으로 설치하는 Amazon Machine Image를 사용합니다. 자세한 내용은 Amazon EC2 설명서를 참조하세요.
- Elastic Load Balancing 로드 밸런서를 사용하여 여러 Amazon EC2 인스턴스 간 트래픽을 처리하고 Amazon EC2 인스턴스에 대한 변경으로부터 애플리케이션을 격리할 수 있습니다. 예를 들어, 로드 밸런서를 사용하는 경우 애플리케이션을 변경하지 않고도 Amazon EC2 인스턴스를 추가 및 삭제할 수 있습니다. 자세한 내용은 Elastic Load Balancing 설명서를 참조하세요.
- CloudFront 배포를 생성할 때 오리진 서버의 도메인 이름으로 로드 밸런서의 URL을 지정합니다. 자세한 내용은 the section called "배포 생성" 섹션을 참조하세요.

## CloudFront 오리진 그룹 사용

예를 들어 고가용성이 필요할 때 시나리오에 대한 오리진 장애 조치를 구성하려면 CloudFront 오리진에 대한 오리진 그룹을 지정할 수 있습니다. 오리진 장애 조치를 사용하여 CloudFront에 대한 기본 오리진과, 기본 오리진이 특정 HTTP 상태 코드 실패 응답을 반환할 때 CloudFront가 자동으로 전환하는 두 번째 오리진을 지정합니다.

원본 그룹을 설정하는 단계를 비롯한 자세한 내용은 the section called "오리진 장애 조치를 통한 고가용성 향상" 단원을 참조하세요.

## Amazon API Gateway 사용

API Gateway를 CloudFront 배포의 사용자 지정 오리진으로 사용할 수 있습니다. 자세한 내용은 다음 항목을 참조하세요.

- Securing Amazon API Gateway with secure ciphers using Amazon CloudFront AWS 블로그 게시물
- How do I set up API Gateway with my own CloudFront distribution? AWS re:Post

# CloudFront 배포에 IPv6 활성화

Amazon CloudFront는 클라이언트에서 AWS 엣지 로케이션까지 IPv4와 IPv6를 모두 지원합니다. CloudFront는 오리진에 대한 IPv6 및 듀얼 스택(IPv4 및 IPv6) 연결도 지원합니다. 이를 통해 엔드 투 엔드 IPv6 전송을 달성할 수 있습니다.

IPv6는 IPv4를 대체하도록 설계된 차세대 인터넷 프로토콜입니다. IPv4는 32비트 주소(예: 192.0.2.44)를 사용하지만 IPv6는 128비트 주소(예: 2001:0db8:85a3::8a2e:0370:7334)를 사용합니다. IPv6는 더 많은 인터넷 연결 디바이스를 수용할 수 있도록 확장된 주소 공간을 제공합니다.

주제

- IPv6 뷰어 요청
- IPv6 오리진 요청

## IPv6 뷰어 요청

일반적으로, 콘텐츠를 액세스하려는 사용자 중에 IPv6 네트워크의 사용자가 있을 경우 IPv6를 사용하도록 설정해야 합니다. 하지만 서명된 URL이나 서명된 쿠키를 사용하여 콘텐츠에 대한 액세스를 제한하며, `IpAddress` 파라미터를 포함하는 사용자 지정 정책을 사용하여, 콘텐츠에 액세스할 수 있는 IP 주소를 제한하려는 경우에는 IPv6를 사용하도록 설정하지 마세요. 일부 콘텐츠에 대한 액세스를 IP 주소에 따라 제한하고, 다른 콘텐츠에 대한 액세스는 제한하지 않으려면(또는 액세스를 제한하되 IP 주소로 제한하지 않으려면), 두 개의 배포를 만들면 됩니다. 사용자 지정 정책을 사용하여 서명된 URL을 만드는 방법에 대한 자세한 내용은 사용자 지정 정책을 사용하여 서명된 URL 생성을 참조합니다. 사용자 지정 정책을 사용하여 서명된 쿠키를 만드는 방법에 대한 자세한 내용은 사용자 지정 정책을 사용하여 서명된 쿠키 설정을 참조합니다.

Route 53 별칭 리소스 레코드 세트를 사용하여 CloudFront 배포로 트래픽을 라우팅할 경우, 다음 조건이 모두 해당되면 두 번째 별칭 리소스 레코드 세트를 만들어야 합니다.

- 배포에 대해 IPv6를 사용하도록 설정함
- 객체의 URL에 대체 도메인 이름을 사용함

자세한 내용은 Amazon Route 53 개발자 안내서에서 도메인 이름을 사용하여 Amazon CloudFront 배포로 트래픽 라우팅을 참조하세요.

Route 53 또는 다른 DNS 서비스를 사용하여 CNAME 리소스 레코드 세트를 만들었을 경우 아무것도 변경할 필요가 없습니다. CNAME 레코드는 최종 사용자 요청의 IP 주소 형식과 상관없이 배포로 트래픽을 라우팅합니다.

IPv6와 CloudFront 액세스 로그를 사용하도록 설정하면 `c-ip` 열에 IPv4 및 IPv6 형식의 값이 포함됩니다. 자세한 내용은 로그 파일 필드 섹션을 참조하세요.

> **Note**
>
> IPv4가 더 우수한 사용자 경험을 제공할 것으로 추천될 경우 CloudFront는 고객에 대해 가용성을 높게 유지하기 위해 IPv4를 사용하여 최종 사용자 요청에 응답합니다. CloudFront가 IPv6를 통해 처리하는 요청의 비율을 확인하려면 배포에 대해 CloudFront 로그 기록을 활성화하여 `c-ip` 열을 구문 분석하세요. 이 열에는 요청을 한 최종 사용자의 IP 주소가 들어 있습니다. 이 비율은 갈수록 커지겠지만 아직 전 세계 모든 최종 사용자 네트워크에서 IPv6가 지원되지는 않으므로 트래픽에서 상대적으로 적은 부분을 차지할 것입니다. 일부 최종 사용자 네트워크는 IPv6를 완벽하게 지원하지만, IPv6를 전혀 지원하지 않는 최종 사용자 네트워크도 있습니다. 최종 사용자 네트워크는 가정용 인터넷이나 무선 통신업체와 유사합니다.
> IPv6에 대한 지원을 자세히 알아보려면 CloudFront FAQ를 참조하세요. 액세스 로그 활성화에 대한 자세한 내용은 표준 로깅 및 로그 접두사 필드를 참조하세요.

## IPv6 오리진 요청

사용자 지정 오리진(Amazon S3 및 VPC 오리진 제외)을 사용하는 경우 배포의 오리진 설정을 사용자 지정하여 CloudFront가 IPv4 또는 IPv6 주소를 사용하여 오리진에 연결하는 방법을 선택할 수 있습니다. 사용자 지정 오리진(Amazon S3 및 VPC 오리진 제외)의 경우 다음과 같은 연결 옵션이 있습니다.

- IPv4 전용(기본값) - CloudFront가 IPv4를 통해 오리진에 연결하는 데 사용하는 기본 구성입니다.
- IPv6 전용 - 오리진 도메인이 IPv6 주소로 확인되어야 합니다. CloudFront는 오리진 연결에 IPv6 주소만 사용합니다.

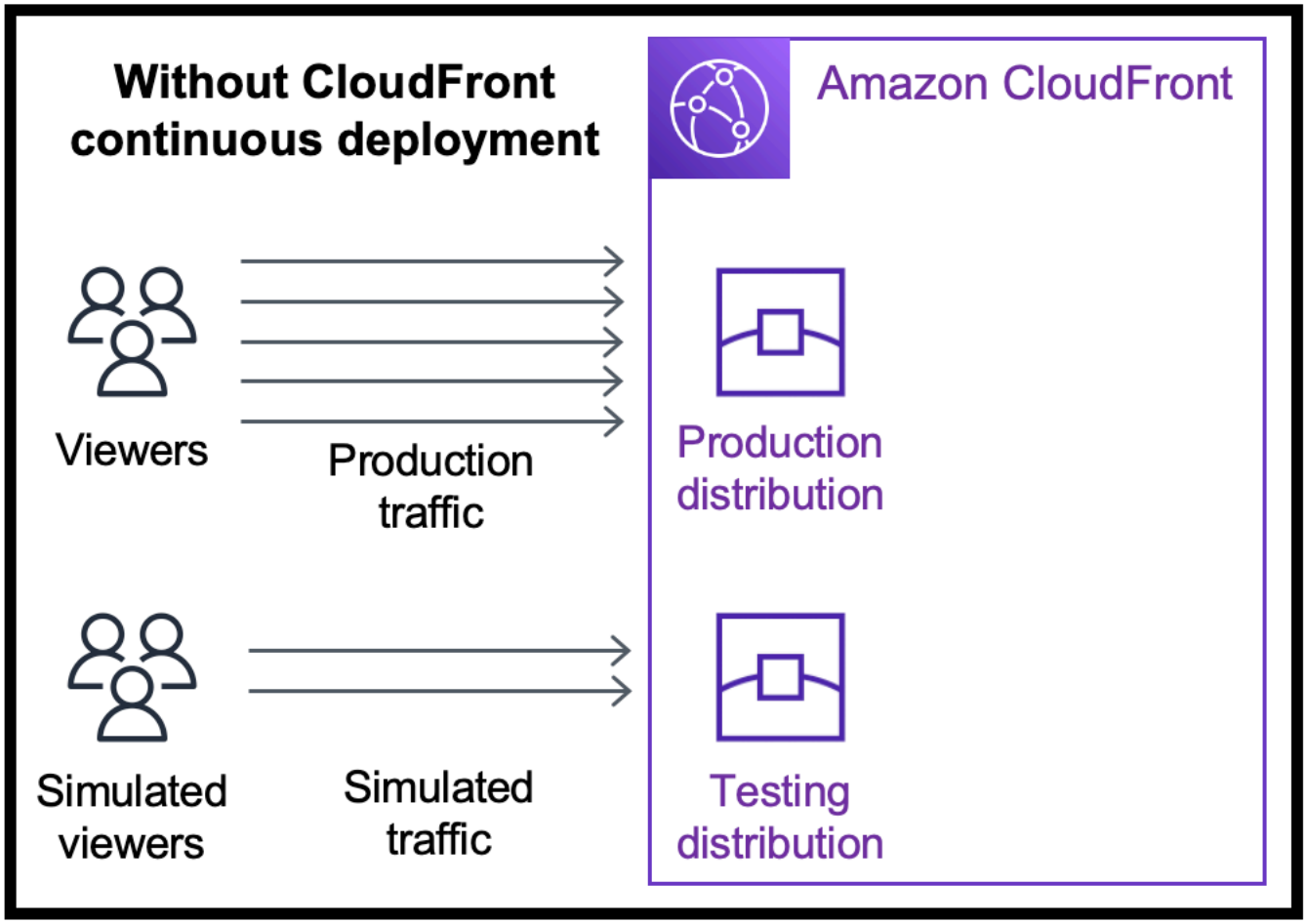
- 듀얼 스택 - IPv4 및 IPv6를 통한 연결을 활성화합니다. CloudFront는 자동으로 IPv4 또는 IPv6 오리진 연결을 선택하여 성능과 가용성을 우선시하므로 CloudFront를 웹 애플리케이션의 IPv6 및 IPv4 듀얼 스택 인터넷 게이트웨이로 사용할 수 있습니다.

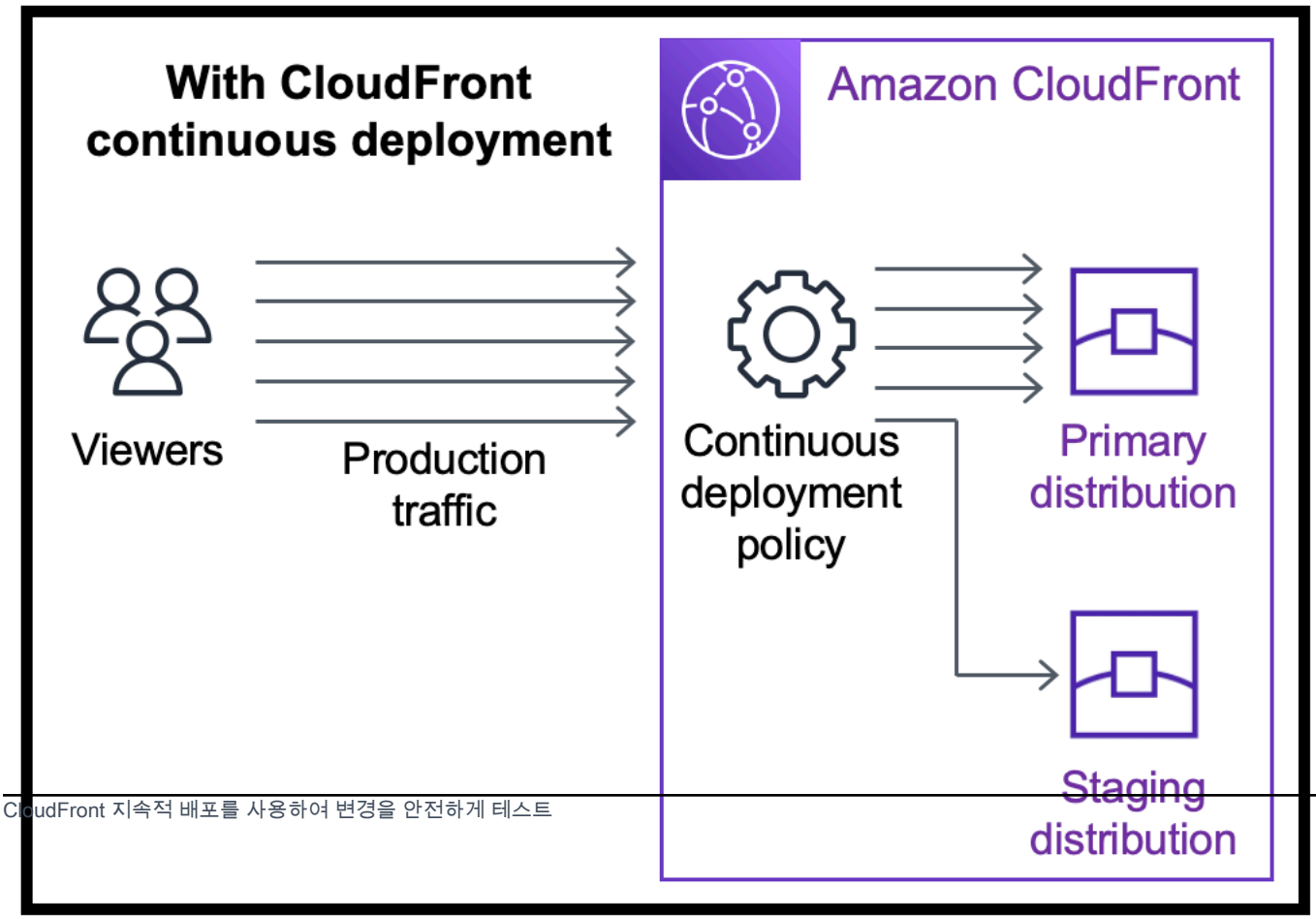
오리진의 네트워크 구성 및 연결 요구 사항에 맞는 옵션을 선택합니다. 자세한 내용은 IPv6용 DNS 설계 및 IPv6 보안 및 모니터링 고려 사항을 참조하세요.

# CloudFront 지속적 배포를 사용하여 CDN 구성 변경을 안전하게 테스트

Amazon CloudFront 지속적 배포를 사용하면 프로덕션 트래픽의 하위 집합으로 먼저 테스트하여 CDN 구성의 변경 사항을 안전하게 배포할 수 있습니다. 스테이징 배포와 지속적 배포 정책을 사용하여 실제 (프로덕션) 최종 사용자의 일부 트래픽을 새 CDN 구성으로 전송하고 예상대로 작동하는지 확인할 수 있습니다. 새 구성의 성능을 실시간으로 모니터링하고 준비가 되면 기본 배포를 통해 모든 트래픽을 처리하도록 새 구성을 승격할 수 있습니다.

다음 다이어그램은 CloudFront 지속적 배포를 사용할 경우의 이점을 보여 줍니다. 이 기능이 없으면 시뮬레이션된 트래픽으로 CDN 구성 변경을 테스트해야 합니다. 지속적 배포를 사용하면 프로덕션 트래픽의 하위 집합을 사용하여 변경 사항을 테스트한 다음 준비가 되면 기본 배포로 변경 사항을 승격할 수 있습니다.


Without CloudFront continuous deployment
Amazon CloudFront
Viewers
Production traffic
Production distribution
Simulated viewers
Simulated traffic
Testing distribution


With CloudFront continuous deployment
Amazon CloudFront
Viewers
Production traffic
Continuous deployment policy
Primary distribution
Staging distribution

다음 주제에서 지속적 배포 작업에 대해 자세히 알아봅니다.

주제


- CloudFront 지속적 배포 워크플로
- 스테이징 배포 및 지속적 배포 정책 사용
- 스테이징 배포 모니터링
- 지속적 배포의 작동 방식 알아보기
- 지속적 배포를 위한 할당량 및 기타 고려 사항


## CloudFront 지속적 배포 워크플로

다음 고급 워크플로는 CloudFront 지속적 배포를 통해 구성 변경을 안전하게 테스트하고 배포하는 방법을 설명합니다.

1. 기본 배포로 사용할 배포를 선택합니다. 기본 배포는 현재 프로덕션 트래픽을 제공하는 배포입니다.
2. 기본 배포에서 스테이징 배포를 생성합니다. 스테이징 배포는 기본 배포의 복사본으로 시작됩니다.
3. 지속적 배포 정책 내에서 트래픽 구성을 생성하고 이를 기본 배포에 연결합니다. 이에 따라 CloudFront가 스테이징 배포로 트래픽을 라우팅하는 방식이 결정됩니다. 요청을 스테이징 배포로 라우팅하는 방법에 대한 자세한 내용은 the section called “요청을 스테이징 배포로 라우팅”을 참조하십시오.
4. 스테이징 배포의 구성을 업데이트합니다. 업데이트할 수 있는 설정에 대한 자세한 내용은 the section called “기본 및 스테이징 배포 업데이트”를 참조하십시오.
5. 스테이징 배포를 모니터링하여 구성 변경이 예상대로 수행되는지 확인합니다. 스테이징 배포를 모니터링하는 방법에 대한 자세한 내용은 the section called “스테이징 배포 모니터링”을 참조하십시오.

   스테이징 배포를 모니터링하면서 다음을 수행할 수 있습니다.

   - 스테이징 배포의 구성을 다시 업데이트하여 구성 변경을 계속 테스트합니다.
   - 연속 배포 정책(트래픽 구성)을 업데이트하여 스테이징 배포에 더 많은 트래픽을 보내거나 더 적게 보냅니다.
6. 스테이징 배포의 성능이 만족스러우면 스테이징 배포의 구성을 기본 배포로 승격합니다. 그러면 스테이징 배포의 구성이 기본 배포에 복사됩니다. 이렇게 하면 지속적 배포 정책도 비활성화되므로 CloudFront는 모든 트래픽을 기본 배포로 라우팅합니다.

스테이징 배포의 성능을 모니터링(5단계)하고 특정 기준이 충족되면 구성을 자동으로 승격(6단계)하는 자동화를 구축할 수 있습니다.

구성을 승격시킨 후 다음에 구성 변경을 테스트할 때 동일한 스테이징 배포를 재사용할 수 있습니다.

CloudFront 콘솔, AWS CLI 또는 CloudFront API에서 스테이징 배포 및 지속적 배포 정책을 사용하는 방법에 대한 자세한 내용은 다음 섹션을 참조하십시오.

# 스테이징 배포 및 지속적 배포 정책 사용

AWS Command Line Interface(AWS CLI) 또는 CloudFront API를 사용하여 CloudFront 콘솔에서 스테이징 배포와 지속적 배포 정책을 생성, 업데이트 및 수정할 수 있습니다.

## 스테이징 배포 및 지속적 배포 정책 생성

다음 절차에서는 스테이징 배포 및 지속적 배포 정책을 생성하는 방법을 보여 줍니다.

Console

AWS Management Console을 사용하여 스테이징 배포 및 지속적 배포 정책을 생성할 수 있습니다.

스테이징 배포 및 연속 배포 정책 생성(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 기본 배포로 사용할 배포를 선택합니다. 기본 배포는 현재 프로덕션 트래픽을 제공하는 배포이며, 여기서 스테이징 배포를 만들게 됩니다.
4. Continuous deployment(지속적 배포) 섹션에서 Create staging distribution(스테이징 배포 생성)을 선택합니다. 그러면 Create staging distribution(스테이징 배포 생성) 마법사가 열립니다.
5. Create staging distribution(스테이징 배포 생성) 마법사에서 다음을 수행합니다.

   a. (선택 사항) 스테이징 배포에 대한 설명을 입력합니다.

   b. 다음을 선택합니다.

   c. 스테이징 배포의 구성을 수정합니다. 업데이트할 수 있는 설정에 대한 자세한 내용은 the section called “기본 및 스테이징 배포 업데이트”를 참조하십시오.

      스테이징 배포의 구성 수정을 마치면 Next(다음)을 선택합니다.

d. 콘솔을 사용하여 Traffic configuration(트래픽 구성)을 지정합니다. 이에 따라 CloudFront가 스테이징 배포로 트래픽을 라우팅하는 방식이 결정됩니다. (CloudFront는 트래픽 구성을 지속적 배포 정책에 저장합니다.)

   트래픽 구성 옵션에 대한 자세한 내용은 the section called "요청을 스테이징 배포로 라우팅"을 참조하세요.

   Traffic configuration(트래픽 구성)을 완료했으면 Next(다음)을 선택합니다.

e. 트래픽 구성을 포함하여 스테이징 배포에 대한 구성을 검토한 다음 Create staging distribution(스테이징 배포 생성)을 선택합니다.

CloudFront 콘솔에서 Create staging distribution(스테이징 배포 생성) 마법사를 완료하면 CloudFront는 다음을 수행합니다.

- 지정한 설정을 사용하여 스테이징 배포를 만듭니다(5c 단계).
- 지정한 트래픽 구성을 사용하여 지속적 배포 정책을 생성합니다(5d단계).
- 스테이징 배포를 만들 때 바탕이 된 기본 배포에 지속적 배포 정책을 연결합니다.

기본 배포의 구성이 연결된 지속적 배포 정책을 사용하여 엣지 로케이션에 배포되면 CloudFront는 트래픽 구성을 기반으로 지정된 트래픽 부분을 스테이징 배포로 전송하기 시작합니다.

CLI

AWS CLI에서 스테이징 배포와 지속적 배포 정책을 생성하려면 다음 절차를 사용합니다.

스테이징 배포 생성(CLI)

1. aws cloudfront get-distribution 및 grep 명령을 함께 사용하면 기본 배포로 사용할 배포의 ETag 값을 얻을 수 있습니다. 기본 배포는 현재 프로덕션 트래픽을 제공하는 배포이며, 여기서 스테이징 배포를 만들게 됩니다.

   다음 명령은 예시를 나타냅니다. 다음 예에서는 *primary_distribution_ID*를 기본 배포의 ID로 바꿉니다.

   ```
   aws cloudfront get-distribution --id primary_distribution_ID | grep 'ETag'
   ```

   다음 단계에 필요하므로 ETag 값을 복사하십시오.

2. aws cloudfront copy-distribution 명령을 사용하여 스테이징 배포를 만들 수 있습니다. 다음 예제 명령에서는 가독성을 높이기 위해 이스케이프 문자(\)와 줄 바꿈을 사용하지만 실제 명령에서는 이러한 문자를 생략해야 합니다. 이 예에서 다음과 같이 합니다.

   - *primary_distribution_ID*를 기본 배포의 ID로 바꿉니다.
   - *primary_distribution_ETag*를 기본 배포의 ETag 값(이전 단계에서 가져온 값)으로 바꿉니다.
   - (선택 사항) *CLI_example*을 원하는 호출자 참조 ID로 바꿉니다.

```
aws cloudfront copy-distribution --primary-distribution-
id primary_distribution_ID \
                                 --if-match primary_distribution_ETag \
                                 --staging \
                                 --caller-reference 'CLI_example'
```

   명령의 출력에는 스테이징 배포와 해당 구성에 대한 정보가 표시됩니다. 스테이징 배포의 CloudFront 도메인 이름은 다음 단계에 필요하므로 복사합니다.

지속적 배포 정책 생성(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 `continuous-deployment-policy.yaml` 명령에 대한 모든 입력 파라미터가 포함된 create-continuous-deployment-policy이라는 파일을 만듭니다. 다음 명령에서는 가독성을 높이기 위해 이스케이프 문자(\)와 줄 바꿈을 사용하지만 실제 명령에서는 이러한 문자를 생략해야 합니다.

```
aws cloudfront create-continuous-deployment-policy --generate-cli-skeleton yaml-
input \
                                                  > continuous-deployment-
policy.yaml
```

2. 방금 생성한 `continuous-deployment-policy.yaml`이라는 파일을 엽니다. 파일을 편집하여 원하는 지속적 배포 정책 설정을 지정한 다음 파일을 저장합니다. 파일을 편집할 때:

   - `StagingDistributionDnsNames` 섹션:
     - `Quantity`의 값을 `1`로 변경합니다.

    - `Items`의 경우 스테이징 배포의 CloudFront 도메인 이름(이전 단계에서 저장한 이름)을 붙여 넣습니다.
  - `TrafficConfig` 섹션:
    - `Type`을 `SingleWeight` 또는 `SingleHeader`로 선택합니다.
    - 다른 유형에 대한 설정을 제거합니다. 예를 들어, 가중치 기반 트래픽 구성을 원하는 경우 `Type`을 `SingleWeight`로 설정한 다음 `SingleHeaderConfig` 설정을 제거합니다.
    - 가중치 기반 트래픽 구성을 사용하려면 `Weight`의 값을 `.01`(1%) 에서 `.15`(15%) 사이의 10진수로 설정합니다.

    `TrafficConfig`의 이러한 옵션에 대한 자세한 내용은 the section called "요청을 스테이징 배포로 라우팅" 및 the section called "가중치 기반 구성을 위한 세션 고정성"을 참조하세요.

3. 다음 명령을 사용하여 `continuous-deployment-policy.yaml` 파일의 입력 파라미터로 지속적 배포 정책을 만듭니다.

```
aws cloudfront create-continuous-deployment-policy --cli-input-yaml file:// continuous-deployment-policy.yaml
```

명령 출력의 `Id` 값을 복사합니다. 이는 지속적 배포 정책 ID이며 다음 단계에서 필요합니다.

기본 배포에 지속적 배포 정책 연결(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 기본 배포에 대한 구성을 이름이 `primary-distribution.yaml`인 파일에 저장합니다. *`primary_distribution_ID`*를 기본 배포의 ID로 바꿉니다.

```
aws cloudfront get-distribution-config --id primary_distribution_ID --output yaml > primary-distribution.yaml
```

2. 방금 생성한 `primary-distribution.yaml`이라는 파일을 엽니다. 파일을 편집하여 다음과 같이 변경합니다.
   - 지속적 배포 정책 ID(이전 단계에서 복사한 ID)를 `ContinuousDeploymentPolicyId` 필드에 붙여 넣습니다.
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

완료되면 파일을 저장합니다.

3. 지속적 배포 정책을 사용하도록 기본 배포를 업데이트하려면 다음 명령을 사용합니다. *primary_distribution_ID*를 기본 배포의 ID로 바꿉니다.

```
aws cloudfront update-distribution --id primary_distribution_ID --cli-input-yaml
 file://primary-distribution.yaml
```

기본 배포의 구성이 연결된 지속적 배포 정책을 사용하여 엣지 로케이션에 배포되면 CloudFront는 트래픽 구성을 기반으로 지정된 트래픽 부분을 스테이징 배포로 전송하기 시작합니다.

API

CloudFront API를 사용하여 스테이징 배포 및 지속적 배포 정책을 생성하려면 다음 API 작업을 사용합니다.

- CopyDistribution
- CreateContinuousDeploymentPolicy

이러한 API 호출에서 지정하는 필드에 대한 자세한 내용은 다음을 참조하세요.

- the section called “요청을 스테이징 배포로 라우팅”
- the section called “가중치 기반 구성을 위한 세션 고정성”
- AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서

스테이징 배포와 지속적 배포 정책을 만든 후에는 UpdateDistribution(기본 배포에서)을 사용하여 지속적 배포 정책을 기본 배포에 연결합니다.

## 스테이징 배포 업데이트

다음 절차에서는 스테이징 배포 및 지속적 배포 정책을 업데이트하는 방법을 보여 줍니다.

Console

기본 배포와 스테이징 배포 모두에 대해 특정 구성을 업데이트할 수 있습니다. 자세한 내용은 기본 및 스테이징 배포 업데이트 섹션을 참조하세요.

스테이징 배포 업데이트(콘솔)

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 기본 배포를 선택합니다. 현재 프로덕션 트래픽을 제공하는 배포이며, 여기서 스테이징 배포를 만들게 됩니다.
4. View staging distribution(스테이징 배포 보기)를 선택합니다.
5. 콘솔을 사용하여 스테이징 배포의 구성을 수정합니다. 업데이트할 수 있는 설정에 대한 자세한 내용은 the section called "기본 및 스테이징 배포 업데이트"를 참조하십시오.

스테이징 배포의 구성이 엣지 로케이션에 배포되는 즉시 스테이징 배포로 라우팅되는 수신 트래픽에 적용됩니다.

CLI

스테이징 배포 업데이트(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 스테이징 배포에 대한 구성을 이름이 `staging-distribution.yaml`인 파일에 저장합니다. *`staging_distribution_ID`*를 스테이징 배포 ID로 바꿉니다.

```
aws cloudfront get-distribution-config --id staging_distribution_ID --output
 yaml > staging-distribution.yaml
```

2. 방금 생성한 `staging-distribution.yaml`이라는 파일을 엽니다. 파일을 편집하여 다음과 같이 변경합니다.
   - 스테이징 배포의 구성을 수정합니다. 업데이트할 수 있는 설정에 대한 자세한 내용은 the section called "기본 및 스테이징 배포 업데이트"를 참조하십시오.
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

   완료되면 파일을 저장합니다.
3. 다음 명령을 사용하여 스테이징 배포의 구성을 업데이트합니다. *`staging_distribution_ID`*를 스테이징 배포 ID로 바꿉니다.

```
aws cloudfront update-distribution --id staging_distribution_ID --cli-input-yaml
 file://staging-distribution.yaml
```

스테이징 배포의 구성이 엣지 로케이션에 배포되는 즉시 스테이징 배포로 라우팅되는 수신 트래픽에 적용됩니다.

API

스테이징 배포의 구성을 업데이트하려면 UpdateDistribution(스테이징 배포에서)을 사용하여 스테이징 배포의 구성을 수정합니다. 업데이트할 수 있는 설정에 대한 자세한 내용은 the section called "기본 및 스테이징 배포 업데이트"를 참조하십시오.

## 지속적 배포 정책 업데이트

다음 절차에서는 지속적 배포 정책을 업데이트하는 방법을 보여 줍니다.

Console

지속적 배포 정책을 업데이트하여 배포의 트래픽 구성을 업데이트할 수 있습니다.

지속적 배포 정책 업데이트(콘솔)

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 기본 배포를 선택합니다. 현재 프로덕션 트래픽을 제공하는 배포로, 스테이징 배포를 만들 때 바탕이 된 배포입니다.
4. Continuous deployment(지속적 배포) 섹션에서 Edit policy(정책 편집)을 선택합니다.
5. 지속적 배포 정책에서 트래픽 구성을 수정합니다. 작업을 마쳤으면 변경 사항 저장을 선택합니다.

기본 배포의 구성이 업데이트된 지속적 배포 정책을 사용하여 엣지 로케이션에 배포되면 CloudFront는 업데이트된 트래픽 구성을 기반으로 지정된 트래픽 부분을 스테이징 배포로 전송하기 시작합니다.

CLI

지속적 배포 정책 업데이트(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 지속적 배포 정책의 구성을 이름이 `continuous-deployment-policy.yaml`인 파일로 저장합니다. *continuous_deployment_policy_ID*를 지속적 배포 정책의 ID로 바꾸십시오. 다음 명령에서는 가독성을 높이기 위해 이스케이프 문자(\)와 줄 바꿈을 사용하지만 실제 명령에서는 이러한 문자를 생략해야 합니다.

```
aws cloudfront get-continuous-deployment-policy-config --
id continuous_deployment_policy_ID \
                                                        --output yaml >
 continuous-deployment-policy.yaml
```

2. 방금 생성한 `continuous-deployment-policy.yaml`이라는 파일을 엽니다. 파일을 편집하여 다음과 같이 변경합니다.

   - 지속적 배포 정책에서 구성을 원하는 대로 수정합니다. 예를 들어 헤더 기반 트래픽 구성에서 가중치 기반 트래픽 구성으로 변경하거나 가중치 기반 구성의 트래픽 비율(가중치)을 변경할 수 있습니다. 자세한 내용은 the section called "요청을 스테이징 배포로 라우팅" 및 the section called "가중치 기반 구성을 위한 세션 고정성"(을)를 참조하세요.
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

   완료되면 파일을 저장합니다.

3. 지속적 배포 정책을 업데이트하려면 다음 명령을 사용합니다. *continuous_deployment_policy_ID*를 지속적 배포 정책의 ID로 바꾸십시오. 다음 명령에서는 가독성을 높이기 위해 이스케이프 문자(\)와 줄 바꿈을 사용하지만 실제 명령에서는 이러한 문자를 생략해야 합니다.

```
aws cloudfront update-continuous-deployment-policy --
id continuous_deployment_policy_ID \
                                                    --cli-input-yaml file://
continuous-deployment-policy.yaml
```

기본 배포의 구성이 업데이트된 지속적 배포 정책을 사용하여 엣지 로케이션에 배포되면 CloudFront는 업데이트된 트래픽 구성을 기반으로 지정된 트래픽 부분을 스테이징 배포로 전송하기 시작합니다.

API

지속적 배포 정책을 업데이트하려면 UpdateContinuousDeploymentPolicy를 사용합니다.

## 스테이징 배포 구성 승격

다음 절차에서는 스테이징 배포 구성을 승격하는 방법을 보여 줍니다.

Console

스테이징 배포를 승격하면 CloudFront는 스테이징 배포의 구성을 기본 배포로 복사합니다. CloudFront는 지속적 배포 정책도 비활성화하며 모든 트래픽을 기본 배포로 라우팅합니다.

구성을 승격시킨 후 다음에 구성 변경을 테스트할 때 동일한 스테이징 배포를 재사용할 수 있습니다.

스테이징 배포의 구성 승격(콘솔)

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 기본 배포를 선택합니다. 현재 프로덕션 트래픽을 제공하는 배포로, 스테이징 배포를 만들 때 바탕이 된 배포입니다.
4. 지속적 배포 섹션에서 승격을 선택합니다.
5. **`confirm`**을 입력한 다음 Promote(승격)을 선택합니다.

CLI

스테이징 배포를 승격하면 CloudFront는 스테이징 배포의 구성을 기본 배포로 복사합니다. CloudFront는 지속적 배포 정책도 비활성화하며 모든 트래픽을 기본 배포로 라우팅합니다.

구성을 승격시킨 후 다음에 구성 변경을 테스트할 때 동일한 스테이징 배포를 재사용할 수 있습니다.

스테이징 배포의 구성 승격(CLI)

- aws cloudfront update-distribution-with-staging-config 명령을 사용하여 스테이징 배포의 구성을 기본 배포로 승격시킵니다. 다음 예제 명령에서는 가독성을 높이기 위해 이스케이프 문자(\)와 줄 바꿈을 사용하지만 실제 명령에서는 이러한 문자를 생략해야 합니다. 이 예에서 다음과 같이 합니다.
  - *primary_distribution_ID*를 기본 배포의 ID로 바꿉니다.
  - *staging_distribution_ID*를 스테이징 배포 ID로 바꿉니다.
  - *primary_distribution_ETag* 및 *staging_distribution_ETag*를 기본 및 스테이징 배포의 ETag 값으로 바꿉니다. 예와 같이 기본 분포의 값이 첫 번째인지 확인하십시오.

```
aws cloudfront update-distribution-with-staging-config --
id primary_distribution_ID \
                                                        --staging-distribution-
id staging_distribution_ID \
                                                        --if-match
 'primary_distribution_ETag,staging_distribution_ETag'
```

API

스테이징 배포의 구성을 기본 배포로 승격시키려면 UpdateDistributionWithStagingConfig를 사용합니다.

## 스테이징 배포 모니터링

스테이징 배포의 성능을 모니터링하려면 CloudFront에서 모든 배포에 제공하는 것과 동일한 지표, 로그 및 보고서를 사용할 수 있습니다. 예제:

- CloudFront 콘솔에서 기본 CloudFront 배포 지표(예: 총 요청 수 및 오류율)를 볼 수 있으며 추가 비용을 지불하고 추가 지표(예: 캐시 적중률 및 상태 코드별 오류 발생률)를 설정할 수 있습니다. 이러한 지표를 기반으로 경보를 생성할 수도 있습니다.
- 표준 로그와 실시간 액세스 로그를 보고 스테이징 배포에서 받은 요청에 대한 자세한 정보를 얻을 수 있습니다. 표준 로그에는 CloudFront가 스테이징 배포로 요청을 라우팅하기 전에 요청이 원래 전송된 기본 배포를 식별하는 데 도움이 되는 두 필드 `primary-distribution-id` 및 `primary-distribution-dns-name`이 있습니다.

- CloudFront 콘솔에서 보고서(예: 캐시 통계 보고서)를 보고 다운로드할 수 있습니다.

# 지속적 배포의 작동 방식 알아보기

다음 주제에서는 CloudFront 지속적 배포의 작동 방식을 설명합니다.

주제


- 요청을 스테이징 배포로 라우팅
- 가중치 기반 구성을 위한 세션 고정성
- 기본 및 스테이징 배포 업데이트
- 캐시를 공유하지 않는 기본 배포와 스테이징 배포


## 요청을 스테이징 배포로 라우팅

CloudFront 지속적 배포를 사용할 경우 최종 사용자 요청에 대한 내용을 변경할 필요가 없습니다. 최종 사용자는 DNS 이름, IP 주소 또는 CNAME을 사용하여 스테이징 배포에 직접 요청을 보낼 수 없습니다. 대신 최종 사용자는 기본 (프로덕션) 배포에 요청을 보내고 CloudFront는 지속적 배포 정책의 트래픽 구성 설정에 따라 이러한 요청 중 일부를 스테이징 배포로 라우팅합니다. 다음과 같은 두 가지 유형의 트래픽 구성이 있습니다.

가중치 기반

가중치 기반 구성은 지정된 비율의 최종 사용자 요청을 스테이징 배포로 라우팅합니다. 가중치 기반 구성을 사용하는 경우 세션 고정성을 활성화하여 CloudFront가 동일한 사용자의 요청을 단일 세션의 일부로 처리하도록 할 수도 있습니다. 자세한 내용은 the section called “가중치 기반 구성을 위한 세션 고정성” 섹션을 참조하세요.

헤더 기반

헤더 기반 구성은 최종 사용자 요청에 특정 HTTP 헤더(헤더 및 값 지정)가 포함된 경우 요청을 스테이징 배포로 라우팅합니다. 지정된 헤더와 값을 포함하지 않는 요청은 기본 배포로 라우팅됩니다. 이 구성은 로컬 테스트나 최종 사용자 요청을 제어할 수 있는 경우에 유용합니다.

> **Note**
>
> 스테이징 배포로 라우팅되는 헤더에는 `aws-cf-cd-` 접두사가 포함되어야 합니다.

## 가중치 기반 구성을 위한 세션 고정성

트래픽을 스테이징 배포로 라우팅하기 위해 가중치 기반 구성을 사용하는 경우 세션 고정성을 활성화하여 CloudFront가 동일한 사용자의 요청을 단일 세션의 일부로 처리하도록 할 수도 있습니다. 세션 고정성을 활성화하면 CloudFront는 쿠키를 설정하여 단일 세션에서 동일한 사용자의 모든 요청이 기본 배포든 스테이징이든 하나의 배포에서 처리되도록 합니다.

세션 고정성을 활성화하면 유휴 기간도 지정할 수 있습니다. 이 시간 동안 사용자가 유휴 상태(요청을 보내지 않음)인 경우 세션이 만료되고 CloudFront는 이 사용자의 향후 요청을 새 세션으로 취급합니다. 유휴 지속 시간을 300(5분)에서 3600(1시간) 사이의 초 수로 지정합니다.

다음과 같은 경우 CloudFront는 모든 세션(활성 세션 포함)을 재설정하고 모든 요청을 새 세션으로 간주합니다.

- 지속적 배포 정책 활성화 또는 비활성화
- 세션 고정성 설정 활성화 또는 비활성화

## 기본 및 스테이징 배포 업데이트

기본 배포에 연속 배포 정책이 연결된 경우 기본 배포와 스테이징 배포 모두에 대해 다음과 같은 구성 변경을 사용할 수 있습니다.

- 기본 캐시 동작을 포함한 모든 캐시 동작 설정
- 모든 오리진 설정(오리진 및 오리진 그룹)
- 사용자 지정 오류 응답(오류 페이지)
- 지리적 제한
- 기본 루트 객체
- 로깅 설정
- 설명(댓글)

또한 캐시 정책, 응답 헤더 정책, CloudFront 함수 또는 Lambda@Edge 함수와 같이 배포 구성에서 참조되는 외부 리소스를 업데이트할 수 있습니다.

### 캐시를 공유하지 않는 기본 배포와 스테이징 배포

기본 배포와 스테이징 배포는 캐시를 공유하지 않습니다. CloudFront가 스테이징 배포에 첫 번째 요청을 보낼 때 해당 캐시는 비어 있습니다. 요청이 스테이징 배포에 도착하면 응답 캐싱을 시작합니다(그렇게 하도록 구성된 경우).

## 지속적 배포를 위한 할당량 및 기타 고려 사항

CloudFront 지속적 배포에는 다음 할당량 및 기타 고려 사항이 적용됩니다.

### 할당량

- AWS 계정당 최대 스테이징 배포 수: 20
- AWS 계정당 최대 지속적 배포 정책 수: 20
- 가중치 기반 구성에서 스테이징 배포로 보낼 수 있는 최대 트래픽 비율: 15%
- 세션 고정 유휴 기간의 최소값 및 최대값: 300~3600초

자세한 내용은 할당량 섹션을 참조하세요.

> **Note**
>
> 지속적 배포를 사용하고 기본 배포가 S3 버킷 액세스를 위한 OAC로 설정된 경우 스테이징 배포에 대한 액세스를 허용하도록 S3 버킷 정책을 업데이트합니다. 예제 S3 버킷 정책은 the section called “CloudFront에 S3 버킷에 액세스할 수 있는 권한 부여” 섹션을 참조합니다.

### AWS WAF 웹 ACL

배포에 지속적 배포를 활성화하는 경우 다음 고려 사항이 AWS WAF에 적용됩니다.

- ACL이 배포와 처음 연결된 경우 배포에 AWS WAF 웹 액세스 제어 목록(ACL)을 연결할 수 없습니다.
- 배포에서 AWS WAF 웹 ACL을 연결 해제할 수 없습니다.

위의 작업을 수행하려면 먼저 프로덕션 배포에 대한 지속적 배포 정책을 삭제해야 합니다. 이렇게 하면 스테이징 배포의 구성도 업데이트됩니다. 자세한 내용은 AWS WAF 보호 사용 섹션을 참조하세요.

### CloudFront가 모든 요청을 기본 배포로 보내는 경우

리소스 사용률이 높은 기간과 같은 특정 상황에서는 지속적 배포 정책에 지정된 내용에 관계없이 CloudFront가 모든 요청을 기본 배포로 보낼 수 있습니다.

CloudFront는 지속적 배포 정책에 지정된 내용에 관계없이 트래픽이 가장 많은 시간대에 기본 배포에 모든 요청을 보냅니다. 최대 트래픽은 배포의 트래픽이 아니라 CloudFront 서비스의 트래픽을 나타냅니다.

### HTTP/3

HTTP/3을 지원하는 배포에서는 지속적 배포를 사용할 수 없습니다.

## 대체 도메인 이름(CNAME)을 추가하여 사용자 지정 URL 사용

배포를 만들 때 CloudFront가 배포에 도메인 이름을 제공합니다(예: d111111abcdef8.cloudfront.net). 제공된 이 도메인 이름을 사용하는 대신 대체 도메인 이름(CNAME이라고도 함)을 사용할 수 있습니다.

www.example.com과 같은 고유한 도메인 이름을 사용하는 방법은 다음 주제를 참조하세요.

주제


- 대체 도메인 이름 사용과 관련된 요구 사항
- 대체 도메인 이름 사용에 대한 제한
- 대체 도메인 이름 사용
- 대체 도메인 이름 이동
- 대체 도메인 이름 제거
- 대체 도메인 이름에서 와일드카드 사용


### 대체 도메인 이름 사용과 관련된 요구 사항

CloudFront 배포에 www.example.com과 같은 대체 도메인 이름을 추가하는 경우 요구 사항은 다음과 같습니다.

대체 도메인 이름은 소문자여야 합니다.

모든 대체 도메인 이름(CNAME)은 소문자여야 합니다.

대체 도메인 이름은 유효한 TLS 인증서에 포함되어야 합니다.

CloudFront 배포에 대체 도메인 이름(CNAME)을 추가하려면 대체 도메인 이름이 포함된 신뢰할 수 있는 유효한 TLS 인증서를 배포에 연결해야 합니다. 그러면 도메인의 인증서에 대한 액세스 권한이 있는 사람만 해당 도메인과 관련된 CloudFront CNAME과 연결할 수 있습니다.

신뢰할 수 있는 인증서는 AWS Certificate Manager(ACM) 또는 다른 유효한 CA(인증 기관)에서 발급하는 인증서입니다. 자체 서명된 인증서를 사용하여 기존 CNAME의 유효성을 검사할 수 있지만 새 CNAME의 유효성은 확인할 수 없습니다. CloudFront는 Mozilla가 지원하는 것과 동일한 인증 기관을 지원합니다. 최신 목록은 Mozilla에 포함된 CA 인증서 목록을 참조하십시오. 서드 파티 CA 사용 시 중간 인증서에 대한 자세한 내용은 중간 인증서 섹션을 참조하시기 바랍니다.

연결한 인증서를 사용하여 와일드카드가 포함된 대체 도메인 이름 등의 대체 도메인 이름을 확인하기 위해 CloudFront는 인증서에서 주제 대체 이름(SAN)을 확인합니다. 추가하는 대체 도메인 이름은 SAN에 포함되어야 합니다.

> **Note**
>
> 한 번에 인증서 하나만 CloudFront 배포에 연결할 수 있습니다.

다음 중 하나를 수행하여 배포에 특정 대체 도메인 이름을 추가할 권한을 입증합니다.

- 대체 도메인 이름(예: product-name.example.com)이 포함된 인증서 연결.
- 도메인 이름의 시작 부분에 * 와일드카드가 포함된 인증서를 추가하여 인증서 하나로 여러 하위 도메인 포괄. 와일드카드를 지정할 때 CloudFront에서 여러 하위 도메인을 대체 도메인 이름으로 추가할 수 있습니다.

다음 예에서는 인증서 작업에서 도메인 이름에 와일드카드를 사용하여 CloudFront에서 특정 대체 도메인 이름을 추가할 수 있는 권한을 부여하는 보여 줍니다.

- marketing.example.com을 대체 도메인 이름으로 추가하려고 합니다. 인증서에 도메인 이름(*.example.com)을 나열합니다. CloudFront에 이 인증서를 연결할 때 해당 수준에서 와일드카드를 대체하는 배포의 대체 도메인 이름을 추가할 수 있습니다(예: marketing.example.com). 또한 예를 들어 다음과 같은 대체 도메인 이름을 추가할 수도 있습니다.
  - product.example.com
  - api.example.com

그러나 와일드카드보다 높거나 낮은 수준의 대체 도메인 이름은 추가할 수 없습니다. 예를 들어 example.com 또는 marketing.product.example.com은 대체 도메인 이름으로 추가할 수 없습니다.

- example.com을 대체 도메인 이름으로 추가하려고 합니다. 이를 수행하려면 배포에 연결하는 인증서에서 도메인 이름 example.com 자체를 나열해야 합니다.
- marketing.product.example.com을 대체 도메인 이름으로 추가하려고 합니다. 이를 수행하려면 인증서에 *.product.example.com을 나열하거나, 인증서에 marketing.product.example.com 자체를 나열하면 됩니다.

DNS 구성 변경 권한

대체 도메인 이름을 추가하는 경우, CNAME 레코드를 생성하여 대체 도메인 이름에 대한 DNS 쿼리를 CloudFront 배포에 라우팅해야 합니다. 이를 수행하려면 사용 중인 대체 도메인 이름의 DNS 서비스 공급자를 통해 CNAME 레코드를 생성할 수 있는 권한이 있어야 합니다. 일반적으로 이 경우 도메인을 소유하고 있지만 도메인 소유자를 위해 애플리케이션을 개발하는 중일 수도 있습니다.

대체 도메인 이름과 HTTPS

최종 사용자가 대체 도메인 이름과 함께 HTTPS를 사용하도록 하려면 추가 구성을 수행해야 합니다. 자세한 내용은 대체 도메인 이름과 HTTPS 사용 단원을 참조합니다.

## 대체 도메인 이름 사용에 대한 제한

대체 도메인 이름 사용에 대한 제한 사항은 다음과 같습니다.

최대 대체 도메인 이름 수

현재 하나의 배포에 추가할 수 있는 대체 도메인 이름의 최대 수를 살펴보고 더 높은 할당량(이전에는 제한이라고 함)을 요청하려면, 배포의 일반 할당량 단원을 참조하세요.

중복 및 중첩 대체 도메인 이름

AWS 계정에서 다른 배포를 소유하더라도 동일한 대체 도메인 이름이 다른 CloudFront 배포 안에 이미 있는 경우에는 대체 도메인 이름을 CloudFront 배포에 추가할 수 없습니다.

하지만 *.example.com 등의 와일드카드 대체 도메인 이름을 추가할 수 있는데 여기에는 www.example.com 등의 비 와일드카드 대체 도메인 이름이 포함(중첩)됩니다. 배포 2개의 대체 도메인 이름이 겹치는 경우 CloudFront는 DNS 레코드가 가리키는 배포에 상관 없이 보다 구체적으로 일치하는 이름을 사용하여 배포에 요청을 보냅니다. 예를 들어, markeeting.domain.com은 *.domain.com보다 더 구체적입니다.

CloudFront 배포를 가리키는 기존 와일드카드 DNS 항목이 있고 보다 구체적인 이름으로 새 CNAME을 추가하려고 할 때 잘못 구성된 DNS 오류가 발생하는 경우 새 CNAME을 추가하려고 하면 CloudFront에서 잘못 구성된 DNS 레코드 오류가 반환됩니다. 섹션을 참조하세요.

Domain Fronting(도메인 프론팅)

CloudFront에는 여러 AWS 계정에서 발생하는 도메인 프론팅에 대한 보호 기능이 있습니다. 이는 표준이 아닌 클라이언트가 한 AWS 계정의 도메인 이름에 대한 TLS/SSL 연결을 생성하지만 다른 AWS 계정에서 관련이 없는 도메인 이름에 대한 HTTPS 요청을 하는 시나리오입니다.

예를 들어 TLS 연결은 www.example.com에 연결한 후 www.example.org에 대한 요청을 발행할 수 있습니다.

요청이 도메인 프론팅인지 확인하기 위해 CloudFront는 다음 검사를 수행합니다.

- SNI 확장이 HTTP 요청 `Host` 헤더와 동일합니다.
- 인증서가 요청의 배포와 동일한 AWS 계정에 속합니다.
- HTTP 요청 `Host`가 TLS 핸드셰이크 중에 제공되는 인증서에 포함됩니다.

이러한 조건 중 하나라도 충족되지 않으면 CloudFront는 요청이 도메인 프론팅이라고 판단합니다. CloudFront는 421 HTTP 오류 응답과 함께 해당 요청을 거부합니다.

> **Note**
>
> 클라이언트가 SNI 확장을 제공하지 않고 대신 기본 *.cloudfront.net 인증서를 받는 경우 CloudFront는 수신되는 요청을 수락합니다.

CloudFront가 요청의 배포를 식별하는 방법

CloudFront는 `Host` 헤더를 기반으로 HTTP 요청의 배포를 식별합니다. CloudFront는 연결 중인 CloudFront IP 주소 또는 TLS 핸드셰이크 중에 제공된 SNI 핸드셰이크에 의존하지 않습니다.

CloudFront는 요청을 수신하면 `Host` 헤더 값을 사용하여 요청을 특정 배포에 매칭합니다.

예를 들어 배포가 두 개 있고 대체 도메인 이름이 다음 엔드포인트로 라우팅되도록 DNS 구성을 업데이트했다고 가정해 보겠습니다.

- primary.example.com이 d111111primary.cloudfront.net를 가리킴
- secondary.example.com이 d222222secondary.cloudfront.net를 가리킴

`curl https://primary.example.com -H "Host: secondary.example.com"`과 같이 https://primary.example.com에 요청을 하지만 `Host` 헤더를 secondary.example.com으로 지정하면 요청이 대신 보조 배포로 라우팅됩니다.

## 도메인의 최상위 노드(Zone Apex)에서 대체 도메인 이름 추가

배포에 대체 도메인 이름을 추가하는 경우, 일반적으로 DNS 구성에서 CNAME 레코드를 생성하여 도메인 이름에 대한 DNS 쿼리를 CloudFront 배포에 라우팅합니다. 하지만 Zone Apex라고 하는 DNS 네임스페이스의 최상위 노드에 대한 CNAME 레코드를 생성할 수 없습니다. DNS 프로토콜이 허용하지 않습니다. 예를 들어, DNS 이름 example.com을 등록하면 zone apex는 example.com입니다. example.com에 대한 CNAME 레코드를 생성할 수는 없지만, www.example.com, newproduct.example.com 등에 대한 CNAME 레코드는 생성할 수 있습니다.

Route 53을 DNS 서비스로 사용하는 경우 CNAME 레코드에 비해 다음과 같은 장점이 있는 별칭 리소스 레코드 세트를 생성할 수 있습니다.

- 최상위 노드(example.com)에서 도메인 이름에 대한 별칭 리소스 레코드 세트를 생성할 수 있습니다.
- 클라이언트가 지원하는 경우 대체 도메인 이름에 대한 HTTPS 레코드를 생성하여 DNS 조회의 일부로 프로토콜 협상을 허용할 수 있습니다. 자세한 내용은 Create alias resource record set 섹션을 참조하세요.
- 별칭 리소스 레코드 세트를 사용할 때는 Route 53 쿼리에 대한 요금을 지불하지 않습니다.

> **Note**
>
> IPv6를 활성화하는 경우 두 개의 별칭 리소스 레코드를 생성해야 합니다. 하나는 IPv6 트래픽(A 레코드)을 라우팅하고 다른 하나는 IPv4 트래픽(AAAA 레코드)을 라우팅합니다. 자세한 내용은 IPv6 활성화(뷰어 요청) 주제에서 모든 배포 설정 참조 단원을 참조하십시오.

자세한 내용은 Amazon Route 53 개발자 안내서에서 도메인 이름을 사용하여 Amazon CloudFront 웹 배포로 트래픽 라우팅을 참조하세요.

DNS에 Route 53을 사용하지 않는 경우 애니캐스트 고정 IP 주소를 요청하여 example.com와 같은 apex 도메인을 CloudFront로 라우팅할 수 있습니다. 자세한 내용은 허용 목록에 사용할 애니캐스트 고정 IP 요청 섹션을 참조하세요.

# 대체 도메인 이름 사용

다음 작업 목록은 CloudFront 콘솔을 사용하여 배포에 대체 도메인 이름을 추가하여 링크에서 CloudFront 도메인 이름 대신에 고유의 도메인 이름을 사용할 수 있도록 하는 방법을 설명합니다. CloudFront API를 사용한 배포 업데이트에 대한 자세한 내용은 배포 구성 단원을 참조하세요.

> **Note**
>
> 최종 사용자가 대체 도메인 이름에 HTTPS를 사용하도록 하려면 대체 도메인 이름과 HTTPS 사용 단원을 참조하십시오.

시작하기 전에: 대체 도메인 이름을 추가하도록 배포를 업데이트하기 전에 다음을 수행해야 합니다.

- 도메인 이름을 Route 53 또는 다른 도메인 등록 기관에 등록합니다.
- 공인 인증 기관(CA)에서 도메인 이름에 적용되는 TLS 인증서를 받습니다. 인증서를 배포에 추가하여 도메인 사용 권한이 있는지 확인합니다. 자세한 내용은 대체 도메인 이름 사용과 관련된 요구 사항 섹션을 참조하세요.

대체 도메인 이름 사용

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 업데이트하려는 배포의 ID를 선택합니다.
3. 일반 탭에서 도메인 추가를 선택합니다.
4. 서비스를 제공할 도메인을 최대 5개까지 입력합니다.
5. 다음을 선택합니다.
6. TLS 인증서의 경우 CloudFront가 AWS 계정의 `us-east-1` AWS 리전에서 도메인에 대한 기존 AWS Certificate Manager(ACM) 인증서를 찾을 수 없다면 인증서를 자동으로 생성하거나 ACM에서 수동으로 생성하도록 선택할 수 있습니다.
7. 인증서가 프로비저닝되면 DNS 공급자로 DNS 레코드를 업데이트하여 도메인 소유권을 입증해야 합니다. DNS 레코드에 입력해야 하는 항목은 CloudFront 콘솔에서 제공됩니다.
8. DNS 레코드를 업데이트한 후 인증서 검증을 선택합니다.
9. 인증서가 검증되면 다음을 선택합니다.
10. 변경 사항을 검토하고 도메인 추가를 선택합니다.

11. 배포에 대한 일반 탭에서 배포 상태가 배포 완료로 변경되었는지 확인합니다. 배포에 대한 업데이트가 배포되기 전에 대체 도메인 이름을 사용하려고 하면 다음 단계에서 생성하는 링크가 작동하지 않을 수도 있습니다.
12. 트래픽을 배포의 CloudFront 도메인 이름(예: d111111abcdef8.cloudfront.net)으로 라우팅하도록 대체 도메인 이름(예: www.exapmle.com)에 대한 DNS 서비스를 구성합니다. 사용하는 방법은 Route 53을 도메인의 DNS 서비스 공급자로 사용하는지, 다른 공급자로 사용하는지에 따라 결정됩니다. 자세한 내용은 CloudFront 표준 배포에 도메인 추가 섹션을 참조하세요.

    Route 53

    별칭 리소스 레코드 세트를 생성합니다. 별칭 리소스 레코드 세트를 사용하면 Route 53 쿼리에 대해서는 요금이 지불되지 않습니다. 또한 DNS에서 CNAME을 허용하지 않는 루트 도메인 이름(example.com)에 대해 별칭 리소스 레코드 세트를 생성할 수 있습니다. 별칭 리소스 레코드 세트 생성에 대한 자세한 내용은 Amazon Route 53 개발자 안내서의 도메인 이름을 사용하여 Amazon CloudFront 웹 배포로 트래픽 라우팅을 참조하세요.

    선택적으로, 클라이언트가 지원하는 경우 대체 도메인 이름에 대한 HTTPS 레코드를 생성하여 DNS 조회의 일부로 프로토콜 협상을 허용할 수 있습니다.

    HTTPS 레코드를 사용하여 별칭 리소스 레코드 세트를 생성하려면(선택 사항)

    1. CloudFront 배포 설정에서 HTTP/2 또는 HTTP/3을 활성화합니다. 자세한 내용은 지원되는 HTTP 버전 및 배포 업데이트(을)를 참조하세요.
    2. Route 53 콘솔에서 별칭 리소스 레코드 세트를 생성합니다. 도메인 이름을 사용하여 Amazon CloudFront 웹 배포로 트래픽 라우팅 절차를 따르세요.
    3. 별칭 리소스 레코드 세트를 생성하는 동안 레코드 유형 HTTPS를 사용하여 별칭 레코드를 생성합니다.

    다른 DNS 서비스 공급자

    DNS 서비스 공급자가 제공하는 방법을 사용하여 도메인에 대한 CNAME 레코드를 추가합니다. 이 새 CNAME 레코드는 DNS 쿼리를 대체 도메인 이름(예: www.example.com)에서 배포의 CloudFront 도메인 이름(예: d111111abcdef8.cloudfront.net)으로 리디렉션합니다. 자세한 내용은 DNS 서비스 공급자가 제공하는 설명서를 참조하십시오.

> **Important**
>
> 대체 도메인 이름에 대한 기존 CNAME 레코드가 이미 있는 경우 해당 레코드를 업데이트하거나 배포의 CloudFront 도메인 이름을 가리키는 새 레코드로 바꿉니다.

13. `dig` 또는 이와 유사한 DNS 도구를 사용하여 이전 단계에서 생성한 DNS 구성이 배포의 도메인 이름을 가리키는지 확인합니다.

    다음 예는 www.example.com 도메인에 대한 `dig` 요청과 응답의 관련 부분을 보여 줍니다.

```
PROMPT> dig www.example.com

; <<> DiG 9.3.3rc2 <<> www.example.com
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15917
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;www.example.com.     IN    A

;; ANSWER SECTION:
www.example.com. 10800 IN CNAME d111111abcdef8.cloudfront.net.
...
```

    Answer Section에서는 CNAME 레코드가 www.example.com에 대한 쿼리를 CloudFront 배포 도메인 이름 d111111abcdef8.cloudfront.net로 라우팅합니다. `CNAME`의 오른쪽에 있는 이름이 CloudFront 배포의 도메인 이름이면 CNAME 레코드가 올바르게 구성된 것입니다. 이 값이 다른 값이면(예: Amazon S3 버킷의 도메인 이름) CNAME 레코드가 잘못 구성된 것입니다. 이 경우에는 7단계로 돌아가서 배포의 도메인 이름을 가리키도록 CNAME 레코드를 수정합니다.

14. 배포의 CloudFront 도메인 이름 대신에 고유 도메인 이름을 사용하는 URL을 방문하여 대체 도메인 이름을 테스트합니다.

15. 애플리케이션에서 CloudFront 배포의 도메인 이름 대신에 대체 도메인 이름을 사용하도록 객체 URL을 변경합니다.

# 대체 도메인 이름 이동

표준 배포 또는 배포 테넌트에 대체 도메인 이름을 추가하려고 하는데 대체 도메인 이름이 이미 다른 리소스와 연결되어 있는 경우 오류 메시지가 표시됩니다.

예를 들어 표준 배포 또는 배포 테넌트에 www.example.com을 추가하려고 하는데 이 대체 도메인 이름이 이미 다른 리소스와 연결되어 있는 경우 `CNAMEAlreadyExists` 오류 메시지(One or more of the CNAMEs you provided are already associated with a different resource)가 표시됩니다.

이 경우 기존 대체 도메인 이름을 한 리소스에서 다른 리소스로 이동할 수 있습니다. 소스 배포와 대상 배포입니다. 표준 배포 및/또는 배포 테넌트 간에 대체 도메인 이름을 이동할 수 있습니다.

대체 도메인 이름을 이동하려면 다음 항목을 참조하세요.

주제

- 대상 표준 배포 또는 배포 테넌트 설정
- 소스 표준 배포 또는 배포 테넌트 찾기
- 대체 도메인 이름 이동

## 대상 표준 배포 또는 배포 테넌트 설정

대체 도메인 이름을 이동하려면 먼저 대상 리소스를 설정해야 합니다. 대체 도메인 이름을 이동하려는 대상 표준 배포 또는 배포 테넌트입니다.

Standard distribution

대상 표준 배포를 설정하려면

1. TLS 인증서를 요청합니다. 이 인증서에는 대체 도메인 이름이 주체 또는 주체 대체 도메인(SAN)으로 포함되거나 이동하려는 대체 도메인 이름에 적용되는 와일드카드(*)가 포함됩니다. 없는 경우 AWS Certificate Manager(ACM) 또는 다른 인증 기관(CA)에 요청하거나 ACM으로 가져올 수 있습니다.

> **Note**
>
> 미국 동부(버지니아 북부)(`us-east-1`) 리전에서 인증서를 요청하거나 가져와야 합니다.

자세한 내용은 AWS Certificate Manager 사용 설명서의 AWS Certificate Manager에서 콘솔을 사용하여 퍼블릭 인증서 요청 또는 인증서 가져오기를 참조하세요.

2. 대상 표준 배포를 생성하지 않은 경우 지금 생성합니다. 표준 배포 생성의 일부로 인증서를 이 표준 배포와 연결합니다. 자세한 내용은 배포 생성 섹션을 참조하세요.

   대상 표준 배포가 이미 있는 경우 인증서를 표준 배포와 연결합니다. 자세한 내용은 배포 업데이트 섹션을 참조하세요.

3. 동일한 AWS 계정 내에서 대체 도메인 이름을 이동하는 경우 이 단계를 건너뜁니다.

   대체 도메인 이름을 AWS 계정 간에 이동하려면 DNS 구성에서 TXT 레코드를 생성해야 합니다. 이 확인 단계는 무단 도메인 전송을 방지하는 데 도움이 됩니다. CloudFront가 이 TXT 레코드를 사용하여 대체 도메인 이름의 소유권을 확인합니다.

   DNS 구성에서 대체 도메인 이름을 대상 표준 배포와 연결하는 DNS TXT 레코드를 생성합니다. TXT 레코드 형식은 도메인 유형에 따라 다를 수 있습니다.

   - 하위 도메인의 경우 대체 도메인 이름 앞에 밑줄(_)을 지정합니다. 다음에 TXT 레코드의 예시가 나와 있습니다.

     ```
     _www.example.com TXT d111111abcdef8.cloudfront.net
     ```

   - apex(또는 루트 도메인)의 경우 도메인 이름 앞에 밑줄과 마침표(_.)를 지정합니다. 다음에 TXT 레코드의 예시가 나와 있습니다.

     ```
     _.example.com TXT d111111abcdef8.cloudfront.net
     ```

Distribution tenant

대상 배포 테넌트를 설정하려면

1. TLS 인증서를 요청합니다. 이 인증서에는 대체 도메인 이름이 주체 또는 주체 대체 도메인(SAN)으로 포함되거나 이동하려는 대체 도메인 이름에 적용되는 와일드카드(*)가 포함됩니다. 없는 경우 AWS Certificate Manager(ACM) 또는 다른 인증 기관(CA)에 요청하거나 ACM으로 가져올 수 있습니다.

> **Note**
>
> 미국 동부(버지니아 북부)(`us-east-1`) 리전에서 인증서를 요청하거나 가져와야 합니다.

자세한 내용은 AWS Certificate Manager 사용 설명서의 AWS Certificate Manager에서 콘솔을 사용하여 퍼블릭 인증서 요청 또는 인증서 가져오기를 참조하세요.

2. 대상 배포 테넌트를 생성하지 않은 경우 지금 생성합니다. 대상 배포를 생성하는 과정에서 인증서를 배포 테넌트와 연결합니다. 자세한 내용은 배포 생성 섹션을 참조하세요.

   대상 표준 배포가 이미 있는 경우 인증서를 배포 테넌트와 연결합니다. 자세한 내용은 도메인 및 인증서 추가(분산 테넌트) 섹션을 참조하세요.

3. 동일한 AWS 계정 내에서 대체 도메인 이름을 이동하는 경우 이 단계를 건너뜁니다.

   대체 도메인 이름을 AWS 계정 간에 이동하려면 DNS 구성에서 TXT 레코드를 생성해야 합니다. 이 확인 단계는 무단 도메인 이전을 방지하는 데 도움이 되며 CloudFront는 이 TXT 레코드를 사용하여 대체 도메인 이름의 소유권을 검증합니다.

   DNS 구성에서 대체 도메인 이름을 대상 배포 테넌트와 연결하는 DNS TXT 레코드를 생성합니다. TXT 레코드 형식은 도메인 유형에 따라 다를 수 있습니다.

   - 하위 도메인의 경우 대체 도메인 이름 앞에 밑줄(_)을 지정합니다. 다음에 TXT 레코드의 예시가 나와 있습니다.

     ```
     _www.example.com TXT d111111abcdef8.cloudfront.net
     ```

   - apex(또는 루트 도메인)의 경우 도메인 이름 앞에 밑줄과 마침표(`_.`)를 지정합니다. 다음에 TXT 레코드의 예시가 나와 있습니다.

     ```
     _.example.com TXT d111111abcdef8.cloudfront.net
     ```

그러고 나서 다음 항목을 참조하여 대체 도메인 이름과 이미 연결된 소스 표준 배포 또는 배포 테넌트를 찾습니다.

## 소스 표준 배포 또는 배포 테넌트 찾기

한 배포(표준 또는 테넌트)에서 다른 배포로 대체 도메인 이름을 이동하려면 먼저 소스 배포를 찾습니다. 대체 도메인 이름이 이미 연결된 리소스입니다. 소스와 대상 배포 리소스의 AWS 계정 ID 모두를 알고 있는 경우 대체 도메인 이름을 이동하는 방법을 결정할 수 있습니다.

> **참고**
>
> - 표준 배포와 배포 테넌트를 모두 지원하는 ListDomainConflicts API 작업을 사용하는 것이 좋습니다.
> - ListConflictingAliases API 작업은 표준 배포만 지원합니다.

다음 예시를 따라 소스 배포(표준 또는 테넌트)를 찾습니다.

list-domain-conflicts

> **Tip**
>
> - 표준 배포의 경우 `cloudfront:GetDistribution` 및 `cloudfront:ListDomainConflicts` 권한이 있어야 합니다.
> - 배포 테넌트의 경우 `cloudfront:GetDistributionTenant` 및 `cloudfront:ListDomainConflicts` 권한이 있어야 합니다.

**`list-domain-conflicts`**를 사용하여 소스 표준 배포 또는 배포 테넌트를 찾으려면

1. 다음 예와 같이 `list-domain-conflicts` 명령을 사용합니다.

   a. *`www.example.com`*을 도메인 이름으로 바꿉니다.

   b. `domain-control-validation-resource`의 경우 이전에 설정한 대상 표준 배포 또는 배포 테넌트의 ID를 지정합니다. 지정된 도메인에 적용되는 인증서와 연결된 표준 배포 또는 배포 테넌트가 있어야 합니다.

   c. 대상 표준 배포 또는 배포 테넌트와 동일한 AWS 계정에 있는 자격 증명을 사용하여 이 명령을 실행합니다.

   요청

이 예시에서는 배포 테넌트를 지정합니다.

```
aws cloudfront list-domain-conflicts \
--domain www.example.com \
--domain-control-validation-resource
 "DistributionTenantId=dt_2x9GhoK0TZRsohWzv1b9It8JABC"
```

응답

명령 출력의 각 도메인 이름에 대해 다음을 볼 수 있습니다.

- 도메인에 연결된 리소스 유형
- 리소스 ID
- 리소스를 소유한 AWS 계정 ID

리소스 ID와 계정 ID는 부분적으로 가려집니다. 이를 통해 계정에 속한 표준 배포 또는 배포 테넌트를 식별하고 소유하지 않은 표준 배포 또는 배포 테넌트의 정보를 보호할 수 있습니다.

```
{
    "DomainConflicts": [
        {
            "Domain": "www.example.com",
            "ResourceType": "distribution-tenant",
            "ResourceId": "***************ohWzv1b9It8JABC",
            "AccountId": "******112233"
        }
    ]
}
```

응답에는 지정한 도메인 이름과 충돌하거나 겹치는 모든 도메인 이름이 나열됩니다.

예제

- *tenant1.example.com*을 지정하는 경우 응답에는 tenant1.example.com 및 겹치는 와일드카드 대체 도메인 이름(*.example.com, 있는 경우)이 포함됩니다.
- **.tenant1.example.com*을 지정하면 응답에 *.tenant1.example.com 및 해당 와일드카드가 적용되는 대체 도메인 이름(예: test.tenant1.example.com, dev.tenant1.example.com 등)이 포함됩니다.

2. 응답에서 이동하려는 대체 도메인 이름의 소스 표준 배포 또는 배포 테넌트를 찾고 AWS 계정 ID를 기록해 둡니다.
3. 소스 표준 배포 또는 배포 테넌트의 계정 ID를 이전 단계에서 대상 표준 배포 또는 배포 테넌트를 생성한 계정 ID와 비교합니다. 그러면 소스와 대상이 동일한 AWS 계정에 있는지 확인할 수 있습니다. 이렇게 하면 대체 도메인 이름을 이동하는 방법을 결정할 수 있습니다.

   자세한 내용은 AWS Command Line Interface 명령 참조의 list-domain-conflicts 명령을 참조하세요.

list-conflicting-aliases (standard distributions only)

> **Tip**
> 대상 표준 배포에 대한 `cloudfront:GetDistribution` 및 `cloudfront:ListConflictingAliases` 권한이 있어야 합니다.

**`list-conflicting-aliases`**를 사용하여 소스 표준 배포를 찾으려면

1. 다음 예와 같이 `list-conflicting-aliases` 명령을 사용합니다.

   a. *`www.example.com`*을 대체 도메인 이름으로 바꾸고 *`EDFDVBD6EXAMPLE`*을 이전에 설정한 대상 표준 배포의 ID로 바꿉니다.

   b. 대상 표준 배포와 동일한 AWS 계정의 자격 증명을 사용하여 이 명령을 실행합니다.

   요청

   이 예시에서는 표준 배포를 지정합니다.

```
aws cloudfront list-conflicting-aliases \
--alias www.example.com \
--distribution-id EDFDVBD6EXAMPLE
```

   응답

   명령의 출력에 있는 각 대체 도메인 이름에 대해 연결된 표준 배포의 ID와 해당 표준 배포를 소유하는 AWS 계정 ID를 볼 수 있습니다. 표준 배포 및 계정 ID는 부분적으로 가려지므로 본인이

소유한 표준 배포와 계정을 식별할 수 있지만 소유하지 않은 배포의 정보를 보호하는 데 도움이 됩니다.

```
{
    "ConflictingAliasesList": {
        "MaxItems": 100,
        "Quantity": 1,
        "Items": [
            {
                "Alias": "www.example.com",
                "DistributionId": "*******EXAMPLE",
                "AccountId": "******112233"
            }
        ]
    }
}
```

응답에는 지정한 도메인 이름과 충돌하거나 겹치는 대체 도메인 이름이 나열됩니다.

예제

- *www.example.com*을 지정하는 경우 응답에는 www.example.com 및 겹치는 와일드카드 대체 도메인 이름(*.example.com, 있는 경우)이 포함됩니다.
- **.example.com*을 지정한다면 응답에는 *.example.com 및 해당 와일드카드가 적용되는 대체 도메인 이름(예: www.example.com, test.example.com, dev.example.com 등)이 명령의 출력에 포함됩니다.

2. 이동하려는 대체 도메인 이름의 표준 배포를 찾고 AWS 계정 ID를 기록해 둡니다. 이 계정 ID를 이전 단계에서 대상 표준 배포를 생성한 계정 ID와 비교합니다. 그러면 이러한 두 표준 배포가 동일한 AWS 계정에 있는지 여부와 대체 도메인 이름을 이동하는 방법을 확인할 수 있습니다.

   자세한 내용은 AWS Command Line Interface 명령 참조의 list-conflicting-aliases 명령을 참조하세요.

그런 다음 대체 도메인 이름을 이동하려면 다음 항목을 참조하세요.

## 대체 도메인 이름 이동

상황에 따라 다음 방법 중에서 선택하여 대체 도메인 이름을 이동합니다.

소스와 대상 배포(표준 또는 테넌트)가 동일한 AWS 계정에 있음

AWS Command Line Interface(AWS CLI)에서 update-domain-association 명령을 사용하여 대체 도메인 이름을 이동합니다.

이 명령은 대체 도메인 이름이 apex 도메인(루트 도메인이라고도 함. 예: example.com)인 경우를 포함해 같은 계정 이동인 경우 작동합니다.

소스와 대상 배포(표준 또는 테넌트)가 서로 다른 AWS 계정에 있음

소스 표준 배포 또는 배포 테넌트에 대한 액세스 권한이 있고 대체 도메인 이름이 apex 도메인이 아니며 해당 대체 도메인 이름과 겹치는 와일드카드를 아직 사용하고 있지 않은 경우, 대체 도메인 이름을 이동하려면 와일드카드를 사용합니다. 자세한 내용은 the section called “와일드카드를 사용하여 대체 도메인 이름 이동” 섹션을 참조하세요.

소스 표준 배포 또는 배포 테넌트의 AWS 계정에 대한 액세스 권한이 없는 경우, update-domain-association 명령을 사용해 대체 도메인 이름을 이동해볼 수 있습니다. 대체 도메인 이름을 이동하려면 소스 표준 배포 또는 배포 테넌트를 비활성화해야 합니다. 추가적인 도움말은 the section called “AWS Support에 문의하여 대체 도메인 이름을 이동” 섹션을 참조하십시오.

> **Note**
>
> associate-alias 명령을 사용할 수 있지만 이 명령은 표준 배포만 지원합니다. Amazon CloudFront API 참조의 AssociateAlias를 참조하세요.

update-domain-association (standard distributions and distribution tenants)

**`update-domain-association`**를 사용하여 대체 도메인 이름을 이동하려면

1. 다음 예와 같이 `update-domain-association` 명령을 사용하세요.

   a. *`example.com`*을 대체 도메인 이름으로 바꾸고 대상 표준 배포 또는 배포 테넌트의 ID를 지정합니다.

   b. 대상 표준 배포 또는 배포 테넌트와 동일한 AWS 계정에 있는 자격 증명을 사용하여 이 명령을 실행합니다.

다음과 같은 제한 사항이 있습니다.

- `cloudfront:UpdateDomainAssociation` 권한 외에도 표준 배포를 업데이트할 수 있는 `cloudfront:UpdateDistribution` 권한이 있어야 합니다. 배포 테넌트를 업데이트하려면 `cloudfront:UpdateDistributionTenant` 권한이 있어야 합니다.
- 소스 배포와 대상 배포(표준 또는 테넌트)가 서로 다른 AWS 계정에 있는 경우 도메인을 이동하려면 먼저 소스를 비활성화해야 합니다.
- 대상 배포는 the section called "대상 표준 배포 또는 배포 테넌트 설정"에 설명된 대로 설정해야 합니다.

요청

```
aws cloudfront update-domain-association \
  --domain "www.example.com" \
  --target-resource DistributionTenantId=dt_9Fd3xTZq7Hl2KABC \
  --if-match E3UN6WX5ABC123
```

응답

```
{
    "ETag": "E7Xp1Y3N9DABC",
    "Domain": "www.example.com",
    "ResourceId": "dt_9Fd3xTZq7Hl2KABC"
}
```

이 명령은 대체 도메인 이름을 소스 표준 배포 또는 배포 테넌트에서 제거하고 대상 표준 배포 또는 배포 테넌트에 추가합니다.

2. 대상 배포가 완전히 배포되면 도메인 이름이 CloudFront 라우팅 엔드포인트를 가리키도록 DNS 구성을 업데이트합니다. 예를 들어 DNS 레코드가 대체 도메인 이름(`www.example.com`)을 CloudFront 제공 도메인 이름인 d111111abcdef8.cloudfront.net로 가리키도록 합니다. 대상이 배포 테넌트인 경우 연결 그룹 엔드포인트를 지정합니다. 자세한 내용은 도메인을 CloudFront로 가리키기 섹션을 참조하세요.

associate-alias (standard distributions only)

**associate-alias**를 사용하여 대체 도메인 이름을 이동하려면

1. 다음 예와 같이 `associate-alias` 명령을 사용하세요.

   a. *www.example.com*을 대체 도메인 이름으로 바꾸고 *EDFDVBD6EXAMPLE*을 대상 표준 배포 ID로 바꿉니다.

   b. 대상 표준 배포와 동일한 AWS 계정의 자격 증명을 사용하여 이 명령을 실행합니다.

   > 다음과 같은 제한 사항이 있습니다.
   >
   > - 표준 대상 배포에 대한 `cloudfront:AssociateAlias` 및 `cloudfront:UpdateDistribution` 권한이 있어야 합니다.
   > - 소스 표준 배포와 대상 표준 배포가 동일한 AWS 계정에 있는 경우 소스 표준 배포에 대한 `cloudfront:UpdateDistribution` 권한이 있어야 합니다.
   > - 소스 표준 배포와 대상 표준 배포가 서로 다른 AWS 계정에 있는 경우 먼저 소스 표준 배포를 비활성화해야 합니다.
   > - 대상 표준 배포는 the section called “대상 표준 배포 또는 배포 테넌트 설정”에 설명된 대로 설정해야 합니다.

   요청

   ```
   aws cloudfront associate-alias \
   --alias www.example.com \
   --target-distribution-id EDFDVBD6EXAMPLE
   ```

   이 명령은 대체 도메인 이름을 소스 표준 배포에서 제거하고 대상 표준 배포로 이동합니다.

2. 대상 표준 배포가 완전히 배포된 후 대체 도메인 이름의 DNS 레코드가 대상 표준 배포의 배포 도메인 이름을 가리키도록 DNS 구성을 업데이트합니다. 예를 들어 DNS 레코드가 대체 도메인 이름(`www.example.com`)을 CloudFront 제공 도메인 이름인 d111111abcdef8.cloudfront.net로 가리키도록 합니다.

자세한 내용은 AWS CLI 명령 참조의 associate-alias 명령을 참조하세요.

### 와일드카드를 사용하여 대체 도메인 이름 이동

소스 배포가 대상 배포와 서로 다른 AWS 계정에 있고 소스 배포가 활성화된 경우, 와일드카드를 사용하여 대체 도메인 이름을 이동할 수 있습니다.

> **Note**
>
> 와일드카드를 사용하여 apex 도메인(예: example.com)을 이동할 수는 없습니다. 소스 배포와 대상 배포가 서로 다른 AWS 계정에 있을 때 apex 도메인을 이동하려면 지원에 문의하세요. 자세한 내용은 the section called “AWS Support에 문의하여 대체 도메인 이름을 이동” 섹션을 참조하세요.

와일드카드를 사용하여 대체 도메인 이름을 이동하려면 다음을 수행합니다.

> **Note**
>
> 이 프로세스에는 배포에 대한 다중 업데이트가 포함됩니다. 각 배포가 최신 변경 사항을 완전히 배포할 때까지 기다린 후 다음 단계를 진행합니다.

1. 대상 배포를 업데이트하여 이동할 대체 도메인 이름을 포함하는 와일드카드 대체 도메인 이름을 추가합니다. 예를 들어 이동할 대체 도메인 이름이 www.example.com인 경우 대체 도메인 이름 *.example.com을 대상 배포에 추가합니다. 이렇게 하려면 대상 배포의 SSL/TLS 인증서에 와일드카드 도메인 이름이 포함되어야 합니다. 자세한 내용은 the section called “배포 업데이트” 단원을 참조하세요.
2. 대상 배포의 도메인 이름을 가리키도록 대체 도메인 이름에 대한 DNS 설정을 업데이트합니다. 예를 들어 이동하려는 대체 도메인 이름이 www.example.com인 경우 대상 배포의 도메인 이름(예: d1111abcdef8.cloudfront.net)으로 트래픽을 라우팅하도록 www.example.com에 대한 DNS 레코드를 업데이트합니다.

   > **Note**
   >
   > DNS 설정을 업데이트해도 대체 도메인 이름이 현재 구성되어 있으므로 대체 도메인 이름이 소스 배포에서 계속 제공됩니다.

3. 소스 배포를 업데이트하여 대체 도메인 이름을 제거합니다. 자세한 내용은 배포 업데이트 단원을 참조하세요.

4. 대상 배포를 업데이트하여 대체 도메인 이름을 추가합니다. 자세한 내용은 배포 업데이트 단원을 참조하세요.
5. dig(또는 유사한 DNS 쿼리 도구)을(를) 사용하여 대체 도메인 이름에 대한 DNS 레코드가 대상 배포의 도메인 이름으로 확인되는지 확인합니다.
6. (선택 사항) 대상 배포를 업데이트하여 와일드카드 대체 도메인 이름을 제거합니다.

AWS Support에 문의하여 대체 도메인 이름을 이동

소스 배포와 대상 배포가 서로 다른 AWS 계정에 있고 소스 배포의 AWS 계정에 대한 액세스 권한이 없거나 소스 배포를 비활성화할 수 없는 경우 지원에 문의하여 대체 도메인 이름을 이동할 수 있습니다.

지원에 문의하여 대체 도메인 이름을 이동하려면 다음을 수행하세요.

1. 대상 배포를 가리키는 DNS TXT 레코드를 포함하여 대상 배포를 설정합니다. 자세한 내용은 대상 표준 배포 또는 배포 테넌트 설정 단원을 참조하세요.
2. 지원에 문의해 도메인 소유권을 확인하고 새 CloudFront 배포로 도메인을 이동시켜 달라고 요청합니다.
3. 대상 배포가 완전히 배포된 후 대체 도메인 이름의 DNS 레코드가 대상 배포의 배포 도메인 이름을 가리키도록 DNS 구성을 업데이트합니다.

## 대체 도메인 이름 제거

도메인 또는 하위 도메인에서 CloudFront 배포로의 트래픽 라우팅을 중단시키고 싶으면 여기 나오는 단계에 따라 DNS 구성과 CloudFront 배포를 모두 업데이트합니다.

배포에서 대체 도메인 이름을 제거하고 DNS 구성을 업데이트하는 것이 중요합니다. 이렇게 하면 나중에 CloudFront 배포에 도메인 이름을 연결하고 싶은 경우에 문제가 발생하는 것을 막을 수 있습니다. 대체 도메인 이름이 이미 하나의 배포에 연결되어 있는 경우에는 또 다른 연결을 설정할 수 없습니다.

> **Note**
>
> 또 다른 배포에 추가할 수 있도록 이 배포에서 대체 도메인 이름을 제거하고 싶으면 대체 도메인 이름 이동의 단계를 따릅니다. 도메인을 제거하는 대신에 여기 나온 단계에 따라 또 다른 배포에 도메인을 추가한 경우에는 도메인이 새 배포에 연결되지 않는 기간이 존재합니다. 왜냐하면 CloudFront가 엣지 로케이션에 대한 업데이트로 전파되고 있기 때문입니다.

배포에서 대체 도메인 이름을 제거하려면

1. 먼저 도메인에 대한 인터넷 트래픽을 CloudFront 배포가 아닌 또 다른 리소스(예: Elastic Load Balancing 로드 밸런서)로 라우팅합니다. 또는 CloudFront로 트래픽을 라우팅하고 있는 DNS 레코드를 삭제할 수 있습니다.

   도메인의 DNS 서비스에 따라 다음 중 하나를 수행합니다.

   - Route 53을 사용 중인 경우에는 별칭 레코드나 CNAME 레코드를 업데이트 또는 삭제합니다. 자세한 내용은 레코드 편집 또는 레코드 삭제를 참조하세요.
   - 또 다른 DNS 서비스 공급자를 사용하고 있는 경우에는 DNS 서비스 공급자가 제공한 방법을 사용하여 CloudFront로 트래픽을 전달하는 CNAME 레코드를 업데이트 또는 삭제합니다. 자세한 내용은 DNS 서비스 공급자가 제공하는 설명서를 참조하십시오.
2. 도메인의 DNS 레코드를 업데이트한 후에는 변경 사항이 전파되어 DNS 해석기가 새 리소스로 트래픽을 라우팅할 때까지 기다립니다. 라우팅이 완료되면 URL에서 도메인을 사용하는 몇 가지 테스트 링크를 생성하여 확인할 수 있습니다.
3. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 연 다음, 다음과 같이 CloudFront 배포를 업데이트하여 도메인 이름을 제거합니다.

   a. 업데이트하려는 배포의 ID를 선택합니다.
   b. [General] 탭에서 [Edit]를 선택합니다.
   c. 대체 도메인 이름 사용(CNAME)에서 배포에서 더 이상 사용하고 싶지 않은 대체 도메인 이름 또는 도메인 이름을 제거합니다.
   d. 예, 편집합니다를 선택합니다.

## 대체 도메인 이름에서 와일드카드 사용

대체 도메인 이름을 추가할 경우 하위 도메인을 개별적으로 추가하는 대신에 도메인 이름 시작 부분에 와일드카드(*)를 사용할 수 있습니다. 예를 들어 대체 도메인 이름으로 *.example.com을 사용하면 example.com으로 끝나는 모든 도메인 이름(예: www.example.com, product-name.example.com, marketing.product-name.example.com 등)을 URL에 사용할 수 있습니다. 객체의 경로는 도메인 이름과 상관없이 동일하며, 예를 들면 다음과 같습니다.

- www.example.com/images/image.jpg
- product-name.example.com/images/image.jpg

- marketing.product-name.example.com/images/image.jpg

와일드카드가 포함된 대체 도메인 이름에 대해 다음 요구 사항을 따르세요.

- 대체 도메인 이름은 별표(*)와 마침표(.)로 시작해야 합니다.
- 와일드카드를 사용하여 하위 도메인 이름의 일부를 대체할 수 없습니다(예: *domain.example.com).
- 도메인 이름의 중간에 있는 하위 도메인은 바꿀 수 없습니다(예: subdomain.*.example.com).
- 와일드카드를 사용하는 대체 도메인 이름을 포함하여 모든 대체 도메인 이름은 인증서의 주체 대체 이름(SAN)으로 보호되어야 합니다.

와일드카드 대체 도메인 이름(예: *.example.com)에는 사용 중인 다른 대체 도메인 이름(예: example.com이 포함될 수 있습니다.

# CloudFront 배포에 WebSocket 사용

Amazon CloudFront는 클라이언트와 서버 사이에 수명이 긴 양방향 연결이 필요할 때 유용한 TCP 기반 프로토콜인 WebSocket의 사용을 지원합니다. 지속적 연결이 실시간 애플리케이션의 요구 사항에 포함되는 경우가 흔히 있습니다. WebSockets를 사용할 수 있는 시나리오에는 채팅 플랫폼, 온라인 협업 공간, 멀티 플레이어 게임 및 금융 거래 플랫폼과 같이 실시간 데이터 피드를 제공하는 서비스 등이 포함됩니다. WebSocket 연결에서는 데이터가 전이중 통신의 양방향으로 흐를 수 있습니다.

WebSocket 기능은 모든 배포판에서 작동하도록 자동으로 활성화됩니다. WebSocket을 사용하려면 배포에 연결된 캐시 동작에서 다음 중 하나를 구성합니다.

- 모든 뷰어 요청 헤더를 오리진에 전달합니다. AllViewer 관리형 오리진 요청 정책을 사용할 수 있습니다.
- 특히 원본 요청 정책에서 `Sec-WebSocket-Key` 및 `Sec-WebSocket-Version` 요청 헤더를 구체적으로 전달하세요.

## WebSocket 프로토콜의 작동 방식

WebSocket 프로토콜은 독립적인 TCP 기반의 프로토콜이며, 이를 통해 HTTP의 오버헤드 및 잠재적인 지연 시간 연장을 방지할 수 있습니다.

WebSocket 연결을 설정하기 위해, 클라이언트가 HTTP의 업그레이드 의미론을 사용하는 정규 HTTP 요청을 전송하여 프로토콜을 변경합니다. 그 후에는 서버가 핸드셰이크를 완료할 수 있습니다.

WebSocket 연결은 개방 상태를 유지하며 클라이언트 또는 서버가 매회 새로운 연결을 설정할 필요 없이 서로에게 데이터 프레임을 전송할 수 있습니다.

기본적으로 WebSocket 프로토콜은 정규 WebSocket 연결에 포트 80을, TLS를 통한 WebSocket 연결에는 포트 443을 사용합니다. CloudFront 뷰어 프로토콜 정책 및 프로토콜(사용자 지정 오리진만 해당)에 대해 선택한 옵션은 WebSocket 연결뿐 아니라 HTTP 트래픽에도 적용됩니다.

## WebSocket 요구 사항

WebSocket 요청은 다음 표준 형식으로 RFC 6455를 준수해야 합니다.

Example예제 클라이언트 요청

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: dGhlIHNhbXBsZSBub25jZQ==
Origin: https://example.com
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13
```

Example샘플 서버 응답

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+xOo=
Sec-WebSocket-Protocol: chat
```

WebSocket 연결이 클라이언트나 서버 또는 네트워크 장애로 인해 끊어진 겨우, 클라이언트 애플리케이션이 서버와의 연결을 다시 시작해야 합니다.

## 권장 WebSocket 헤더

WebSocket을 사용할 때 예상치 못한 압축 관련 문제를 방지하려면 오리진 요청 정책에 다음 헤더를 포함하는 것이 좋습니다.

- `Sec-WebSocket-Key`
- `Sec-WebSocket-Version`
- `Sec-WebSocket-Protocol`

- `Sec-WebSocket-Accept`
- `Sec-WebSocket-Extensions`

> **Note**
>
> 현재 CloudFront는 HTTP/1.1 프로토콜을 통한 WebSocket 연결만 지원합니다.

# 허용 목록에 사용할 애니캐스트 고정 IP 요청

CloudFront에서 배포에 사용할 애니캐스트 고정 IP를 요청할 수 있습니다. 애니캐스트 고정 IP 목록에는 IPv4 IP 주소만 포함하거나 IPv4 및 IPv6 IP 주소 둘 다 포함될 수 있습니다. 이러한 고정 IP는 AWS 계정 전용이며 여러 리전에 분산되어 있습니다.

네트워크 공급자의 허용 목록에 추가할 21개의 애니캐스트 고정 IP 주소를 요청하여 애플리케이션에 액세스하는 사용자의 데이터 요금을 면제하거나, 아웃바운드 보안 방화벽 내에서 이러한 고정 IP를 사용하여 승인된 애플리케이션과의 트래픽 교환을 제어할 수 있습니다. 애니캐스트 고정 IP 목록은 하나 이상의 배포와 함께 사용할 수 있습니다.

CloudFront 배포로 직접 apex 도메인(예: example.com)을 라우팅하려는 경우 이 사용 사례용으로 3개의 애니캐스트 고정 IP 주소를 요청할 수 있습니다. 그런 다음 DNS에 A 레코드를 추가하여 apex 도메인이 CloudFront를 가리키도록 합니다.

애니캐스트 고정 IP는 서버 이름 표시(SNI)와 함께 작동합니다. 자세한 내용은 SNI를 사용하여 HTTPS 요청 제공(대부분 클라이언트에 적용) 섹션을 참조하세요.

## 사전 조건

CloudFront 배포에서 애니캐스트 고정 IP 목록을 사용하려면 배포의 가격 등급에 대해 모든 엣지 로케이션 사용을 선택해야 합니다. 요금에 대한 자세한 내용은 CloudFront 요금을 참조하시기 바랍니다.

## 애니캐스트 고정 IP 목록 요청

CloudFront 배포에 사용할 애니캐스트 고정 IP 목록을 요청합니다.

애니캐스트 고정 IP 목록을 요청하려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.

2. 왼쪽 탐색 창에서 고정 IP를 선택합니다.
3. 요청에서 CloudFront 지원 엔지니어링 팀에 문의할 수 있는 링크를 선택합니다.
4. 워크로드 정보(초당 요청 바이트 수 및 초당 요청 수)를 제공합니다.
5. CloudFront 지원 엔지니어링 팀에서는 요청을 검토합니다. 검토 프로세스는 최대 2일이 걸릴 수 있습니다.

요청이 승인되면 애니캐스트 고정 IP 목록을 만들어 하나 이상의 배포와 연결할 수 있습니다.

## 애니캐스트 고정 IP 목록 생성

시작하기 전에 이전 섹션에 설명된 대로 애니캐스트 고정 IP 목록을 요청합니다.

애니캐스트 고정 IP 목록을 만들려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 고정 IP를 선택합니다.
3. 애니캐스트 IP 목록 생성을 선택합니다.
4. 이름에 이름을 입력합니다.
5. 고정 IP 사용 사례에서 적절한 사용 사례를 선택합니다.
6. IP 주소 유형에서 다음 옵션 중 하나를 지정합니다.
   - IPv4 - IPv4 주소 목록만 할당
   - 듀얼 스택 - IPv4 및 IPv6 주소 목록 할당
7. 서비스 약관 및 요금을 검토하고 제출을 선택합니다.

고정 IP 목록이 만들어지면 고정 IP 목록 세부 정보 페이지에서 할당된 IP 주소를 볼 수 있습니다. 배포를 고정 IP 목록과 연결할 수도 있습니다.

## 애니캐스트 고정 IP 목록을 기존 배포와 연결

시작하기 전에 이전 섹션에 설명된 대로 애니캐스트 고정 IP 목록을 요청하고 만듭니다.

다음 배포 설정이 애니캐스트 고정 IP 목록과 호환되는지 확인합니다.

- 요금 계층에는 모든 엣지 로케이션 사용(최상의 성능) 설정이 있습니다.

- IPv6가 활성화된 경우 듀얼 스택 애니캐스트 고정 IP 목록을 연결할 수 있습니다. IPv4 주소만 있는 애니캐스트 정적 IP 목록은 IPv6가 활성화된 배포에 연결할 수 없습니다.

애니캐스트 고정 IP 목록을 기존 배포와 연결하려면 다음과 같이 합니다.

- 다음 중 하나를 수행하세요.
  - 고정 IP 목록 세부 정보 페이지에서 고정 IP 목록을 연결합니다.
    1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
    2. 왼쪽 탐색 창에서 고정 IP를 선택합니다.
    3. 고정 IP 목록의 이름을 선택합니다.
    4. 배포 연결을 선택합니다.
    5. 하나 이상의 배포를 선택하고 배포 연결을 선택합니다.
  - 배포 세부 정보 페이지에서 다음과 같이 고정 IP 목록을 연결합니다.
    1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
    2. 왼쪽 탐색 창에서 배포를 선택합니다.
    3. 배포의 이름을 선택합니다.
    4. 일반 탭의 설정에서 편집을 선택합니다.
    5. 애니캐스트 IP 목록에서 이 배포에 사용할 애니캐스트 고정 IP 목록을 선택합니다.
    6. 변경 사항 저장을 선택합니다.

## 애니캐스트 고정 IP 목록을 새 배포와 연결

시작하기 전에 이전 섹션에 설명된 대로 애니캐스트 고정 IP 목록을 요청하고 만듭니다.

애니캐스트 고정 IP 목록을 새 배포와 연결하려면 다음과 같이 합니다.

- 새 배포를 생성합니다. 자세한 내용은 콘솔에서 CloudFront 배포 생성 섹션을 참조하세요. 설정에서 애니캐스트 고정 IP 목록을 사용하려면 다음을 선택해야 합니다.
  - 애니캐스트 IP 목록의 드롭다운 목록에서 애니캐스트 고정 IP 목록을 선택합니다.
  - 가격 분류에서 모든 엣지 로케이션에서 사용(최고의 성능)을 선택합니다.

- 참고: 애니캐스트 고정 IP가 IPv4만 사용하고 듀얼 스택은 사용하지 않는 경우 IPv6에서 끄기를 선택합니다.

배포 생성을 완료합니다. 필요에 따라 애니캐스트 고정 IP 목록에 필요하지 않은 다른 설정 및 구성을 선택할 수 있습니다.

애니캐스트 고정 IP 목록과 관련된 할당량에 대한 자세한 내용은 AWS 일반 참조의 Amazon CloudFront endpoints and quotas를 참조하시기 바랍니다.

## 애니캐스트 고정 IP 목록을 연결 그룹과 연결

시작하기 전에 이전 섹션에 설명된 대로 애니캐스트 고정 IP 목록을 요청하고 만듭니다.

애니캐스트 고정 IP 목록을 새 연결 그룹과 연결하려면

1. 설정에서 연결 그룹을 활성화했는지 확인합니다.
2. 연결 그룹을 생성합니다. 자세한 내용을 알아보려면 사용자 지정 연결 그룹 생성 섹션을 참조하세요.
3. 설정에서 애니캐스트 고정 IP 목록을 사용하려면 다음을 선택해야 합니다.
   - 애니캐스트 IP 목록의 드롭다운 목록에서 애니캐스트 고정 IP 목록을 선택합니다.
4. 연결 그룹 생성을 완료합니다.

> **Note**
> 애니캐스트 고정 IP가 IPv4만 사용하고 듀얼 스택은 사용하지 않는 경우 IPv6에서 끄기를 선택합니다.

애니캐스트 고정 IP 목록과 관련된 할당량에 대한 자세한 내용은 Amazon Web Services 일반 참조의 Amazon CloudFront endpoints and quotas를 참조하시기 바랍니다.

## 애니캐스트 고정 IP 목록 업데이트

애니캐스트 고정 IP 주소를 생성하고 배포에 연결한 후 애니캐스트 고정 IP 목록의 IP 주소 유형을 변경할 수 있습니다.

애니캐스트 고정 IP 목록을 업데이트하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 고정 IP를 선택합니다.
3. 고정 IP 목록의 이름을 선택합니다.
4. 편집을 선택합니다.
5. IP 주소 유형에서 다음 옵션 중 하나를 지정합니다.
   - IPv4 - IPv4 주소 목록만 할당
   - 듀얼 스택 - IPv4 및 IPv6 주소 목록 할당

   > **Note**
   > 연결된 배포에서 이미 IPv6를 활성화한 경우 IPv4를 선택할 수 없습니다. 이렇게 하려면 애니캐스트 고정 IP의 IP 주소 유형을 업데이트하기 전에 IPv6를 비활성화합니다. 자세한 내용은 CloudFront 배포에 IPv6 활성화 섹션을 참조하세요.

6. 제출을 선택하여 변경 사항을 저장하고 애니캐스트 고정 IP 목록을 업데이트합니다.

## IPAM을 사용하여 CloudFront로 자체 IP 가져오기

이 자습서에서는 IPAM을 사용하여 CloudFront 애니캐스트 정적 IP 목록에 대한 BYOIP CIDR을 관리하는 방법을 알아봅니다.

주제


- 애니캐스트 고정 IP의 BYOIP란 무엇입니까?
- 이 기능을 사용하는 이유는 무엇인가요?
- 사전 조건
- 1단계: 애니캐스트 고정 IP 목록 요청
- 2단계: 애니캐스트 고정 IP 목록 생성
- 3단계: CloudFront 배포 생성
- 4단계: CloudFront 리소스와 연결
- 5단계: 마이그레이션 준비

- 6단계: CIDR을 전역에 보급

## 애니캐스트 고정 IP의 BYOIP란 무엇입니까?

CloudFront는 글로벌 서비스에 대한 IPAM의 BYOIP를 통해 자체 IPv4 주소 가져오기를 지원합니다. IPAM의 통합 인터페이스를 통해 고객은 자체 IP 주소(BYOIP)를 사용하여 전용 IP 주소 풀을 생성하고 CloudFront 배포에 할당하는 동시에 AWS 전 세계 콘텐츠 전송 네트워크를 활용하여 애플리케이션과 콘텐츠를 제공할 수 있습니다. IP 주소는 애니캐스트 라우팅을 사용하여 여러 CloudFront 엣지 로케이션에서 동시에 광고됩니다.

## 이 기능을 사용하는 이유는 무엇인가요?

허용 목록에서 네트워크 액세스를 제어하여 다음을 수행합니다.

- 승인된 뷰어에 대한 데이터 요금을 면제하기 위해 네트워크 공급자와의 IP 주소 허용 목록
- 승인된 애플리케이션으로만 트래픽을 제한하도록 아웃바운드 보안 방화벽 구성

### 운영 및 마이그레이션 간소화

- 고정 IP를 가리키는 A 레코드를 추가하여 Apex 도메인(example.com)을 CloudFront로 직접 라우팅
- IP 인프라 또는 방화벽 구성을 업데이트하지 않고 다른 CDN에서 마이그레이션
- 파트너 및 클라이언트와 기존 IP 허용 목록 유지
- 여러 CloudFront 배포에서 단일 애니캐스트 고정 IP 목록 공유

### 일관된 브랜딩

- AWS로 이동할 때 일관된 브랜딩을 위해 기존 IP 주소 공간을 유지합니다.

## 사전 조건

CloudFront 배포에서 애니캐스트 고정 IP 목록을 사용하려면 배포의 가격 등급에 대해 모든 엣지 로케이션 사용을 선택해야 합니다. 요금에 대한 자세한 내용은 CloudFront 요금을 참조하시기 바랍니다. Bring Your Own IP(BYOIP)의 경우 배포 또는 연결 그룹에 대해 IPv6도 비활성화해야 합니다.

시작하기 전에 다음 단계를 완료합니다.

- IPAM 설정: IPAM을 계정과 통합 및 IPAM 생성을 참조하세요.

- 도메인 확인: 도메인 제어를 확인합니다.
- 최상위 풀 생성: IPAM에 자체 IPv4 CIDR 가져오기의 1~2단계를 따릅니다.
- CloudFront에서 사용할 글로벌 로캘로 IPAM 풀을 생성합니다. 자세한 내용은 IPAM을 사용하여 자체 IP를 CloudFront로 가져오기를 참조하세요.

> **Note**
>
> 3개의 /24 IPv4 CIDR 블록이 필요합니다.

## 1단계: 애니캐스트 고정 IP 목록 요청

CloudFront 배포에 사용할 애니캐스트 고정 IP 목록을 요청합니다.

애니캐스트 고정 IP 목록을 요청하려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 고정 IP를 선택합니다.
3. 요청에서 CloudFront 지원 엔지니어링 팀에 문의할 수 있는 링크를 선택합니다.
4. 워크로드 정보(초당 요청 바이트 수 및 초당 요청 수)를 제공합니다.
5. CloudFront 지원 엔지니어링 팀에서는 요청을 검토합니다. 검토 프로세스는 최대 2일이 걸릴 수 있습니다.
6. 요청이 승인되면 애니캐스트 고정 IP 목록을 만들어 하나 이상의 배포와 연결할 수 있습니다.

## 2단계: 애니캐스트 고정 IP 목록 생성

시작하기 전에 이전 섹션에 설명된 대로 애니캐스트 고정 IP 목록을 요청합니다.

애니캐스트 고정 IP 목록을 만들려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 고정 IP를 선택합니다.
3. 애니캐스트 IP 목록 생성을 선택합니다.
4. 이름에 이름을 입력합니다.

5. 고정 IP 사용 사례에서 사용 사례로 BYOIP를 선택합니다.

다음 단계는 표준 리전 BYOIP 프로세스와 다르며 글로벌 서비스에 대한 패턴을 설정합니다.

AWS CLI

최신 버전의 AWS CLI를 설치 또는 업데이트합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

1. CIDR 블록이 프로비저닝된 IPAM 풀의 IpamPoolArn을 검색합니다. 자세한 내용은 AWS CLI만 사용하여 IPAM으로 고유 퍼블릭 IPv4 CIDR 가져오기를 참조하세요.
2. CIDR 블록 및 IPAM 구성을 사용하여 애니캐스트 IP 목록을 생성합니다.

```
aws cloudfront create-anycast-ip-list \
    --name byoip-aip-1 \
    --ip-count 3 \
    --region us-east-1 \
    --ip-address-type ipv4 \
    --ipam-cidr-configs
 '[{"Cidr":"1.1.1.0/24","IpamPoolArn":"arn:aws:ec2::123456789012:ipam-pool/ipam-
pool-005d58a8aa8147abc"},
{"Cidr":"2.2.2.0/24","IpamPoolArn":"arn:aws:ec2::123456789012:ipam-pool/ipam-
pool-005d58a8aa8147abc"},
{"Cidr":"3.3.3.0/24","IpamPoolArn":"arn:aws:ec2::123456789012:ipam-pool/ipam-
pool-005d58a8aa8147abc"}]'
```

> **Note**
> 풀에서 특정 IP 주소를 선택할 수 없습니다. CloudFront는 이를 자동으로 수행합니다.

## 3단계: CloudFront 배포 생성

CloudFront의 경우 지침에 따라 표준 배포를 생성하거나 다중 테넌트 배포를 사용할 수 있습니다.

## 4단계: CloudFront 리소스와 연결

- 애니캐스트 고정 IP 목록을 기존 배포와 연결
- 애니캐스트 고정 IP 목록을 새 배포와 연결

- 애니캐스트 고정 IP 목록을 연결 그룹과 연결

### 5단계: 마이그레이션 준비

자세한 내용은 Amazon VPC 사용 설명서의 4단계: 마이그레이션 준비를 참조하세요.

### 6단계: CIDR을 전역에 보급

자세한 내용은 Amazon VPC 사용 설명서의 5단계: CIDR 글로벌 알림을 참조하세요.

## CloudFront 배포에 gRPC 사용

Amazon CloudFront는 HTTP/2를 기반으로 구축된 오픈 소스 원격 프로시저 호출(RPC) 프레임워크인 gRPC를 지원합니다. gRPC는 양방향 스트리밍 및 바이너리 프로토콜 버퍼 페이로드를 제공하므로 지연 시간이 짧은 통신이 필요한 애플리케이션에 적합합니다.

CloudFront는 gRPC 요청을 수신하여 오리진에 직접 프록시 처리합니다. CloudFront를 사용하여 다음 4가지 유형의 gRPC 서비스를 프록시 처리할 수 있습니다.

- 단항 RPC
- 서버 스트리밍 RPC
- 클라이언트 스트리밍 RPC
- 양방향 스트리밍 RPC

## CloudFront에서 gRPC가 작동하는 방식

CloudFront에서 gRPC를 구성하려면 gRPC 서비스를 제공하는 오리진을 배포의 오리진으로 설정합니다. 비 gRPC 서비스와 gRPC 서비스를 모두 제공하는 오리진을 사용할 수 있습니다. CloudFront는 수신 요청이 gRPC 요청인지 아니면 `Content-Type` 헤더를 기반으로 하는 HTTP/HTTPS 요청인지 확인합니다. 요청의 `Content-Type` 헤더 값이 `application/grpc`인 경우 요청이 gRPC 요청으로 간주되며 CloudFront는 해당 요청을 오리진으로 프록시 처리합니다.

> **Note**
>
> 배포에서 gRPC 요청을 처리하도록 사용 설정하려면 HTTP/2를 지원되는 HTTP 버전 중 하나로 포함하고 `POST`를 비롯한 HTTP 메서드를 허용해야 합니다. CloudFront는 보안(HTTPS 기반) gRPC 연결만 지원하므로 gRPC 오리진 엔드포인트는 HTTPS를 지원하도록 구성되어야

합니다. gRPC는 엔드 투 엔드 HTTPS만 지원합니다. 사용자 지정 오리진을 사용하는 경우 프로토콜 설정이 HTTPS를 지원하는지 확인합니다.

배포에서 gRPC 지원을 사용 설정하려면 다음 단계를 완료합니다.

1. 배포의 캐시 동작을 업데이트하여 POST 메서드를 포함한 HTTP 메서드를 허용합니다.
2. POST 메서드를 선택한 후 나타나는 gRPC 확인란을 선택합니다.
3. HTTP/2를 지원되는 HTTP 버전 중 하나로 지정합니다.

자세한 내용은 다음 항목을 참조하세요.

- HTTP/2를 통한 gRPC 요청 허용
- Amazon CloudFront API 참조의 GrpcConfig

gRPC는 캐시할 수 없는 API 트래픽에만 사용되므로 캐시 구성은 gRPC 요청에 영향을 주지 않습니다. gRPC 오리진으로 전송된 gRPC 요청에 사용자 지정 헤더를 추가하는 데 오리진 요청 정책을 사용할 수 있습니다. CloudFront와 함께 AWS WAF를 사용하여 gRPC 배포에 대한 액세스를 관리하고, 봇을 제어하고, 웹 익스플로잇으로부터 gRPC 애플리케이션을 보호할 수 있습니다. CloudFront gRPC는 CloudFront Functions을 지원합니다.

HTTPS 상태 외에도 gRPC 응답과 함께 grpc-status를 받게 됩니다. grpc-status에 사용할 수 있는 값 목록은 Status codes and their use in gRPC를 참조하시기 바랍니다.

> **참고**
>
> gRPC는 다음의 CloudFront 기능을 지원하지 않습니다.
>
> - 사용자 지정 오류 응답
> - gRPC는 POST 메서드를 사용하므로 오리진 장애 조치는 gRPC에서 지원되지 않습니다. CloudFront는 최종 사용자 요청의 HTTP 메서드가 GET, HEAD 또는 OPTIONS인 경우에만 보조 오리진으로 장애 조치합니다.
> - CloudFront는 gRPC 요청을 오리진으로 직접 프록시 처리하고 리전 엣지 캐시(REC)를 우회합니다. gRPC는 REC를 우회하기 때문에 gRPC는 Lambda@Edge 또는 Origin Shield를 지원하지 않습니다.
> - gRPC는 AWS WAF 요청 본문 검사 규칙을 지원하지 않습니다. 배포의 웹 ACL에서 이러한 규칙을 사용 설정한 경우 gRPC를 사용하는 모든 요청은 요청 본문 검사 규칙을 무시합니다.

모든 다른 AWS WAF 규칙은 계속 적용됩니다. 자세한 내용은 배포에 AWS WAF 활성화 섹션을 참조하세요.

# CloudFront에서 공유 리소스 작업

Amazon CloudFront는 AWS Resource Access Manager(AWS RAM)와 통합되어 리소스 공유를 가능하게 합니다. AWS RAM은 일부 CloudFront 리소스를 다른 AWS 계정과 공유하거나 AWS Organizations를 통해 공유할 수 있도록 합니다. AWS RAM을 사용하여 리소스 공유로 생성한 사용자 소유 리소스를 공유할 수 있습니다. 리소스 공유는 공유할 리소스와 공유 대상 소비자를 지정합니다. 소비자에는 다음이 포함될 수 있습니다.

- AWS Organizations의 조직 내부 또는 외부의 특정 AWS 계정.
- AWS Organizations에서 조직 내부의 조직 단위
- AWS Organizations의 전체 조직

AWS RAM에 대한 자세한 내용은 AWS RAM 사용 설명서를 참조하세요.

이 항목에서는 소유한 리소스를 공유하는 방법과 공유 리소스를 사용하는 방법을 설명합니다.

내용


- 리소스 공유를 위한 사전 조건
- VPC 오리진 공유
- 공유 VPC 오리진 사용
- 공유 VPC 오리진 식별
- 공유 VPC 오리진 공유 해제
- 공유 VPC 오리진에 대한 책임 및 권한
- 결제 및 측정
- 공유 리소스 할당량


## 리소스 공유를 위한 사전 조건

- 리소스 공유에 대한 읽기 전용 액세스 권한을 부여하려면 AWSRAMDefaultPermissionCloudFront 관리형 정책이 있어야 합니다. 자세한 내용은 AWSRAMDefaultPermissionCloudFront 섹션을 참조하세요.
- VPC 오리진을 공유하려면 AWS 계정에서 이를 소유하고 있어야 합니다. 즉, 계정에서 리소스를 할당하거나 프로비저닝해야 합니다. 자신과 공유된 리소스는 공유할 수 없습니다.

- AWS Organizations의 조직 또는 조직 단위와 리소스를 공유하려면, AWS Organizations과의 공유를 활성화해야 합니다. 자세한 내용은 AWS RAM 사용 설명서의 AWS Organizations과(와) 공유 활성화를 참조하세요.

# VPC 오리진 공유

> **Note**
>
> 현재 CloudFront는 VPC 오리진 공유를 지원합니다. 아직 생성하지 않은 경우 VPC 오리진으로 액세스 제한 섹션을 참조하세요.

소유한 VPC 오리진을 다른 AWS 계정과 공유하는 경우 해당 리소스를 CloudFront 배포의 오리진으로 사용할 수 있습니다.

VPC 오리진을 공유하려면 리소스 공유에 추가해야 합니다. 리소스 공유는 AWS 계정 전반에서 리소스를 공유할 수 있게 해주는 AWS RAM 리소스입니다.

리소스 공유는 다음을 지정합니다.

- 공유하려는 리소스
- 공유 대상 소비자
- 리소스에 대한 권한을 결정하는 서비스의 관리형 정책

CloudFront 콘솔을 사용하여 VPC 오리진을 공유하면 기존 리소스 공유에 추가합니다. 리소스 공유가 아직 없는 경우 CloudFront 콘솔에서 VPC 오리진을 공유할 때 리소스 공유를 생성할 수 있습니다. AWS RAM 콘솔 또는 AWS CLI를 사용하여 별도로 생성할 수도 있습니다.

VPC 오리진을 다른 AWS 계정 및 AWS Organizations와 공유할 수 있습니다.

- 리소스를 AWS 조직과 공유하는 경우 해당 특정 조직의 모든 소비자가 VPC 오리진에 액세스할 수 있습니다.
- AWS 계정 또는 소속되지 않은 조직과 공유하는 경우 소비자는 리소스 공유를 수락하라는 초대를 받게 됩니다. 수락되면 VPC 오리진을 사용할 수 있습니다.

CloudFront 콘솔, AWS RAM 콘솔 또는 AWS CLI를 사용하여 소유한 VPC 오리진을 공유할 수 있습니다.

CloudFront 콘솔을 사용하여 리소스 공유를 생성하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 VPC 오리진을 선택합니다.
3. 하나 이상의 리소스를 선택하고 VPC 오리진 공유를 선택합니다.
4. 리소스 공유 생성을 선택합니다.
5. 이름에 리소스 공유의 이름을 입력합니다.
6. 위탁자 유형에서 다음 옵션 중 하나를 선택합니다.
   - AWS 계정 - 특정 AWS 계정에 대한 액세스 권한을 부여합니다.
   - 조직 단위 - 특정 조직 단위(OU)에 대한 액세스 권한을 부여합니다.
   - 조직 - 하위 OU 및 AWS 계정을 포함하여 전체 조직에 대한 액세스 권한을 부여합니다.

   a. AWS 계정를 선택한 경우 계정 ID 번호를 입력합니다. 새 계정 추가를 선택하여 최대 5개의 AWS 계정을 추가할 수 있습니다.
   b. 조직 단위를 선택한 경우 OU 단위 ARN을 입력합니다. OU는 1개만 입력할 수 있습니다.
   c. 조직을 선택한 경우 조직 ARN을 입력합니다. 하나의 조직만 입력할 수 있습니다.
7. 리소스 공유를 선택합니다.

   기본적으로 CloudFront는 리소스 공유에 AWSRAMDefaultPermissionCloudFront AWS 관리형 정책을 적용합니다. 이 정책은 리소스 공유에 대한 읽기 전용 작업을 허용하므로 소비 계정은 공유 리소스를 업데이트하거나 삭제할 수 없습니다. 리소스 공유에서 이 정책을 편집하거나 제거할 수 없습니다.

   > **Tip**
   > 리소스 공유를 생성한 후 AWS RAM 콘솔에서 AWS 계정을 추가할 수 있습니다. 자세한 내용은 AWS RAM 사용 설명서의 AWS RAM에서 리소스 공유 업데이트를 참조하세요.

CloudFront 콘솔을 사용하여 소유한 VPC 오리진을 공유하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 VPC 오리진을 선택합니다.
3. 리소스를 선택하고 VPC 오리진 공유를 선택합니다.

4. VPC 오리진 공유 페이지에서 이 VPC 오리진을 추가할 기존 리소스 공유를 선택할 수 있습니다.
5. 리소스 공유를 선택합니다.

   리소스 세부 정보 페이지의 공유 대상에서 VPC 오리진이 다음 세부 정보와 공유되는 것을 확인할 수 있습니다.

   - 리소스 공유 이름
   - 공유 상태
   - 최근 수정 날짜

리소스 공유를 생성하고 소비 계정과 공유한 후 12시간 내에 초대를 수락해야 합니다. 자세한 내용은 AWS RAM 사용 설명서의 공유 리소스 초대 수락 및 거부를 참조하세요.

> **Important**
>
> 소비 계정이 CloudFront 배포에 VPC 오리진을 사용할 수 있도록 하려면 VPC 오리진의 Elastic Load Balancing 또는 Amazon EC2 엔드포인트도 제공해야 합니다.

AWS RAM 콘솔을 사용하여 소유한 VPC 오리진을 공유하려면

리소스 공유를 생성한 다음 리소스 공유에 추가할 CloudFront 리소스를 선택합니다. 자세한 내용은 AWS RAM 사용 설명서의 리소스 공유 생성을 참조하세요.

AWS CLI를 사용하여 소유한 VPC 오리진을 공유하려면

create-resource-share 명령을 사용합니다.

# 공유 VPC 오리진 사용

공유 VPC 오리진을 사용하려면 초대를 받는 계정이 리소스 공유를 수락해야 합니다. 미국 동부(버지니아 북부) 리전의 AWS Resource Access Manager 콘솔로 이동하여 보류 중 탭에서 보류 중인 요청을 수락하면 됩니다. 자세한 내용은 AWS RAM 사용 설명서의 공유 리소스 수락을 참조하세요.

리소스 공유를 수락한 후 VPC 오리진을 CloudFront 배포의 오리진으로 사용할 수 있습니다.

공유 VPC 오리진을 사용하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.

2. 탐색 창의 배포에서 다음 중 하나를 수행합니다.
   - 새로운 배포의 경우 배포 생성을 선택합니다.
   - 기존 배포의 경우 배포 ID를 선택합니다.
3. 오리진 유형에서 VPC 오리진을 선택한 다음 공유된 VPC 오리진을 지정합니다.
4. VPC 오리진 엔드포인트에 Amazon EC2 인스턴스 또는 Elastic Load Balancing 로드 밸런서의 프라이빗 DNS 이름 또는 오리진 도메인을 입력합니다. 이 값이 아직 없는 경우 VPC 오리진을 소유한 AWS 계정에서 값을 가져와야 합니다. 이 엔드포인트가 아직 없는 경우 VPC 오리진을 소유한 AWS 계정에서 값을 가져올 수 있습니다.
5. 콘솔 단계의 나머지를 따라 배포를 생성 또는 업데이트합니다.

# 공유 VPC 오리진 식별

소유자와 소비자는 CloudFront 콘솔 및 AWS CLI를 사용하여 공유된 VPC 오리진을 식별할 수 있습니다.

CloudFront 콘솔을 사용하여 공유 VPC 오리진을 식별하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 VPC 오리진을 선택합니다. 소유자 ID 열을 사용하여 리소스가 속한 AWS 계정을 식별할 수 있습니다.
3. 리소스를 선택합니다.
4. 리소스 세부 정보 페이지의 공유 대상에서 VPC 오리진이 다음 세부 정보와 공유되는 것을 확인할 수 있습니다.
   - 리소스 공유 이름
   - 공유 상태
   - 최근 수정 날짜

# 공유 VPC 오리진 공유 해제

리소스를 공유 해제하면 AWS 계정(소비 계정)이 더 이상 새 배포에 해당 리소스를 사용하거나 기존 배포를 업데이트할 수 없습니다.

> **Note**
>
> 리소스를 공유 해제하면 해당 리소스를 계속 사용 중인 기존 배포는 활성 상태로 유지되며 트래픽을 계속 처리합니다. 그러나 공유되지 않은 리소스가 오리진으로 제거될 때까지 이러한 배포를 편집할 수 없습니다. 공유를 해제하기 전에 모든 소비 계정이 공유되지 않은 리소스 사용을 중지하는지 확인하는 것이 좋습니다.

소유하고 있는 공유된 VPC 오리진을 공유 해제하려면 리소스 공유에서 제거해야 합니다. 이를 위해 CloudFront 콘솔, AWS RAM 콘솔 또는 AWS CLI를 사용할 수 있습니다.

CloudFront 콘솔을 사용하여 소유한 공유 VPC 오리진의 공유를 해제하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 VPC 오리진을 선택합니다.
3. 리소스를 선택하고 공유 해제를 선택합니다.
4. 리소스 공유 해제 대화 상자의 세부 정보를 검토한 다음 공유 해제를 선택합니다. 나열된 보안 주체는 더 이상 공유 리소스에 액세스할 수 없습니다.

AWS RAM 콘솔을 사용하여 소유한 공유 VPC 오리진의 공유를 해제하려면

AWS RAM 사용 설명서에서 리소스 공유 업데이트를 참조하세요.

AWS CLI를 사용하여 소유한 공유 VPC 오리진의 공유를 해제하려면

disassociate-resource-share 명령을 사용합니다.

# 공유 VPC 오리진에 대한 책임 및 권한

## 소유자에 대한 권한

리소스 소유 계정인 경우 공유를 해제하거나 삭제하기 전에 소비하는 계정이 리소스 사용을 중지해야 합니다.

## 소비자에 대한 권한

계정은 공유 리소스를 CloudFront 배포의 오리진으로 사용할 수 있지만 리소스를 편집하거나 삭제할 수는 없습니다. 기본적으로 AWSRAMDefaultPermissionCloudFront AWS 관리형 정책은 공유 계정(리소스를 소유한 계정)의 리소스 공유에 적용됩니다.

## AWSRAMDefaultPermissionCloudFront

CloudFront에서 리소스 공유를 생성하면 CloudFront는 AWSRAMDefaultPermissionCloudFront AWS 관리형 정책을 사용하여 리소스 공유에 적용합니다. 이 정책은 리소스 소유자에서 소비 계정으로 공유할 수 있는 읽기 전용 권한을 CloudFront 리소스에 부여합니다.

AWS RAM의 권한 관리에 대한 자세한 내용은 AWS Resource Access Manager 사용 설명서의 AWS RAM의 권한 관리를 참조하세요.

# 결제 및 측정

VPC 오리진을 다른 AWS 계정과 공유하는 데는 추가 요금이 부과되지 않습니다. 공유 VPC 오리진을 사용하는 배포의 트래픽 사용 비용은 배포를 소유한 소비 계정으로 이동합니다.

# 공유 리소스 할당량

CloudFront는 AWS RAM에 지정된 것과 동일한 리소스 공유 할당량을 사용합니다. CloudFront 콘솔에서 최대 5개의 AWS 계정, 1개의 OU 또는 1개의 조직을 추가할 수 있습니다. 더 추가하려면 AWS RAM 콘솔 또는 AWS RAM API를 사용합니다.

자세한 내용은 AWS RAM 사용 안내서의 Service quotas for AWS RAM를 참조하세요.

# 캐싱 및 가용성

CloudFront를 사용하여 오리진 서버에서 직접 응답해야 하는 요청의 수를 줄일 수 있습니다. CloudFront 캐싱을 사용하면 사용자에게 더 가까운 CloudFront 엣지 로케이션에서 더 많은 객체가 제공됩니다. 따라서 오리진 서버에 걸리는 부하가 줄어들고 지연 시간이 단축됩니다.

엣지 캐시에서 CloudFront가 처리할 수 있는 요청이 많을수록 객체의 최신 버전 또는 고유한 버전을 얻기 위해 CloudFront가 오리진에 전달해야 하는 최종 사용자 요청 수가 줄어듭니다. 오리진에 최대한 적게 요청하도록 CloudFront를 최적화하려면 CloudFront Origin Shield 사용을 고려해 보십시오. 자세한 내용은 Amazon CloudFront Origin Shield 사용 단원을 참조하세요.

모든 요청과 비교하여 CloudFront 캐시에서 직접 처리되는 요청의 비율을 캐시 적중률이라고 합니다. CloudFront 콘솔에서 최종 사용자 요청의 적중, 누락, 오류 백분율을 확인할 수 있습니다. 자세한 내용은 CloudFront 캐시 통계 보고서 보기 단원을 참조하세요.

적중률에 영향을 주는 요인은 매우 많습니다. CloudFront 캐시에서 직접 처리되는 요청의 비율 증가(캐시 적중률)의 지침을 수행하여 CloudFront 배포 구성을 조정하고 적중률을 높일 수 있습니다.

CloudFront가 제공하게 하려는 콘텐츠를 추가하고 제거하는 방법을 알아보려면 CloudFront가 배포하는 콘텐츠 추가, 제거 또는 교체 단원을 참조하십시오.

주제


- CloudFront 캐시에서 직접 처리되는 요청의 비율 증가(캐시 적중률)
- Amazon CloudFront Origin Shield 사용
- CloudFront 오리진 장애 조치를 통한 고가용성 최적화
- 콘텐츠가 캐시에 유지되는 기간(만료) 관리
- 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱
- 쿠키 기반의 콘텐츠 캐싱
- 요청 헤더 기반의 콘텐츠 캐싱


## CloudFront 캐시에서 직접 처리되는 요청의 비율 증가(캐시 적중률)

오리진 서버에서 콘텐츠를 제공하는 대신 CloudFront 캐시에서 직접 콘텐츠를 제공하는 최종 사용자 요청의 비율을 높임으로써 성능을 향상시킬 수 있습니다. 이를 캐시 적중률 개선이라고 합니다.

다음 섹션에서는 캐시 적중률을 개선하는 방법을 설명합니다.

주제


- CloudFront에서 객체를 캐싱하는 시간 지정
- Origin Shield 사용
- 쿼리 문자열 파라미터 기반 캐싱
- 쿠키 값 기반 캐싱
- 요청 헤더 기반 캐싱
- 압축이 불필요할 때 Accept-Encoding 헤더 제거
- HTTP를 통한 미디어 콘텐츠 제공


## CloudFront에서 객체를 캐싱하는 시간 지정

캐시 적중률을 높이기 위해 객체에 Cache-Control max-age 명령을 추가하도록 오리진을 구성하고 `max-age`에 실질적으로 가장 긴 값을 지정할 수 있습니다. 캐시 주기가 짧을수록 CloudFront에서는 요청을 오리진에 더 자주 보내서 객체가 변경되었는지 여부를 확인하고 최신 버전을 가져옵니다. `max-age`를 `stale-while-revalidate` 및 `stale-if-error` 지시문으로 보완하여 특정 조건에서 캐시 적중률을 더욱 개선할 수 있습니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 섹션을 참조하세요.

## Origin Shield 사용

CloudFront Origin Shield는 오리진 앞에 캐싱의 추가 계층을 제공하기 때문에 CloudFront 배포의 캐시 적중률을 개선하는 데 도움이 됩니다. Origin Shield를 사용하면 CloudFront의 모든 캐싱 계층에서 오리진에 이르는 모든 요청은 단일 위치에서 이루어집니다. CloudFront는 Origin Shield의 단일 오리진 요청을 사용하여 각 객체를 검색할 수 있으며 CloudFront 캐시의 다른 모든 계층(엣지 로케이션 및 리전 엣지 캐시)은 Origin Shield에서 객체를 검색할 수 있습니다.

자세한 내용은 Amazon CloudFront Origin Shield 사용 단원을 참조하세요.

## 쿼리 문자열 파라미터 기반 캐싱

쿼리 문자열 파라미터를 기반으로 캐싱하도록 CloudFront를 구성한 경우, 다음을 수행하면 캐싱 성능을 개선할 수 있습니다.

- 오리진에서 고유한 객체를 반환하는 쿼리 문자열 파라미터만 전달하도록 CloudFront를 구성합니다.
- 동일 파라미터의 전체 인스턴스에 대해서는 대문자 또는 소문자만 사용합니다. 예를 들어 하나의 요청에 `parameter1=A`가 포함되어 있고 다른 요청에는 `parameter1=a`가 포함된 경우, CloudFront에서는 요청에 `parameter1=A`가 포함되었을 때와 `parameter1=a`가 포함되었을 때 각각 별도의

요청을 오리진에 전달합니다. 그런 다음 CloudFront는 오리진에서 개별적으로 반환된 해당 객체를 별도로 캐싱합니다. 이는 이러한 객체가 실제로는 동일하더라도 마찬가지입니다. `A` 또는 `a`만 사용하는 경우, CloudFront에서는 오리진에 더 적은 요청을 전달하게 됩니다.

- 파라미터를 같은 순서로 나열합니다. 다른 순서로 나열하는 경우, 객체에 대한 하나의 요청에 쿼리 문자열 `parameter1=a&parameter2=b`가 포함되어 있고 같은 객체의 다른 요청에는 `parameter2=b&parameter1=a`가 포함되어 있다면, CloudFront에서는 두 요청을 오리진에 모두 전달하고 해당 객체가 실제로는 동일한 경우라도 이를 개별적으로 캐싱합니다. 파라미터에 언제나 같은 순서를 사용한다면 CloudFront에서는 오리진에 더 적은 요청을 전달하게 됩니다.

자세한 내용은 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱 단원을 참조하세요. CloudFront에서 오리진에 전달한 쿼리 문자열을 검토하려는 경우, CloudFront 로그 파일에서 `cs-uri-query` 열의 값을 확인합니다. 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하세요.

## 쿠키 값 기반 캐싱

쿠키 값을 기반으로 캐싱하도록 CloudFront를 구성한 경우, 다음을 수행하면 캐싱 성능을 개선할 수 있습니다.

- 전체 쿠키를 전달하는 대신 지정된 쿠키만 전달하도록 CloudFront를 구성합니다. CloudFront가 오리진에 전달하도록 구성하는 쿠키의 경우 CloudFront에서 쿠키 이름과 값의 모든 조합을 전달합니다. 그런 다음 오리진이 반환하는 객체를 별도로 캐시합니다(객체가 모두 동일한 경우도 마찬가지).

  예를 들어, 최종 사용자가 각 요청에 두 개의 쿠키를 포함했다고 가정하면, 각 쿠키는 3가지 값을 가질 수 있으며 이 모든 쿠키 값의 조합이 가능합니다. CloudFront에서는 각 객체의 오리진에 최대 9개의 다양한 요청을 전달합니다. 오리진에서 오직 하나의 쿠키를 기반으로 서로 다른 버전의 객체를 반환하는 경우, CloudFront에서는 오리진에 필요한 것 이상의 요청을 전달하게 되며 불필요하게 동일한 버전의 객체를 여러 번 캐싱하게 됩니다.
- 정적 콘텐츠와 동적 콘텐츠의 캐시 동작을 별도로 만들고 오리진에는 동적 콘텐츠에 대해서만 쿠키를 전달하도록 CloudFront를 구성합니다.

  예를 들어, 배포에 대한 하나의 캐시 동작이 있고 `.js` 파일 및 `.css` 파일과 같은 거의 변동이 없는 두 동적 콘텐츠에 대한 배포를 사용 중이라고 가정합니다. CloudFront에서는 쿠키 값을 바탕으로 `.css` 파일의 개별 버전을 캐싱하여 각 CloudFront 엣지 로케이션에서 각각의 새 쿠키 값 또는 쿠키 값들의 조합에 대한 요청을 오리진에 전달하도록 합니다.

  경로 패턴이 `*.css`이고 CloudFront에서 쿠키 값에 따라 캐싱하지 않는 캐시 동작을 만든 경우, CloudFront에서는 `.css` 파일에 대한 요청 중 엣지 로케이션에서 수신하는 지정된 `.css` 파일에 대한 첫 번째 요청과 `.css` 파일이 만료된 후의 첫 번째 요청만 오리진에 전달합니다.

- 가능한 경우, 각 사용자(예: 사용자 ID)별로 쿠키 값이 고유한 동적 콘텐츠에 대해서와 더 작은 수의 고유 값에 따라 달라지는 동적 콘텐츠에 대해서 별도의 캐시 동작을 만듭니다.

자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하세요. CloudFront에서 오리진에 전달한 쿠키를 검토하려는 경우, CloudFront 로그 파일에서 `cs(Cookie)` 열의 값을 확인합니다. 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하세요.

## 요청 헤더 기반 캐싱

요청 헤더를 기반으로 캐싱하도록 CloudFront를 구성한 경우, 다음을 수행하면 캐싱 성능을 개선할 수 있습니다.

- 전체 헤더를 기반으로 전달 및 캐싱하는 대신 지정된 헤더를 기반으로만 전달하고 캐싱하도록 CloudFront를 구성합니다. 지정한 헤더의 경우 CloudFront에서 헤더 이름과 값의 모든 조합을 전달합니다. 그런 다음 오리진이 반환하는 객체를 별도로 캐시합니다(객체가 모두 동일한 경우도 마찬가지).

  > **Note**
  >
  > CloudFront에서는 다음 주제에서 지정한 헤더를 오리진에 항상 전달합니다.
  >
  > - CloudFront에서 요청을 처리하고 Amazon S3 오리진 서버에 요청을 전달하는 방법> CloudFront에서 제거하거나 업데이트하는 HTTP 요청 헤더
  > - CloudFront에서 요청을 처리하고 사용자 지정 오리진 서버에 요청을 전달하는 방법 > HTTP 요청 헤더 및 CloudFront 동작(사용자 지정 및 Amazon S3 오리진)

  요청 헤더에 따라 캐싱하도록 CloudFront를 구성한 경우, CloudFront에서 헤더 값에 따라 객체를 캐싱하는 경우를 제외하고는 CloudFront에서 전달하는 헤더를 변경하지 않습니다.

- 큰 수의 고유 값을 갖는 요청 헤더를 기반으로 캐싱하지 않도록 하십시오.

  예를 들어 사용자 디바이스에 따라 서로 다른 크기의 이미지를 제공하려는 경우, 매우 큰 값을 가질 수 있는 `User-Agent` 헤더를 기반으로 캐싱하도록 CloudFront를 구성하지 마십시오. 대신 CloudFront 디바이스 유형 헤더인 `CloudFront-Is-Desktop-Viewer`, `CloudFront-Is-Mobile-Viewer`, `CloudFront-Is-SmartTV-Viewer` 및 `CloudFront-Is-Tablet-Viewer`를 기반으로 캐싱하도록 CloudFront를 구성하십시오. 또한 태블릿과 데스크톱에 대해 같은 버전의 이미지를 반환하려는 경우, `CloudFront-Is-Tablet-Viewer` 헤더가 아닌 `CloudFront-Is-Desktop-Viewer` 헤더만 전달하세요.

자세한 내용은 요청 헤더 기반의 콘텐츠 캐싱 섹션을 참조하세요.

## 압축이 불필요할 때 `Accept-Encoding` 헤더 제거

압축이 오리진이나 CloudFront에서 지원되지 않거나 압축 자체가 불가한 콘텐츠라서 압축 활성화가 어렵다면 다음과 같이 Custom Origin Header를 설정하는 오리진에 배포의 캐시 동작을 연결하여 캐시 적중률을 높일 수 있습니다.

- Header name(헤더 이름)`Accept-Encoding`:
- Header value(헤더 값): (공백으로 유지)

이 구성을 사용하면 CloudFront가 캐시 키에서 `Accept-Encoding` 헤더를 제거하며 오리진 요청에 해당 헤더를 포함하지 않습니다. 이 구성은 CloudFront가 해당 오리진에서의 배포를 통해 제공하는 모든 콘텐츠에 적용됩니다.

## HTTP를 통한 미디어 콘텐츠 제공

온디맨드 비디오(VOD) 및 스트리밍 비디오 콘텐츠를 최적화하는 방법에 대한 자세한 내용은 CloudFront를 사용한 온디맨드 비디오 및 라이브 스트리밍 비디오 단원을 참조하십시오.

# Amazon CloudFront Origin Shield 사용

CloudFront Origin Shield는 CloudFront 캐싱 인프라의 추가 계층으로 오리진의 부하를 최소화하고 가용성을 높이며 운영 비용을 절감하는 데 도움이 됩니다. CloudFront Origin Shield를 사용하면 다음과 같은 이점을 얻을 수 있습니다.

향상된 캐시 적중률

Origin Shield는 오리진 앞에 캐싱의 추가 계층을 제공하기 때문에 CloudFront 배포의 캐시 적중률을 개선하는 데 도움이 됩니다. Origin Shield를 사용하면 모든 CloudFront의 캐싱 계층에서 오리진에 이르는 모든 요청이 Origin Shield를 통과하여 캐시 적중 가능성을 높입니다. CloudFront는 Origin Shield에서 오리진으로의 단일 오리진 요청을 사용하여 각 객체를 검색할 수 있으며 CloudFront 캐시의 다른 모든 계층(엣지 로케이션 및 리전 엣지 캐시)은 Origin Shield에서 객체를 검색할 수 있습니다.

오리진 부하 감소

Origin Shield는 동일한 객체에 대해 오리진으로 전송되는 동시 요청 수를 추가로 줄일 수 있습니다. Origin Shield의 캐시에 없는 콘텐츠에 대한 요청은 동일한 객체에 대한 다른 요청과 통합되어 오리진에 대한 요청이 하나만 발생합니다. 오리진에서 요청을 적게 처리하면 최대 부하 또는 예기치 않

은 트래픽 급증 시에도 오리진의 가용성을 유지할 수 있으며, JIT(Just-in-time) 패키징, 이미지 변환 및 DTO(데이터 송신)와 같은 비용을 절감할 수 있습니다.

향상된 네트워크 성능

오리진에 대한 대기 시간이 가장 짧은 AWS 리전에서 Origin Shield를 활성화하면 향상된 네트워크 성능을 얻을 수 있습니다. AWS 리전의 오리진의 경우 CloudFront 네트워크 트래픽은 오리진에 이르기까지 처리량이 높은 CloudFront 네트워크에 남아 있습니다. AWS 외부에 있는 오리진의 경우 CloudFront 네트워크 트래픽은 오리진에 대한 연결 대기 시간이 짧은 Origin Shield에 이르기까지 CloudFront 네트워크에 남아 있습니다.

Origin Shield 사용 시 추가 요금이 발생합니다. 자세한 내용은 CloudFront 요금을 참조하십시오.

> **Note**
>
> Origin Shield는 gRPC 요청에서 지원되지 않습니다. gRPC를 지원하는 배포에 Origin Shield가 사용 설정된 경우 gRPC 요청은 계속 작동합니다. 하지만 요청은 Origin Shield를 통과하지 않고 gRPC 오리진으로 직접 프록시됩니다. 자세한 내용은 CloudFront 배포에 gRPC 사용 섹션을 참조하세요.

주제


- Origin Shield 사용 사례
- Origin Shield의 AWS 리전 선택
- Origin Shield 사용
- Origin Shield 비용 추정
- Origin Shield 고가용성
- Origin Shield가 다른 CloudFront 기능과 상호 작용하는 방법


## Origin Shield 사용 사례

CloudFront Origin Shield는 다음과 같은 많은 사용 사례에 도움이 될 수 있습니다.

- 서로 다른 지리적 리전에 분산되어 있는 최종 사용자
- 라이브 스트리밍 또는 즉석 이미지 처리를 위한 JIT(Just-in-time) 패키징을 제공하는 오리진
- 용량 또는 대역폭 제약이 있는 온프레미스 오리진

- 여러 CDN(콘텐츠 전송 네트워크)을 사용하는 워크로드

Origin Shield는 오리진으로 프록시되는 동적 콘텐츠, 캐시 가능성이 낮은 콘텐츠, 자주 요청되지 않는 콘텐츠 등 다른 경우에는 적합하지 않을 수 있습니다.

다음 섹션에서는 다음과 같은 사용 사례에 대한 Origin Shield의 이점에 대해 설명합니다.

사용 사례

- 서로 다른 지리적 리전의 최종 사용자
- 여러 CDN

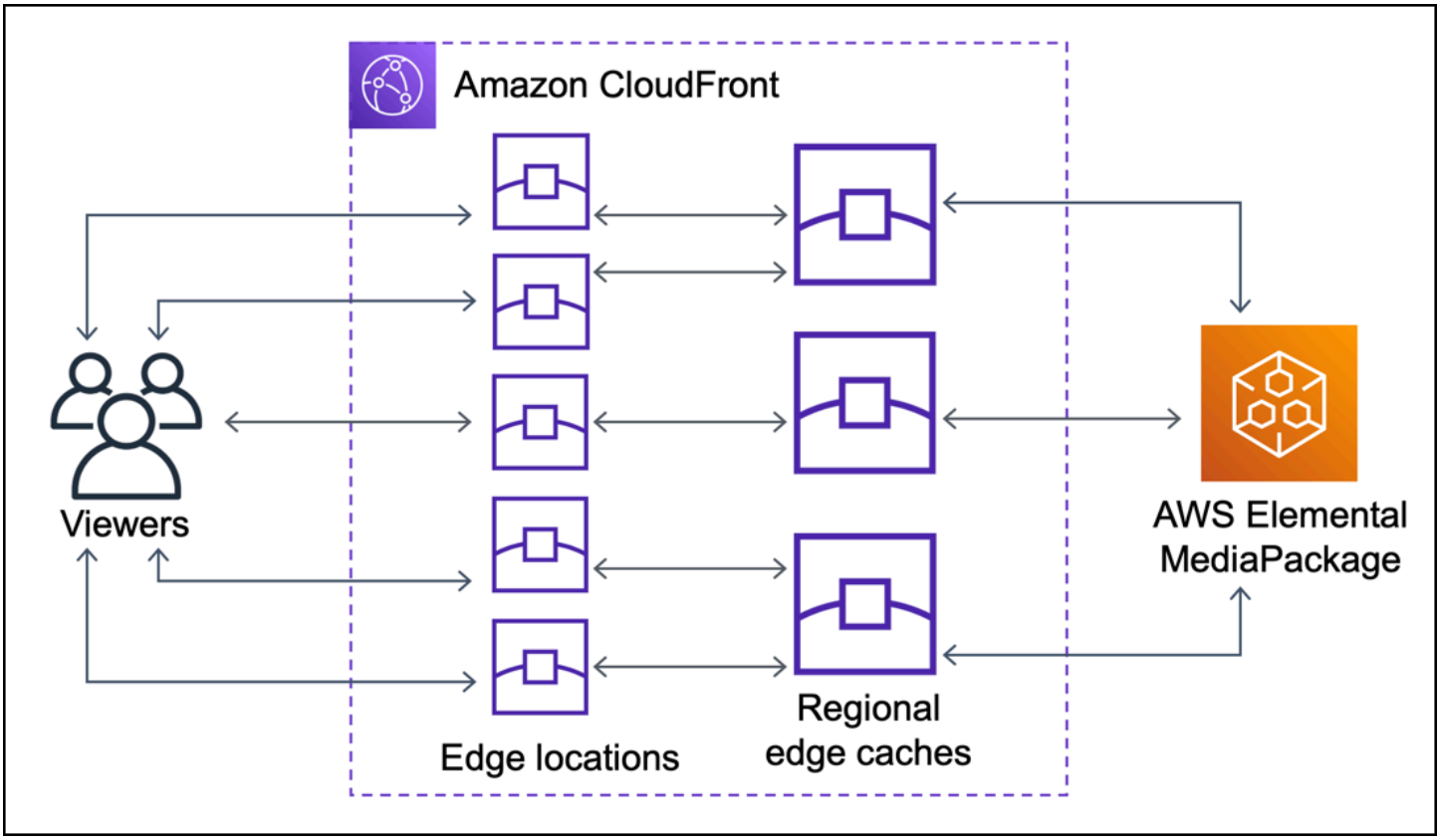
## 서로 다른 지리적 리전의 최종 사용자

Amazon CloudFront를 사용하면 CloudFront가 캐시에서 처리할 수 있는 요청이 오리진으로 이동하지 않기 때문에 본질적으로 오리진의 부하를 줄일 수 있습니다. CloudFront의 엣지 로케이션의 글로벌 네트워크 외에도 리전 엣지 캐시는 미드티어 캐싱 계층 역할을 하여 가까운 지리적 리전의 최종 사용자에 대한 캐시 적중을 제공하고 오리진 요청을 통합합니다. 최종 사용자 요청은 먼저 가까운 CloudFront 엣지 로케이션으로 라우팅되며 객체가 해당 위치에 캐싱되지 않으면 요청이 리전 엣지 캐시로 전송됩니다.

최종 사용자가 서로 다른 지리적 리전에 있는 경우 요청을 서로 다른 리전 엣지 캐시를 통해 라우팅할 수 있으며, 각 캐시에서는 동일한 콘텐츠에 대한 요청을 오리진에 보낼 수 있습니다. 하지만 Origin Shield를 사용하면 리전 엣지 캐시와 오리진 사이에 추가 캐싱 계층이 생깁니다. 모든 리전 엣지 캐시의 모든 요청은 Origin Shield를 통과하여 오리진의 부하를 더욱 줄입니다. 다음 다이어그램에서는 이 개념을 보여줍니다. 다음 다이어그램에서 오리진은 AWS Elemental MediaPackage입니다.

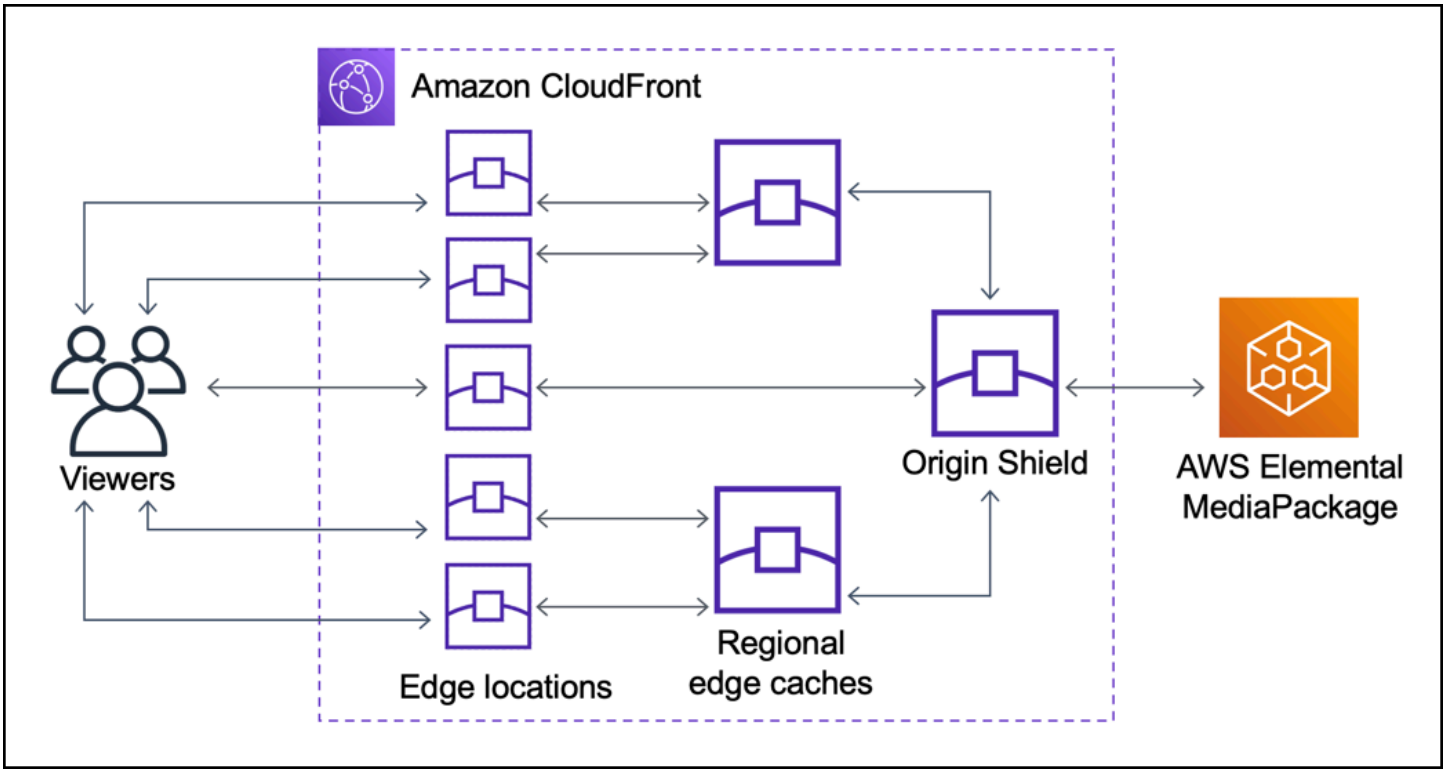
Origin Shield 미사용

Origin Shield를 사용하지 않으면 다음 다이어그램과 같이 오리진에서 동일한 콘텐츠에 대한 중복 요청을 수신할 수 있습니다.




Origin Shield 사용

Origin Shield를 사용하면 다음 다이어그램과 같이 오리진의 부하를 줄일 수 있습니다.




## 여러 CDN

라이브 동영상 이벤트 또는 인기 있는 주문형 콘텐츠를 제공하기 위해 여러 CDN(콘텐츠 전송 네트워크)을 사용할 수 있습니다. 여러 CDN을 사용하면 특정 이점을 얻을 수 있지만 오리진에서는 동일한 콘텐츠에 대해 여러 개의 중복 요청을 수신할 수 있습니다. 각 요청은 서로 다른 CDN 또는 동일한 CDN 내의 다른 위치에서 발생합니다. 이러한 중복 요청은 오리진의 가용성에 부정적인 영향을 미치거나 JIT(Just-in-time) 패키징 또는 DTO(데이터 송신)와 같은 프로세스에 추가적인 운영 비용을 야기할 수 있습니다.

다른 CDN의 오리진으로 CloudFront 배포를 사용하여 Origin Shield를 결합하면 다음과 같은 이점을 얻을 수 있습니다.

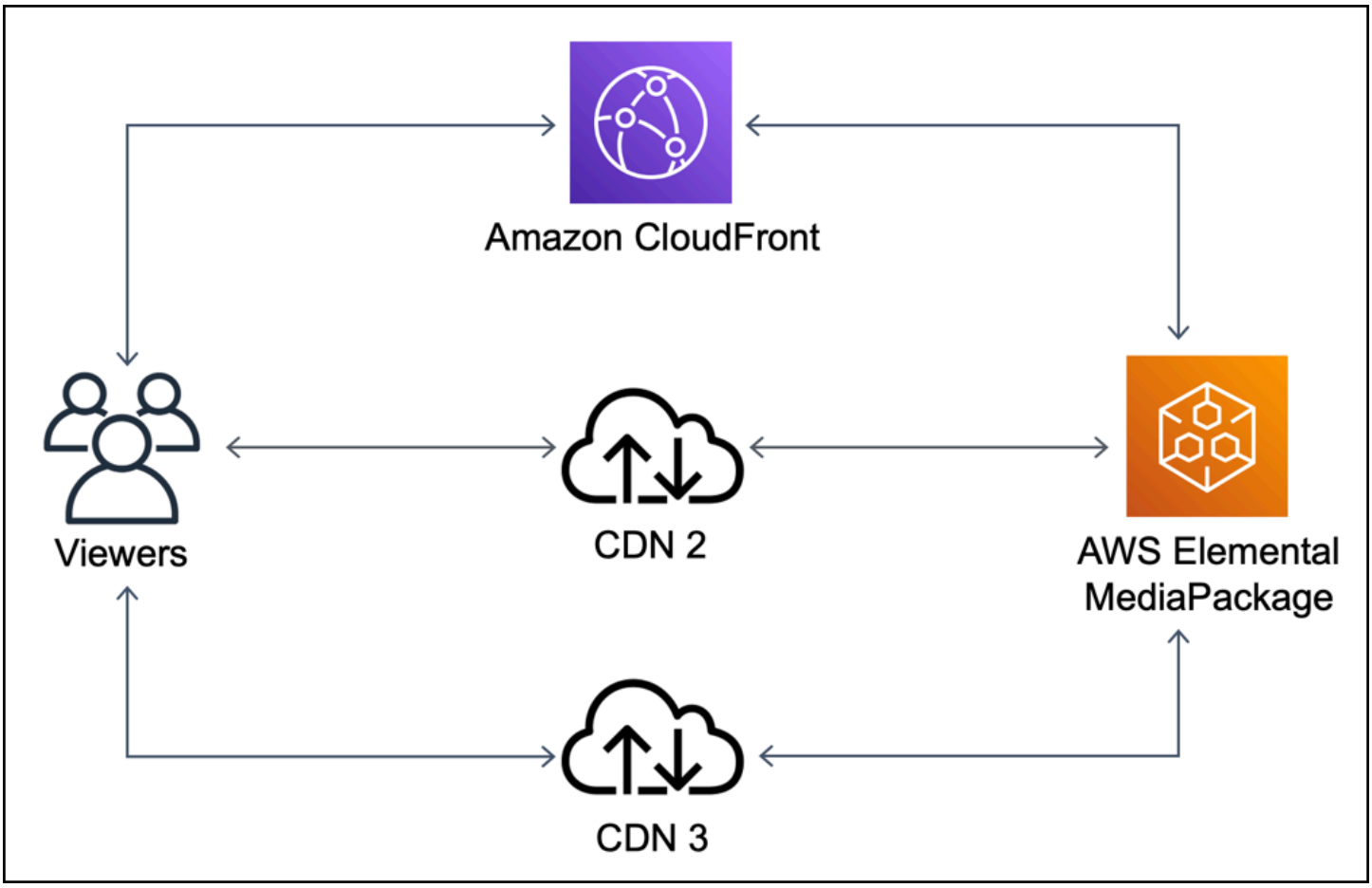
- 오리진에서 수신되는 중복 요청 수가 적기 때문에 여러 CDN을 사용할 때의 부정적인 영향을 줄일 수 있습니다.
- CDN 전반의 공통 캐시 키 및 오리진 방향 기능을 위한 중앙 집중식 관리.
- 네트워크 성능이 향상되었습니다. 다른 CDN의 네트워크 트래픽은 가까운 CloudFront 엣지 로케이션에서 종료되어 로컬 캐시에서 적중을 제공할 수 있습니다. 요청된 객체가 엣지 로케이션 캐시에 없는 경우 오리진에 대한 요청은 오리진에 대한 높은 처리량과 짧은 대기 시간을 제공하는 Origin Shield까지 CloudFront 네트워크에 남아 있습니다. 요청된 객체가 Origin Shield의 캐시에 있는 경우 오리진에 대한 요청은 완전히 방지됩니다.

> **Important**
>
> 다중 CDN 아키텍처에서 Origin Shield를 사용하는 데 관심이 있고 요금을 할인받고 있는 경우 AWS에 문의하거나 AWS 영업 담당자에게 연락하여 자세한 내용을 알아보세요. 추가 요금이 적용될 수 있습니다.

다음 다이어그램은 여러 CDN으로 인기 있는 라이브 동영상 이벤트를 제공할 때 이 구성을 통해 오리진의 부하를 최소화하는 방법을 보여줍니다. 다음 다이어그램에서 오리진은 AWS Elemental MediaPackage입니다.

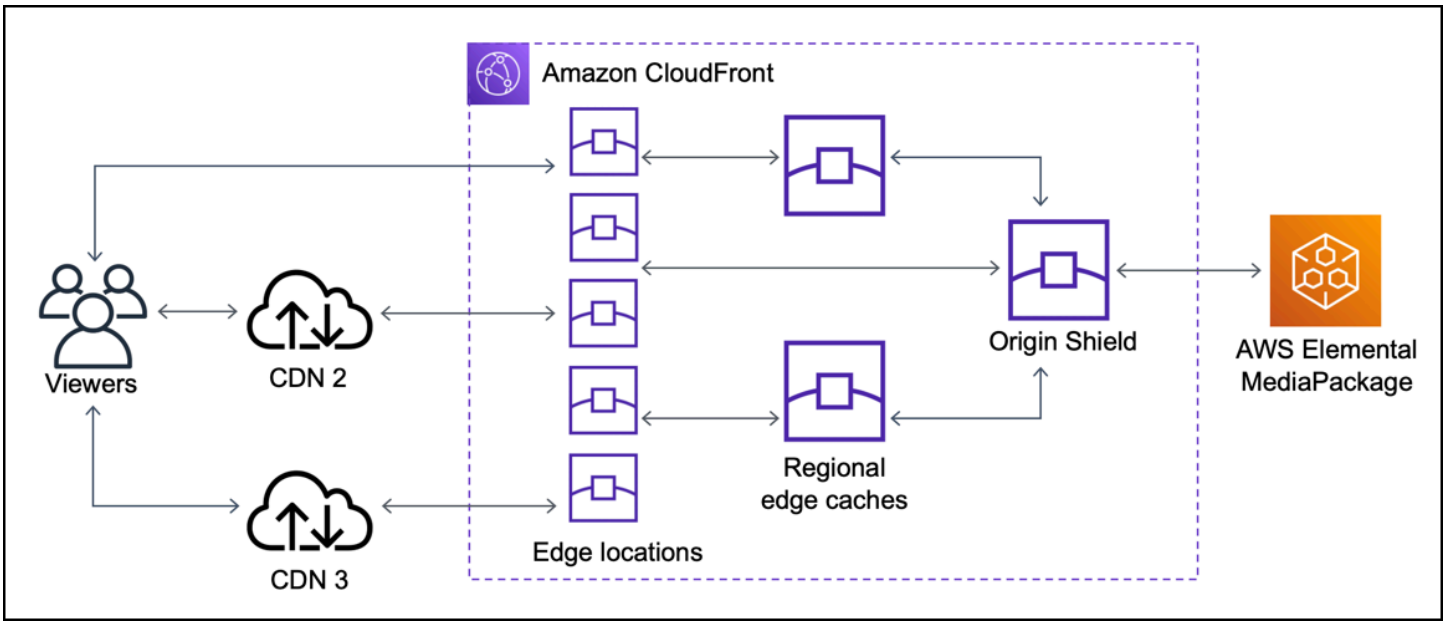
Origin Shield 미사용(여러 CDN)

Origin Shield를 사용하지 않으면 다음 다이어그램과 같이 오리진에서 동일한 콘텐츠에 대해 각각 서로 다른 CDN에서 오는 여러 개의 중복 요청을 수신할 수 있습니다.




Origin Shield 사용(여러 CDN)

Origin Shield를 사용하고 다른 CDN의 오리진으로 CloudFront를 함께 사용하면 다음 다이어그램과 같이 오리진의 부하를 줄일 수 있습니다.




## Origin Shield의 AWS 리전 선택

Amazon CloudFront는 CloudFront에 리전 엣지 캐시가 있는 AWS 리전에서 Origin Shield를 제공합니다. Origin Shield를 활성화하면 Origin Shield에 대해 AWS 리전을 선택합니다. 오리진에 대한 지연 시간이 가장 짧은 AWS 리전을 선택해야 합니다. Origin Shield는 AWS 리전에 있는 오리진 및 AWS에 없는 오리진과 함께 사용할 수 있습니다.

### AWS 리전에 있는 오리진의 경우

오리진이 AWS 리전에 있는 경우 먼저 CloudFront가 Origin Shield를 제공하는 리전에 오리진이 있는지 확인합니다. CloudFront는 다음과 같은 AWS 리전에서 Origin Shield를 제공합니다.

- 미국 동부(오하이오) – `us-east-2`
- 미국 동부(버지니아 북부) – `us-east-1`
- 미국 서부(오레곤) – `us-west-2`
- 아시아 태평양(뭄바이) – `ap-south-1`
- 아시아 태평양(서울) – `ap-northeast-2`
- 아시아 태평양(싱가포르) – `ap-southeast-1`
- 아시아 태평양(시드니) – `ap-southeast-2`
- 아시아 태평양(도쿄) – `ap-northeast-1`

- 유럽(프랑크푸르트) – eu-central-1
- 유럽(아일랜드) – eu-west-1
- 유럽(런던) – eu-west-2
- 남아메리카(상파울루) – sa-east-1
- 중동(UAE) - me-central-1

CloudFront가 Origin Shield를 제공하는 AWS 리전에 오리진이 있는 경우

CloudFront가 Origin Shield를 제공하는 AWS 리전에 오리진이 있는 경우(이전 목록 참조) 오리진과 동일한 리전에서 Origin Shield를 활성화합니다.

CloudFront가 Origin Shield를 제공하는 AWS 리전에 오리진이 없는 경우

CloudFront가 Origin Shield를 제공하는 AWS 리전에 오리진이 없는 경우 다음 표를 참조하여 Origin Shield를 활성화할 리전을 결정합니다.

| 오리진이 다음에 있는 경우 | Origin Shield 활성화 |
| --- | --- |
| 미국 서부(캘리포니아 북부) – us-west-1 | 미국 서부(오레곤) – us-west-2 |
| 아프리카(케이프타운) – af-south-1 | 유럽(아일랜드) – eu-west-1 |
| 아시아 태평양(홍콩) – ap-east-1 | 아시아 태평양(싱가포르) – ap-southeast-1 |
| 캐나다(중부) – ca-central-1 | 미국 동부(버지니아 북부) – us-east-1 |
| 유럽(밀라노) – eu-south-1 | 유럽(프랑크푸르트) – eu-central-1 |
| 유럽(파리) – eu-west-3 | 유럽(런던) – eu-west-2 |
| 유럽(스톡홀름) – eu-north-1 | 유럽(런던) – eu-west-2 |
| 중동(바레인) – me-south-1 | 아시아 태평양(뭄바이) – ap-south-1 |

## 외부에 있는 오리진의 경우AWS

Origin Shield는 온프레미스에 있거나 AWS 리전에 있지 않은 오리진과 함께 사용할 수 있습니다. 이 경우 오리진에 대한 지연 시간이 가장 짧은 AWS 리전에서 Origin Shield를 활성화합니다. 오리진에 대한

지연 시간이 가장 짧은 AWS 리전이 확실하지 않은 경우 다음 제안 사항을 사용하여 결정할 수 있습니다.

- 오리진의 지리적 위치를 기준으로 오리진에 대한 지연 시간이 가장 짧은 AWS 리전을 대략적으로 알아보려면 이전 표를 참조하십시오.
- 오리진과 지리적으로 가까운 몇 가지 다른 AWS 리전에서 Amazon EC2 인스턴스를 시작하고, `ping`을 사용하여 해당 리전과 오리진 간의 일반적인 네트워크 대기 시간을 측정하는 몇 가지 테스트를 실행할 수 있습니다.

## Origin Shield 사용

Origin Shield를 활성화하면 캐시 적중률을 개선하고 오리진의 부하를 줄이며 성능을 향상시킬 수 있습니다. Origin Shield를 활성화하려면 CloudFront 배포에서 오리진 설정을 변경합니다. Origin Shield는 오리진의 속성입니다. CloudFront 배포의 각 오리진에 대해 Origin Shield가 해당 오리진에 대해 최상의 성능을 제공하는 AWS 리전에서 Origin Shield를 별도로 활성화할 수 있습니다.

CloudFront 콘솔, CloudFormation 또는 CloudFront API를 사용하여 Origin Shield를 활성화할 수 있습니다.

Console

기존 오리진에 대해 Origin Shield를 활성화하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 업데이트할 오리진이 있는 배포를 선택합니다.
3. 오리진 탭을 선택합니다.
4. 업데이트할 오리진을 선택한 다음 편집을 선택합니다.
5. Origin Shield 활성화에 대해 예를 선택합니다.
6. Origin Shield 리전의 경우 Origin Shield를 활성화하려는 AWS 리전을 선택합니다. 리전 선택에 대한 도움말은 Origin Shield의 AWS 리전 선택 단원을 참조하십시오.
7. 변경 사항 저장을 선택합니다.

배포 상태가 배포됨이면 Origin Shield가 준비됩니다. 이 작업에는 몇 분이 걸립니다.

새 오리진에 대해 Origin Shield를 활성화하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 기존 배포에서 새 오리진을 생성하려면 다음을 수행합니다.

   1. 오리진을 생성할 배포를 선택합니다.
   2. 오리진 생성을 선택한 다음 3단계로 이동합니다.

   새 표준 배포에서 새 오리진을 생성하려면 다음을 수행합니다.

   1. 각 단계에 따라 콘솔에서 표준 배포를 생성합니다. 자세한 내용은 콘솔에서 CloudFront 배포 생성 섹션을 참조하세요.
   2. 설정 섹션에서 오리진 설정 사용자 지정을 선택합니다. 3단계로 이동합니다.
3. Origin Shield 활성화에 대해 예를 선택합니다.
4. Origin Shield 리전의 경우 Origin Shield를 활성화하려는 AWS 리전을 선택합니다. 리전 선택에 대한 도움말은 Origin Shield의 AWS 리전 선택 단원을 참조하십시오.
5. 콘솔에서 각 단계에 따라 오리진 또는 배포 생성을 완료합니다.

배포 상태가 배포됨이면 Origin Shield가 준비됩니다. 이 작업에는 몇 분이 걸립니다.

CloudFormation

CloudFormation에서 Origin Shield를 활성화하려면 `OriginShield` 리소스의 `Origin` 속성 유형에서 `AWS::CloudFront::Distribution` 속성을 사용합니다. `OriginShield` 속성을 기존 `Origin`에 추가하거나 새 `Origin`을 생성할 때 포함할 수 있습니다.

다음 예제에서는 미국 서부(오레곤) 리전(`OriginShield`)에서 `us-west-2`를 활성화하는 구문을 YAML 형식으로 보여줍니다. 리전 선택에 대한 도움말은 the section called “Origin Shield의 AWS 리전 선택” 단원을 참조하십시오. 이 예제에서는 전체 `Origin` 리소스가 아니라 `AWS::CloudFront::Distribution` 속성 유형만 보여줍니다.

```
Origins:
- DomainName: 3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com
  Id: Example-EMP-3ae97e9482b0d011
  OriginShield:
    Enabled: true
    OriginShieldRegion: us-west-2
  CustomOriginConfig:
```

```
    OriginProtocolPolicy: match-viewer
    OriginSSLProtocols: TLSv1
```

자세한 내용은 AWS CloudFormation 사용 설명서의 리소스 및 속성 참조 섹션에서 AWS::CloudFront::Distribution Origin을 참조하세요.

API

AWS SDK 또는 AWS Command Line Interface(AWS CLI)를 사용하여 CloudFront API에서 Origin Shield를 활성화하려면 `OriginShield` 유형을 사용합니다. `OriginShield`의 `Origin`에서 `DistributionConfig`를 지정합니다. `OriginShield` 유형에 대한 자세한 내용은 Amazon CloudFront API 참조에서 다음 정보를 참조하십시오.

- OriginShield(유형)
- 오리진(유형)
- DistributionConfig(유형)
- UpdateDistribution(작업)
- CreateDistribution(작업)

이러한 유형 및 작업을 사용하기 위한 특정 구문은 SDK, CLI 또는 API 클라이언트에 따라 다릅니다. 자세한 내용은 SDK, CLI 또는 클라이언트에 대한 참조 설명서를 참조하십시오.

## Origin Shield 비용 추정

증분 계층인 Origin Shield로 이동하는 요청 수를 기준으로 Origin Shield에 대한 요금이 발생합니다.

오리진으로 프록시되는 동적(캐시할 수 없음) 요청의 경우 Origin Shield는 항상 증분 계층입니다. 동적 요청에는 `PUT`, `POST`, `PATCH`, `DELETE`와 같은 HTTP 메서드가 사용됩니다.

`GET` 및 `HEAD`는 TTL 설정이 3600초 미만인 요청은 동적 요청으로 간주됩니다. 또한, `GET` 및 `HEAD`는 캐싱을 비활성화한 요청도 동적 요청으로 간주됩니다.

동적 요청에 대한 Origin Shield 요금을 추정하려면 다음 공식을 사용하십시오.

총 동적 요청 수 x 요청 10,000개당 Origin Shield 요금/10,000

HTTP 메서드 , `GET`, `HEAD`, 및 `OPTIONS`를 사용하는 비동적 요청의 경우 Origin Shield가 증분 계층이 되기도 합니다. Origin Shield를 활성화하면 Origin Shield에 대해 AWS 리전을 선택합니다. Origin Shield와 동일한 리전에 있는 리전 엣지 캐시로 자연스럽게 이동하는 요청의 경우 Origin Shield는 증분

계층이 아닙니다. 이러한 요청에는 Origin Shield 요금은 발생하지 않습니다. Origin Shield와 다른 리전에서 리전 엣지 캐시로 이동한 다음 Origin Shield로 이동하는 요청의 경우 Origin Shield는 증분 계층입니다. 이러한 요청에는 Origin Shield 요금이 발생합니다.

캐시 가능한 요청에 대한 Origin Shield의 요금을 추정하려면 다음 공식을 사용하십시오.

캐시 가능한 총 요청 수 x(1 - 캐시 적중률) x 다른 리전의 리전 엣지 캐시에서 Origin Shield로 이동하는 요청의 비율 x 요청 10,000개당 Origin Shield 요금/10,000

Origin Shield에 대한 요청 10,000개당 요금에 대한 자세한 내용은 CloudFront 요금을 참조하십시오.

## Origin Shield 고가용성

Origin Shield는 CloudFront의 리전 엣지 캐시 기능을 활용합니다. 이러한 각 엣지 캐시는 Auto-Scaling Amazon EC2 인스턴스의 플릿과 함께 최소 3개의 가용 영역을 사용하여 AWS 리전에서 빌드됩니다. 또한 CloudFront 위치와 Origin Shield 간의 연결에서는 각 요청에 활성 오류 추적 기능을 사용해 주 Origin Shield 위치를 사용할 수 없으면 요청을 보조 Origin Shield 위치로 자동 라우팅합니다.

## Origin Shield가 다른 CloudFront 기능과 상호 작용하는 방법

다음 섹션에서는 Origin Shield가 다른 CloudFront 기능과 상호 작용하는 방식에 대해 설명합니다.

### Origin Shield 및 CloudFront 로깅

Origin Shield가 요청을 처리한 시기를 확인하려면 다음 중 하나를 활성화해야 합니다.

- CloudFront 표준 로그(액세스 로그) 표준 로그는 무료로 제공됩니다.
- CloudFront 실시간 액세스 로그. 실시간 액세스 로그 사용에 대한 추가 요금이 발생합니다. Amazon CloudFront 요금을 참조하십시오.

Origin Shield의 캐시 적중은 CloudFront 로그의 `OriginShieldHit` 필드로 `x-edge-detailed-result-type`로 표시됩니다. Origin Shield는 Amazon CloudFront의 리전 엣지 캐시를 활용합니다. 요청이 CloudFront 엣지 로케이션에서 Origin Shield 역할을 하는 리전 엣지 캐시로 라우팅되는 경우, 로그에서 `Hit`가 아닌 `OriginShieldHit`로 보고됩니다.

### Origin Shield 및 오리진 그룹

Origin Shield는 CloudFront 오리진 그룹과 호환됩니다. Origin Shield는 오리진의 속성이므로 오리진이 오리진 그룹의 일부인 경우에도 요청은 항상 각 오리진에 대해 Origin Shield를 통과합니다. 지정된 요청에 대해 CloudFront는 기본 오리진의 Origin Shield를 통해 오리진 그룹의 기본 오리진으로 요청을 라

우팅합니다. 요청이 실패하면(오리진 그룹 장애 조치 기준에 따라) CloudFront는 보조 오리진의 Origin Shield를 통해 요청을 보조 오리진으로 라우팅합니다.

### Origin Shield 및 Lambda@Edge

Origin Shield는 Lambda@Edge 함수의 기능에 영향을 미치지 않지만 해당 함수가 실행되는 AWS 리전에 영향을 줄 수 있습니다.

Lambda@Edge와 함께 Origin Shield를 사용하면 Origin Shield가 활성화된 AWS 리전에서 오리진 방향 트리거(오리진 요청 및 오리진 응답)가 실행됩니다. 기본 Origin Shield 위치를 사용할 수 없고 CloudFront가 요청을 보조 Origin Shield 위치로 라우팅하는 경우 Lambda@Edge 오리진 연결 트리거는 보조 Origin Shield 위치를 사용하도록 전환합니다.

최종 사용자 방향 트리거는 영향을 받지 않습니다.

## CloudFront 오리진 장애 조치를 통한 고가용성 최적화

고가용성을 필요로 하는 시나리오에 대한 오리진 장애 조치를 사용하여 이제 CloudFront를 설정할 수 있습니다. 시작하려면 2개의 오리진(기본 오리진 및 보조 오리진)이 포함된 오리진 그룹을 만듭니다. 기본 오리진을 사용할 수 없거나 실패를 나타내는 특정 HTTP 응답 상태 코드를 반환하는 경우 CloudFront는 자동으로 보조 오리진으로 전환합니다.

오리진 장애 조치를 설정하려면 최소 2개의 오리진이 있는 배포가 전제되어야 합니다. 다음으로, 2개의 오리진이 포함된 배포에 대한 오리진 그룹을 만들고 그 중 1개의 오리진을 기본 오리진으로 설정합니다. 마지막으로, 오리진 그룹을 사용하도록 캐시 동작을 생성하거나 업데이트합니다.

오리진 그룹을 설정하고 특정 오리진 장애 조치 옵션을 구성하는 단계는 오리진 그룹 생성 단원을 참조하십시오.

캐시 동작에 대한 오리진 장애 조치를 구성한 후에, CloudFront가 최종 사용자의 요청에 대해 다음을 수행합니다.

- 캐시 적중이 있는 경우 CloudFront는 요청된 객체를 반환합니다.
- 캐시 누락이 있는 경우, CloudFront는 오리진 그룹 내에서 기본 오리진으로 요청을 라우팅합니다.
- 기본 원본이 HTTP 2xx 또는 3xx 상태 코드와 같이 장애 조치용으로 구성되지 않은 상태 코드를 반환하면 CloudFront는 요청된 객체를 뷰어에게 제공합니다.
- 다음과 같은 문제가 발생하는 경우:
  - 기본 오리진이 장애 조치를 위해 구성된 HTTP 상태 코드를 반환합니다.
  - CloudFront가 기본 오리진에 연결하지 못함(503이 장애 조치 코드로 설정된 경우)

- 기본 오리진의 응답은 너무 오래 걸립니다(시간 초과). (504가 장애 조치 코드로 설정된 경우)

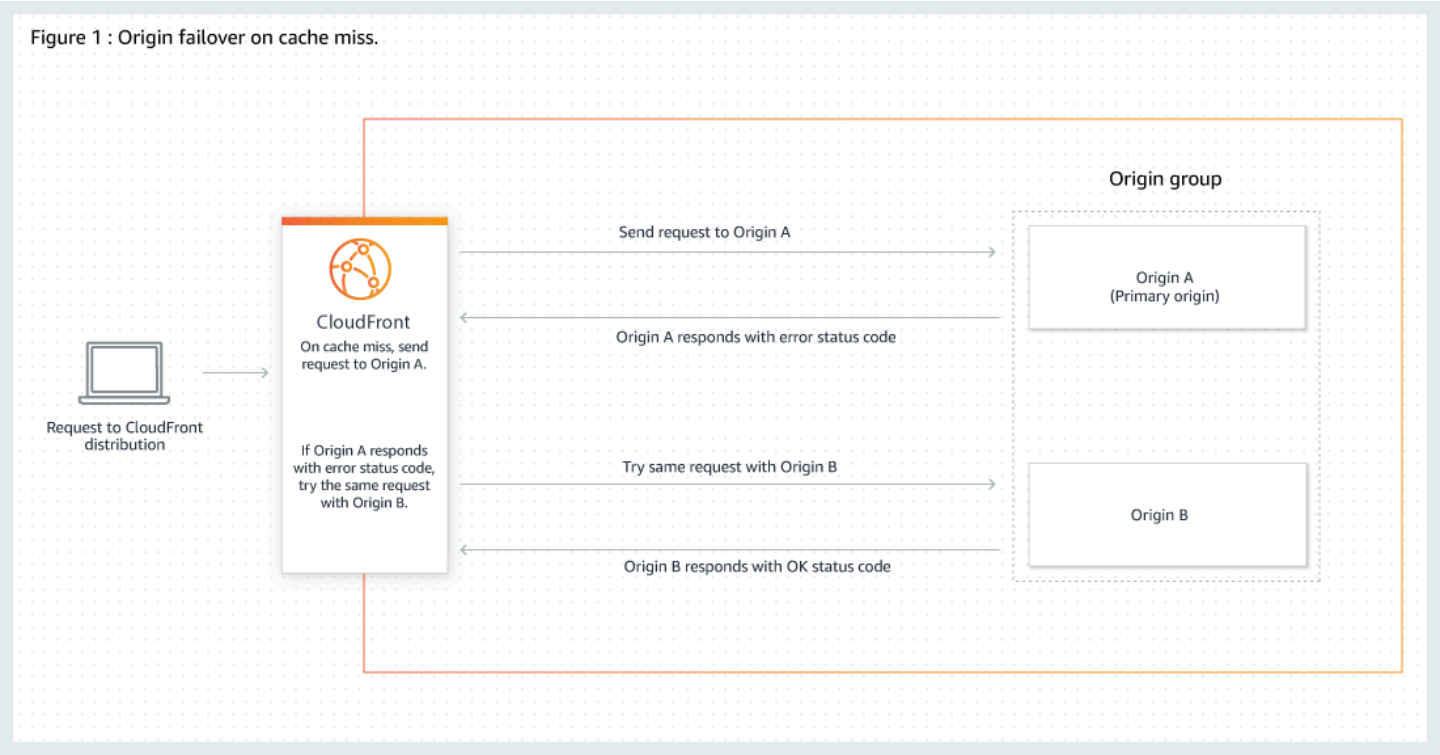
그러면 CloudFront는 요청을 오리진 그룹의 보조 오리진으로 라우팅합니다.

> **Note**
>
> 비디오 콘텐츠 스트리밍과 같은 일부 사용 사례의 경우 CloudFront가 보조 오리진으로 신속하게 장애 조치하도록 할 수 있습니다. CloudFront가 보조 오리진으로 장애 조치하는 속도를 조정하려면 오리진 제한 시간 및 시도 횟수 제어 단원을 참조하십시오.

CloudFront는 이전 요청이 보조 오리진으로 장애 조치되는 경우에도 수신되는 모든 요청을 기본 오리진으로 라우팅합니다. CloudFront는 기본 오리진에 대한 요청이 실패한 후에만 보조 오리진에 요청을 보냅니다.

CloudFront는 최종 사용자 요청의 HTTP 메서드가 `GET`, `HEAD` 또는 `OPTIONS`인 경우에만 보조 오리진으로 장애 조치합니다. 최종 사용자가 다른 HTTP 메서드(예: `POST`, `PUT` 등)를 보내면 CloudFront는 장애 조치를 수행하지 않습니다.

다음 다이어그램에서는 오리진 장애 조치의 작동 방식을 보여 줍니다.




## 주제

- 오리진 그룹 생성
- 오리진 제한 시간 및 시도 횟수 제어
- Lambda@Edge 함수와 함께 오리진 장애 조치 사용
- 오리진 장애 조치와 함께 사용자 지정 오류 페이지 사용

## 오리진 그룹 생성

오리진 그룹을 생성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 오리진 그룹을 만들려는 분포를 선택합니다.
3. 오리진 탭을 선택합니다.
4. 배포에 둘 이상의 원본이 있는지 확인합니다. 그렇지 않으면 두 번째 원본을 추가합니다.
5. 원본(Origins) 탭의 원본 그룹(Origin groups) 창에서 원본 그룹 생성(Create origin group)을 선택합니다.
6. 오리진 그룹에 대한 오리진을 선택합니다. 오리진을 추가한 후 화살표를 사용하여 우선순위를 설정합니다. 즉, 기본 오리진과 보조 오리진을 지정합니다.
7. 원본 그룹의 이름을 입력합니다.
8. 장애 조치 기준으로 사용할 HTTP 상태 코드를 선택합니다. 상태 코드 400, 403, 404, 416, 500, 502, 503 또는 504의 어떠한 조합도 선택할 수 있습니다. 지정한 상태 코드 중 하나가 포함된 응답을 CloudFront에서 수신하면 보조 오리진으로 장애 조치됩니다.

   > **Note**
   >
   > CloudFront는 최종 사용자 요청의 HTTP 메서드가 `GET`, `HEAD` 또는 `OPTIONS`인 경우에만 보조 오리진으로 장애 조치합니다. 최종 사용자가 다른 HTTP 메서드(예: `POST`, `PUT` 등)를 보내면 CloudFront는 장애 조치를 수행하지 않습니다.

9. 오리진 선택 기준에서 배포가 뷰어 요청을 라우팅할 때 오리진을 선택하는 방법을 지정합니다. 다음 옵션을 선택할 수 있습니다.

   Default

   CloudFront는 설정 페이지에서 지정한 기본 오리진 우선순위를 사용합니다.

미디어 품질 점수

CloudFront는 이 점수를 추적하고 사용하여 요청을 전달할 첫 번째 오리진을 확인합니다. 또한 이를 통해 CloudFront는 미디어 품질 점수를 확인하기 위해 오리진 그룹의 대체 오리진에 비동기식 HEAD 요청을 할 수 있습니다. AWS Elemental MediaPackage v2 오리진의 경우에만 이 옵션을 선택할 수 있습니다. 자세한 내용은 미디어 품질 인식 복원력 섹션을 참조하세요.

10. 원본 그룹 생성(Create origin group)을 선택합니다.

분산 캐시 동작에 대한 오리진 그룹을 오리진으로 지정해야 합니다. 자세한 내용은 이름 섹션을 참조하세요.

## 오리진 제한 시간 및 시도 횟수 제어

기본적으로 CloudFront는 오리진 그룹의 기본 오리진에 최대 30초(각각 10초 동안 연결 시도 3회) 동안 연결을 시도한 후 보조 오리진으로 장애 조치합니다. 비디오 콘텐츠 스트리밍과 같은 일부 사용 사례의 경우 CloudFront가 보조 오리진으로 보다 신속하게 장애 조치하도록 할 수 있습니다. 다음 설정을 조정하여 CloudFront가 보조 오리진으로 장애 조치하는 속도에 영향을 줄 수 있습니다. 오리진이 보조 오리진이거나 오리진 그룹의 일부가 아닌 오리진인 경우 이러한 설정은 CloudFront가 HTTP 504 응답을 최종사용자에게 얼마나 빨리 반환하는지에 영향을 줍니다.

더욱 신속하게 장애 조치하려면 더 짧은 연결 제한 시간, 더 적은 연결 시도 횟수 또는 둘 다를 지정합니다. 사용자 지정 오리진(정적 웹 사이트 호스팅으로 구성된 Amazon S3 버킷 오리진 포함)의 경우 오리진 응답 제한 시간을 조정할 수도 있습니다.

오리진 연결 제한 시간

오리진 연결 제한 시간 설정은 오리진에 대한 연결을 설정하려고 할 때 CloudFront의 대기 시간에 영향을 줍니다. 기본적으로 CloudFront는 연결을 설정하기 위해 10초 동안 대기하지만 1~10초를 지정할 수 있습니다. 자세한 내용은 연결 제한 시간 단원을 참조하세요.

오리진 연결 시도 횟수

오리진 연결 시도 횟수 설정은 CloudFront가 오리진에 연결을 시도하는 횟수에 영향을 줍니다. 기본적으로 CloudFront에서는 세 번 연결을 시도하지만 1~3회를 지정할 수 있습니다. 자세한 내용은 연결 시도 단원을 참조하세요.

사용자 지정 오리진(정적 웹 사이트 호스팅으로 구성된 Amazon S3 버킷 포함)의 경우 이 설정은 또한 오리진 응답 제한 시간이 발생했을 때 CloudFront가 오리진에서 응답을 얻으려고 시도하는 횟수에 영향을 미칩니다.

### 오리진 응답 제한 시간

오리진 읽기 제한 시간이라고도 하는 오리진 응답 제한 시간은 CloudFront가 오리진으로부터 응답을 수신(또는 전체 응답을 수신)하기 위해 대기하는 시간에 영향을 줍니다. 기본적으로 CloudFront는 30초 동안 대기하지만 1~120초를 지정할 수 있습니다. 자세한 내용은 응답 제한 시간 섹션을 참조하세요.

## 이러한 설정을 변경하는 방법

CloudFront 콘솔에서 이러한 설정을 변경하려면

- 새 오리진 또는 새 배포의 경우 리소스를 생성할 때 이러한 값을 지정하십시오.
- 기존 배포의 기존 오리진의 경우 오리진을 편집할 때 이 값을 지정하십시오.

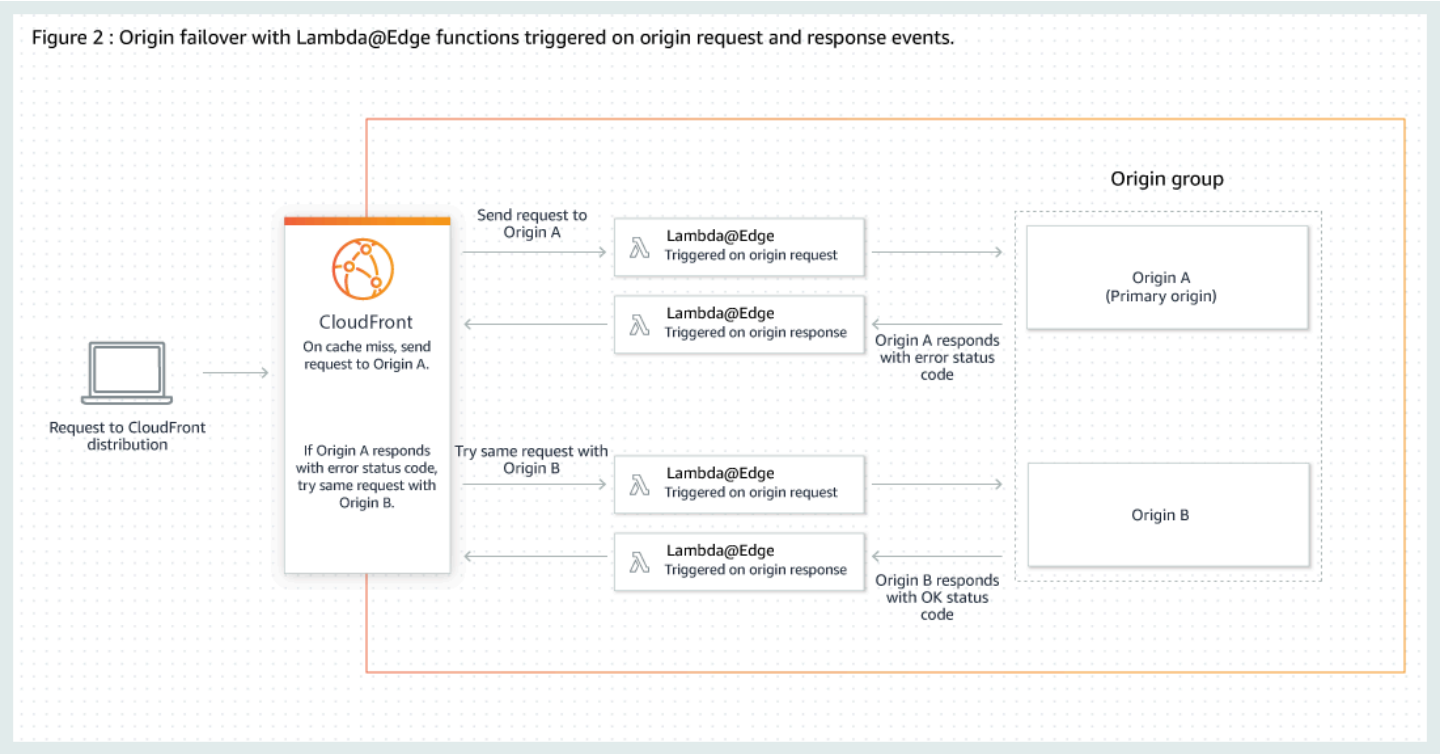
자세한 내용은 모든 배포 설정 참조 단원을 참조하세요.

# Lambda@Edge 함수와 함께 오리진 장애 조치 사용

Lambda@Edge 함수를 오리진 그룹에 설정한 CloudFront 배포와 함께 사용할 수 있습니다. Lambda 함수를 사용하려면, 캐시 동작을 생성할 때 오리진 그룹에 대한 오리진 요청 또는 오리진 응답 트리거에 이 함수를 지정합니다. 오리진 그룹과 함께 Lambda@Edge 함수를 사용하는 경우, 단일 최종 사용자 요청에 대해 함수를 두 번 트리거할 수 있습니다. 예를 들어 다음 시나리오를 고려해 보십시오.

1. 오리진 요청 트리거를 사용하여 Lambda@Edge 함수를 생성합니다.
2. Lambda 함수는 CloudFront가 (캐시 누락 시) 기본 오리진에 요청을 보낼 때 한 번 트리거됩니다.
3. 기본 오리진은 장애 조치를 위해 구성된 HTTP 상태 코드로 응답합니다.
4. Lambda 함수는 CloudFront가 보조 오리진에 동일한 요청을 보낼 때 다시 트리거됩니다.

다음 다이어그램에서는 오리진 요청이나 응답 트리거에 Lambda@Edge 함수가 포함된 경우 오리진 장애 조치가 작동하는 방식을 보여 줍니다.




Lambda@Edge 트리거 사용에 대한 자세한 내용은 the section called "Lambda@Edge 함수에 대한 트리거 추가" 단원을 참조하십시오.

DNS 장애 조치 관리에 대한 자세한 내용은 Amazon Route 53 개발자 안내서의 DNS 장애 조치 구성을 참조합니다.

## 오리진 장애 조치와 함께 사용자 지정 오류 페이지 사용

사용자 지정 오류 페이지를 오리진 장애 조치에 대해 설정되지 않은 오리진에 사용하는 경우와 유사한 방식으로 사용자 지정 오류 페이지를 오리진 그룹에 사용할 수 있습니다.

오리진 장애 조치를 사용할 때, CloudFront를 기본 오리진 또는 보조 오리진에 대해(또는 양쪽 모두에 대해) 사용자 지정 오류 페이지를 반환하도록 구성할 수 있습니다.

- 기본 오리진에 대해 사용자 지정 오류 페이지 반환 — 장애 조치를 위해 구성되지 않은 HTTP 상태 코드를 기본 오리진에서 반환하는 경우 CloudFront는 사용자 지정 오류 페이지를 최종 사용자에게 반환합니다.
- 보조 오리진에 대해 사용자 지정 오류 페이지 반환 — CloudFront가 보조 오리진에서 실패 상태 코드를 수신하면 CloudFront는 사용자 지정 오류 페이지를 반환합니다.

CloudFront에서 사용자 지정 오류 페이지를 사용하는 방법에 대한 자세한 내용은 사용자 지정 오류 응답 생성 단원을 참조하십시오.

## 콘텐츠가 캐시에 유지되는 기간(만료) 관리

CloudFront에서 다른 요청을 오리진에 전달하기 전에 파일을 CloudFront 캐시에 보관하는 시간을 제어할 수 있습니다. 이 기간을 단축함으로써 동적 콘텐츠를 제공할 수 있습니다. 이 기간이 늘어나면 파일이 엣지 캐시에서 바로 제공될 가능성이 높으므로 사용자에게 제공되는 성능이 향상됩니다. 보관 기간이 늘어나면 오리진에 걸리는 부하 역시 줄어듭니다.

일반적으로 CloudFront는 지정한 캐시 기간이 지날 때까지, 즉 파일이 만료될 때까지 엣지 로케이션에서 파일을 제공합니다. 파일이 만료된 후 다음에 엣지 로케이션에서 이 파일에 대한 요청을 받으면, CloudFront에서는 이 요청을 오리진으로 전달하여 캐시에 최신 버전의 파일이 포함되어 있는지 확인합니다. 오리진의 응답은 파일 변경 여부에 따라 달라집니다.

- CloudFront 캐시에 이미 최신 버전이 있는 경우 오리진에서는 상태 코드 `304 Not Modified`를 반환합니다.
- CloudFront 캐시에 최신 버전이 없는 경우 오리진에서는 상태 코드 `200 OK`와 최신 버전의 파일을 반환합니다.

엣지 로케이션의 파일이 자주 요청되지 않는 경우 CloudFront에서는 만료 날짜 전에 이 파일을 제거하여 보다 최근에 요청된 파일을 위한 공간을 마련할 수 있습니다.

배포의 캐시 정책을 업데이트하여 캐시 기간을 관리하는 것이 좋습니다. 캐시 정책 사용을 옵트아웃하는 경우 기본 TTL(Time to Live)은 24시간이지만 다음 설정을 업데이트하여 기본 설정을 재정의할 수 있습니다.

- 동일한 경로 패턴과 일치하는 모든 파일에 대해 캐시 기간을 변경하려면 캐시 동작에 대한 Minimum TTL(최소 TTL), Maximum TTL(최대 TTL) 및 Default TTL(기본 TTL)의 CloudFront 설정을 변경할 수 있습니다. 개별 설정에 관한 자세한 내용은 Minimum TTL, Maximum TTL 및 기본 TTL 섹션을 참조하세요.
- 개별 파일의 캐시 기간을 변경하려면 `max-age` 또는 `s-maxage` 지시문이 있는 `Cache-Control` 헤더 또는 `Expires` 헤더를 파일에 추가하도록 오리진을 구성할 수 있습니다. 자세한 내용은 헤더를 사용하여 개별 객체의 캐시 기간 제어 섹션을 참조하세요.

최소 TTL(Minimum TTL), 기본 TTL(Default TTL) 및 최대 TTL(Maximum TTL)이 `max-age` 및 `s-maxage` 명령 및 `Expires` 헤더 필드와 상호 작용하는 방식에 대한 자세한 내용은 the section called "CloudFront에서 객체를 캐싱하는 시간 지정" 섹션을 참조하세요.

또한 CloudFront 에서 다른 요청을 오리진에 전달하여 요청된 객체의 가져오기를 다시 시도하기 전에 오류(예: `404 Not Found`)가 CloudFront 캐시에 보관되는 시간 역시 제어할 수 있습니다. 자세한 내용은 the section called "CloudFront에서 오리진의 HTTP 4xx 및 5xx 상태 코드를 처리하는 방법" 섹션을 참조하세요.

주제

- 헤더를 사용하여 개별 객체의 캐시 기간 제어
- 오래된(만료된) 콘텐츠 제공
- CloudFront에서 객체를 캐싱하는 시간 지정
- Amazon S3 콘솔을 사용하여 객체에 헤더 추가

## 헤더를 사용하여 개별 객체의 캐시 기간 제어

`Cache-Control` 및 `Expires` 헤더를 사용하여 객체가 캐시에 머무르는 시간을 제어할 수 있습니다. 최소 TTL, 기본 TTL 및 최대 TTL에 대한 설정은 캐시 기간에도 영향을 주지만 캐시 기간에 영향을 미치는 방식에 대한 개요는 다음과 같습니다.

- `Cache-Control max-age` 명령을 사용하면 CloudFront가 오리진 서버에서 객체를 다시 가져오기 전에 이 객체를 캐시에 보관하려는 시간(초)을 지정할 수 있습니다. CloudFront가 지원하는 최소 만료 시간은 0초입니다. 최대값은 100년입니다. 다음 형식으로 값을 지정합니다.

  ```
  Cache-Control: max-age=#
  ```

  예를 들어 다음 명령은 CloudFront에 3,600초(1시간) 동안 관련 객체를 캐시에 보관하도록 지시합니다.

  ```
  Cache-Control: max-age=3600
  ```

  브라우저 캐시에 보관하는 기간과는 다른 기간 동안 객체를 CloudFront 엣지 캐시에 보관하려는 경우, `Cache-Control max-age` 및 `Cache-Control s-maxage` 명령을 함께 사용할 수 있습니다. 자세한 내용은 CloudFront에서 객체를 캐싱하는 시간 지정 단원을 참조하세요.
- `Expires` 헤더 필드를 사용하면 다음과 같이 RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1 Section 3.3.1, Full Date에 지정된 형식을 사용하여 만료 날짜와 시간을 지정할 수 있습니다.

```
Sat, 27 Jun 2015 23:59:59 GMT
```

`Expires` 헤더 필드 대신 `Cache-Control max-age` 명령을 사용하여 객체 캐싱을 제어하는 것이 좋습니다. `Cache-Control max-age` 및 `Expires` 모두에 대해 값을 지정한 경우, CloudFront에서는 `Cache-Control max-age`의 값만 사용합니다.

자세한 내용은 CloudFront에서 객체를 캐싱하는 시간 지정 단원을 참조하세요.

최종 사용자의 `Cache-Control` 요청에 HTTP `Pragma` 또는 `GET` 헤더 필드를 사용하여 CloudFront가 객체에 대한 오리진 서버로 돌아가도록 할 수는 없습니다. CloudFront에서는 최종 사용자 요청의 헤더 필드를 무시합니다.

`Cache-Control` 및 `Expires` 헤더 필드에 대한 자세한 내용은 RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1의 다음 단원을 참조하십시오.

- 섹션 14.9 Cache Control
- 섹션 14.21 Expires

## 오래된(만료된) 콘텐츠 제공

CloudFront는 `Stale-While-Revalidate` 및 `Stale-If-Error` 캐시 제어 지시문을 지원합니다. 이러한 지시문을 사용하여 뷰어가 기한 경과 콘텐츠를 사용할 수 있는 기간을 지정할 수 있습니다.

주제

- Stale-While-Revalidate
- Stale-If-Error
- 두 지시문 모두 사용

### Stale-While-Revalidate

이 지시문을 통해 CloudFront는 오리진에서 새 버전을 비동기적으로 가져오는 동안 캐시에서 기한 경과 콘텐츠를 제공할 수 있습니다. 이렇게 하면 뷰어가 백그라운드 가져오기를 기다릴 필요 없이 엣지 로케이션에서 즉시 응답을 받을 때 지연 시간이 향상됩니다. 새 콘텐츠는 향후 요청을 위해 백그라운드에 로드됩니다.

Example예시: Stale-While-Revalidate

CloudFront는 이러한 지시문을 사용하도록 `Cache-Control` 헤더를 설정할 때 다음을 수행합니다.

```
Cache-Control: max-age=3600, stale-while-revalidate=600
```

1. CloudFront는 1시간(`max-age=3600`) 동안 응답을 캐싱합니다.
2. 이 기간 이후에 요청이 이루어지면 CloudFront는 기한 경과 콘텐츠를 제공하는 동시에 캐시된 콘텐츠를 재검증하고 새로 고침하라는 요청을 오리진에 전송합니다.
3. 콘텐츠가 재검증되는 동안 CloudFront는 최대 10분(`stale-while-revalidate=600`) 동안 기한 경과 콘텐츠를 제공합니다.

> **Note**
>
> CloudFront는 `stale-while-revalidate` 지시문의 값 또는 CloudFront 최대 TTL의 값 중 더 적은 값까지 기한 경과 콘텐츠를 제공합니다. 최대 TTL 기간이 지나면 `stale-while-revalidate` 값에 관계없이 엣지 캐시에서 기한 경과 객체를 사용할 수 없습니다.

## `Stale-If-Error`

이 지시문을 통해 CloudFront는 오리진에 연결할 수 없거나 500에서 600 사이의 오류 코드를 반환하는 경우 캐시에서 기한 경과 콘텐츠를 제공할 수 있습니다. 이를 통해 뷰어가 오리진 가동 중단 중에도 콘텐츠에 액세스할 수 있습니다.

Example예시: Stale-If-Error

CloudFront는 이러한 지시문을 사용하도록 `Cache-Control` 헤더를 설정할 때 다음을 수행합니다.

```
Cache-Control: max-age=3600, stale-if-error=86400
```

1. CloudFront는 1시간(`max-age=3600`) 동안 응답을 캐싱합니다.
2. 이 기간 이후에 오리진이 다운되거나 오류가 반환되는 경우 CloudFront는 최대 24시간(`stale-if-error=86400`) 동안 기한 경과 콘텐츠를 계속 제공합니다.
3. 사용자 지정 오류 응답이 구성된 경우 CloudFront는 지정된 `stale-if-error` 기간 내에 오류가 발생하면 먼저 기한 경과 콘텐츠를 제공하려고 시도합니다. 기한 경과 콘텐츠를 사용할 수 없으면

CloudFront는 해당 오류 상태 코드에 대해 구성된 사용자 지정 오류 응답을 제공합니다. 자세한 내용은 사용자 지정 오류 응답 생성 섹션을 참조하세요.

> **Notes**
>
> - CloudFront는 `stale-if-error` 지시문의 값 또는 CloudFront 최대 TTL의 값 중 더 적은 값까지 기한 경과 콘텐츠를 제공합니다. 최대 TTL 기간이 지나면 `stale-if-error` 값에 관계없이 엣지 캐시에서 기한 경과 객체를 사용할 수 없습니다.
> - `stale-if-error` 또는 사용자 지정 오류 응답을 구성하지 않으면 CloudFront는 요청된 객체가 엣지 캐시에 있는지 여부에 따라 기한 경과 객체를 반환하거나 오류 응답을 뷰어에게 다시 전달합니다. 자세한 내용은 사용자 지정 오류 페이지를 구성하지 않은 경우 CloudFront에서 오류를 처리하는 방법 섹션을 참조하세요.

## 두 지시문 모두 사용

`stale-while-revalidate` 및 `stale-if-error`는 지연 시간을 줄이고 오리진이 응답하거나 복구하기 위한 버퍼를 추가하도록 함께 사용할 수 있는 독립적인 캐시 제어 지시문입니다.

Example예: 두 지시문 모두 사용

CloudFront는 다음 지시문을 사용하도록 `Cache-Control` 헤더를 설정할 경우 다음을 수행합니다.

```
Cache-Control: max-age=3600, stale-while-revalidate=600, stale-if-error=86400
```

1. CloudFront는 1시간(`max-age=3600`) 동안 응답을 캐싱합니다.
2. 이 기간 이후에 요청이 이루어지면 CloudFront는 콘텐츠가 재검증되는 동안 최대 10분(`stale-while-revalidate=600`)간 기한 경과 콘텐츠를 제공합니다.
3. CloudFront가 콘텐츠 재검증을 시도하는 동안 오리진 서버에서 오류가 반환되는 경우 CloudFront는 최대 24시간(`stale-if-error=86400`) 동안 기한 경과 콘텐츠를 계속 제공합니다.

캐싱은 성능과 최신 상태 사이의 균형을 유지하는 것입니다. `stale-while-revalidate` 및 `stale-if-error`와 같은 지시문을 사용하면 성능과 사용자 경험을 향상할 수 있지만 원하는 콘텐츠의 최신 상태에 맞게 구성을 조정해야 합니다. 오래된 콘텐츠 지시문은 콘텐츠를 새로 고쳐야 하지만 최신 버전이 필요하지 않은 사용 사례에 가장 적합합니다. 또한 콘텐츠가 전혀 또는 거의 변경되지 않는 경우

`stale-while-revalidate`로 인해 불필요한 네트워크 요청이 추가될 수 있습니다. 대신 캐시 기간을 길게 설정하는 것이 좋습니다.

## CloudFront에서 객체를 캐싱하는 시간 지정

CloudFront에서 오리진에 다른 요청을 보내기 전에 객체를 캐시에 보관하는 시간을 제어하려면 다음을 수행할 수 있습니다.

- CloudFront 배포의 캐시 동작에서 최소값, 최대값 및 기본 TTL 값을 설정합니다. 캐시 동작에 연결된 캐시 정책(권장) 또는 레거시 캐시 설정에서 이러한 값을 설정할 수 있습니다.
- 오리진의 응답에 `Cache-Control` 또는 `Expires` 헤더를 포함합니다. 또한 이러한 헤더는 CloudFront로 다른 요청을 보내기 전에 브라우저에서 브라우저 캐시에 객체를 유지하는 기간을 결정하는 데 도움이 됩니다.

다음 표에는 오리진에서 전송된 `Cache-Control` 및 `Expires` 헤더와 캐시 동작의 TTL 설정이 캐싱에 미치는 영향이 설명되어 있습니다.

| 오리진 헤더 | 최소 TTL = 0 | 최소 TTL > 0 |
| --- | --- | --- |
| 오리진에서 **`Cache-Control: max-age`** 지시문을 객체에 추가 | CloudFront 캐싱<br><br>CloudFront는 `Cache-Control: max-age` 지시문의 값 또는 CloudFront 최대 TTL의 값 중 더 작은 값 동안 객체를 캐싱합니다.<br><br>브라우저 캐싱<br><br>브라우저는 `Cache-Control: max-age` 지시문의 값 동안 객체를 캐싱합니다. | CloudFront 캐싱<br><br>CloudFront 캐싱은 CloudFront 최소 TTL과 최대 TTL 및 `Cache-Control max-age` 지시문의 값에 따라 달라집니다.<br><br>• 최소 TTL < `max-age` < 최대 TTL인 경우 CloudFront는 `Cache-Control: max-age` 지시문의 값 동안 객체를 캐싱합니다.<br>• `max-age` < 최소 TTL인 경우 CloudFront는 CloudFront 최 |

| 오리진 헤더 | 최소 TTL = 0 | 최소 TTL > 0 |
| --- | --- | --- |
| | | 소 TTL의 값 동안 객체를 캐싱합니다.<br>• `max-age` > 최대 TTL인 경우 CloudFront는 CloudFront 최대 TTL의 값 동안 객체를 캐싱합니다.<br><br>브라우저 캐싱<br><br>브라우저는 `Cache-Control: max-age` 지시문의 값 동안 객체를 캐싱합니다. |
| 오리진에서 **`Cache-Control: max-age`** 지시문을 객체에 추가하지 않음 | CloudFront 캐싱<br><br>CloudFront는 CloudFront 기본 TTL의 값 동안 객체를 캐싱합니다.<br><br>브라우저 캐싱<br><br>브라우저에 따라 다릅니다. | CloudFront 캐싱<br><br>CloudFront는 CloudFront 최소 TTL 또는 기본 TTL의 값 중 더 큰 값 동안 객체를 캐싱합니다.<br><br>브라우저 캐싱<br><br>브라우저에 따라 다릅니다. |

| 오리진 헤더 | 최소 TTL = 0 | 최소 TTL > 0 |
| --- | --- | --- |
| 오리진에서 **`Cache-Control: max-age`** 및 **`Cache-Control: s-maxage`** 지시문을 객체에 추가 | CloudFront 캐싱<br><br>CloudFront는 `Cache-Control: s-maxage` 지시문의 값 또는 CloudFront 최대 TTL의 값 중 더 작은 값 동안 객체를 캐싱합니다.<br><br>브라우저 캐싱<br><br>브라우저는 `Cache-Control max-age` 지시문의 값 동안 객체를 캐싱합니다. | CloudFront 캐싱<br><br>CloudFront 캐싱은 CloudFront 최소 TTL과 최대 TTL 및 `Cache-Control: s-maxage` 지시문의 값에 따라 달라집니다.<br><br>• 최소 TTL < `s-maxage` < 최대 TTL인 경우 CloudFront는 `Cache-Control: s-maxage` 지시문의 값 동안 객체를 캐싱합니다.<br>• `s-maxage` < 최소 TTL인 경우 CloudFront는 CloudFront 최소 TTL의 값 동안 객체를 캐싱합니다.<br>• `s-maxage` > 최대 TTL인 경우 CloudFront는 CloudFront 최대 TTL의 값 동안 객체를 캐싱합니다.<br><br>브라우저 캐싱<br><br>브라우저는 `Cache-Control: max-age` 지시문의 값 동안 객체를 캐싱합니다. |

| 오리진 헤더 | 최소 TTL = 0 | 최소 TTL > 0 |
| --- | --- | --- |
| 오리진에서 **`Expires`** 헤더를 객체에 추가 | CloudFront 캐싱<br><br>CloudFront는 `Expires` 헤더의 날짜 또는 CloudFront 최대 TTL의 값 중 더 빨리 도래하는 객체를 캐싱합니다.<br><br>브라우저 캐싱<br><br>브라우저는 `Expires` 헤더의 날짜까지 객체를 캐싱합니다. | CloudFront 캐싱<br><br>CloudFront 캐싱은 CloudFront 최소 TTL과 최대 TTL 및 `Expires` 지시문의 값에 따라 달라집니다.<br><br>• 최소 TTL < `Expires` < 최대 TTL인 경우 CloudFront는 `Expires` 헤더의 날짜 및 시간까지 객체를 캐싱합니다.<br>• `Expires` < 최소 TTL인 경우 CloudFront는 CloudFront 최소 TTL의 값 동안 객체를 캐싱합니다.<br>• `Expires` > 최대 TTL인 경우 CloudFront는 CloudFront 최대 TTL의 값 동안 객체를 캐싱합니다.<br><br>브라우저 캐싱<br><br>브라우저는 `Expires` 헤더의 날짜와 시간까지 객체를 캐싱합니다. |

| 오리진 헤더 | 최소 TTL = 0 | 최소 TTL > 0 |
| --- | --- | --- |
| 오리진에서 **`Cache-Control: no-cache`**, **`no-store`** 및/또는 **`private`** 지시문을 객체에 추가 | CloudFront 및 브라우저에서는 헤더의 설정을 따릅니다. | CloudFront 캐싱<br><br>CloudFront는 CloudFront 최소 TTL의 값 동안 객체를 캐싱합니다. 이 표 아래의 경고를 참조하세요.<br><br>브라우저 캐싱<br><br>브라우저에서는 헤더의 설정을 따릅니다. |

**Warning**

- 최소 TTL이 0보다 큰 경우 CloudFront는 `Cache-Control: no-cache`, `no-store` 및/또는 `private` 지시문이 오리진 헤더에 있더라도 캐시 정책의 최소 TTL을 사용합니다.
  - 오리진에 연결할 수 있는 경우 CloudFront는 오리진에서 객체를 가져와 최종 사용자에게 반환합니다.
  - 오리진에 연결할 수 없고 최소 또는 최대 TTL이 0보다 큰 경우 CloudFront는 이전에 오리진에서 가져온 객체를 제공합니다.

  이 동작을 방지하려면 오리진에서 반환된 객체에 `Cache-Control: stale-if-error=0` 지시문을 포함합니다. 이렇게 하면 향후 요청 시 오리진에 연결할 수 없을 때 CloudFront가 이전에 오리진에서 가져온 객체를 반환하는 대신 오류를 반환합니다.
- 오리진 헤더에 `Cache-Control: no-cache`, `no-store` 및/또는 `private` 명령이 포함된 경우 CloudFront는 S3 오리진에서 HTTP 501 상태 코드(구현되지 않음)를 캐싱하지 않습니다. 이는 최소 TTL 설정이 0보다 큰 경우에도 S3 오리진의 기본 동작입니다.

CloudFront 콘솔을 사용하여 배포에 대한 설정을 변경하는 방법에 대한 자세한 내용은 배포 업데이트 섹션을 참조하세요. CloudFront API를 사용하여 배포에 대한 설정을 변경하는 방법에 대한 자세한 내용은 UpdateDistribution을 참조하세요.

## Amazon S3 콘솔을 사용하여 객체에 헤더 추가

Amazon S3 객체에 `Cache-Control` 또는 `Expires` 헤더 필드를 추가할 수 있습니다. 이렇게 하려면 객체의 메타데이터 필드를 수정합니다.

Amazon S3 객체에 **`Cache-Control`** 또는 **`Expires`** 헤더 필드 추가

1. Amazon S3 사용 설명서의 Amazon S3 콘솔에서 객체 메타데이터 편집 주제에 있는 시스템 정의 메타데이터 바꾸기 섹션의 절차를 따릅니다.
2. [키(Key)]에서 추가할 헤더의 이름(Cache-Control 또는 Expires)을 선택합니다.
3. [값(Value)]에 헤더 값을 입력합니다. 예를 들어, `Cache-Control` 헤더의 경우 `max-age=86400`을 입력할 수 있습니다. `Expires`의 경우 만료 날짜 및 시간(예: `Wed, 30 Jun 2021 09:28:00 GMT`)을 입력할 수 있습니다.
4. 나머지 절차에 따라 메타데이터 변경 사항을 저장합니다.

# 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱

일부 웹 애플리케이션은 쿼리 문자열을 사용하여 오리진에 정보를 전송합니다. 쿼리 문자열은 ? 문자 다음에 오는 웹 요청의 일부이며 이 문자열은 & 문자로 구분된 하나 이상의 파라미터를 포함할 수 있습니다. 다음 예의 쿼리 문자열에는 *`color=red`*와 *`size=large`*의 두 가지 파라미터가 포함되어 있습니다.

`https://d111111abcdef8.cloudfront.net/images/image.jpg?`*`color=red&size=large`*

배포의 경우 CloudFront에서 쿼리 문자열을 오리진에 전달하도록 선택할 수 있으며, 콘텐츠를 모든 파라미터를 기준으로 캐싱할지 아니면 일부 파라미터를 기준으로 캐싱할지를 선택합니다. 이 기능이 유용한 이유를 알아보려면 다음 예제를 살펴보십시오.

웹사이트가 5개 언어로 제공되는 경우를 가정하겠습니다. 5개 웹사이트 버전의 디렉터리 구조와 파일 이름은 동일합니다. 사용자가 웹사이트를 볼 때 CloudFront로 전달되는 요청에는 사용자가 선택한 언어를 기준으로 하는 언어 쿼리 문자열 파라미터가 포함됩니다. 쿼리 문자열을 오리진으로 전달하고 언어 파라미터를 기준으로 캐싱하도록 CloudFront를 구성할 수 있습니다. 웹 서버에서 선택한 언어에 해당하는 특정 페이지의 버전을 반환하도록 구성할 경우 CloudFront는 언어 쿼리 문자열 파라미터 값을 기준으로 각 언어 버전을 별도로 캐싱합니다.

이 예에서 웹 사이트의 메인 페이지가 `main.html`인 경우 다음과 같은 5개 요청이 발생하면 CloudFront에서 언어 쿼리 문자열 파라미터의 각 값에 대해 한 번씩 `main.html`을 다섯 번 캐싱합니다.

- `https://d111111abcdef8.cloudfront.net/main.html?language=de`
- `https://d111111abcdef8.cloudfront.net/main.html?language=en`
- `https://d111111abcdef8.cloudfront.net/main.html?language=es`
- `https://d111111abcdef8.cloudfront.net/main.html?language=fr`
- `https://d111111abcdef8.cloudfront.net/main.html?language=jp`

다음을 참조하십시오.

- 일부 HTTP 서버는 쿼리 문자열 파라미터를 처리하지 않으므로 파라미터 값을 기준으로 객체의 다른 버전을 반환하지 않습니다. 이러한 오리진의 경우 CloudFront에서 쿼리 문자열 파라미터를 오리진으로 전달하도록 구성할 경우 CloudFront는 오리진에서 모든 파라미터 값에 대해 CloudFront에 동일한 객체 버전을 반환할 경우에도 파라미터 값을 기준으로 캐싱합니다.
- 쿼리 문자열 파라미터가 위의 예제에서 설명한 대로 작동하려면 쿼리 문자열 파라미터 간에 & 문자를 구분 기호로 사용해야 합니다. 다른 구분 기호를 사용하는 경우에는 캐싱 기준으로 CloudFront에서 지정하는 파라미터 및 쿼리 문자열에서 파라미터가 나타나는 순서에 따라 결과가 달라질 수 있습니다.

  다음 예는 다른 구분 기호를 사용하고, `color` 파라미터에 따라서만 캐싱하도록 CloudFront를 구성할 경우의 결과를 보여줍니다.

  - 다음 요청에서 CloudFront는 `color` 파라미터 값을 기준으로 콘텐츠를 캐싱하지만 CloudFront는 이 값을 *`red;size=large`*로 해석합니다.

    ```
    https://d111111abcdef8.cloudfront.net/images/
    image.jpg?color=red;size=large
    ```

  - 다음 요청에서 CloudFront는 콘텐츠를 캐싱하지만 쿼리 문자열 파라미터를 기준으로 캐싱하지 않습니다. 그 이유는 CloudFront에서 `color` 파라미터를 기준으로 캐싱하도록 구성했지만 CloudFront는 다음 문자열에 *`large;color=red`* 값을 가진 `size` 파라미터만 포함된 것으로 해석하기 때문입니다.

    ```
    https://d111111abcdef8.cloudfront.net/images/
    image.jpg?size=large;color=red
    ```

CloudFront에서 다음 중 하나를 수행하도록 구성할 수 있습니다.

- 쿼리 문자열을 오리진에 전혀 전달하지 않음. 쿼리 문자열을 전달하지 않을 경우 CloudFront는 쿼리 문자열 파라미터를 기준으로 캐싱하지 않습니다.

- 쿼리 문자열을 오리진으로 전달하며 쿼리 문자열의 모든 파라미터를 기준으로 캐싱함
- 쿼리 문자열을 오리진으로 전달하며 쿼리 문자열 중 지정된 파라미터를 기준으로 캐싱함

자세한 내용은 the section called “캐싱 최적화” 섹션을 참조하세요.

주제

- 쿼리 문자열 전달 및 캐싱에 대한 콘솔 및 API 설정
- 캐싱 최적화
- 쿼리 문자열 파라미터 및 CloudFront 표준 로그(액세스 로그)

## 쿼리 문자열 전달 및 캐싱에 대한 콘솔 및 API 설정

CloudFront 콘솔에서 배포를 생성하면 CloudFront는 오리진 유형에 따라 쿼리 문자열 전달 및 캐싱을 구성합니다. 선택적으로 이러한 설정을 수동으로 편집할 수 있습니다. 자세한 내용은 the section called “모든 배포 설정”에서 다음 설정을 참조하세요.

- the section called “쿼리 문자열 전달 및 캐싱”
- the section called “쿼리 문자열 허용 목록”

CloudFront API를 사용하여 쿼리 문자열 전달 및 캐싱을 구성하려면 Amazon CloudFront API 참조의 CachePolicy 및 OriginRequestPolicy를 참조하세요.

## 캐싱 최적화

쿼리 문자열 파라미터를 기반으로 캐싱하도록 CloudFront를 구성하는 경우 다음 단계를 수행하여 CloudFront에서 오리진에 전달되는 요청 수를 줄일 수 있습니다. CloudFront 엣지 로케이션에서 객체를 제공하면 사용자에게 보다 엣지 로케이션에서 객체가 제공되므로 오리진 서버의 부하를 줄이고 지연 시간을 단축할 수 있습니다.

원본이 다른 버전의 객체를 반환하는 파라미터만 기반으로 캐시

웹 애플리케이션에서 CloudFront로 각 쿼리 문자열 파라미터를 전달할 때마다 CloudFront는 모든 파라미터 값에 대해 오리진에 요청을 전달하며 모든 파라미터 값에 대해 별도의 객체 버전을 캐싱합니다. 이는 오리진에서 파라미터 값과 상관없이 항상 같은 객체를 반환하는 경우라도 그렇습니다. 복수 파라미터의 경우 요청 수와 객체 수를 곱합니다.

오리진에서 다른 버전을 반환하는 쿼리 문자열 파라미터만 기준으로 캐싱하도록 CloudFront를 구성하고 각 파라미터를 기준으로 캐싱하는 이점을 주의 깊게 고려하는 것이 좋습니다. 예를 들어, 소매 웹사이트가 있다고 가정해봅시다. 6가지 색상의 자켓 사진이 있고 이 자켓은 10가지 사이즈로 제공됩니다. 가지고 있는 자켓 사진은 다른 색상을 보여주지만 다른 사이즈는 보여주지 않습니다. 캐싱을 최적화하려면 CloudFront에서 크기 파라미터가 아닌 색상 파라미터만 기준으로 캐싱하도록 구성해야 합니다. 이렇게 하면 CloudFront에서 캐시로부터 요청을 제공할 수 있는 가능성이 높아지고, 그에 따라 성능이 향상되며 오리진에 걸리는 부하가 줄어듭니다.

파라미터를 항상 동일한 순서로 나열

쿼리 문자열에서 파라미터의 순서는 중요합니다. 다음 예에서 쿼리 문자열은 파라미터 순서가 다른 것을 제외하고 동일합니다. 이 경우 CloudFront는 image.jpg에 대한 별도의 두 가지 요청을 오리진에 전달하고 객체의 두 가지 별도 버전을 캐싱합니다.

- `https://d111111abcdef8.cloudfront.net/images/image.jpg?color=red&size=large`
- `https://d111111abcdef8.cloudfront.net/images/image.jpg?size=large&color=red`

사전순과 같이 파라미터 이름을 항상 동일한 순서로 나열하는 것이 좋습니다.

파라미터 이름 및 값에 대해 항상 동일한 대소문자 사용

CloudFront는 쿼리 문자열 파라미터를 기준으로 캐싱할 때 파라미터 이름과 값의 대소문자를 고려합니다. 다음 예에서 쿼리 문자열은 파라미터 이름과 값의 대소문자가 다른 것을 제외하고 동일합니다. 이 경우 CloudFront는 image.jpg에 대한 별도의 네 가지 요청을 오리진에 전달하고 객체의 네 가지 별도 버전을 캐싱합니다.

- `https://d111111abcdef8.cloudfront.net/images/image.jpg?color=red`
- `https://d111111abcdef8.cloudfront.net/images/image.jpg?color=Red`
- `https://d111111abcdef8.cloudfront.net/images/image.jpg?Color=red`
- `https://d111111abcdef8.cloudfront.net/images/image.jpg?Color=Red`

파라미터 이름과 값을 모두 소문자로 하는 경우와 같이 대소문자를 일관적으로 사용하는 것이 좋습니다.

서명된 URL과 충돌하는 파라미터 이름은 사용하지 않음

서명된 URL을 사용하여 콘텐츠에 대한 액세스를 제한하는 경우(신뢰할 수 있는 서명자를 배포에 추가한 경우), CloudFront에서는 나머지 URL을 오리진에 전달하기 전에 다음 쿼리 문자열 파라미터를 제거합니다.

- `Expires`
- `Key-Pair-Id`
- `Policy`
- `Signature`

서명된 URL을 사용하면서 CloudFront에서 쿼리 문자열을 오리진에 전달하도록 구성하려는 경우 자체 쿼리 문자열 파라미터를 `Expires`, `Key-Pair-Id`, `Policy` 또는 `Signature`라는 이름으로 지정할 수 없습니다.

## 쿼리 문자열 파라미터 및 CloudFront 표준 로그(액세스 로그)

로깅을 활성화한 경우 CloudFront는 쿼리 문자열 파라미터를 포함해 전체 URL을 로깅합니다. 오리진에 쿼리 문자열을 전달하도록 CloudFront를 구성했는지와 관계없이 로깅합니다. CloudFront 로깅에 대한 자세한 내용은 the section called “액세스 로그(표준 로그)” 단원을 참조하십시오.

# 쿠키 기반의 콘텐츠 캐싱

CloudFront에서는 요청 및 응답을 처리할 때나 엣지 로케이션의 객체를 캐싱할 때 쿠키를 기본적으로 고려하지 않습니다. `Cookie` 헤더의 내용을 제외하고 동일한 두 개의 요청을 CloudFront가 수신하면 기본적으로 CloudFront는 요청을 동일하게 처리하고 두 요청에 대해 동일한 객체를 반환합니다.

최종 사용자 요청의 쿠키 일부 또는 전체를 오리진으로 전달하고, 전달되는 쿠키 값에 따라 별도의 객체 버전을 캐싱하도록 CloudFront를 구성할 수 있습니다. 이렇게 하면 CloudFront에서 최종 사용자 요청에서 전달하도록 구성된 일부 또는 전체 쿠키를 사용하여 캐시의 객체를 고유하게 식별합니다.

예를 들어, `locations.html`에 대한 요청에 `country` 또는 `uk` 값을 가진 `fr` 쿠키가 포함되어 있다고 가정합니다. `country` 쿠키 값을 기반으로 객체를 캐싱하도록 CloudFront를 구성할 경우, CloudFront에서는 `locations.html`에 대한 요청을 오리진에 전달하고 `country` 쿠키 및 쿠키 값을 포함합니다. 오리진에서는 `locations.html`을 반환하고 CloudFront에서는 `country` 쿠키의 값이 `uk`인 요청에 대해 한 번, 값이 `fr`인 요청에 대해 한 번, 객체를 캐싱합니다.

> **Important**
>
> Amazon S3 및 일부 HTTP 서버에서는 쿠키를 처리하지 않습니다. 쿠키를 처리하지 않거나 쿠키에 따라 다른 응답을 제공하지 않는 오리진에 쿠키를 전달하도록 CloudFront를 구성하지 마십시오. 그렇지 않으면 CloudFront가 동일한 객체에 대해 더 많은 요청을 오리진에 전달할 수 있으므로 성능이 저하되고 오리진의 로드가 증가합니다. 앞의 예제를 예로 들면, 오리진이

country 쿠키를 처리하지 않거나 locations.html 쿠키의 값에 관계없이 항상 동일한 버전의 country을 CloudFront로 반환하는 경우 해당 쿠키를 전달하도록 CloudFront를 구성하지 마십시오.
반대로 사용자 지정 오리진이 특정 쿠키에 의존하거나 쿠키에 따라 다른 응답을 보내는 경우 해당 쿠키를 오리진에 전달하도록 CloudFront를 구성해야 합니다. 그렇지 않을 경우 CloudFront는 오리진에 요청을 전달하기 전에 쿠키를 제거합니다.

쿠키 전달을 구성하려면 배포의 캐시 동작을 업데이트합니다. 캐시 동작에 대한 자세한 내용은 캐시 동작 설정(특히 쿠키 전달 및 허용 목록 쿠키 단원)를 참조하십시오.

다음 중 하나를 수행하도록 각 캐시 동작을 구성할 수 있습니다.

- 오리진에 모든 쿠키 전달 – CloudFront가 오리진에 요청을 전달할 때 최종 사용자가 보내는 모든 쿠키를 포함시킵니다. 오리진에서 응답을 반환할 때 CloudFront는 최종 사용자 요청의 쿠키 이름 및 쿠키 값을 사용하여 응답을 캐싱합니다. 오리진 응답에 Set-Cookie 헤더가 포함되면 CloudFront는 요청된 객체와 함께 최종 사용자에게 해당 헤더를 반환합니다. 또한 CloudFront는 오리진에서 반환된 객체와 함께 Set-Cookie 헤더를 캐싱하고 모든 캐시 적중 시 최종 사용자에게 해당 Set-Cookie 헤더를 보냅니다.
- 지정한 쿠키 세트 전달 – CloudFront는 요청을 원본에 전달하기 전에 뷰어가 보내는 쿠키 중 허용 목록에 없는 모든 쿠키를 제거합니다. CloudFront는 뷰어 요청에 나열된 쿠키 이름과 값을 사용하여 응답을 캐시합니다. 오리진 응답에 Set-Cookie 헤더가 포함되면 CloudFront는 요청된 객체와 함께 최종 사용자에게 해당 헤더를 반환합니다. 또한 CloudFront는 오리진에서 반환된 객체와 함께 Set-Cookie 헤더를 캐싱하고 모든 캐시 적중 시 최종 사용자에게 해당 Set-Cookie 헤더를 보냅니다.

  쿠키 이름에 와일드카드를 지정하는 방법에 대한 자세한 내용은 허용 목록 쿠키 단원을 참조하십시오.

  각 캐시 동작에 대해 전달할 수 있는 쿠키 이름 수에 대한 현재 할당량을 확인하거나 더 높은 할당량을 요청하려면 쿼리 문자열에 대한 할당량(레거시 캐시 설정)을 참조하십시오.
- 오리진에 쿠키를 전달하지 않음 – CloudFront는 최종 사용자가 보낸 쿠키에 따라 객체를 캐싱하지 않습니다. 또한 CloudFront는 오리진에 요청을 전달하기 전에 쿠키를 제거하고 최종 사용자에게 응답을 반환하기 전에 응답에서 Set-Cookie 헤더를 제거합니다. 이는 오리진 리소스를 사용하는 최적의 방법이 아니므로 이 캐시 동작을 선택할 때는 오리진이 기본적으로 오리진 응답에 쿠키를 포함하지 않도록 해야 합니다.

전달하려는 쿠키를 지정할 때는 다음 사항에 유의하십시오.

액세스 로그

요청 및 쿠키를 로그하도록 CloudFront를 구성하면 CloudFront에서 원본에 쿠키를 전달하지 않도록 구성하거나 특정 쿠키만 전달하도록 CloudFront를 구성한 경우에도 CloudFront는 모든 쿠키와 모든 쿠키 속성을 로그합니다. CloudFront 로깅에 대한 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하십시오.

대소문자 구분

쿠키 이름과 값은 모두 대소문자를 구분합니다. 예를 들어 CloudFront가 모든 쿠키를 전달하도록 구성되어 있고 동일한 객체에 대한 두 개의 최종 사용자 요청에 대소문자를 제외하고 동일한 쿠키가 있는 경우 CloudFront는 객체를 두 번 캐싱합니다.

CloudFront에서 쿠키 정렬

CloudFront가 쿠키(전체 또는 일부)를 전달하도록 구성된 경우 CloudFront에서는 요청을 원본에 전달하기 전에 쿠키 이름을 기준으로 쿠키를 자연스러운 순서로 정렬합니다.

> **Note**
>
> $ 문자로 시작하는 쿠키 이름은 지원되지 않습니다. CloudFront는 오리진에 요청을 전달하기 전에 쿠키를 제거합니다. $ 문자를 제거하거나 쿠키 이름의 시작 부분에 다른 문자를 지정할 수 있습니다.

### `If-Modified-Since` 및 `If-None-Match`

CloudFront가 쿠키(전체 또는 일부)를 전달하도록 구성된 경우 `If-Modified-Since` 및 `If-None-Match` 조건부 요청이 지원되지 않습니다.

표준 이름-값 페어 형식 필요

CloudFront에서는 값이 표준 이름-값 페어 형식을 따를 경우에만 쿠키 헤더를 전달합니다(예: `"Cookie: cookie1=value1; cookie2=value2"`).

**`Set-Cookie` 헤더 캐싱 비활성화**

CloudFront가 쿠키를 원본에 전달하도록 구성된 경우(전체 또는 특정 쿠키인지 여부에 관계없이) 원본 응답에서 수신된 `Set-Cookie` 헤더도 캐시합니다. CloudFront는 원래 최종 사용자의 응답에 이러한 `Set-Cookie` 헤더를 포함하며 CloudFront 캐시에서 제공되는 후속 응답에도 이러한 헤더를 포함합니다.

오리진에서 쿠키를 받지만 CloudFront가 오리진 응답에 `Set-Cookie` 헤더를 캐싱하지 않도록 하려면 `Cache-Control`를 필드 이름으로 지정하는 `no-cache` 지시문이 있는`Set-Cookie` 헤더를 추가하도록 오리진을 구성합니다. 예를 들면 `Cache-Control: no-cache="Set-Cookie"`입니다. 자세한 내용은 Hypertext Transfer Protocol(HTTP/1.1): 캐싱 표준의 응답 캐시 제어 지시문을 참조하십시오.

쿠키 이름의 최대 길이

특정 쿠키를 원본으로 전달하도록 CloudFront를 구성하는 경우 전달하도록 CloudFront를 구성하는 모든 쿠키 이름의 총 바이트 수는 512에서 전달하는 쿠키 수를 뺀 값을 초과할 수 없습니다. 예를 들어 오리진에 10개의 쿠키를 전달하도록 CloudFront를 구성하는 경우, 이 10개의 쿠키 이름들을 합친 길이는 502바이트(512-10)를 초과할 수 없습니다.

오리진에 전체 쿠키를 전달하도록 CloudFront를 구성하는 경우, 쿠키 이름의 길이는 문제가 되지 않습니다.

CloudFront에서 오리진에 쿠키를 전달하도록 CloudFront 콘솔을 사용하여 웹 배포를 업데이트하는 방법에 대한 자세한 내용은 배포 업데이트 단원을 참조하십시오. CloudFront API를 사용한 배포 업데이트에 대한 자세한 내용은 Amazon CloudFront API 참조의 배포 업데이트를 참조하십시오.

# 요청 헤더 기반의 콘텐츠 캐싱

CloudFront에서 헤더를 오리진으로 전송할지, 최종 사용자 요청의 헤더 값에 따라 지정된 객체의 개별 버전을 캐싱할지 여부를 선택할 수 있습니다. 이렇게 하면 사용자가 사용하는 디바이스, 최종 사용자의 위치, 최종 사용자가 사용하는 언어 및 다양한 조건에 따라 서로 다른 버전의 콘텐츠를 제공할 수 있습니다.

주제


- 헤더 및 배포 – 개요
- 캐싱의 기반이 되는 헤더 선택
- CORS 설정을 준수하도록 CloudFront 구성
- 디바이스 유형을 기반으로 캐싱하도록 구성
- 뷰어 언어를 기반으로 캐싱하도록 구성
- 뷰어 언어를 기반으로 캐싱하도록 구성
- 요청의 프로토콜을 기반으로 캐싱하도록 구성
- 압축 파일에 대한 캐싱 구성
- 헤더를 기반으로 한 캐싱이 성능에 미치는 영향

- 헤더 및 헤더 값의 대소문자가 캐싱에 미치는 영향
- CloudFront에서 최종 사용자에게 반환하는 헤더

## 헤더 및 배포 – 개요

기본적으로 CloudFront에서는 엣지 로케이션의 객체를 캐싱할 때 헤더를 고려하지 않습니다. 오리진에서 두 개의 객체를 반환하고 이들 객체는 요청 헤더의 값에만 차이가 있는 경우, CloudFront에서는 한 가지 버전의 객체만 캐싱합니다.

오리진에 헤더를 전달하도록 CloudFront를 구성할 수 있으며, 이러한 경우 CloudFront에서는 하나 이상의 요청 헤더 값에 따라 여러 버전의 객체를 캐싱합니다. 특정 헤더의 값에 따라 객체를 캐싱하기 위한 CloudFront를 구성하려면, 배포에 대한 캐시 동작 설정을 지정해야 합니다. 자세한 내용은 선택한 요청 헤더 기반의 캐시 단원을 참조하십시오.

예를 들어, `logo.jpg`에 대한 최종 사용자 요청에 `Product` 또는 `Acme` 값을 가진 사용자 지정 `Apex` 헤더가 포함되어 있다고 가정합니다. `Product` 헤더 값을 기반으로 객체를 캐싱하도록 CloudFront를 구성하는 경우, CloudFront에서는 `logo.jpg`에 대한 요청을 오리진에 전달하고 `Product` 헤더 및 헤더 값을 포함합니다. CloudFront에서는 `logo.jpg` 헤더의 값이 `Product`인 요청에 대해 한 번, 값이 `Acme`인 요청에 대해 한 번 `Apex`를 캐싱합니다.

다음 중 하나를 수행하도록 배포의 각 캐시 동작을 구성할 수 있습니다.

- 오리진에 전체 헤더를 전달합니다.

> **Note**
>
> 레거시 캐시 설정 - 전체 헤더를 오리진에 전달하도록 CloudFront를 구성한 경우, CloudFront에서는 이 캐시 동작과 연결된 객체를 캐싱하지 않습니다. 대신 각 요청을 오리진에 보냅니다.

- 지정한 헤더 목록을 전달합니다. CloudFront에서는 모든 지정된 헤더의 값을 기반으로 객체를 캐싱합니다. 또한 CloudFront에서는 기본적으로 전달하는 헤더를 모두 전달하지만 지정한 헤더를 기반으로만 객체를 캐싱합니다.
- 기본 헤더만 전달합니다. 이 구성의 경우 CloudFront에서는 요청 헤더의 값을 기반으로 객체를 캐싱하지 않습니다.

각 캐시 동작에 대해 전달할 수 있는 헤더 수에 대한 현재 할당량을 확인하거나 또는 더 높은 할당량을 요청하려면 헤더에 대한 할당량을 참조하십시오.

CloudFront에서 오리진에 헤더를 전달하도록 CloudFront 콘솔을 사용하여 웹 배포를 업데이트하는 방법에 대한 자세한 내용은 배포 업데이트 단원을 참조하십시오. CloudFront API를 사용한 기존 배포 업데이트에 대한 자세한 내용은 Amazon CloudFront API 참조의 배포 업데이트를 참조하십시오.

## 캐싱의 기반이 되는 헤더 선택

오리진에 전달할 수 있고 CloudFront에서 캐싱의 기반이 되는 헤더는 오리진이 Amazon S3 버킷인지 사용자 지정 오리진인지에 따라 달라집니다.

- Amazon S3 – 다수의 특정 헤더를 기반으로 객체를 전달하고 캐싱하도록 CloudFront를 구성할 수 있습니다(다음 예외 목록 참조). 그러나 교차 원본 리소스 공유(CORS)를 구현해야 하거나 원본 대면 이벤트에서 Lambda@Edge를 사용하여 콘텐츠를 개인화하려는 경우가 아니면 Amazon S3 원본으로 헤더를 전달하지 않는 것이 좋습니다.
  - CORS를 구성하려면 CloudFront가 CORS에 대해 활성화된 웹 사이트용 콘텐츠를 배포할 수 있도록 헤더를 전달해야 합니다. 자세한 내용은 CORS 설정을 준수하도록 CloudFront 구성 단원을 참조하세요.
  - Amazon S3 오리진에 전달된 헤더를 사용하여 콘텐츠를 개인 설정하려면 Lambda@Edge 함수를 작성 및 추가하고 이를 오리진 방향 이벤트에서 트리거되는 CloudFront 배포에 연결합니다. 콘텐츠를 개인 설정하기 위해 헤더를 처리하는 방법에 대한 자세한 내용은 국가 또는 디바이스 유형 헤더별 콘텐츠 개인화 - 예제 단원을 참조하십시오.

    추가 헤더를 전달하면 캐시 적중률이 감소할 수 있으므로 콘텐츠를 개인화하는 데 사용하지 않는 헤더는 전달하지 않는 것이 좋습니다. 즉, CloudFront는 모든 요청의 비율로서 엣지 캐시에서 제공되는 다수의 요청을 처리할 수 없습니다.
- 사용자 지정 오리진 – 다음과 같이 요청 헤더 값을 기반으로 캐싱하도록 CloudFront를 구성할 수 있습니다.
  - `Connection`
  - `Cookie` – 쿠키를 기반으로 전달 및 캐싱을 수행하려는 경우, 배포에 개별 설정을 사용합니다. 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하세요.
  - `Host (for Amazon S3 origins)`
  - `Proxy-Authorization`
  - `TE`
  - `Upgrade`

`Date` 및 `User-Agent` 헤더의 값에 따라 객체를 캐싱하도록 CloudFront를 구성할 수 있지만, 권장되지는 않습니다. 이러한 헤더는 여러 가지 값을 가질 수 있으며, 헤더의 값에 따른 캐싱으로 인해 CloudFront에서 오리진에 전달되는 요청이 눈에 띄게 증가할 수 있습니다.

HTTP 요청 헤더의 전체 목록과 CloudFront에서 이를 처리하는 방법은 HTTP 요청 헤더 및 CloudFront 동작(사용자 지정 및 Amazon S3 오리진) 단원을 참조하십시오.

## CORS 설정을 준수하도록 CloudFront 구성

Amazon S3 버킷 또는 사용자 지정 오리진에서 CORS를 활성화한 경우에는 CORS 설정을 준수하도록 전달할 특정 헤더를 선택해야 합니다. 오리진(Amazon S3 또는 사용자 지정)과 `OPTIONS` 응답의 캐싱 여부에 따라 서로 다른 헤더를 전달해야 합니다.

Amazon S3

- `OPTIONS` 응답을 캐싱하고 싶으면 다음을 수행하세요.
  - `OPTIONS` 응답에 대해 캐싱을 활성화하는 기본 캐시 동작 설정에 대한 옵션을 선택합니다.
  - `Origin`, `Access-Control-Request-Headers` 및 `Access-Control-Request-Method` 헤더를 전달하도록 CloudFront를 구성합니다.
- `OPTIONS` 응답을 캐싱하고 싶지 않으면 오리진에서 요청된 다른 모든 헤더와 함께 `Origin` 헤더를 전달하도록 CloudFront를 구성합니다(예: `Access-Control-Request-Headers`, `Access-Control-Request-Method`, 또는 기타).

사용자 지정 오리진 – 오리진에서 요구하는 나머지 헤더와 함께 `Origin` 헤더를 전달합니다.

CORS를 기반으로 응답을 캐싱하도록 CloudFront를 구성하려면 캐시 정책을 사용하여 헤더를 전달하도록 CloudFront를 구성해야 합니다. 자세한 내용은 정책으로 캐시 키 제어 섹션을 참조하세요.

CORS 및 Amazon S3에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 교차 원본 리소스 공유(CORS) 사용을 참조하세요.

## 디바이스 유형을 기반으로 캐싱하도록 구성

CloudFront에서 사용자가 콘텐츠를 보기 위해 사용 중인 디바이스에 따라 여러 버전의 객체를 캐싱하려는 경우, 다음과 같이 적용 가능한 헤더를 사용자 지정 오리진에 전달하도록 CloudFront를 구성합니다.

- `CloudFront-Is-Desktop-Viewer`

- `CloudFront-Is-Mobile-Viewer`
- `CloudFront-Is-SmartTV-Viewer`
- `CloudFront-Is-Tablet-Viewer`

`User-Agent` 헤더의 값에 따라 CloudFront에서는 요청을 오리진에 전달하기 전에 이러한 헤더의 값을 `true` 또는 `false`로 설정합니다. 디바이스가 둘 이상의 범주에 해당하는 경우 둘 이상의 값이 `true`일 수 있습니다. 예를 들어 일부 태블릿 디바이스의 경우 CloudFront에서는 `CloudFront-Is-Mobile-Viewer`와 `CloudFront-Is-Tablet-Viewer`를 모두 `true`로 설정할 수 있습니다.

## 뷰어 언어를 기반으로 캐싱하도록 구성

CloudFront에서 요청에 지정된 언어에 따라 여러 버전의 객체를 캐싱하려는 경우, `Accept-Language` 헤더를 오리진에 전달하도록 CloudFront를 구성합니다.

## 뷰어 언어를 기반으로 캐싱하도록 구성

CloudFront에서 요청이 시작된 국가에 따라 여러 버전의 객체를 캐싱하려는 경우, `CloudFront-Viewer-Country` 헤더를 오리진에 전달하도록 CloudFront를 구성합니다. CloudFront에서는 자동으로 요청이 시작된 IP 주소를 두 자로 된 국가 코드로 변환합니다. 코드와 국가 이름으로 정렬이 가능하고 사용이 간편한 국가 코드 목록에 대해서는 Wikipedia 항목 ISO 3166-1 alpha-2를 참조하십시오.

## 요청의 프로토콜을 기반으로 캐싱하도록 구성

CloudFront에서 HTTP 또는 HTTPS 중 요청의 프로토콜에 따라 여러 버전의 객체를 캐싱하려는 경우, `CloudFront-Forwarded-Proto` 헤더를 오리진에 전달하도록 CloudFront를 구성합니다.

## 압축 파일에 대한 캐싱 구성

원본이 Brotli 압축을 지원하는 경우 `Accept-Encoding` 헤더를 기반으로 캐시할 수 있습니다. 오리진이 이 헤더를 기반으로 하여 다른 콘텐츠를 제공하는 경우에만 `Accept-Encoding`을 기반으로 캐싱하도록 구성합니다.

## 헤더를 기반으로 한 캐싱이 성능에 미치는 영향

하나 이상의 헤더를 기반으로 캐싱하도록 CloudFront를 구성하고 그러한 헤더가 둘 이상의 값을 가질 수 있는 경우, CloudFront에서는 동일한 객체에 대해 더 많은 요청을 오리진 서버에 전달합니다. 이는 성능을 저하시키고 오리진 서버에 걸리는 부하를 가중시킵니다. 오리진 서버에서 지정된 헤더의 값에

상관없이 동일한 객체를 반환하는 경우, 해당 헤더를 기반으로 캐싱하도록 CloudFront를 구성하지 않는 것이 좋습니다.

둘 이상의 헤더를 전달하도록 CloudFront를 구성하는 경우, 최종 사용자 요청의 헤더 순서는 헤더의 값이 동일하다면 캐싱에 영향을 주지 않습니다. 예를 들어 하나의 요청에 A:1,B:2 헤더가 포함되어 있고 다른 요청에는 B:2,A:1이 포함되어 있는 경우, CloudFront에서는 하나의 객체 사본만 캐싱합니다.

## 헤더 및 헤더 값의 대소문자가 캐싱에 미치는 영향

CloudFront에서 헤더 값에 따라 캐싱을 수행할 경우 헤더 이름의 대소문자는 구별하지 않지만 헤더 값의 대소문자는 구별합니다.

- 최종 사용자 요청에 `Product:Acme` 및 `product:Acme`가 모두 포함된 경우, CloudFront에서는 객체를 한 번만 캐싱합니다. 이들은 헤더 이름의 대소문자만 다른 경우로, 캐싱에는 영향을 미치지 않습니다.
- 최종 사용자 요청에 `Product:Acme` 및 `Product:acme`가 모두 포함된 경우, CloudFront에서는 객체를 두 번 캐싱합니다. 이는 일부 요청에서는 값이 `Acme`이고 다른 요청에서는 `acme`이기 때문입니다.

## CloudFront에서 최종 사용자에게 반환하는 헤더

특정 헤더를 전달하고 캐싱하도록 CloudFront를 구성하는 것은 CloudFront에서 최종 사용자에게 반환하는 헤더의 종류에 영향을 미치지 않습니다. CloudFront에서는 오리진에서 가져온 헤더를 모두 반환하며, 여기에는 몇 가지 예외 사항이 적용됩니다. 자세한 내용은 관련 주제를 참조하십시오.

- Amazon S3 오리진 – CloudFront에서 제거하거나 업데이트하는 HTTP 응답 헤더 단원을 참조하십시오.
- 사용자 지정 오리진 – CloudFront에서 제거하거나 교체하는 HTTP 응답 헤더단원을 참조하십시오.

# 정책으로 캐시 키 제어

CloudFront 캐시 정책을 사용하여 CloudFront가 CloudFront 엣지 로케이션에 캐싱되는 객체의 캐시 키에 포함하는 HTTP 헤더, 쿠키 및 쿼리 문자열을 지정할 수 있습니다. 캐시 키는 캐시에 있는 모든 객체의 고유 식별자이며, 뷰어의 HTTP 요청이 캐시 적중으로 나타나는지 여부를 결정합니다.

캐시 적중은 최종 사용자 요청이 이전 요청과 동일한 캐시 키를 생성하고 해당 캐시 키의 객체가 엣지 로케이션 캐시에 있고 유효할 때 발생합니다. 캐시 적중이 있는 경우, 객체가 CloudFront 엣지 로케이션에서 최종 사용자에게 제공되므로 다음과 같은 이점이 있습니다.

- 오리진 서버의 부하 감소
- 최종 사용자에 대한 지연 시간 감소

캐시 키에 더 적은 수의 값을 포함하면 캐시 적중 가능성이 높아집니다. 캐시 적중률이 높아지므로(뷰어 요청 비율이 높을수록 캐시 적중이 발생함) 웹 사이트 또는 애플리케이션에서 더 나은 성능을 얻을 수 있습니다. 자세한 내용은 캐시 키 이해 섹션을 참조하세요.

캐시 키를 제어하려면 CloudFront 캐시 정책을 사용합니다. CloudFront 배포의 하나 이상의 캐시 동작에 캐시 정책을 연결합니다.

또한 캐시 정책을 사용하여 CloudFront 캐시의 객체에 대한 유지 시간(TTL) 설정을 지정하고 CloudFront에서 압축 객체를 요청하고 캐시하도록 할 수 있습니다.

> **Note**
>
> gRPC 트래픽은 캐시할 수 없으므로 캐시 설정은 gRPC 요청에 영향을 주지 않습니다. 자세한 내용은 CloudFront 배포에 gRPC 사용 섹션을 참조하세요.

주제


- 캐시 정책 이해
- 캐시 정책 생성
- 관리형 캐시 정책 사용
- 캐시 키 이해

# 캐시 정책 이해

캐시 정책을 사용하면 캐시 키에 포함된 값(URL 쿼리 문자열, HTTP 헤더 및 쿠키)을 제어하여 캐시 적중률을 향상시킬 수 있습니다. CloudFront 에서는 캐시 정책에 대해 미리 정의된 캐시 정책(관리형 정책이라고 함)을 제공합니다. 이러한 관리형 정책을 사용하거나 필요에 따라 자체 캐시 정책을 만들 수 있습니다. 관리형 정책에 대한 자세한 내용은 관리형 캐시 정책 사용 단원을 참조합니다.

캐시 정책에는 정책 정보, TTL(Time To Live) 설정 및 캐시 키 설정으로 분류되는 다음 설정이 포함되어 있습니다.

## 정책 정보

이름

캐시 정책을 식별하는 이름입니다. 콘솔에서 이름을 사용하여 캐시 정책을 캐시 동작에 연결합니다.

설명

캐시 정책에 대한 설명입니다. 이는 선택 사항이지만 캐시 정책의 용도를 식별하는 데 도움이 될 수 있습니다.

## TTL(Time To Live) 설정

TTL(Time To Live) 설정은 `Cache-Control` 및 `Expires` HTTP 헤더(오리진 응답에 있는 경우)와 함께 작동하여 CloudFront 캐시의 객체가 유효한 상태를 유지하는 기간을 결정합니다.

Minimum TTL

객체가 업데이트되었는지 여부를 확인하기 위해 CloudFront에서 오리진을 검사하기 전에 객체를 CloudFront 캐시에 유지할 최소 시간(초)입니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 섹션을 참조하세요.

> **Warning**
>
> 최소 TTL이 0보다 큰 경우 오리진 헤더에 `Cache-Control: no-cache`, `no-store` 또는 `private` 지시문이 있더라도 CloudFront는 적어도 캐시 정책의 최소 TTL에 지정된 기간 동안 콘텐츠를 캐싱합니다.

Maximum TTL

객체가 업데이트되었는지 여부를 확인하기 위해 CloudFront에서 오리진을 검사하기 전에 객체를 CloudFront 캐시에 유지할 최대 시간(초)입니다. CloudFront는 오리진이 객체와 함께 `Cache-Control` 또는 `Expires` 헤더를 전송하는 경우에만 이 설정을 사용합니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 섹션을 참조하세요.

기본 TTL

객체가 업데이트되었는지 여부를 확인하기 위해 CloudFront에서 오리진을 검사하기 전에 객체를 CloudFront 캐시에 유지할 기본 시간(초)입니다. CloudFront는 오리진이 객체와 함께 `Cache-Control` 또는 `Expires` 헤더를 전송하지 않는 경우에만 이 설정의 값을 객체의 TTL로 사용합니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 단원을 참조합니다.

> **Note**
>
> 최소 TTL, 최대 TTL 및 기본 TTL 설정이 모두 0으로 설정된 경우 CloudFront 캐싱이 비활성화됩니다.

## 캐시 키 설정

캐시 키 설정은 CloudFront에서 캐시 키에 포함하는 최종 사용자 요청의 값을 지정합니다. 이 값에는 URL 쿼리 문자열, HTTP 헤더 및 쿠키가 포함될 수 있습니다. 캐시 키에 포함하는 값은 CloudFront가 오리진으로 보내는 요청(오리진 요청이라고 함)에 자동으로 포함됩니다. 캐시 키에 영향을 주지 않고 오리진 요청을 제어하는 방법에 대한 자세한 내용은 정책을 통한 오리진 요청 제어 단원을 참조합니다.

캐시 키 설정은 다음과 같습니다.

- 헤더
- 쿠키
- 쿼리 문자열
- 압축 지원

헤더

CloudFront에서 캐시 키 및 오리진 요청에 포함하는 최종 사용자 요청의 HTTP 헤더입니다. 헤더의 경우 다음 설정 중 하나를 선택할 수 있습니다.

- None(없음) – 최종 사용자 요청의 HTTP 헤더가 캐시 키에 포함되어 있지 않고 오리진 요청에 자동으로 포함되지 않습니다.
- Include the following headers(다음 헤더 포함) – 캐시 키에 포함되고 오리진 요청에 자동으로 포함되는 뷰어 요청의 HTTP 헤더를 지정합니다.

Include the following headers(다음 헤더 포함) 설정을 사용하는 경우 HTTP 헤더를 해당 값이 아닌 이름으로 지정합니다. 예를 들어 다음 HTTP 헤더를 고려해 보세요.

```
Accept-Language: en-US,en;q=0.5
```

이 경우 헤더를 `Accept-Language: en-US,en;q=0.5`로 지정하지 않고 `Accept-Language`으로 지정합니다. 그러나 CloudFront는 캐시 키 및 오리진 요청에 해당 값을 포함한 전체 헤더를 포함합니다.

CloudFront에서 생성한 특정 헤더를 캐시 키에 포함할 수도 있습니다. 자세한 내용은 the section called “CloudFront 요청 헤더 추가” 단원을 참조합니다.

쿠키

CloudFront에서 캐시 키 및 오리진 요청에 포함하는 최종 사용자 요청의 쿠키입니다. 쿠키의 경우 다음 설정 중 하나를 선택할 수 있습니다.

- None(없음) – 최종 사용자 요청의 쿠키가 캐시 키에 포함되어 있지 않고 오리진 요청에 자동으로 포함되지 않습니다.
- All(모두) – 최종 사용자 요청의 모든 쿠키가 캐시 키에 포함되고 오리진 요청에 자동으로 포함됩니다.
- Include specified cookies(지정된 쿠키 포함) – 캐시 키에 포함되고 오리진 요청에 자동으로 포함되는 뷰어 사용자 요청의 쿠키를 지정합니다.
- Include all cookies except(일부 쿠키 제외) – 캐시 키에 포함되지 않고 오리진 요청에 자동으로 포함되지 않는 뷰어 요청의 쿠키를 지정합니다. 지정한 쿠키를 제외한 다른 모든 쿠키는 캐시 키에 포함되고 오리진 요청에 자동으로 포함됩니다.

Include specified cookies(지정된 쿠키 포함) 또는 Include all cookies except(일부 쿠키 제외) 설정을 사용하는 경우 값이 아닌 이름으로 쿠키를 지정합니다. 예를 들어 다음 `Cookie` 헤더를 고려해 보십시오.

```
Cookie: session_ID=abcd1234
```

이 경우 쿠키를 `session_ID=abcd1234`로 지정하지 않고 `session_ID`로 지정합니다. 그러나 CloudFront는 캐시 키 및 오리진 요청에 해당 값을 포함한 전체 쿠키를 포함합니다.

### 쿼리 문자열

CloudFront에서 캐시 키 및 오리진 요청에 포함하는 최종 사용자 요청의 URL 쿼리 문자열입니다. 쿼리 문자열의 경우 다음 설정 중 하나를 선택할 수 있습니다.

- None(없음) – 최종 사용자 요청의 쿼리 문자열이 캐시 키에 포함되어 있지 않고 오리진 요청에 자동으로 포함되지 않습니다.
- All(모두) – 최종 사용자 요청의 모든 쿼리 문자열이 캐시 키에 포함되고 오리진 요청에도 자동으로 포함됩니다.
- Include specified query strings(지정된 쿼리 문자열 포함) – 캐시 키에 포함되고 오리진 요청에 자동으로 포함되는 뷰어 요청의 쿼리 문자열을 지정합니다.
- Include all query strings except(일부 쿼리 문자열 제외) – 캐시 키에 포함되지 않고 오리진 요청에 자동으로 포함되지 않는 뷰어 요청의 쿼리 문자열을 지정합니다. 지정한 쿼리 문자열을 제외한 다른 모든 쿼리 문자열은 캐시 키에 포함되며 오리진 요청에 자동으로 포함됩니다.

Include specified query strings(지정된 쿼리 문자열 포함) 또는 Include all query strings except(일부 쿼리 문자열 제외) 설정을 사용하는 경우 값이 아닌 이름으로 쿼리 문자열을 지정합니다. 예를 들어 다음 URL 경로를 고려해 보세요.

```
/content/stories/example-story.html?split-pages=false
```

이 경우 쿼리 문자열을 `split-pages=false`가 아닌 `split-pages`로 지정합니다. 그러나 CloudFront는 캐시 키 및 오리진 요청에 해당 값을 포함한 전체 쿼리 문자열을 포함합니다.

> **Note**
>
> 캐시 키 설정의 경우 CloudFront는 헤더, 쿼리 문자열 및 쿠키의 별표 문자(*)를 와일드카드가 아닌 리터럴 문자열로 취급합니다.

### 압축 지원

이러한 설정을 사용하면 최종 사용자가 지원할 때 CloudFront에서 Gzip 또는 Brotli 압축 형식으로 압축된 객체를 요청하고 캐시할 수 있습니다. 이 설정을 사용하면 CloudFront 압축이 작동할 수도 있습니다. 최종 사용자는 `Accept-Encoding` HTTP 헤더를 사용하여 이러한 압축 형식에 대한 지원을 나타냅니다.

> **Note**
>
> Chrome 및 Firefox 웹 브라우저는 HTTPS를 사용하여 요청이 전송될 때만 Brotli 압축을 지원합니다. 이러한 브라우저는 HTTP 요청이 있는 Brotli를 지원하지 않습니다.

다음 중 하나에 해당하는 경우 이러한 설정을 활성화합니다.

- 오리진에서 최종 사용자가 지원할 때 Gzip 압축 객체를 반환합니다(요청에는 값이 `gzip`인 `Accept-Encoding` HTTP 헤더가 포함되어 있음). 이 경우 Gzip 활성화 설정(CloudFront API, AWS SDK AWS CLI 또는 CloudFormation에서 `EnableAcceptEncodingGzip`을 `true`로 설정)을 사용합니다.
- 오리진에서는 최종 사용자가 지원할 때 Brotli 압축 객체를 반환합니다(요청에는 값이 `br`인 `Accept-Encoding` HTTP 헤더가 포함되어 있음). 이 경우 Brotli 활성화 설정(CloudFront API, AWS SDK AWS CLI 또는 CloudFormation에서 `EnableAcceptEncodingBrotli`를 `true`로 설정)을 사용합니다.
- 이 캐시 정책이 연결된 캐시 동작은 CloudFront 압축을 사용하여 구성됩니다. 이 경우 Gzip이나 Brotli 또는 둘 다에 대해 캐싱을 활성화할 수 있습니다. CloudFront 압축이 활성화된 경우 두 형식 모두에 대해 캐싱을 활성화하면 인터넷으로 내보내는 데이터 송신 비용을 절감할 수 있습니다.

> **Note**
>
> 이러한 압축 형식 중 하나 또는 둘 모두에 대해 캐싱을 활성화한 경우 동일한 캐시 동작과 연결된 오리진 요청 정책에 `Accept-Encoding` 헤더를 포함하지 마십시오. CloudFront는 이러한 형식 중 하나에 대해 캐싱이 활성화된 경우 항상 오리진 요청에 `Accept-Encoding` 헤더를 포함하므로 이 헤더를 오리진 요청 정책에 포함해도 아무런 효과가 없습니다.

오리진 서버가 Gzip 또는 Brotli 압축 객체를 반환하지 않거나 캐시 동작이 CloudFront 압축으로 구성되지 않은 경우 압축된 객체에 대해 캐싱을 활성화하지 마세요. 이렇게 하면 캐시 적중률이 감소할 수 있습니다.

다음은 이러한 설정이 CloudFront 배포에 미치는 영향을 설명한 것입니다. 다음 시나리오는 모두 최종 사용자 요청에 `Accept-Encoding` 헤더가 포함되어 있다고 가정합니다. 최종 사용자 요청에 `Accept-Encoding` 헤더가 포함되지 않은 경우 CloudFront는 이 헤더를 캐시 키에 포함하지 않고 해당 오리진 요청에 포함하지 않습니다.

두 압축 형식에 대해 압축된 객체 캐싱이 활성화된 경우

최종 사용자가 Gzip과 Brotli를 모두 지원하는 경우, 즉 최종 사용자 요청의 `Accept-Encoding` 헤더에 `gzip` 및 `br` 값이 모두 있으면 CloudFront에서는 다음을 수행합니다.

- 헤더를 `Accept-Encoding: br,gzip`으로 정규화하고 캐시 키에 정규화된 헤더를 포함합니다. 캐시 키는 최종 사용자가 보낸 `Accept-Encoding` 헤더에 있던 다른 값을 포함하지 않습니다.
- 엣지 로케이션에 요청과 일치하고 만료되지 않은 Brotli 또는 Gzip 압축 객체가 있는 경우 엣지 로케이션에서 최종 사용자에게 객체를 반환합니다.
- 엣지 로케이션의 캐시에 요청과 일치하고 만료되지 않은 Brotli 또는 Gzip 압축 객체가 없는 경우 CloudFront는 정규화된 헤더(`Accept-Encoding: br,gzip`)를 해당 오리진 요청에 포함합니다. 오리진 요청에는 최종 사용자가 보낸 `Accept-Encoding` 헤더에 있던 다른 값이 포함되지 않습니다.

최종 사용자가 하나의 압축 형식을 지원하지만 다른 압축 형식은 지원하지 않는 경우(예: `gzip`이 최종 사용자 요청의 `Accept-Encoding` 헤더에 있는 값이지만 `br`은 아닌 경우) CloudFront에서는 다음을 수행합니다.

- 헤더를 `Accept-Encoding: gzip`으로 정규화하고 캐시 키에 정규화된 헤더를 포함합니다. 캐시 키는 최종 사용자가 보낸 `Accept-Encoding` 헤더에 있던 다른 값을 포함하지 않습니다.
- 엣지 로케이션의 캐시에 요청과 일치하고 만료되지 않은 Gzip 압축 객체가 있는 경우 엣지 로케이션에서 최종 사용자에게 객체를 반환합니다.
- 엣지 로케이션의 캐시에 요청과 일치하고 만료되지 않은 Gzip 압축 객체가 없는 경우 CloudFront는 해당 오리진 요청에 정규화된 헤더(`Accept-Encoding: gzip`)를 포함합니다. 오리진 요청에는 최종 사용자가 보낸 `Accept-Encoding` 헤더에 있던 다른 값이 포함되지 않습니다.

최종 사용자가 Brotli를 지원하지만 Gzip은 지원하지 않는 경우 CloudFront에서 수행할 작업을 이해하려면 앞의 예제에서 두 압축 형식을 서로 바꿉니다.

최종 사용자가 Brotli 또는 Gzip을 지원하지 않는 경우, 즉 최종 사용자 요청의 `Accept-Encoding` 헤더에 `br` 또는 `gzip` 값이 포함되지 않으면 CloudFront에서는 다음을 수행합니다.

- 캐시 키에 `Accept-Encoding` 헤더를 포함하지 않습니다.
- 해당 오리진 요청에 `Accept-Encoding: identity`를 포함합니다. 오리진 요청에는 최종 사용자가 보낸 `Accept-Encoding` 헤더에 있던 다른 값이 포함되지 않습니다.

하나의 압축 형식에 대해 압축된 객체 캐싱이 활성화되지만 다른 압축 형식에 대해서는 활성화되지 않는 경우

최종 사용자가 캐싱이 활성화된 형식을 지원하는 경우(예: Gzip에 대해 압축된 객체 캐싱이 활성화되어 있고 최종 사용자가 Gzip을 지원하는 경우 (`gzip`이 최종 사용자 요청의 `Accept-Encoding` 헤더에 있는 값 중 하나)) CloudFront에서는 다음을 수행합니다.

- 헤더를 `Accept-Encoding: gzip`으로 정규화하고 캐시 키에 정규화된 헤더를 포함합니다.
- 엣지 로케이션의 캐시에 요청과 일치하고 만료되지 않은 Gzip 압축 객체가 있는 경우 엣지 로케이션에서 최종 사용자에게 객체를 반환합니다.
- 엣지 로케이션의 캐시에 요청과 일치하고 만료되지 않은 Gzip 압축 객체가 없는 경우 CloudFront는 해당 오리진 요청에 정규화된 헤더(`Accept-Encoding: gzip`)를 포함합니다. 오리진 요청에는 최종 사용자가 보낸 `Accept-Encoding` 헤더에 있던 다른 값이 포함되지 않습니다.

이 시나리오에서는 Brotli에 대한 압축 객체 캐싱이 활성화되지 않기 때문에 최종 사용자가 Gzip과 Brotli(최종 사용자 요청의 `Accept-Encoding` 헤더에는 `gzip` 및 `br` 값이 모두 포함됨)를 모두 지원하는 경우와 동일합니다.

압축된 객체 캐싱이 Gzip이 아닌 Brotli에 대해 활성화되어 있는 경우 CloudFront에서 수행할 작업을 이해하려면 앞의 예제에서 두 압축 형식을 서로 바꿉니다.

최종 사용자가 캐싱이 활성화된 압축 형식을 지원하지 않는 경우(최종 사용자 요청의 `Accept-Encoding` 헤더에 해당 형식에 대한 값이 포함되지 않음) CloudFront에서는 다음을 수행합니다.

- 캐시 키에 `Accept-Encoding` 헤더를 포함하지 않습니다.
- 해당 오리진 요청에 `Accept-Encoding: identity`를 포함합니다. 오리진 요청에는 최종 사용자가 보낸 `Accept-Encoding` 헤더에 있던 다른 값이 포함되지 않습니다.

압축 객체 캐싱이 두 압축 형식 모두에 대해 비활성화된 경우

압축 객체 캐싱이 두 압축 형식 모두에 대해 비활성화된 경우 `Accept-Encoding` 헤더는 CloudFront에서 최종 사용자 요청에 있는 다른 HTTP 헤더와 동일하게 처리됩니다. 기본적으로 캐시 키에 포함되지 않으며 오리진 요청에 포함되지 않습니다. 캐시 정책의 헤더 목록 또는 다른 HTTP 헤더와 마찬가지로 오리진 요청 정책에 포함할 수 있습니다.

# 캐시 정책 생성

캐시 정책을 사용하면 캐시 키에 포함된 값(URL 쿼리 문자열, HTTP 헤더 및 쿠키)을 제어하여 캐시 적중률을 향상시킬 수 있습니다. AWS Command Line Interface(AWS CLI) 또는 CloudFront API를 사용하여 CloudFront 콘솔에서 캐시 정책을 생성할 수 있습니다.

캐시 정책을 생성한 후 CloudFront 배포의 하나 이상의 캐시 동작에 연결합니다.

Console

캐시 정책을 생성하는 방법(콘솔)

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudfront/v4/home?#/policies의 CloudFront 콘솔에서 [정책(Policies)] 페이지를 엽니다.
2. Create cache policy(캐시 정책 생성)를 선택합니다.
3. 이 캐시 정책에 대해 원하는 설정을 선택합니다. 자세한 내용은 캐시 정책 이해 섹션을 참조하세요.
4. 마친 후에는 Create(생성)를 선택합니다.

캐시 정책을 생성한 후 캐시 동작에 연결할 수 있습니다.

캐시 정책을 기존 배포에 연결하는 방법(콘솔)

1. https://console.aws.amazon.com/cloudfront/v4/home#/distributions의 CloudFront 콘솔에서 [Distributions(배포)] 페이지를 엽니다.
2. 업데이트할 배포를 선택한 다음 Behaviors(동작) 탭을 선택합니다.
3. 업데이트할 캐시 동작을 선택한 다음 Edit(편집)를 선택합니다.

   또는 새 캐시 동작을 생성하려면 Create Behavior(동작 생성)를 선택합니다.
4. Cache key and origin request(캐시 키 및 오리진 요청) 섹션에서 Cache policy and origin request policy(캐시 정책 및 오리진 요청 정책)가 선택되어 있는지 확인합니다.
5. Cache Policy(캐시 정책)의 경우 이 캐시 동작에 연결할 캐시 정책을 선택합니다.
6. 페이지 하단에서 Save changes(변경 사항 저장)를 선택합니다.

캐시 정책을 새 배포에 연결하는 방법(콘솔)

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.

2. 배포 생성을 선택합니다.
3. Cache key and origin request(캐시 키 및 오리진 요청) 섹션에서 Cache policy and origin request policy(캐시 정책 및 오리진 요청 정책)가 선택되어 있는지 확인합니다.
4. Cache Policy(캐시 정책)의 경우 이 배포의 기본 캐시 동작에 연결할 캐시 정책을 선택합니다.
5. 오리진, 기본 캐시 동작 및 기타 배포 설정에 대해 원하는 설정을 선택합니다. 자세한 내용은 모든 배포 설정 참조 섹션을 참조하세요.
6. 완료되면 Create Distribution(배포 생성)을 선택합니다.

CLI

AWS Command Line Interface(AWS CLI)를 사용하여 캐시 정책을 생성하려면 aws cloudfront create-cache-policy 명령을 사용합니다. 각 개별 파라미터를 명령줄 입력으로 지정하는 대신 입력 파일을 사용하여 명령의 입력 파라미터를 제공할 수 있습니다.

캐시 정책을 생성하는 방법(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 `cache-policy.yaml` 명령에 대한 모든 입력 파라미터가 포함된 create-cache-policy이라는 파일을 만듭니다.

```
aws cloudfront create-cache-policy --generate-cli-skeleton yaml-input > cache-policy.yaml
```

2. 방금 생성한 `cache-policy.yaml`이라는 파일을 엽니다. 파일을 편집하여 원하는 캐시 정책 설정을 지정한 다음 파일을 저장합니다. 파일에서 선택적 필드를 제거할 수 있지만 필수 필드는 제거하지 마세요.

   캐시 정책 설정에 대한 자세한 내용은 캐시 정책 이해 단원을 참조합니다.

3. 다음 명령을 사용하여 `cache-policy.yaml` 파일의 입력 파라미터로 캐시 정책을 만듭니다.

```
aws cloudfront create-cache-policy --cli-input-yaml file://cache-policy.yaml
```

   명령의 출력에 있는 `Id` 값을 기록해 둡니다. 캐시 정책 ID인 이 값은 캐시 정책을 CloudFront 배포의 캐시 동작에 연결하는 데 필요합니다.

캐시 정책을 기존 배포에 연결하는 방법(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 업데이트할 CloudFront 배포에 대한 배포 구성을 저장합니다. *distribution_ID*를 배포 ID로 바꿉니다.

```
aws cloudfront get-distribution-config --id distribution_ID --output yaml >
 dist-config.yaml
```

2. 방금 생성한 `dist-config.yaml`이라는 파일을 엽니다. 캐시 정책을 사용하도록 업데이트하려는 각 캐시 동작을 다음과 같이 변경하여 파일을 편집합니다.

   - 캐시 동작에서 `CachePolicyId`이라는 필드를 추가합니다. 필드 값에는 정책을 만든 후 기록한 캐시 정책 ID를 사용합니다.
   - 캐시 동작에서 `MinTTL`, `MaxTTL`, `DefaultTTL` 및 `ForwardedValues` 필드를 제거합니다. 이러한 설정은 캐시 정책에 지정되므로 이러한 필드와 캐시 정책을 동일한 캐시 동작에 포함할 수 없습니다.
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

   완료되면 파일을 저장합니다.

3. 캐시 정책을 사용하도록 배포를 업데이트하려면 다음 명령을 사용합니다. *distribution_ID*를 배포 ID로 바꿉니다.

```
aws cloudfront update-distribution --id distribution_ID --cli-input-yaml file://
dist-config.yaml
```

캐시 정책을 새 배포에 연결하는 방법(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 `distribution.yaml` 명령에 대한 모든 입력 파라미터가 포함된 create-distribution이라는 파일을 만듭니다.

```
aws cloudfront create-distribution --generate-cli-skeleton yaml-input >
 distribution.yaml
```

2. 방금 생성한 `distribution.yaml`이라는 파일을 엽니다. 기본 캐시 동작의 `CachePolicyId` 필드에 정책을 생성한 후 기록한 캐시 정책 ID를 입력합니다. 파일을 계속 편집하여 원하는 배포 설정을 지정한 다음 완료되면 파일을 저장합니다.

   배포 설정에 대한 자세한 내용은 모든 배포 설정 참조 단원을 참조하세요.
3. 다음 명령을 사용하여 `distribution.yaml` 파일의 입력 파라미터로 배포를 만듭니다.

```
aws cloudfront create-distribution --cli-input-yaml file://distribution.yaml
```

API

CloudFront API를 사용하여 캐시 정책을 생성하려면 CreateCachePolicy를 사용합니다. 이 API 호출에서 지정하는 필드에 대한 자세한 내용은 캐시 정책 이해 및 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

캐시 정책을 생성한 후 다음 API 호출 중 하나를 사용하여 캐시 동작에 연결할 수 있습니다.

- 기존 배포의 캐시 동작에 연결하려면 UpdateDistribution을 사용합니다.
- 새 배포의 캐시 동작에 연결하려면 CreateDistribution을 사용합니다.

이 두 API 호출에 대해 캐시 동작 내의 `CachePolicyId` 필드에 캐시 정책의 ID를 제공합니다. 이러한 API 호출에서 지정하는 다른 필드에 대한 자세한 내용은 모든 배포 설정 참조와 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

# 관리형 캐시 정책 사용

CloudFront 에서는 배포의 캐시 동작에 연결할 수 있는 일련의 관리형 캐시 정책을 제공합니다. 관리형 캐시 정책을 사용하면 자체 캐시 정책을 작성하거나 유지 관리할 필요가 없습니다. 관리형 정책은 특정 사용 사례에 최적화된 설정을 사용합니다.

관리형 캐시 정책을 사용하려면 배포의 캐시 동작에 이 정책을 연결합니다. 이 프로세스는 캐시 정책을 생성할 때와 동일하지만 새 캐시 정책을 생성하는 대신 관리형 캐시 정책 중 하나를 연결하기만 하면 됩니다. 이름(콘솔 사용) 또는 ID(AWS CLI 또는 SDK 사용)로 정책을 연결합니다. 이름과 ID는 다음 섹션에 나열되어 있습니다.

자세한 내용은 캐시 정책 생성 섹션을 참조하세요.

다음 항목에서는 사용할 수 있는 관리형 캐시 정책에 대해 설명합니다.

주제


- Amplify
- CachingDisabled
- CachingOptimized
- CachingOptimizedForUncompressedObjects
- Elemental-MediaPackage
- UseOriginCacheControlHeaders
- UseOriginCacheControlHeaders-QueryStrings


## Amplify

CloudFront 콘솔에서 이 정책 보기

이 정책은 AWS Amplify 웹 앱인 오리진과 함께 사용하도록 설계되었습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

```
2e54312d-136d-493c-8eb9-b001f22f67d2
```

이 정책에는 다음 설정이 포함되어 있습니다.

- 최소 TTL: 2초
- 최대 TTL: 600초(10분)
- 기본 TTL: 2초
- 캐시 키에 포함된 헤더:
  - `Authorization`
  - `CloudFront-Viewer-Country`
  - `Host`

  압축된 객체 캐시 설정이 활성화되어 있기 때문에 정규화된 `Accept-Encoding` 헤더도 포함됩니다. 자세한 내용은 압축 지원을 참조하세요.
- 캐시 키에 포함된 쿠키: 모든 쿠키가 포함됩니다.

- 캐시 키에 포함된 쿼리 문자열: 모든 쿼리 문자열이 포함됩니다.
- 압축된 객체 캐시 설정: 활성화됨. 자세한 내용은 압축 지원을 참조하세요.

> **Warning**
> 이 정책의 최소 TTL이 0보다 크므로 오리진 헤더에 `Cache-Control: no-cache`, `no-store` 또는 `private` 지시문이 있더라도 CloudFront는 적어도 캐시 정책의 최소 TTL에 지정된 기간 동안 콘텐츠를 캐싱합니다.

### AWS Amplify Hosting 캐시 정책

Amplify는 다음과 같은 관리형 캐시 정책을 사용하여 고객 애플리케이션의 기본 캐시 구성을 최적화합니다.

- Amplify-Default
- Amplify-DefaultNoCookies
- Amplify-ImageOptimization
- Amplify-StaticContent

> **Note**
> 이러한 정책은 Amplify에서만 사용됩니다. 이러한 정책을 배포에 사용하지 않는 것이 좋습니다.

Amplify 호스팅 애플리케이션의 캐시 구성 관리에 대한 자세한 내용은 Amplify Hosting 사용 설명서의 Managing cache configuration을 참조하세요.

## CachingDisabled

CloudFront 콘솔에서 이 정책 보기

이 정책은 캐싱을 비활성화합니다. 이 정책은 동적 콘텐츠 및 캐시할 수 없는 요청에 유용합니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`4135ea2d-6df8-44a3-9df3-4b5a84be39ad`

이 정책에는 다음 설정이 포함되어 있습니다.

- 최소 TTL: 0초
- 최대 TTL: 0초
- 기본 TTL: 0초
- 캐시 키에 포함된 헤더: 없음
- 캐시 키에 포함된 쿠키: 없음
- 캐시 키에 포함된 쿼리 문자열: 없음
- 압축된 객체 캐시 설정: 사용 안 함

## CachingOptimized

CloudFront 콘솔에서 이 정책 보기

이 정책은 CloudFront가 캐시 키에 포함된 값을 최소화하여 캐시 효율성을 최적화하도록 설계되었습니다. CloudFront는 캐시 키에 쿼리 문자열이나 쿠키를 포함하지 않으며 정규화된 `Accept-Encoding` 헤더만 포함합니다. 이렇게 하면 오리진에서 객체를 반환하거나 CloudFront 엣지 압축이 활성화된 경우 CloudFront에서 Gzip 및 Brotli 압축 형식의 객체를 별도로 캐시할 수 있습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`658327ea-f89d-4fab-a63d-7e88639e58f6`

이 정책에는 다음 설정이 포함되어 있습니다.

- 최소 TTL: 1초.
- 최대 TTL: 31,536,000초(365일).
- 기본 TTL: 86,400초(24시간).
- 캐시 키에 포함된 헤더: 명시적으로 포함되지 않습니다. 압축된 객체 캐시 설정이 활성화되어 있기 때문에 정규화된 `Accept-Encoding` 헤더가 포함됩니다. 자세한 내용은 압축 지원을 참조하세요.
- 캐시 키에 포함된 쿠키: 없음.
- 캐시 키에 포함된 쿼리 문자열: 없음.
- 압축된 객체 캐시 설정: 활성화됨. 자세한 내용은 압축 지원을 참조하세요.

> **Warning**
> 이 정책의 최소 TTL이 0보다 크므로 오리진 헤더에 `Cache-Control: no-cache, no-store` 또는 `private` 지시문이 있더라도 CloudFront는 적어도 캐시 정책의 최소 TTL에 지정된 기간 동안 콘텐츠를 캐싱합니다.

## CachingOptimizedForUncompressedObjects

CloudFront 콘솔에서 이 정책 보기

이 정책은 캐시 키에 포함된 값을 최소화하여 캐시 효율성을 최적화하도록 설계되었습니다. 쿼리 문자열, 헤더 또는 쿠키가 포함되지 않습니다. 이 정책은 이전 정책과 동일하지만 압축된 객체 캐시 설정을 비활성화합니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`b2884449-e4de-46a7-ac36-70bc7f1ddd6d`

이 정책에는 다음 설정이 포함되어 있습니다.

- 최소 TTL: 1초.
- 최대 TTL: 31,536,000초(365일).
- 기본 TTL: 86,400초(24시간).
- 캐시 키에 포함된 헤더: 없음
- 캐시 키에 포함된 쿠키: 없음
- 캐시 키에 포함된 쿼리 문자열: 없음
- 압축된 객체 캐시 설정: 사용 안 함

> **Warning**
> 이 정책의 최소 TTL이 0보다 크므로 오리진 헤더에 `Cache-Control: no-cache, no-store` 또는 `private` 지시문이 있더라도 CloudFront는 적어도 캐시 정책의 최소 TTL에 지정된 기간 동안 콘텐츠를 캐싱합니다.

## Elemental-MediaPackage

CloudFront 콘솔에서 이 정책 보기

이 정책은 AWS Elemental MediaPackage 엔드포인트인 오리진과 함께 사용하도록 설계되었습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`08627262-05a9-4f76-9ded-b50ca2e3a84f`

이 정책에는 다음 설정이 포함되어 있습니다.

- 최소 TTL: 0초
- 최대 TTL: 31,536,000초(365일).
- 기본 TTL: 86,400초(24시간).
- 캐시 키에 포함된 헤더:
  - `Origin`

  압축된 객체 캐시 설정이 Gzip에 대해 활성화되어 있기 때문에 정규화된 `Accept-Encoding` 헤더도 포함됩니다. 자세한 내용은 압축 지원을 참조하세요.
- 캐시 키에 포함된 쿠키: 없음
- 캐시 키에 포함된 쿼리 문자열:
  - `aws.manifestfilter`
  - `start`
  - `end`
  - `m`
- 압축된 객체 캐시 설정: Gzip에 대해 활성화되었습니다. 자세한 내용은 압축 지원을 참조하세요.

## UseOriginCacheControlHeaders

CloudFront 콘솔에서 이 정책 보기

이 정책은 `Cache-Control` HTTP 응답 헤더를 반환하고 쿼리 문자열에 있는 값에 따라 다른 콘텐츠를 제공하지 않는 오리진에 사용하도록 설계되었습니다. 오리진이 이 쿼리 문자열에 있는 값을 기반으로 하여 다른 콘텐츠를 제공하는 경우 UseOriginCacheControlHeaders-QueryStrings를 사용합니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`83da9c7e-98b4-4e11-a168-04f0df8e2c65`

이 정책에는 다음 설정이 포함되어 있습니다.

- 최소 TTL: 0초
- 최대 TTL: 31,536,000초(365일).
- 기본 TTL: 0초
- 캐시 키에 포함된 헤더:
  - `Host`
  - `Origin`
  - `X-HTTP-Method-Override`
  - `X-HTTP-Method`
  - `X-Method-Override`

  압축된 객체 캐시 설정이 활성화되어 있기 때문에 정규화된 `Accept-Encoding` 헤더도 포함됩니다. 자세한 내용은 압축 지원을 참조하세요.
- 캐시 키에 포함된 쿠키: 모든 쿠키가 포함됩니다.
- 캐시 키에 포함된 쿼리 문자열: 없음.
- 압축된 객체 캐시 설정: 활성화됨. 자세한 내용은 압축 지원을 참조하세요.

## UseOriginCacheControlHeaders-QueryStrings

CloudFront 콘솔에서 이 정책 보기

이 정책은 `Cache-Control` HTTP 응답 헤더를 반환하고 쿼리 문자열에 있는 값에 따라 다른 콘텐츠를 제공하는 오리진에 사용하도록 설계되었습니다. 오리진이 이 쿼리 문자열에 있는 값을 기반으로 하여 다른 콘텐츠를 제공하지 는 경우 UseOriginCacheControlHeaders를 사용합니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`4cc15a8a-d715-48a4-82b8-cc0b614638fe`

이 정책에는 다음 설정이 포함되어 있습니다.

- 최소 TTL: 0초
- 최대 TTL: 31,536,000초(365일).

- 기본 TTL: 0초
- 캐시 키에 포함된 헤더:
  - `Host`
  - `Origin`
  - `X-HTTP-Method-Override`
  - `X-HTTP-Method`
  - `X-Method-Override`

  압축된 객체 캐시 설정이 활성화되어 있기 때문에 정규화된 `Accept-Encoding` 헤더도 포함됩니다. 자세한 내용은 압축 지원을 참조하세요.
- 캐시 키에 포함된 쿠키: 모든 쿠키가 포함됩니다.
- 캐시 키에 포함된 쿼리 문자열: 모든 쿼리 문자열이 포함됩니다.
- 압축된 객체 캐시 설정: 활성화됨. 자세한 내용은 압축 지원을 참조하세요.

# 캐시 키 이해

캐시 키는 CloudFront 엣지 로케이션에 대한 최종 사용자 요청으로 캐시 적중이 발생하는지 여부를 확인합니다. 캐시 키는 캐시에 있는 객체의 고유 식별자입니다. 각 캐시 객체에는 고유한 캐시 키가 있습니다.

캐시 적중은 최종 사용자 요청이 이전 요청과 동일한 캐시 키를 생성하고 해당 캐시 키의 객체가 엣지 로케이션 캐시에 있고 유효할 때 발생합니다. 캐시 적중이 있는 경우, 요청된 객체가 CloudFront 엣지 로케이션에서 최종 사용자에게 제공되므로 다음과 같은 이점이 있습니다.

- 오리진 서버의 부하 감소
- 최종 사용자에 대한 지연 시간 감소

캐시 적중률이 높으면(최종 사용자 요청 비율이 높을수록 캐시 적중이 발생함) 웹 사이트 또는 애플리케이션 성능이 향상될 수 있습니다. 캐시 적중률을 향상시키는 한 가지 방법은 캐시 키에 필요한 최소 값만 포함시키는 것입니다. 자세한 내용은 다음 단원을 참조하세요.

캐시 정책을 사용하여 캐시 키의 값(URL 쿼리 문자열, HTTP 헤더 및 쿠키)을 수정할 수 있습니다. (뷰어 요청에서 Lambda@Edgefunction 또는 CloudFront Functions을 사용하여 캐시 키를 수정할 수도 있습니다.) 캐시 키를 수정하기 전에 애플리케이션의 설계 방식과 최종 사용자 요청의 특성에 따라 다양한 응답을 제공할 수 있는 시기 및 방법을 이해하는 것이 중요합니다. 최종 사용자 요청 값으로 오리진

에서 반환하는 응답을 확인하는 경우 해당 값을 캐시 키에 포함해야 합니다. 그러나 오리진이 반환하는 응답에 영향을 미치지 않는 값을 캐시 키에 포함하면 중복 객체를 캐시하게 될 수 있습니다.

## 기본 캐시 키

기본적으로 CloudFront 배포의 캐시 키에는 다음 정보가 포함됩니다.

- CloudFront 배포의 도메인 이름(예: d111111abcdef8.cloudfront.net)
- 요청된 객체의 URL 경로(예: `/content/stories/example-story.html`)

> **Note**
>
> `OPTIONS` 메서드는 `OPTIONS` 요청에 대한 캐시 키에 포함됩니다. 이는 `OPTIONS` 요청에 대한 응답이 `GET` 및 `HEAD` 요청에 대한 응답과 별도로 캐시됨을 의미합니다.

최종 사용자 요청의 다른 값은 기본적으로 캐시 키에 포함되지 않습니다. 웹 브라우저의 다음 HTTP 요청을 검토합니다.

```
            GET /content/stories/example-story.html?ref=0123abc&split-pages=false
HTTP/1.1
            Host: d111111abcdef8.cloudfront.net
            User-Agent: Mozilla/5.0 Gecko/20100101 Firefox/68.0
            Accept: text/html,*/*
            Accept-Language: en-US,en
            Cookie: session_id=01234abcd
            Referer: https://news.example.com/
```

CloudFront 엣지 로케이션에 이 최종 사용자 요청이 수신되면 CloudFront는 캐시 키를 사용하여 캐시 적중 여부를 확인합니다. 기본적으로 요청의 다음 구성 요소만 캐시 키 `/content/stories/example-story.html` 및 `d111111abcdef8.cloudfront.net`에 포함됩니다. 요청된 객체가 캐시에 없으면(캐시 누락) CloudFront는 오리진에 요청을 보내 객체를 가져옵니다. 객체를 가져온 후 CloudFront는 최종 사용자에게 반환하고 엣지 로케이션의 캐시에 저장합니다.

CloudFront가 캐시 키에 의해 확인된 동일한 객체에 대한 다른 요청을 수신하면 CloudFront는 오리진에 요청을 보내지 않고 캐싱된 객체를 최종 사용자에게 즉시 제공합니다. 예를 들어, 이전 요청 이후에 들어오는 다음 HTTP 요청을 검토합니다.

```
            GET /content/stories/example-story.html?ref=xyz987&split-pages=true
HTTP/1.1
            Host: d111111abcdef8.cloudfront.net
            User-Agent: Mozilla/5.0 AppleWebKit/537.36 Chrome/83.0.4103.116
            Accept: text/html,*/*
            Accept-Language: en-US,en
            Cookie: session_id=wxyz9876
            Referer: https://rss.news.example.net/
```

이 요청은 이전 요청과 동일한 객체에 대한 요청이지만 이전 요청과는 다릅니다. URL 쿼리 문자열, `User-Agent` 및 `Referer` 헤더 및 `session_id` 쿠키가 다릅니다. 그러나 기본적으로 이러한 값은 캐시 키의 일부가 아니므로 이 두 번째 요청으로 인해 캐시 적중이 발생합니다.

## 캐시 키 사용자 지정

경우에 따라 캐시 적중이 줄어들더라도 캐시 키에 더 많은 정보를 포함하고 싶을 수도 있습니다. 캐시 정책을 사용하여 캐시 키에 포함할 항목을 지정합니다.

예를 들어 오리진 서버에서 최종 사용자 요청의 `Accept-Language` HTTP 헤더를 사용하여 최종 사용자 언어에 따라 다른 콘텐츠를 반환하는 경우 이 헤더를 캐시 키에 포함할 수 있습니다. 이렇게 하면 CloudFront에서는 이 헤더를 사용하여 캐시 적중을 확인하고 오리진 요청(캐시 누락이 있을 때 CloudFront가 오리진으로 보내는 요청)에 헤더를 포함합니다.

캐시 키에 추가 값을 포함하면 최종 사용자 요청에서 발생할 수 있는 변형 때문에 CloudFront에서 중복 객체를 캐시할 수 있습니다. 예를 들어 최종 사용자는 `Accept-Language` 헤더에 대해 다음 값 중 하나를 보낼 수 있습니다.

- `en-US,en`
- `en,en-US`
- `en-US, en`
- `en-US`

이 모든 다양한 값은 최종 사용자 언어가 영어라는 것을 나타내지만 변형으로 인해 CloudFront에서 동일한 객체를 여러 번 캐시할 수 있습니다. 이렇게 하면 캐시 적중을 줄이고 오리진 요청 수를 늘릴 수 있습니다. 캐시 키에 `Accept-Language` 헤더를 포함하지 않고 다른 언어로 된 콘텐츠에 서로 다른 URL을 사용하도록 웹 사이트 또는 애플리케이션을 구성하여 이러한 중복을 피할 수 있습니다(예: `/en-US/content/stories/example-story.html`).

캐시 키에 포함하려는 특정 값의 경우, 최종 사용자 요청에 나타날 수 있는 해당 값의 변형 수를 이해하는지 확인해야 합니다. 캐시 키에 특정 요청 값을 포함하는 것은 큰 의미가 없습니다. 예를 들어 `User-Agent` 헤더에는 수천 개의 고유한 변형이 있을 수 있으므로 일반적으로 캐시 키에 포함하기에는 적합하지 않습니다. 사용자별 또는 세션별 값이 있으며 수천(또는 수백만) 개의 요청에 걸쳐 고유한 쿠키도 캐시 키에 포함하기에는 적합하지 않습니다. 캐시 키에 이러한 값을 포함하면 고유한 각 변형으로 인해 캐시에 객체의 다른 복사본이 생성됩니다. 이 객체 복사본이 고유하지 않거나 객체 수가 너무 많아서 객체마다 캐시 적중 횟수가 적으면 다른 방법을 고려해 보는 것이 좋습니다. 변수가 많은 값을 캐시 키에서 제외하거나 객체를 캐시에 사용할 수 없도록 표시할 수 있습니다.

캐시 키를 사용자 지정할 때는 주의하세요. 경우에 따라 중복 객체를 캐시하고 캐시 적중률을 낮추며 오리진 요청 횟수를 늘리는 등의 의도하지 않은 결과가 발생할 수 있습니다. 오리진 웹 사이트 또는 애플리케이션에서 분석, 원격 측정 또는 기타 용도에 대한 최종 사용자 요청의 특정 값을 수신해야 하지만 이 값이 오리진에서 반환하는 객체를 변경하지 않는 경우, 오리진 요청 정책을 사용하여 이 값을 오리진 요청에 포함하지만 캐시 키에는 포함하지 않습니다.

# 정책을 통한 오리진 요청 제어

CloudFront에 대한 최종 사용자 요청이 캐시 누락로 나타나는 경우(요청된 객체가 엣지 로케이션에 캐싱되지 않음) CloudFront는 오리진에 요청을 보내 객체를 검색합니다. 이를 오리진 요청이라고 합니다. 오리진 요청에는 다음과 같은 최종 사용자 요청의 정보가 항상 포함됩니다.

- URL 경로(경로 전용, URL 쿼리 문자열 또는 도메인 이름 제외)
- 요청 본문(있는 경우)
- CloudFront에서 모든 오리진 요청에 자동으로 포함하는 HTTP 헤더(`Host`, `User-Agent`, `X-Amz-Cf-Id` 등)

URL 쿼리 문자열, HTTP 헤더 및 쿠키와 같은 최종 사용자 요청의 기타 정보는 기본적으로 오리진 요청에 포함되지 않습니다. (예외: 레거시 캐시 설정의 경우 CloudFront는 기본적으로 헤더를 오리진에 전달합니다.) 그러나 분석 또는 원격 분석을 위해 데이터를 수집하는 등 오리진에서 이러한 다른 정보 중 일부를 수신할 수 있습니다. 오리진 요청 정책을 사용하여 오리진 요청에 포함된 정보를 제어할 수 있습니다.

오리진 요청 정책은 캐시 키를 제어하는 캐시 정책과 별개입니다. 이를 통해 오리진에서 추가 정보를 수신하고 적절한 캐시 적중률(캐시 적중으로 나타나는 뷰어 요청의 비율)을 유지할 수 있습니다. 오리진 요청에 포함되는 정보(오리진 요청 정책 사용)와 캐시 키에 포함되는 정보(캐시 정책 사용)를 별도로 제어하면 됩니다.

두 종류의 정책은 별개이지만 서로 관련이 있습니다. 캐시 정책을 사용하여 캐시 키에 포함하는 모든 URL 쿼리 문자열, HTTP 헤더 및 쿠키는 오리진 요청에 자동으로 포함됩니다. 오리진 요청 정책을 사용하여 오리진 요청에 포함하지만 캐시 키에는 포함하지 않을 정보를 지정합니다. 캐시 정책과 마찬가지로 오리진 요청 정책을 CloudFront 배포에 있는 하나 이상의 캐시 동작에 연결합니다.

오리진 요청 정책을 사용하여 최종 사용자 요청에 포함되지 않은 오리진 요청에 HTTP 헤더를 추가할 수도 있습니다. 이러한 추가 헤더는 오리진 요청을 보내기 전에 CloudFront에서 추가되며, 헤더 값은 최종 사용자 요청에 따라 자동으로 결정됩니다. 자세한 내용은 the section called “CloudFront 요청 헤더 추가” 섹션을 참조하세요.

주제


- 오리진 요청 정책 이해
- 오리진 요청 정책 생성
- 관리형 오리진 요청 정책 사용

- CloudFront 요청 헤더 추가
- 오리진 요청 정책과 캐시 정책이 함께 작동하는 방식 이해

# 오리진 요청 정책 이해

CloudFront에서는 일반 사용 사례에 대해 미리 정의된 오리진 요청 정책(관리형 정책이라고 함)을 제공합니다. 이러한 관리형 정책을 사용하거나 필요에 따라 자체 오리진 요청 정책을 생성할 수 있습니다. 관리형 정책에 대한 자세한 내용은 관리형 오리진 요청 정책 사용 단원을 참조합니다.

오리진 요청 정책에는 정책 정보 및 오리진 요청 설정으로 분류되는 다음과 같은 설정이 포함되어 있습니다.

## 정책 정보

이름

오리진 요청 정책을 식별하는 이름입니다. 콘솔에서 이름을 사용하여 오리진 요청 정책을 캐시 동작에 연결합니다.

설명

오리진 요청 정책에 대한 설명입니다. 이는 선택 사항입니다.

## 오리진 요청 설정

오리진 요청 설정은 CloudFront에서 오리진으로 보내는 요청(오리진 요청이라고 함)에 포함된 최종 사용자 요청의 값을 지정합니다. 이 값에는 URL 쿼리 문자열, HTTP 헤더 및 쿠키가 포함될 수 있습니다. 지정한 값은 오리진 요청에 포함되지만 캐시 키에는 포함되지 않습니다. 캐시 키 제어에 대한 자세한 내용은 정책으로 캐시 키 제어 단원을 참조하세요.

헤더

CloudFront에서 오리진 요청에 포함하는 최종 사용자 요청의 HTTP 헤더입니다. 헤더의 경우 다음 설정 중 하나를 선택할 수 있습니다.

- 없음(None) - 최종 사용자 요청의 HTTP 헤더가 오리진 요청에 포함되지 않습니다.
- 모든 최종 사용자 헤더(All viewer header) - 최종 사용자 요청의 모든 HTTP 헤더가 오리진 요청에 포함됩니다.
- All viewer headers and the following CloudFront headers(모든 뷰어 헤더 및 다음의 CloudFront 헤더) - 뷰어 요청의 모든 HTTP 헤더가 오리진 요청에 포함됩니다. 또한 오리진 요청에 추가

할 CloudFront 헤더를 지정합니다. CloudFront 헤더에 대한 자세한 내용은 the section called "CloudFront 요청 헤더 추가" 단원을 참조하세요.

- Include the following headers(다음 헤더 포함) - 오리진 요청에 포함할 HTTP 헤더를 지정합니다.

> **Note**
>
> Origin 커스텀 헤더 설정에 이미 포함되어 있는 헤더는 지정하지 마세요. 자세한 내용은 사용자 지정 헤더를 오리진 요청에 추가하도록 CloudFront 구성 섹션을 참조하세요.

- 다음을 제외한 모든 뷰어 헤더 - 오리진 요청에 포함하지 않을 HTTP 헤더를 지정합니다. 지정된 헤더를 제외한 뷰어 요청의 다른 HTTP 헤더가 모두 포함됩니다.

모든 뷰어 헤더 및 다음 CloudFront 헤더, 다음 헤더 포함 또는 다음을 제외한 모든 뷰어 헤더 설정을 사용하는 경우 HTTP 헤더를 이름으로만 지정합니다. CloudFront는 오리진 요청에 값을 비롯한 전체 헤더를 포함합니다.

> **Note**
>
> 뷰어의 `Host` 헤더를 제거하기 위해 다음을 제외한 모든 뷰어 헤더 설정을 사용하면 CloudFront는 오리진의 도메인 이름이 포함된 새 `Host` 헤더를 오리진 요청에 추가합니다.

쿠키

CloudFront에서 오리진 요청에 포함하는 최종 사용자 요청의 쿠키입니다. 쿠키의 경우 다음 설정 중 하나를 선택할 수 있습니다.

- 없음(None) - 최종 사용자 요청의 쿠키가 오리진 요청에 포함되지 않습니다.
- 모두(All) - 최종 사용자 요청의 모든 쿠키가 오리진 요청에 포함됩니다.
- 다음 쿠키 포함 – 오리진 요청에 포함할 뷰어 요청의 쿠키를 지정합니다.
- 다음을 제외한 모든 쿠키 - 뷰어 요청의 쿠키 중 오리진 요청에 포함하지 않을 쿠키를 지정합니다. 뷰어 요청의 다른 쿠키가 모두 포함됩니다.

다음 쿠키 포함 또는 다음을 제외한 모든 쿠키 설정을 사용하는 경우 이름으로만 쿠키를 지정합니다. CloudFront는 오리진 요청에 값을 비롯한 전체 쿠키를 포함합니다.

쿼리 문자열

CloudFront에서 오리진 요청에 포함하는 최종 사용자 요청의 URL 쿼리 문자열입니다. 쿼리 문자열의 경우 다음 설정 중 하나를 선택할 수 있습니다.

- 없음(None) - 최종 사용자 요청의 쿼리 문자열이 오리진 요청에 포함되지 않습니다.
- 모두(All) - 최종 사용자 요청의 모든 쿼리 문자열이 오리진 요청에 포함됩니다.
- 다음 쿼리 문자열 포함 – 오리진 요청에 포함할 뷰어 요청의 쿼리 문자열을 지정합니다.
- 다음을 제외한 모든 쿼리 문자열 – 오리진 요청에 포함하지 않을 뷰어 요청의 쿼리 문자열을 지정합니다. 다른 모든 쿼리 문자열이 포함됩니다.

다음 쿼리 문자열 포함 또는 다음을 제외한 모든 쿼리 문자열 설정을 사용하는 경우 이름으로만 쿼리 문자열을 지정합니다. CloudFront는 오리진 요청에 값을 비롯한 전체 쿼리 문자열을 포함합니다.

## 오리진 요청 정책 생성

오리진 요청 정책을 사용하여 CloudFront에서 오리진으로 보내는 요청에 포함된 값(URL 쿼리 문자열, HTTP 헤더 및 쿠키)을 제어할 수 있습니다. AWS Command Line Interface(AWS CLI) 또는 CloudFront API를 사용하여 CloudFront 콘솔에서 오리진 요청 정책을 생성할 수 있습니다.

오리진 요청 정책을 생성한 후 CloudFront 배포의 하나 이상의 캐시 동작에 이를 연결합니다.

오리진 요청 정책은 필요하지 않습니다. 캐시 동작에 오리진 요청 정책이 연결되어 있지 않으면 오리진 요청에는 캐시 정책에 지정된 모든 값이 포함되지만 그 이상은 포함되지 않습니다.

> **Note**
>
> 오리진 요청 정책을 사용하려면 캐시 동작도 캐시 정책을 사용해야 합니다. 캐시 정책 없이 캐시 동작에 오리진 요청 정책을 사용할 수 없습니다.

Console

오리진 요청 정책을 생성하는 방법(콘솔)

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudfront/v4/home?#/policies의 CloudFront 콘솔에서 [정책(Policies)] 페이지를 엽니다.
2. Origin request(오리진 요청)를 선택한 다음 Create origin request policy(오리진 요청 정책 생성)를 선택합니다.
3. 이 오리진 요청 정책에 대해 원하는 설정을 선택합니다. 자세한 내용은 오리진 요청 정책 이해 섹션을 참조하세요.

4. 마친 후에는 Create(생성)를 선택합니다.

오리진 요청 정책을 생성한 후 이를 캐시 동작에 연결할 수 있습니다.

오리진 요청 정책을 기존 배포에 연결하는 방법(콘솔)

1. https://console.aws.amazon.com/cloudfront/v4/home#/distributions의 CloudFront 콘솔에서 [Distributions(배포)] 페이지를 엽니다.
2. 업데이트할 배포를 선택한 다음 Behaviors(동작) 탭을 선택합니다.
3. 업데이트할 캐시 동작을 선택한 다음 Edit(편집)를 선택합니다.

   또는 새 캐시 동작을 생성하려면 Create Behavior(동작 생성)를 선택합니다.
4. Cache key and origin request(캐시 키 및 오리진 요청) 섹션에서 Cache policy and origin request policy(캐시 정책 및 오리진 요청 정책)가 선택되어 있는지 확인합니다.
5. Origin Request Policy(오리진 요청 정책)의 경우 이 캐시 동작에 연결할 오리진 요청 정책을 선택합니다.
6. 페이지 하단에서 Save changes(변경 사항 저장)를 선택합니다.

오리진 요청 정책을 새 배포에 연결하는 방법(콘솔)

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포 생성을 선택합니다.
3. Cache key and origin request(캐시 키 및 오리진 요청) 섹션에서 Cache policy and origin request policy(캐시 정책 및 오리진 요청 정책)가 선택되어 있는지 확인합니다.
4. Origin Request Policy(오리진 요청 정책)의 경우 이 배포의 기본 캐시 동작에 연결할 오리진 요청 정책을 선택합니다.
5. 오리진, 기본 캐시 동작 및 기타 배포 설정에 대해 원하는 설정을 선택합니다. 자세한 내용은 모든 배포 설정 참조 섹션을 참조하세요.
6. 완료되면 Create Distribution(배포 생성)을 선택합니다.

CLI

AWS Command Line Interface(AWS CLI)를 사용하여 오리진 요청 정책을 생성하려면 aws cloudfront create-origin-request-policy 명령을 사용합니다. 각 개별 파라미터를 명령줄 입력으로 지정하는 대신 입력 파일을 사용하여 명령의 입력 파라미터를 제공할 수 있습니다.

오리진 요청 정책을 생성하는 방법(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 `origin-request-policy.yaml` 명령에 대한 모든 입력 파라미터가 포함된 create-origin-request-policy이라는 파일을 만듭니다.

```
aws cloudfront create-origin-request-policy --generate-cli-skeleton yaml-input >
 origin-request-policy.yaml
```

2. 방금 생성한 `origin-request-policy.yaml`이라는 파일을 엽니다. 파일을 편집하여 원하는 오리진 요청 정책 설정을 지정한 다음 파일을 저장합니다. 파일에서 선택적 필드를 제거할 수 있지만 필수 필드는 제거하지 마세요.

   오리진 요청 정책 설정에 대한 자세한 내용은 오리진 요청 정책 이해 단원을 참조하세요.

3. 다음 명령을 사용하여 `origin-request-policy.yaml` 파일의 입력 파라미터를 사용하여 오리진 요청 정책을 만듭니다.

```
aws cloudfront create-origin-request-policy --cli-input-yaml file://origin-
request-policy.yaml
```

   명령의 출력에 있는 `Id` 값을 기록해 둡니다. 오리진 요청 정책 ID인 이 값은 오리진 요청 정책을 CloudFront 배포의 캐시 동작에 연결하는 데 필요합니다.

오리진 요청 정책을 기존 배포에 연결하는 방법(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 업데이트할 CloudFront 배포에 대한 배포 구성을 저장합니다. *`distribution_ID`*를 배포 ID로 바꿉니다.

```
aws cloudfront get-distribution-config --id distribution_ID --output yaml >
 dist-config.yaml
```

2. 방금 생성한 `dist-config.yaml`이라는 파일을 엽니다. 오리진 요청 정책을 사용하도록 업데이트하려는 각 캐시 동작을 다음과 같이 변경하여 파일을 편집합니다.

   - 캐시 동작에서 `OriginRequestPolicyId`이라는 필드를 추가합니다. 필드 값에는 정책을 만든 후 기록한 오리진 요청 정책 ID를 사용합니다.
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

완료되면 파일을 저장합니다.

3. 오리진 요청 정책을 사용하도록 배포를 업데이트하려면 다음 명령을 사용합니다. *distribution_ID*를 배포 ID로 바꿉니다.

```
aws cloudfront update-distribution --id distribution_ID --cli-input-yaml file://
dist-config.yaml
```

오리진 요청 정책을 새 배포에 연결(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 `distribution.yaml` 명령에 대한 모든 입력 파라미터가 포함된 create-distribution이라는 파일을 만듭니다.

```
aws cloudfront create-distribution --generate-cli-skeleton yaml-input >
 distribution.yaml
```

2. 방금 생성한 `distribution.yaml`이라는 파일을 엽니다. 기본 캐시 동작의 `OriginRequestPolicyId` 필드에 정책을 생성한 후 기록한 오리진 요청 정책 ID를 입력합니다. 파일을 계속 편집하여 원하는 배포 설정을 지정한 다음 완료되면 파일을 저장합니다.

   배포 설정에 대한 자세한 내용은 모든 배포 설정 참조 단원을 참조하세요.

3. 다음 명령을 사용하여 `distribution.yaml` 파일의 입력 파라미터로 배포를 만듭니다.

```
aws cloudfront create-distribution --cli-input-yaml file://distribution.yaml
```

API

CloudFront API를 사용하여 오리진 요청 정책을 생성하려면 CreateOriginRequestPolicy를 사용합니다. 이 API 호출에서 지정하는 필드에 대한 자세한 내용은 오리진 요청 정책 이해 및 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

오리진 요청 정책을 생성한 후 다음 API 호출 중 하나를 사용하여 캐시 동작에 연결할 수 있습니다.

- 기존 배포의 캐시 동작에 연결하려면 UpdateDistribution을 사용합니다.

- 새 배포의 캐시 동작에 연결하려면 CreateDistribution을 사용합니다.

이 두 API 호출에 대해 캐시 동작 내의 `OriginRequestPolicyId` 필드에 오리진 요청 정책의 ID를 제공합니다. 이러한 API 호출에서 지정하는 다른 필드에 대한 자세한 내용은 모든 배포 설정 참조와 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

# 관리형 오리진 요청 정책 사용

CloudFront에서는 배포의 캐시 동작에 연결할 수 있는 일련의 관리형 오리진 요청 정책을 제공합니다. 관리형 오리진 요청 정책을 사용하면 오리진 요청 정책을 직접 작성하거나 유지할 필요가 없습니다. 관리형 정책은 특정 사용 사례에 최적화된 설정을 사용합니다.

관리형 오리진 요청 정책을 사용하려면 배포의 캐시 동작에 이 정책을 연결합니다. 이 프로세스는 오리진 요청 정책을 생성할 때와 동일하지만 새 정책을 생성하는 대신 관리형 오리진 요청 정책 중 하나를 연결하기만 하면 됩니다. 이름(콘솔 사용) 또는 ID(AWS CLI 또는 SDK 사용)로 정책을 연결합니다. 이름과 ID는 다음 섹션에 나열되어 있습니다.

자세한 내용은 오리진 요청 정책 생성 섹션을 참조하세요.

다음 항목에서는 사용할 수 있는 관리형 오리진 요청 정책에 대해 설명합니다.

주제

- AllViewer
- AllViewerAndCloudFrontHeaders-2022-06
- AllViewerExceptHostHeader
- CORS-CustomOrigin
- CORS-S3Origin
- Elemental-MediaTailor-PersonalizedManifests
- HostHeaderOnly
- UserAgentRefererHeaders

## AllViewer

CloudFront 콘솔에서 이 정책 보기

이 정책에는 뷰어 요청의 모든 값(헤더, 쿠키, 쿼리 문자열)이 포함됩니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`216adef6-5c7f-47e4-b989-5492eafa07d3`

이 정책에는 다음 설정이 포함되어 있습니다.

- 오리진 요청에 포함된 헤더: 최종 사용자 요청의 모든 헤더
- 오리진 요청에 포함된 쿠키: 모두
- 오리진 요청에 포함된 쿼리 문자열: 모두

## AllViewerAndCloudFrontHeaders-2022-06

CloudFront 콘솔에서 이 정책 보기

이 정책에는 뷰어 요청의 모든 값(헤더, 쿠키, 쿼리 문자열)과 2022년 6월까지 릴리스된 모든 CloudFront 헤더(2022년 6월 이후 릴리스된 CloudFront 헤더는 포함되지 않음)가 포함됩니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`33f36d7e-f396-46d9-90e0-52428a34d9dc`

이 정책에는 다음 설정이 포함되어 있습니다.

- 오리진 요청에 포함된 헤더: 최종 사용자 요청의 모든 헤더 및 다음 CloudFront 헤더:
  - `CloudFront-Forwarded-Proto`
  - `CloudFront-Is-Android-Viewer`
  - `CloudFront-Is-Desktop-Viewer`
  - `CloudFront-Is-IOS-Viewer`
  - `CloudFront-Is-Mobile-Viewer`
  - `CloudFront-Is-SmartTV-Viewer`
  - `CloudFront-Is-Tablet-Viewer`
  - `CloudFront-Viewer-Address`
  - `CloudFront-Viewer-ASN`
  - `CloudFront-Viewer-City`
  - `CloudFront-Viewer-Country`
  - `CloudFront-Viewer-Country-Name`

  - `CloudFront-Viewer-Country-Region`
  - `CloudFront-Viewer-Country-Region-Name`
  - `CloudFront-Viewer-Http-Version`
  - `CloudFront-Viewer-Latitude`
  - `CloudFront-Viewer-Longitude`
  - `CloudFront-Viewer-Metro-Code`
  - `CloudFront-Viewer-Postal-Code`
  - `CloudFront-Viewer-Time-Zone`
  - `CloudFront-Viewer-TLS`
- 오리진 요청에 포함된 쿠키: 모두
- 오리진 요청에 포함된 쿼리 문자열: 모두

## AllViewerExceptHostHeader

CloudFront 콘솔에서 이 정책 보기

이 정책에는 뷰어 요청의 ***Host*** 헤더가 포함되지 않지만 뷰어 요청의 다른 값(헤더, 쿠키, 쿼리 문자열)은 모두 포함됩니다.

이 정책에는 HTTP 프로토콜, HTTP 버전, TLS 버전, 모든 디바이스 유형 및 최종 사용자 위치 헤더에 대한 추가 CloudFront 요청 헤더도 포함됩니다.

이 정책은 Amazon API Gateway 및 AWS Lambda 함수 URL 오리진과 함께 사용하기 위한 것입니다. 이러한 오리진에서는 `Host` 헤더에 CloudFront 배포의 도메인 이름이 아니라 오리진 도메인 이름이 포함될 것으로 예상합니다. 뷰어 요청의 `Host` 헤더를 이러한 오리진에 전달하면 오리진이 작동하지 않을 수 있습니다.

> **Note**
>
> 뷰어의 `Host` 헤더를 제거하기 위해 이러한 관리형 오리진 요청 정책을 사용하면 CloudFront는 오리진의 도메인 이름이 포함된 새 `Host` 헤더를 오리진 요청에 추가합니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`b689b0a8-53d0-40ab-baf2-68738e2966ac`

이 정책에는 다음 설정이 포함되어 있습니다.

- 오리진 요청에 포함된 헤더: `Host` 헤더를 제외한 뷰어 요청의 모든 헤더
- 오리진 요청에 포함된 쿠키: 모두
- 오리진 요청에 포함된 쿼리 문자열: 모두

## CORS-CustomOrigin

CloudFront 콘솔에서 이 정책 보기

이 정책에는 오리진이 사용자 지정 오리진일 때 CORS(Cross-Origin Resource Sharing) 요청을 활성화하는 헤더가 포함됩니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`59781a5b-3903-41f3-afcb-af62929ccde1`

이 정책에는 다음 설정이 포함되어 있습니다.

- 오리진 요청에 포함된 헤더:
  - `Origin`
- 오리진 요청에 포함된 쿠키: 없음
- 오리진 요청에 포함된 쿼리 문자열: 없음

## CORS-S3Origin

CloudFront 콘솔에서 이 정책 보기

이 정책에는 오리진이 Amazon S3 버킷일 때 CORS(Cross-Origin Resource Sharing) 요청을 활성화하는 헤더가 포함됩니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`88a5eaf4-2fd4-4709-b370-b4c650ea3fcf`

이 정책에는 다음 설정이 포함되어 있습니다.

- 오리진 요청에 포함된 헤더:
  - `Origin`

  - `Access-Control-Request-Headers`
  - `Access-Control-Request-Method`
- 오리진 요청에 포함된 쿠키: 없음
- 오리진 요청에 포함된 쿼리 문자열: 없음

## Elemental-MediaTailor-PersonalizedManifests

CloudFront 콘솔에서 이 정책 보기

이 정책은 AWS Elemental MediaTailor 엔드포인트인 오리진과 함께 사용하도록 설계되었습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`775133bc-15f2-49f9-abea-afb2e0bf67d2`

이 정책에는 다음 설정이 포함되어 있습니다.

- 오리진 요청에 포함된 헤더:
  - `Origin`
  - `Access-Control-Request-Headers`
  - `Access-Control-Request-Method`
  - `User-Agent`
  - `X-Forwarded-For`
- 오리진 요청에 포함된 쿠키: 없음
- 오리진 요청에 포함된 쿼리 문자열: 모두

## HostHeaderOnly

CloudFront 콘솔에서 이 정책 보기

이 정책에는 오리진 요청의 `Host` 헤더만 포함됩니다. 쿼리 문자열이나 쿠키가 포함되지 않습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`bf0718e1-ba1e-49d1-88b1-f726733018ae`

이 정책에는 다음 설정이 포함되어 있습니다.

- 오리진 요청에 포함된 헤더: Host
- 오리진 요청에 포함된 쿠키: 없음
- 오리진 요청에 포함된 쿼리 문자열: 없음

## UserAgentRefererHeaders

CloudFront 콘솔에서 이 정책 보기

이 정책에는 `User-Agent` 및 `Referer` 헤더만 포함됩니다. 쿼리 문자열이나 쿠키가 포함되지 않습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`acba4595-bd28-49b8-b9fe-13317c0390fa`

이 정책에는 다음 설정이 포함되어 있습니다.

- 오리진 요청에 포함된 헤더:
  - `User-Agent`
  - `Referer`
- 오리진 요청에 포함된 쿠키: 없음
- 오리진 요청에 포함된 쿼리 문자열: 없음

# CloudFront 요청 헤더 추가

CloudFront가 최종 사용자의 요청에서 원본 또는 엣지 기능으로 특정 HTTP 헤더를 추가하도록 CloudFront를 구성할 수 있습니다. 이러한 HTTP 헤더의 값은 최종 사용자 또는 최종 사용자 요청의 특성을 기반으로 합니다. 헤더는 최종 사용자의 디바이스 유형, IP 주소, 지리적 위치, 요청 프로토콜 (HTTP 또는 HTTPS), HTTP 버전 및 TLS 연결 세부 정보, JA3 핑거프린트 및 JA4 핑거프린트를 제공할 수 있습니다. 또한 WebSocket 헤더를 전달하도록 배포의 캐시 동작을 구성할 수 있습니다. 자세한 내용은 CloudFront 배포에 WebSocket 사용 섹션을 참조하세요.

이러한 헤더를 사용하면 정보를 확인하기 위해 직접 코드를 작성할 필요 없이 오리진 또는 엣지 함수에서 최종 사용자에 대한 정보를 수신할 수 있습니다. 오리진에서 이러한 헤더 정보를 기반으로 서로 다른 응답을 반환하는 경우, 캐시 키에 해당 응답을 포함시켜 CloudFront에서 별도로 캐시할 수 있습니다. 예를 들어, 오리진에서는 최종 사용자가 거주하는 국가에 따라 특정 언어로 된 콘텐츠나 특정 디바이스 유형에 맞춘 콘텐츠로 응답할 수 있습니다. 오리진에서 로그 파일에 이러한 헤더를 쓸 수도 있습

니다. 이 헤더를 사용하여 최종 사용자의 위치, 사용 중인 디바이스 유형 등에 대한 정보를 확인할 수 있습니다.

캐시 키에 헤더를 포함하려면 캐시 정책을 사용합니다. 자세한 내용은 정책으로 캐시 키 제어 및 the section called "캐시 키 이해" 섹션을 참조하세요.

원본에서 이러한 헤더를 수신하지만 캐시 키에는 포함하지 않으려면 원본 요청 정책을 사용하십시오. 자세한 내용은 정책을 통한 오리진 요청 제어 섹션을 참조하세요.

주제


- 디바이스 유형 헤더
- 뷰어 위치 헤더
- 뷰어의 헤더 구조를 결정하기 위한 헤더
- TLS 관련 헤더
- 기타 CloudFront 헤더


## 디바이스 유형 헤더

다음 헤더를 추가하여 최종 사용자의 디바이스 유형을 확인할 수 있습니다. `User-Agent` 헤더 값에 따라 CloudFront에서는 해당 헤더 값을 `true` 또는 `false`로 설정합니다. 디바이스가 둘 이상의 범주에 속하는 경우 둘 이상의 값이 `true`일 수 있습니다. 예를 들어 일부 태블릿 디바이스의 경우 CloudFront에서는 `CloudFront-Is-Mobile-Viewer`와 `CloudFront-Is-Tablet-Viewer`를 모두 `true`로 설정합니다.

- `CloudFront-Is-Android-Viewer` – CloudFront에서 최종 사용자가 Android 운영 체제를 사용하는 디바이스라고 확인되면 `true`로 설정합니다.
- `CloudFront-Is-Desktop-Viewer` – CloudFront에서 최종 사용자가 데스크톱 디바이스라고 확인되면 `true`로 설정합니다.
- `CloudFront-Is-IOS-Viewer` – CloudFront에서 최종 사용자가 iPhone, iPod touch, iPad 디바이스와 같은 Apple 모바일 운영 체제를 사용하는 디바이스라고 확인되면 `true`로 설정합니다.
- `CloudFront-Is-Mobile-Viewer` – CloudFront에서 최종 사용자가 모바일 디바이스라고 확인되면 `true`로 설정합니다.
- `CloudFront-Is-SmartTV-Viewer` – CloudFront에서 최종 사용자가 스마트 TV라고 확인되면 `true`로 설정합니다.
- `CloudFront-Is-Tablet-Viewer` – CloudFront에서 최종 사용자가 태블릿이라고 확인되면 `true`로 설정합니다.

# 뷰어 위치 헤더

다음 헤더를 추가하여 최종 사용자의 위치를 확인합니다. CloudFront는 최종 사용자의 IP 주소에 따라 이러한 헤더의 값을 확인합니다. 이 헤더 값에 ASCII 문자가 아닌 문자가 있는 경우, CloudFront는 RFC 3986의 섹션 1.2에 따라 문자를 퍼센트 인코딩합니다.

- `CloudFront-Viewer-Address` - 최종 사용자의 IP 주소와 요청의 소스 포트를 포함합니다. 예를 들어, 헤더 값 `198.51.100.10:46532`는 최종 사용자의 IP 주소가 198.51.100.10이고 요청 소스 포트가 46532임을 의미합니다.
- `CloudFront-Viewer-ASN` - 최종 사용자의 Autonomous System Number(ASN)를 포함합니다.

> **Note**
>
> `CloudFront-Viewer-Address` 및 `CloudFront-Viewer-ASN`을 오리진 요청 정책에는 추가할 수 있지만 캐시 정책에는 추가할 수 없습니다.

- `CloudFront-Viewer-Country` – 최종 사용자 국가의 두 자로 된 국가 코드가 들어 있습니다. 국가 코드 목록은 ISO 3166-1 alpha-2 단원을 참조하세요.
- `CloudFront-Viewer-City` – 최종 사용자의 도시 이름을 포함합니다.

다음 헤더를 추가하면 CloudFront는 AWS 네트워크에서 시작된 요청을 제외한 모든 요청에 해당 헤더를 적용합니다.

- `CloudFront-Viewer-Country-Name` – 최종 사용자 국가의 이름을 포함합니다.
- `CloudFront-Viewer-Country-Region` – 최종 사용자 리전을 나타내는 코드(최대 3자)를 포함합니다. 리전은 ISO 3166-2코드의 첫 번째 수준의 하위 항목(가장 폭넓거나 가장 덜 구체적)입니다.
- `CloudFront-Viewer-Country-Region-Name` – 최종 사용자 리전의 이름을 포함합니다. 리전은 ISO 3166-2코드의 첫 번째 수준의 하위 항목(가장 폭넓거나 가장 덜 구체적)입니다.
- `CloudFront-Viewer-Latitude` – 최종 사용자의 대략적인 위도를 포함합니다.
- `CloudFront-Viewer-Longitude` – 최종 사용자의 대략적인 경도를 포함합니다.
- `CloudFront-Viewer-Metro-Code` – 최종 사용자의 메트로 코드를 포함합니다. 이는 최종 사용자가 미국에 있을 때만 표시됩니다.
- `CloudFront-Viewer-Postal-Code` – 최종 사용자의 우편 번호를 포함합니다.
- `CloudFront-Viewer-Time-Zone` IANA 표준 시간대 데이터베이스 형식(예: `America/Los_Angeles`)으로 최종 사용자의 표준 시간대를 포함합니다.

> **Note**
>
> `CloudFront-Viewer-City`, `CloudFront-Viewer-Metro-Code` 및 `CloudFront-Viewer-Postal-Code`를 모든 IP 주소에 사용할 수 있는 것은 아닙니다. 일부 IP 주소는 해당 정보를 얻을 수 있을 만큼 구체적으로 지리적 위치를 파악할 수 없습니다.

## 뷰어의 헤더 구조를 결정하기 위한 헤더

다음 헤더를 추가하여 전송한 헤더를 기반으로 최종 사용자를 식별할 수 있습니다. 예를 들어, 다양한 브라우저가 특정 순서로 HTTP 헤더를 전송할 수 있습니다. `User-Agent` 헤더에 지정된 브라우저가 해당 브라우저의 예상 헤더 순서와 일치하지 않는 경우 요청을 거부할 수 있습니다. 또한 `CloudFront-Viewer-Header-Count` 값이 `CloudFront-Viewer-Header-Order`의 헤더 수와 일치하지 않는 경우 요청을 거부할 수 있습니다.

- `CloudFront-Viewer-Header-Order` — 최종 사용자의 헤더 이름을 요청된 순서대로 콜론으로 구분하여 포함합니다. 예를 들면 `CloudFront-Viewer-Header-Order: Host:User-Agent:Accept:Accept-Encoding`입니다. 7,680자 제한을 초과하는 헤더는 잘립니다.
- `CloudFront-Viewer-Header-Count` — 최종 사용자 헤더의 총 수를 포함합니다.

## TLS 관련 헤더

다음 헤더를 추가하여 뷰어의 JA3 지문, JA4 지문, TLS 연결 세부 정보를 확인할 수 있습니다.

- `CloudFront-Viewer-JA3-Fingerprint` — 최종 사용자의 JA3 지문을 포함합니다. JA3 지문은 해당 요청이 알려진 클라이언트에서 온 것인지, 맬웨어인지 악성 봇인지 또는 예상되는(허용 목록에 있는) 응용 프로그램에서 온 것인지 판단하는 데 도움이 됩니다.
- `CloudFront-Viewer-JA4-Fingerprint` - 뷰어의 JA4 지문이 포함되어 있습니다. JA3 지문과 마찬가지로 JA4 지문을 사용하면 해당 요청이 알려진 클라이언트에서 온 것인지, 멀웨어인지, 악성 봇인지 또는 예상되는(허용 목록에 있는) 애플리케이션에서 유입된 것인지 확인할 수 있습니다. 지문을 사용하여 HTTP 요청을 검사할 때 적용할 알려진 좋은 행위자와 나쁜 행위자의 데이터베이스를 구축할 수 있습니다. 그런 다음 애플리케이션 웹 서버 또는 Lambda@Edge 및 CloudFront Functions에서 헤더 값을 검사하여 헤더 값을 알려진 멀웨어 지문 목록과 비교하여 악성 클라이언트를 차단할 수 있습니다.
- `CloudFront-Viewer-TLS` - 최종 사용자와 CloudFront 간의 연결에 사용된 SSL/TLS 버전, 암호, SSL/TLS 핸드셰이크 정보를 포함합니다. 헤더 값은 다음 형식입니다.

```
SSL/TLS_version:cipher:handshake_information
```

*handshake_information*에 대해 헤더는 다음 값 중 하나를 포함할 수 있습니다.

- `fullHandshake` - SSL/TLS 세션에 대해 전체 핸드셰이크가 수행되었습니다.
- `sessionResumed` - 이전 SSL/TLS 세션이 재개되었습니다.
- `connectionReused` - 이전 SSL/TLS 연결이 재사용되었습니다.

다음은 이 헤더에 대한 몇 가지 예시 값입니다.

```
TLSv1.3:TLS_AES_128_GCM_SHA256:sessionResumed
```

```
TLSv1.2:ECDHE-ECDSA-AES128-GCM-SHA256:connectionReused
```

```
TLSv1.1:ECDHE-RSA-AES128-SHA256:fullHandshake
```

```
TLSv1:ECDHE-RSA-AES256-SHA:fullHandshake
```

이 헤더 값에 있을 수 있는 SSL/TLS 버전 및 암호의 전체 목록은 the section called “최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호” 단원을 참조하세요.

> **Notes**
>
> - JA3 및 JA4 지문은 SSL/TLS `Client Hello` 패킷에서 파생됩니다. 이는 HTTPS 요청에만 존재합니다.
> - 이러한 TLS 관련 헤더의 경우 오리진 요청 정책에 추가할 수 있지만 캐시 정책에는 추가할 수 없습니다.

## 기타 CloudFront 헤더

다음 헤더를 추가하여 뷰어의 원본 요청 URI, 원본 요청 쿼리 문자열 파라미터와 값, 프로토콜, 버전을 확인할 수 있습니다.

- `CloudFront-Error-Uri` - 뷰어로부터 받은 원본 요청 URI를 포함합니다.

- `CloudFront-Error-Args` - 원본 요청 쿼리 문자열 파라미터 및 값을 포함합니다.
- `CloudFront-Forwarded-Proto` – 최종 사용자의 요청 프로토콜(HTTP 또는 HTTPS)을 포함합니다.
- `CloudFront-Viewer-Http-Version` – 최종 사용자 요청의 HTTP 버전을 포함합니다.

## 오리진 요청 정책과 캐시 정책이 함께 작동하는 방식 이해

CloudFront 오리진 요청 정책을 사용하여 CloudFront가 오리진으로 보내는 요청, 즉 오리진 요청을 제어할 수 있습니다. 오리진 요청 정책을 사용하려면 동일한 캐시 동작에 캐시 정책을 연결해야 합니다. 캐시 정책 없이 캐시 동작에 오리진 요청 정책을 사용할 수 없습니다. 자세한 내용은 정책을 통한 오리진 요청 제어 단원을 참조하십시오.

오리진 요청 정책과 캐시 정책은 함께 작동하여 CloudFront가 오리진 요청에 포함하는 값을 결정합니다. 캐시 정책을 사용하여 캐시 키에 지정하는 모든 URL 쿼리 문자열, HTTP 헤더 및 쿠키는 오리진 요청에 자동으로 포함됩니다. 오리진 요청 정책에 지정하는 모든 추가 쿼리 문자열, 헤더 및 쿠키도 오리진 요청에 포함됩니다(캐시 키에는 포함되지 않음).

오리진 요청 정책과 캐시 정책에는 서로 충돌하는 것처럼 보일 수 있는 설정이 있습니다. 예를 들어 한 정책에서는 특정 값을 허용하고 다른 정책에서는 차단할 수 있습니다. 다음 표에서는 오리진 요청 정책과 캐시 정책의 설정을 함께 사용할 때 CloudFront가 오리진 요청에 포함하는 값을 설명합니다. 이러한 설정은 일반적으로 모든 유형의 값(쿼리 문자열, 헤더 및 쿠키)에 적용됩니다. 단, 캐시 정책에서 모든 헤더를 지정하거나 헤더 차단 목록을 사용할 수는 없습니다.

| | 오리진 요청 정책 | | | |
|---|---|---|---|---|
| | None(없음) | 모두 | 허용 목록 | 차단 목록 |
| 캐시 정책 | | | | |
| None(없음) | 모든 오리진 요청에 포함된 기본값을 제외하고 뷰어 요청의 값은 오리진 요청에 포함되지 않습니다. 자세한 내용은 정책을 통한 오리진 | 뷰어 요청의 모든 값이 오리진 요청에 포함됩니다. | 오리진 요청 정책에 지정된 값만 오리진 요청에 포함됩니다. | 오리진 요청 정책에 지정된 값을 제외하고 뷰어 요청의 모든 값이 오리진 요청에 포함됩니다. |

| | 오리진 요청 정책 | | | |
|---|---|---|---|---|
| | None(없음) | 모두 | 허용 목록 | 차단 목록 |
| | 요청 제어 단원을 참조하십시오. | | | |
| 모두<br><br>참고: 캐시 정책의 모든 헤더를 지정할 수는 없습니다. | 뷰어 요청의 모든 쿼리 문자열과 쿠키가 오리진 요청에 포함됩니다. | 뷰어 요청의 모든 값이 오리진 요청에 포함됩니다. | 뷰어 요청의 모든 쿼리 문자열 및 쿠키와 오리진 요청 정책에 지정된 모든 헤더가 오리진 요청에 포함됩니다. | 오리진 요청 정책에 차단 목록에 지정된 것을 비롯하여 뷰어 요청의 모든 쿼리 문자열 및 쿠키가 오리진 요청에 포함됩니다. 캐시 정책 설정은 오리진 요청 정책 차단 목록을 재정의합니다. |
| 허용 목록 | 뷰어 요청에 지정된 값만 오리진 요청에 포함됩니다. | 뷰어 요청의 모든 값이 오리진 요청에 포함됩니다. | 캐시 정책 또는 오리진 요청에 지정된 값이 오리진 요청에 포함됩니다. | 캐시 정책에 지정된 값은 오리진 요청 정책 차단 목록에 지정되어 있더라도 오리진 요청에 포함됩니다. 캐시 정책 허용 목록은 오리진 요청 정책 차단 목록을 재정의합니다. |

| | 오리진 요청 정책 | | | |
|---|---|---|---|---|
| | None(없음) | 모두 | 허용 목록 | 차단 목록 |
| 차단 목록<br><br>참고: 캐시 정책 차단 목록의 헤더를 지정할 수는 없습니다. | 지정된 것을 제외하고 뷰어 요청의 모든 쿼리 문자열과 쿠키가 오리진 요청에 포함됩니다. | 뷰어 요청의 모든 값이 오리진 요청에 포함됩니다. | 오리진 요청 정책에 지정된 값은 캐시 정책 차단 목록에 지정되어 있더라도 오리진 요청에 포함됩니다. 오리진 요청 정책 허용 목록은 캐시 정책 차단 목록을 재정의합니다. | 캐시 정책 또는 오리진 요청 정책에 지정된 값을 제외하고 뷰어 요청의 모든 값이 오리진 요청에 포함됩니다. |

# 정책을 통해 CloudFront 응답에서 HTTP 헤더 추가 또는 제거

뷰어(웹 브라우저 및 기타 클라이언트)에 보내는 응답에 HTTP 헤더를 수정하도록 CloudFront를 구성할 수 있습니다. CloudFront는 최종 사용자에게 응답을 보내기 전에 오리진에서 수신한 헤더를 제거하거나 응답에 헤더를 추가할 수 있습니다. 이러한 변경 시에는 코드를 작성하거나 원본을 변경할 필요가 없습니다.

예를 들어 `X-Powered-By` 및 `Vary` 등의 헤더를 제거하여 CloudFront가 최종 사용자에게 보내는 응답에 이러한 헤더를 포함하지 않도록 할 수 있습니다. 또는 다음과 같은 HTTP 헤더를 추가할 수 있습니다.

- 브라우저 캐싱을 제어하는 `Cache-Control` 헤더.
- 교차 원본 리소스 공유(CORS)를 사용 설정하는 `Access-Control-Allow-Origin` 헤더. 또한 기타 CORS 헤더를 추가할 수도 있습니다.
- `Strict-Transport-Security`, `Content-Security-Policy`, `X-Frame-Options` 등과 같은 공통 보안 헤더 집합.
- CloudFront를 통한 요청과 응답의 성능 및 라우팅과 관련된 정보를 제공하는 `Server-Timing` 헤더.

CloudFront에서 HTTP 응답에서 추가 또는 제거하는 헤더를 지정하려면 응답 헤더 정책을 사용합니다. 응답 헤더 정책을 캐시 동작에 연결하면 CloudFront는 캐시 동작과 일치하는 요청에 대해 보내는 HTTP 응답에 정책의 헤더를 추가합니다. CloudFront는 CloudFront가 캐시에서 제공하는 응답과 CloudFront가 원본에서 전달하는 응답에 헤더를 수정합니다. 원본 응답에 응답 헤더 정책에 추가된 헤더가 하나 이상 포함된 경우 정책은 CloudFront가 원본에서 수신한 헤더를 사용할지 아니면 응답 헤더 정책에 있는 헤더로 덮어쓸지 지정할 수 있습니다.

> **Note**
>
> `Cache-Control`과 같이 브라우저 캐싱을 제어하는 헤더를 응답 헤더 정책에 추가하면 CloudFront는 이러한 헤더를 뷰어 응답에만 추가합니다. 이러한 헤더는 CloudFront가 요청된 객체를 캐싱하는 방식에 영향을 주지 않습니다.

CloudFront는 일반적인 사용 사례에 대해 관리형 정책이라고 하는 사전 정의된 응답 헤더 정책을 제공합니다. 이러한 관리형 정책을 사용하거나 고유의 정책을 만들 수 있습니다. AWS 계정의 여러 배포에서 여러 캐시 동작에 단일 응답 헤더 정책을 연결할 수 있습니다.

자세한 내용은 다음 자료를 참조하세요.

주제

- 응답 헤더 정책 이해
- 응답 헤더 정책 생성
- 관리형 응답 헤더 정책 사용

# 응답 헤더 정책 이해

응답 헤더 정책을 사용하여 Amazon CloudFront가 사용자에게 보내는 응답에서 추가 또는 제거하는 HTTP 헤더를 지정할 수 있습니다. 응답 헤더 정책 및 사용 이유에 대한 자세한 내용은 정책을 통해 응답 헤더 추가 또는 제거 단원을 참조하세요.

다음 항목에서는 응답 헤더 정책의 설정에 대해 설명합니다. 설정은 다음 항목에 나와 있는 범주로 그룹화됩니다.

주제

- 정책 세부 정보(메타데이터)
- CORS 헤더
- 보안 헤더
- 사용자 지정 헤더
- 헤더 제거
- Server-Timing 헤더

## 정책 세부 정보(메타데이터)

정책 세부 정보 설정에는 응답 헤더 정책에 대한 메타데이터가 포함되어 있습니다.

- 이름 — 응답 헤더 정책을 식별하는 이름입니다. 콘솔에서 이름을 사용하여 정책을 캐시 동작에 연결합니다.
- 설명(선택 사항) — 응답 헤더 정책을 설명하는 설명입니다. 이는 선택 사항이지만 캐시 정책의 용도를 식별하는 데 도움이 될 수 있습니다.

# CORS 헤더

교차 원본 리소스 공유(CORS) 설정을 사용하면 응답 헤더 정책에 CORS 헤더를 추가하고 구성할 수 있습니다.

이 목록은 응답 헤더 정책에서 설정 및 유효한 값을 지정하는 방법에 중점을 둡니다. 이러한 각 헤더와 실제 CORS 요청 및 응답에 사용되는 방법에 대한 자세한 내용은 MDN 웹 문서 및 CORS 프로토콜 사양의 원본 간 리소스 공유를 참조하세요.

Access-Control-Allow-Credentials

이는 부울 설정(`true` 또는 `false`)으로서 CloudFront에서 CORS 요청에 대한 응답으로 `Access-Control-Allow-Credentials` 헤더를 추가할지 여부를 결정합니다. 이 설정이 `true`로 설정된 경우, CloudFront는 CORS 요청에 대한 응답으로 `Access-Control-Allow-Credentials: true` 헤더를 추가합니다. 그렇지 않으면 CloudFront가 응답에 이 헤더를 추가하지 않습니다.

Access-Control-Allow-Headers

CloudFront가 CORS 실행 전 요청에 대한 응답에서 `Access-Control-Allow-Headers` 헤더 값으로 사용하는 헤더 이름을 지정합니다. 이 설정의 유효한 값으로는 HTTP 헤더 이름 또는 모든 헤더가 허용됨을 나타내는 와일드카드 문자(`*`)가 포함됩니다.

> **Note**
>
> `Authorization` 헤더는 와일드카드를 사용할 수 없으며 명시적으로 나열되어야 합니다.

유효한 와일드카드 문자 사용의 예시

| 예제 | 매칭함 | 매칭하지 않음 |
| --- | --- | --- |
| `x-amz-*` | `x-amz-test`<br>`x-amz-` | `x-amz` |
| `x-*-amz` | `x-test-amz`<br>`x--amz` | |
| `*` | `Authorization` 을 제외한 모든 헤더 | `Authorization` |

Access-Control-Allow-Methods

CloudFront가 CORS 실행 전 요청에 대한 응답에서 `Access-Control-Allow-Methods` 헤더 값으로 사용하는 HTTP 메서드를 지정합니다. 유효한 값은 `GET`, `DELETE`, `HEAD`, `OPTIONS`, `PATCH`, `POST`, `PUT` 또는 `ALL`입니다. `ALL`는 나열된 모든 HTTP 메서드를 포함하는 특수 값입니다.

Access-Control-Allow-Origin

CloudFront에서 `Access-Control-Allow-Origin` 응답 헤더에 사용할 수 있는 값을 지정합니다. 이 설정의 유효한 값에는 특정 원본(예: `http://www.example.com`) 또는 모든 원본이 허용됨을 나타내는 와일드카드 문자(`*`)가 포함됩니다.

> **Notes**
>
> - 와일드카드 문자(`*`)는 하위 도메인의 가장 왼쪽에 사용할 수 있습니다 (`*.example.org`).
> - 와일드카드 문자(`*`)는 다음 위치에 사용할 수 없습니다.
>  - 상위 수준 도메인(`example.*`)
>  - 하위 도메인 오른쪽(`test.*.example.org`) 또는 원하는 하위 도메인 내 (`*test.example.org`)
>  - 용어 내부(`exa*mple.org`)

와일드카드 문자를 사용하는 방법의 예는 다음 표를 참조하세요.

| 예제 | 매칭함 | 매칭하지 않음 |
| --- | --- | --- |
| `http://*.example.org` | `http://www.example.org`<br>`http://test.example.org` | **`https`**`://test.example.org`<br>**`https`**`://test.example.org:123`<br>`http://test.example.org:123` |
| `*.example.org` | `test.example.org` | `http://test.example.org:123` |

| 예제 | 매칭함 | 매칭하지 않음 |
| --- | --- | --- |
| | test.test.example.org<br><br>.example.org<br><br>http://test.example.org<br><br>https://test.example.org | https://test.example.org:123 |
| example.org | http://example.org<br><br>https://example.org | |
| http://example.org | | **https**://example.org<br><br>http://example.org:123 |
| http://example.org:* | http://example.org:123<br><br>http://example.org | |
| http://example.org:1*3 | http://example.org:123<br><br>http://example.org:1893<br><br>http://example.org:13 | |
| *.example.org:1* | test.example.org:123 | |

Access-Control-Expose-Headers

CloudFront가 CORS 요청에 대한 응답에서 `Access-Control-Expose-Headers` 헤더 값으로 사용하는 헤더 이름을 지정합니다. 이 설정의 유효한 값으로는 HTTP 헤더 이름 또는 와일드카드 문자(*)가 포함됩니다.

Access-Control-Max-Age

CloudFront가 CORS 실행 전 요청에 대한 응답에서 `Access-Control-Max-Age` 헤더 값으로 사용하는 시간(초)입니다.

원본 재정의

오리진의 응답에 정책에도 포함된 CORS 헤더 중 하나가 포함된 경우 CloudFront가 작동하는 방식을 결정하는 부울 설정입니다.

- `true`로 설정되고 오리진 응답에 정책에도 있는 CORS 헤더가 포함되어 있으면 CloudFront는 응답에 정책의 CORS 헤더를 추가합니다. 그런 다음 CloudFront는 해당 응답을 최종 사용자에게 보냅니다. CloudFront는 오리진에서 수신한 헤더는 무시합니다.
- `false`로 설정되고 오리진 응답에 CORS 헤더가 포함되어 있으면(CORS 헤더가 정책에 포함되어 있는지 여부에 상관없이) CloudFront는 응답에 오리진에서 수신한 CORS 헤더를 포함합니다. CloudFront는 최종 사용자에게 보내는 응답에 정책의 CORS 헤더를 추가하지 않습니다.

## 보안 헤더

보안 헤더 설정을 사용하면 응답 헤더 정책에서 여러 보안 관련 HTTP 응답 헤더를 추가하고 구성할 수 있습니다.

이 목록은 응답 헤더 정책에서 설정 및 유효한 값을 지정하는 방법을 설명합니다. 이러한 각 헤더와 실제 HTTP 응답에서 사용되는 방법에 대한 자세한 내용은 MDN 웹 문서 링크를 참조하세요.

Content-Security-Policy

CloudFront가 `Content-Security-Policy` 응답 헤더의 값으로 사용하는 콘텐츠 보안 정책 지시문을 지정합니다.

이 헤더 및 유효한 정책 지시문에 대한 자세한 내용은 MDN 웹 문서의 Content-Security-Policy를 참조하십시오.

> **Note**
>
> `Content-Security-Policy` 헤더 값은 1783자로 제한됩니다.

참조자 정책

CloudFront가 `Referrer-Policy` 응답 헤더의 값으로 사용하는 참조자 정책 지시문을 지정합니다. 이 설정에 유효한 값은 `no-referrer`, `no-referrer-when-downgrade`, `origin`, `origin-when-cross-origin`, `same-origin`, `strict-origin`, `strict-origin-when-cross-origin` 또는 `unsafe-url`입니다.

이 헤더 및 이러한 지시문에 대한 자세한 내용은 MDN 웹 문서의 Referrer-Policy를 참조하십시오.

Strict-Transport-Security

CloudFront가 `Strict-Transport-Security` 응답 헤더의 값으로 사용하는 지시문 및 설정을 지정합니다. 이 설정의 경우 다음을 별도로 지정합니다.

- CloudFront가 이 헤더의 `max-age` 지시문에 대한 값으로 사용하는 시간(초)
- CloudFront가 이 헤더 값에 `preload` 지시문을 포함하는지 여부를 결정하는 `preload`에 대한 부울 설정(`true` 또는 `false`)
- CloudFront가 이 헤더 값에 `includeSubDomains` 지시문을 포함하는지 여부를 결정하는 `includeSubDomains`에 대한 부울 설정(`true` 또는 `false`)

이 헤더 및 이러한 지시문에 대한 자세한 내용은 MDN 웹 문서의 Strict-Transport-Security를 참조하십시오.

X-Content-Type-Options

이는 부울 설정(`true` 또는 `false`)으로서 CloudFront에서 응답으로 `X-Content-Type-Options` 헤더를 추가할지 여부를 결정합니다. 이 설정이 `true`인 경우, CloudFront는 응답에 `X-Content-Type-Options: nosniff` 헤더를 추가합니다. 그렇지 않으면 CloudFront는 이 헤더를 추가하지 않습니다.

이 헤더에 대한 자세한 내용은 MDN 웹 문서의 X-Content-Type-Options를 참조하십시오.

X-Frame-Options

CloudFront가 `X-Frame-Options` 응답 헤더의 값으로 사용하는 지시문을 지정합니다. 이 설정에 유효한 값은 `DENY` 또는 `SAMEORIGIN`입니다.

이 헤더 및 이러한 지시문에 대한 자세한 내용은 MDN 웹 문서의 X-Frame-Options를 참조하십시오.

X-XSS-Protection

CloudFront가 `X-XSS-Protection` 응답 헤더의 값으로 사용하는 지시문 및 설정을 지정합니다. 이 설정의 경우 다음을 별도로 지정합니다.

- `0`(XSS 필터링 비활성화) 또는 `1`(XSS 필터링 활성화) `X-XSS-Protection` 설정
- CloudFront가 이 헤더 값에 `mode=block` 지시문을 포함하는지 여부를 결정하는 `block`에 대한 부울 설정(`true` 또는 `false`)
- CloudFront가 이 헤더 값에 `report=`*`reporting URI`* 지시문을 포함하는지 여부를 결정하는 보고 URI

`block`에 대해 `true`를 지정하거나 보고 URI를 지정할 수 있지만 둘 다 함께 지정할 수는 없습니다. 이 헤더 및 이러한 지시문에 대한 자세한 내용은 MDN 웹 문서의 X-XSS-Protection를 참조하십시오.

원본 재정의

이러한 각 보안 헤더 설정에는 원본의 응답에 해당 헤더가 포함되어 있을 때 CloudFront가 작동하는 방식을 결정하는 부울 설정(`true` 또는 `false`)이 포함되어 있습니다.

이 설정이 `true`로 설정되고 원본 응답에 헤더가 포함되어 있으면 CloudFront는 뷰어에게 보내는 응답에 정책의 헤더를 추가합니다. 그리고 원본에서 수신한 헤더는 무시합니다.

이 설정이 `false`로 설정되고 원본 응답에 헤더가 포함된 경우 CloudFront는 뷰어에게 보내는 응답에 원본에서 수신한 헤더를 포함합니다.

원본 응답에 헤더가 포함되어 있지 않으면 CloudFront는 뷰어에게 보내는 응답에 정책의 헤더를 추가합니다. 이 설정이 `true`이든 `false`이든 관계없이 CloudFront는 이 작업을 수행합니다.

## 사용자 지정 헤더

사용자 지정 헤더 설정을 사용하면 응답 헤더 정책에 사용자 지정 HTTP 헤더를 추가하고 구성할 수 있습니다. CloudFront는 이러한 헤더를 뷰어에게 반환하는 모든 응답에 추가합니다. 값을 지정하는 것은 선택 사항이지만 각 사용자 지정 헤더에 헤더 값도 지정합니다. CloudFront가 값 없이 응답 헤더를 추가할 수 있기 때문입니다.

각 사용자 정의 헤더에는 고유한 원본 재정의 설정도 있습니다.

- 이 설정이 `true`로 설정되고 원본 응답에 정책에 있는 사용자 지정 헤더가 포함되어 있으면 CloudFront는 뷰어에게 보내는 응답에 정책의 사용자 지정 헤더를 추가합니다. 그리고 원본에서 수신한 헤더는 무시합니다.
- 이 설정이 `false`이고 원본 응답에 정책에 있는 사용자 지정 헤더가 포함되어 있으면 CloudFront는 뷰어에게 보내는 응답에 원본에서 수신한 사용자 지정 헤더를 포함합니다.
- 원본 응답에 정책에 있는 사용자 지정 헤더가 포함되어 있지 않으면 CloudFront는 뷰어에게 보내는 응답에 정책의 사용자 지정 헤더를 추가합니다. 이 설정이 `true`이든 `false`이든 관계없이 CloudFront는 이 작업을 수행합니다.

## 헤더 제거

CloudFront가 오리진으로부터 받는 응답에서 제거할 헤더를 지정하여 CloudFront가 최종 사용자에게 보내는 응답에 헤더가 포함되지 않도록 할 수 있습니다. CloudFront는 객체가 CloudFront의 캐시에서 제공되든 오리진에서 제공되든 관계없이 최종 사용자에게 보내는 모든 응답에서 헤더를 제거합니다. 예를 들어 `X-Powered-By` 또는 `Vary` 같이 브라우저에 사용되지 않는 헤더를 제거하여 CloudFront가 최종 사용자에게 보내는 응답에서 이러한 헤더를 제거하도록 할 수 있습니다.

응답 헤더 정책을 사용하여 제거할 헤더를 지정하면 CloudFront는 먼저 헤더를 제거한 다음 응답 헤더 정책의 다른 섹션(CORS 헤더, 보안 헤더, 사용자 지정 헤더 등)에 지정된 헤더를 추가합니다. 제거할 헤더를 지정하고 정책의 다른 섹션에 동일한 헤더를 추가하는 경우 CloudFront는 최종 사용자에게 보내는 응답에 해당 헤더를 포함합니다.

> **Note**
>
> 응답 헤더 정책을 사용하여 CloudFront가 오리진에서 수신한 `Server` 및 `Date` 헤더를 제거하여 이러한 헤더(오리진에서 수신됨)가 CloudFront가 최종 사용자에게 보내는 응답에 포함되지 않도록 할 수 있습니다. 하지만 이렇게 하면 CloudFront가 최종 사용자에게 보내는 응답에 해당 버전을 추가합니다. CloudFront가 추가하는 `Server` 헤더의 헤더 값은 `CloudFront`입니다.

### 제거할 수 없는 헤더

응답 헤더 정책을 사용하여 다음 헤더를 제거할 수 없습니다. 응답 헤더 정책(API `ResponseHeadersPolicyRemoveHeadersConfig`)의 Remove headers(헤더 제거) 섹션에서 이러한 헤더를 지정하면 오류가 발생합니다.

- `Connection`

- Content-Encoding
- Content-Length
- Expect
- Host
- Keep-Alive
- Proxy-Authenticate
- Proxy-Authorization
- Proxy-Connection
- Trailer
- Transfer-Encoding
- Upgrade
- Via
- Warning
- X-Accel-Buffering
- X-Accel-Charset
- X-Accel-Limit-Rate
- X-Accel-Redirect
- X-Amz-Cf-.*
- X-Amzn-Auth
- X-Amzn-Cf-Billing
- X-Amzn-Cf-Id
- X-Amzn-Cf-Xff
- X-Amzn-ErrorType
- X-Amzn-Fle-Profile
- X-Amzn-Header-Count
- X-Amzn-Header-Order
- X-Amzn-Lambda-Integration-Tag
- X-Amzn-RequestId
- X-Cache
- X-Edge-.*

- `X-Forwarded-Proto`
- `X-Real-Ip`

## Server-Timing 헤더

`Server-Timing` 헤더 설정을 사용하여 CloudFront에서 보낸 HTTP 응답에서 `Server-Timing` 헤더를 사용 설정합니다. 이 헤더를 사용하여 CloudFront 및 원본의 동작 및 성능에 대한 인사이트를 제공하는 지표를 볼 수 있습니다. 예를 들어 캐시 적중률을 제공한 캐시 계층을 확인할 수 있습니다. 또는 캐시 누락이 발생한 경우 원본에서 첫 번째 바이트 대기 시간을 확인할 수 있습니다. `Server-Timing` 헤더의 지표는 문제를 해결하거나 CloudFront 또는 원본 구성의 효율성을 테스트하는 데 도움이 될 수 있습니다.

CloudFront에서 `Server-Timing` 헤더 사용에 대한 자세한 내용은 다음 주제를 참조하세요.

`Server-Timing` 헤더를 사용 설정하려면 응답 헤더 정책을 생성(또는 편집)합니다.

주제

- 샘플링 속도 및 Pragma 요청 헤더
- 원본에서 수신한 Server-Timing 헤더
- Server-Timing 헤더 지표
- Server-Timing 헤더 예시

### 샘플링 속도 및 Pragma 요청 헤더

응답 헤더 정책에서 `Server-Timing` 헤더를 사용 설정할 때 샘플링 속도도 지정합니다. 샘플링 속도는 CloudFront가 `Server-Timing` 헤더를 추가할 응답의 비율을 지정하는 0~100(포함)의 숫자입니다. 샘플링 속도를 100으로 설정하면 CloudFront는 응답 헤더 정책이 연결된 캐시 동작과 일치하는 모든 요청에 대한 HTTP 응답에 `Server-Timing` 헤더를 추가합니다. 이 값을 50으로 설정하면 CloudFront가 캐시 동작과 일치하는 요청에 대한 응답의 50%에 헤더를 추가합니다. 샘플링 속도는 소수점 이하 4자리까지 0~100의 숫자로 설정할 수 있습니다.

샘플링 속도가 100보다 낮은 숫자로 설정되면 CloudFront가 `Server-Timing` 헤더를 추가하는 응답을 제어할 수 없고 비율만 제어할 수 있습니다. 그러나 HTTP 요청에서 값이 `server-timing`으로 설정된 `Pragma` 헤더를 추가하여 해당 요청에 대한 응답에서 `Server-Timing` 헤더를 수신할 수 있습니다. 이는 샘플링 속도가 무엇으로 설정되어 있든 상관없이 작동합니다. 샘플링 속도가 0으로 설정된 경우에도 요청에 `Pragma: server-timing` 헤더가 포함된 경우 CloudFront는 `Server-Timing` 헤더를 응답에 추가합니다.

## 원본에서 수신한 Server-Timing 헤더

캐시 누락이 있고 CloudFront가 요청을 원본에 전달할 때 원본은 CloudFront에 대한 응답에 `Server-Timing` 헤더를 포함할 수 있습니다. 이 경우 CloudFront는 원본에서 수신한 `Server-Timing` 헤더에 지표를 추가합니다. CloudFront가 뷰어에게 보내는 응답은 원본에서 온 값과 CloudFront가 추가한 지표를 포함하는 하나의 `Server-Timing` 헤더를 포함합니다. 원본에서 수신한 헤더 값은 끝에 있거나 CloudFront가 헤더에 추가하는 두 지표 세트 사이에 있을 수 있습니다.

캐시 적중이 있는 경우 CloudFront가 뷰어에게 보내는 응답은 헤더 값에 CloudFront 지표만 포함하는 하나의 `Server-Timing` 헤더를 포함합니다(원본에서 수신한 값은 포함되지 않음).

## Server-Timing 헤더 지표

CloudFront가 HTTP 응답에 `Server-Timing` 헤더를 추가할 때 헤더 값에는 CloudFront 및 원본의 동작 및 성능에 대한 인사이트를 제공하는 하나 이상의 지표가 포함됩니다. 다음 목록에는 모든 지표와 해당 잠재적 값이 나와 있습니다. `Server-Timing` 헤더에는 CloudFront를 통한 요청 및 응답의 특성에 따라 이러한 지표 중 일부만 포함됩니다.

이러한 지표 중 일부는 이름만 있는(값은 없음) `Server-Timing` 헤더에 포함됩니다. 나머지에는 이름과 값이 있습니다. 지표에 값이 있으면 이름과 값이 세미콜론(`;`)으로 구분됩니다. 헤더에 둘 이상의 지표가 포함된 경우 지표는 쉼표(`,`)로 구분됩니다.

cdn-cache-hit

CloudFront가 원본에 요청하지 않고 캐시에서 응답을 제공했습니다.

cdn-cache-refresh

CloudFront가 캐시된 객체가 여전히 유효한지 확인하기 위해 원본에 요청을 보낸 후 캐시에서 응답을 제공했습니다. 이 경우 CloudFront는 원본에서 전체 객체를 검색하지 않았습니다.

cdn-cache-miss

CloudFront가 캐시에서 응답을 제공하지 않았습니다. 이 경우 CloudFront는 응답을 반환하기 전에 원본에서 전체 객체를 요청했습니다.

cdn-pop

요청을 처리한 CloudFront 접속 지점(POP)을 설명하는 값이 포함되어 있습니다.

cdn-rid

요청에 대한 CloudFront 고유 식별자가 있는 값이 포함되어 있습니다. 지원를 통해 문제를 해결할 때 이 요청 식별자(RID)를 사용할 수 있습니다.

cdn-hit-layer

CloudFront가 원본에 요청하지 않고 캐시에서 응답을 제공하는 경우 이 지표가 표시됩니다. 다음 값 중 하나를 포함합니다.

- EDGE – CloudFront가 POP 위치에서 캐시된 응답을 제공했습니다.
- REC - CloudFront가 리전 엣지 캐시(REC) 위치에서 캐시된 응답을 제공했습니다.
- Origin Shield – CloudFront가 Origin Shield 역할을 하는 REC에서 캐시된 응답을 제공했습니다.

cdn-upstream-layer

CloudFront가 원본에서 전체 객체를 요청하는 경우 이 지표가 표시되며 이 지표에는 다음 값 중 하나가 포함됩니다.

- EDGE – POP 위치에서 요청을 원본으로 직접 보냈습니다.
- REC – REC 위치에서 요청을 원본으로 직접 보냈습니다.
- Origin Shield – Origin Shield 역할을 하는 REC가 요청을 원본으로 직접 보냈습니다.

cdn-upstream-dns

원본에 대한 DNS 레코드를 검색하는 데 소요된 시간(밀리초) 값이 포함됩니다. 값이 0이면 CloudFront에서 캐시된 DNS 결과를 사용했거나 기존 연결을 재사용했음을 나타냅니다.

cdn-upstream-connect

원본 DNS 요청이 완료된 시점과 원본에 대한 TCP(및 해당하는 경우 TLS) 연결이 완료된 시점 사이의 시간(밀리초) 값이 포함됩니다. 값이 0이면 CloudFront에서 기존 연결을 재사용했음을 나타냅니다.

cdn-upstream-fbl

원본 HTTP 요청이 완료된 시점과 원본의 응답에서 첫 번째 바이트가 수신된 시점(첫 번째 바이트 대기 시간) 사이의 밀리초 값이 포함됩니다.

cdn-downstream-fbl

엣지 로케이션에서 요청 수신이 완료된 시점과 사용자에게 응답의 첫 번째 바이트가 전송된 시점 사이의 밀리초 값이 포함됩니다.

## Server-Timing 헤더 예시

다음은 `Server-Timing` 헤더 설정이 사용 설정된 경우 뷰어가 CloudFront에서 수신할 수 있는 `Server-Timing` 헤더의 예입니다.

### Example- 캐시 누락

다음 예제는 요청된 객체가 CloudFront 캐시에 없을 때 뷰어가 수신할 수 있는 `Server-Timing` 헤더를 보여줍니다.

```
Server-Timing: cdn-upstream-layer;desc="EDGE",cdn-upstream-dns;dur=0,cdn-upstream-connect;dur=114,cdn-upstream-fbl;dur=177,cdn-cache-miss,cdn-pop;desc="PHX50-C2",cdn-rid;desc="yNPsyYn7skvTzwWkq3Wcc8Nj_foxUjQUe9H1ifslzWhb0w7aLbFvGg==",cdn-downstream-fbl;dur=436
```

이 `Server-Timing` 헤더는 다음을 나타냅니다.

- 원본 요청이 CloudFront 접속 지점 (POP) 위치(`cdn-upstream-layer;desc="EDGE"`)에서 전송되었습니다.
- CloudFront가 원본에 대해 캐시된 DNS 결과를 사용했습니다(`cdn-upstream-dns;dur=0`).
- CloudFront가 원본에 대한 TCP(및 해당하는 경우 TLS) 연결을 완료하는 데 114밀리초가 소요되었습니다(`cdn-upstream-connect;dur=114`).
- 요청을 완료한 후 CloudFront가 원본에서 응답의 첫 번째 바이트를 수신하는 데 177밀리초가 소요되었습니다(`cdn-upstream-fbl;dur=177`).
- 요청된 객체가 CloudFront의 캐시에 없었습니다(`cdn-cache-miss`).
- 요청이 코드 `PHX50-C2`(`cdn-pop;desc="PHX50-C2"`)로 식별된 엣지 로케이션에서 수신되었습니다.
- 이 요청에 대한 CloudFront 고유 ID는 `yNPsyYn7skvTzwWkq3Wcc8Nj_foxUjQUe9H1ifslzWhb0w7aLbFvGg==`(`cdn-rid;desc="yNPsyYn7skvTzwWkq3Wcc8Nj_foxUjQUe9H1ifslzWhb0w7aLbFvGg==")`이었습니다.
- 최종 뷰어 요청을 받은 후 CloudFront가 뷰어에게 응답의 첫 번째 바이트를 보내는 데 436밀리초가 소요되었습니다(`cdn-downstream-fbl;dur=436`).

### Example- 캐시 적중률

다음 예제는 요청된 객체가 CloudFront 캐시에 있을 때 뷰어가 수신할 수 있는 `Server-Timing` 헤더를 보여줍니다.

```
Server-Timing: cdn-cache-hit,cdn-pop;desc="SEA19-C1",cdn-rid;desc="nQBz4aJU2kP9iC3KHEq7vFxfMozu-VYBwGzkW9diOpeVc7xsrLKj-g==",cdn-hit-layer;desc="REC",cdn-downstream-fbl;dur=137
```

이 `Server-Timing` 헤더는 다음을 나타냅니다.

- 요청된 객체가 캐시에 있었습니다(`cdn-cache-hit`).
- 요청이 코드 `SEA19-C1`(`cdn-pop;desc="SEA19-C1"`)로 식별된 엣지 로케이션에서 수신되었습니다.
- 이 요청에 대한 CloudFront 고유 ID는 `nQBz4aJU2kP9iC3KHEq7vFxfMozu-VYBwGzkW9diOpeVc7xsrLKj-g==`(`cdn-rid;desc="nQBz4aJU2kP9iC3KHEq7vFxfMozu-VYBwGzkW9diOpeVc7xsrLKj-g==")`이었습니다.
- 요청된 객체가 리전 엣지 캐시(REC) 위치에 캐시되었습니다(`cdn-hit-layer;desc="REC"`).
- 최종 뷰어 요청을 받은 후 CloudFront가 뷰어에게 응답의 첫 번째 바이트를 보내는 데 137밀리초가 소요되었습니다(`cdn-downstream-fbl;dur=137`).

# 응답 헤더 정책 생성

응답 헤더 정책을 사용하여 Amazon CloudFront가 HTTP 응답에서 추가 또는 제거하는 HTTP 헤더를 지정할 수 있습니다. 응답 헤더 정책 및 사용 이유에 대한 자세한 내용은 정책을 통해 응답 헤더 추가 또는 제거 단원을 참조하세요.

CloudFront 콘솔에서 응답 헤더 정책을 생성할 수 있습니다. 다른 방법으로 AWS CloudFormation, AWS Command Line Interface(AWS CLI) 또는 CloudFront API를 사용하여 정책을 생성할 수 있습니다. 응답 헤더 정책을 생성한 후 CloudFront 배포의 하나 이상의 캐시 동작에 연결합니다.

사용자 지정 응답 헤더 정책을 생성하기 전에 관리형 응답 헤더 정책 중 하나가 사용 사례에 맞는지 확인합니다. 사용 사례에 맞는 경우 캐시 동작에 연결할 수 있습니다. 이렇게 하면 자체 응답 헤더 정책을 생성하거나 관리할 필요가 없습니다.

Console

응답 헤더 정책을 생성하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home#/policies/responseHeaders에서 CloudFront 콘솔의 정책 페이지의 응답 헤더(Response headers) 탭으로 이동합니다.
2. 응답 헤더 정책 생성(Create response headers policy)을 선택합니다.
3. 응답 헤더 정책 생성(Create response headers policy) 양식에서 다음을 수행합니다.

a. 세부 정보(Details) 패널에서 응답 헤더 정책의 이름과 정책의 목적을 설명하는 설명(선택 사항)을 입력합니다.

b. 교차 원본 리소스 공유(CORS) 패널에서 CORS 구성(Configure CORS) 토글을 선택하고 정책에 추가할 CORS 헤더를 구성합니다. 구성된 헤더가 CloudFront가 원본서 수신하는 헤더를 재정의하도록 하려면 원본 재정의(Origin override) 확인란을 선택합니다.

   CORS 헤더 설정에 대한 자세한 내용은 the section called “CORS 헤더”를 참조하십시오.

c. 보안 헤더(Security headers) 패널에서 토글을 선택하고 정책에 추가할 각 보안 헤더를 구성합니다.

   보안 헤더 설정에 대한 자세한 내용은 the section called “보안 헤더”를 참조하십시오.

d. 사용자 지정 헤더(Custom headers) 패널에서 정책에 포함할 사용자 지정 헤더를 추가합니다.

   사용자 지정 헤더 설정에 대한 자세한 내용은 the section called “사용자 지정 헤더”를 참조하십시오.

e. 헤더 제거 패널에서 CloudFront로 하여금 오리진 응답에서 제거하고 최종 사용자에게 보내는 응답에는 포함하지 않게 하려는 헤더의 이름을 추가합니다.

   헤더 제거 설정에 대한 자세한 내용은 the section called “헤더 제거”를 참조하십시오.

f. Server-Timing 헤더 패널에서 활성화 토글을 선택하고 샘플링 속도(0~100의 숫자)를 입력합니다.

   `Server-Timing` 헤더에 대한 자세한 내용은 the section called “Server-Timing 헤더” 섹션을 참조하십시오.

4. 생성(Create)을 선택하여 정책을 생성합니다.

응답 헤더 정책을 생성한 후 CloudFront 배포의 하나의 캐시 동작에 연결할 수 있습니다.

기존 배포에 응답 헤더 정책을 연결하려면(콘솔)

1. https://console.aws.amazon.com/cloudfront/v4/home#/distributions의 CloudFront 콘솔에서 [Distributions(배포)] 페이지를 엽니다.
2. 업데이트할 배포를 선택한 다음 Behaviors(동작) 탭을 선택합니다.
3. 업데이트할 캐시 동작을 선택한 다음 편집(Edit)을 선택합니다.

   또는 새 캐시 동작을 생성하려면 Create Behavior(동작 생성)를 선택합니다.

4. 응답 헤더 정책(Response headers policy)에서 캐시 동작에 추가할 정책을 선택합니다.
5. 변경 사항 저장(Save changes)을 선택하여 캐시 동작을 업데이트합니다. 새 캐시 동작을 생성하는 경우 동작 생성(Create behavior)을 선택합니다.

새 배포에 응답 헤더 정책을 연결하려면(콘솔)

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포 생성을 선택합니다.
3. 응답 헤더 정책(Response headers policy)에서 캐시 동작에 추가할 정책을 선택합니다.
4. 배포에 대한 다른 설정을 선택합니다. 자세한 내용은 the section called “모든 배포 설정” 단원을 참조하십시오.
5. 배포 생성(Create distribution)을 선택하여 배포를 생성합니다.

## CloudFormation

CloudFormation를 사용하여 응답 헤더 정책을 생성하려면 `AWS::CloudFront::ResponseHeadersPolicy` 리소스 유형을 사용하십시오. 다음 예는 응답 헤더 정책을 생성하기 위한 YAML 형식의 CloudFormation 템플릿 구문을 보여줍니다.

```
Type: AWS::CloudFront::ResponseHeadersPolicy
Properties:
  ResponseHeadersPolicyConfig:
    Name: EXAMPLE-Response-Headers-Policy
    Comment: Example response headers policy for the documentation
    CorsConfig:
      AccessControlAllowCredentials: false
      AccessControlAllowHeaders:
        Items:
          - '*'
      AccessControlAllowMethods:
        Items:
          - GET
          - OPTIONS
      AccessControlAllowOrigins:
        Items:
          - https://example.com
          - https://docs.example.com
      AccessControlExposeHeaders:
        Items:
```

```
          - '*'
      AccessControlMaxAgeSec: 600
      OriginOverride: false
    CustomHeadersConfig:
      Items:
        - Header: Example-Custom-Header-1
          Value: value-1
          Override: true
        - Header: Example-Custom-Header-2
          Value: value-2
          Override: true
    SecurityHeadersConfig:
      ContentSecurityPolicy:
        ContentSecurityPolicy: default-src 'none'; img-src 'self'; script-src
 'self'; style-src 'self'; object-src 'none'; frame-ancestors 'none'
        Override: false
      ContentTypeOptions: # You don't need to specify a value for 'X-Content-Type-
Options'.
                          # Simply including it in the template sets its value to
 'nosniff'.
        Override: false
      FrameOptions:
        FrameOption: DENY
        Override: false
      ReferrerPolicy:
        ReferrerPolicy: same-origin
        Override: false
      StrictTransportSecurity:
        AccessControlMaxAgeSec: 63072000
        IncludeSubdomains: true
        Preload: true
        Override: false
      XSSProtection:
        ModeBlock: true # You can set ModeBlock to 'true' OR set a value for
 ReportUri, but not both
        Protection: true
        Override: false
    ServerTimingHeadersConfig:
      Enabled: true
      SamplingRate: 50
    RemoveHeadersConfig:
      Items:
        - Header: Vary
```

```
        - Header: X-Powered-By
```

자세한 내용은 AWS CloudFormation 사용 설명서의 AWS::CloudFront::ResponseHeadersPolicy를 참조하십시오.

CLI

AWS Command Line Interface(AWS CLI)를 사용하여 응답 헤더 정책을 생성하려면 aws cloudfront create-response-headers-policy 명령을 사용합니다. 각 개별 파라미터를 명령줄 입력으로 지정하는 대신 입력 파일을 사용하여 명령의 입력 파라미터를 제공할 수 있습니다.

응답 헤더 정책을 생성하려면(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 이름이 `response-headers-policy.yaml`인 파일을 생성합니다. 이 파일에는 create-response-headers-policy 명령에 대한 모든 입력 파라미터가 들어 있습니다.

   ```
   aws cloudfront create-response-headers-policy --generate-cli-skeleton yaml-input
    > response-headers-policy.yaml
   ```

2. 방금 생성한 `response-headers-policy.yaml` 파일을 엽니다. 파일을 편집하여 정책 이름과 원하는 응답 헤더 정책 구성을 지정한 다음 파일을 저장합니다.

   응답 헤더 정책 설정에 대한 자세한 내용은 the section called “응답 헤더 정책 이해”를 참조하세요.

3. 응답 헤더 정책을 생성하려면 다음 명령을 사용합니다. 생성하는 정책은 `response-headers-policy.yaml` 파일의 입력 파라미터를 사용합니다.

   ```
   aws cloudfront create-response-headers-policy --cli-input-yaml file://response-
   headers-policy.yaml
   ```

   명령 출력에 있는 `Id` 값을 기록해 둡니다. 이는 응답 헤더 정책 ID입니다. CloudFront 배포의 캐시 동작에 정책을 연결하는 데 필요합니다.

기존 배포에 응답 헤더 정책을 연결하려면(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 업데이트할 CloudFront 배포에 대한 배포 구성을 저장합니다. *`distribution_ID`*를 배포 ID로 바꿉니다.

```
aws cloudfront get-distribution-config --id distribution_ID --output yaml >
 dist-config.yaml
```

2. 방금 생성한 `dist-config.yaml`이라는 파일을 엽니다. 파일을 편집하여 응답 헤더 정책을 사용하도록 캐시 동작을 다음과 같이 변경합니다.

   - 캐시 동작에서 `ResponseHeadersPolicyId`라는 필드를 추가합니다. 필드 값에는 정책을 생성한 후 기록해 둔 응답 헤더 정책 ID를 사용합니다.
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

   완료되면 파일을 저장합니다.

3. 다음 명령을 사용하여 응답 헤더 정책을 사용하도록 배포를 업데이트합니다. *distribution_ID*를 배포 ID로 바꿉니다.

```
aws cloudfront update-distribution --id distribution_ID --cli-input-yaml file://
dist-config.yaml
```

새로운 배포에 응답 헤더 정책을 연결하려면(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 이름이 `distribution.yaml`인 파일을 생성합니다. 이 파일에는 create-distribution 명령에 대한 모든 입력 파라미터가 들어 있습니다.

```
aws cloudfront create-distribution --generate-cli-skeleton yaml-input >
 distribution.yaml
```

2. 방금 생성한 `distribution.yaml` 파일을 엽니다. 기본 캐시 동작의 `ResponseHeadersPolicyId` 필드에 정책을 생성한 후 기록한 응답 정책 ID를 입력합니다. 파일을 계속 편집하여 원하는 배포 설정을 지정한 다음 완료되면 파일을 저장합니다.

   배포 설정에 대한 자세한 내용은 모든 배포 설정 참조 단원을 참조하세요.

3. 다음 명령을 사용하여 `distribution.yaml` 파일의 입력 파라미터로 배포를 만듭니다.

```
aws cloudfront create-distribution --cli-input-yaml file://distribution.yaml
```

API

CloudFront API로 응답 헤더 정책을 생성하려면 CreateResponseHeadersPolicy를 사용하십시오. 이 API 호출에서 지정하는 필드에 대한 자세한 내용은 the section called "응답 헤더 정책 이해" 및 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

응답 헤더 정책을 생성한 후 다음 API 호출 중 하나를 사용하여 캐시 동작에 연결할 수 있습니다.

- 기존 배포의 캐시 동작에 연결하려면 UpdateDistribution을 사용합니다.
- 새 배포의 캐시 동작에 연결하려면 CreateDistribution을 사용합니다.

이 두 API 호출 모두에 대해 캐시 동작 내부의 `ResponseHeadersPolicyId` 필드에 요청 헤더 정책 ID를 제공합니다. 이러한 API 호출에서 지정하는 다른 필드에 대한 자세한 내용은 모든 배포 설정 참조와 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

# 관리형 응답 헤더 정책 사용

CloudFront 응답 헤더 정책을 사용하면 Amazon CloudFront가 사용자에게 보내는 응답에서 제거하거나 추가하는 HTTP 헤더를 지정할 수 있습니다. 응답 헤더 정책 및 사용 이유에 대한 자세한 내용은 정책을 통해 응답 헤더 추가 또는 제거 단원을 참조하세요.

CloudFront는 CloudFront 배포의 캐시 동작에 연결할 수 있는 관리형 응답 헤더 정책을 제공합니다. 관리형 응답 헤더 정책을 사용하면 자체 정책을 작성하거나 유지 관리할 필요가 없습니다. 관리형 정책은 일반 사용 사례에 대한 HTTP 응답 헤더 집합을 포함합니다.

관리 응답 헤더 정책을 사용하려면 배포의 캐시 동작에 연결합니다. 이 프로세스는 사용자 지정 응답 헤더 정책을 생성할 때와 동일합니다. 그러나 새 정책을 만드는 대신 관리형 정책 중 하나를 연결합니다. 이름(콘솔 사용) 또는 ID(CloudFormation, AWS CLI 또는 AWS SDK 사용)로 정책을 연결합니다. 이름과 ID는 다음 섹션에 나열되어 있습니다.

자세한 내용은 the section called "응답 헤더 정책 생성" 단원을 참조하십시오.

다음 항목에서는 사용할 수 있는 관리형 응답 헤더 정책에 대해 설명합니다.

주제

- CORS-and-SecurityHeadersPolicy
- CORS-With-Preflight
- CORS-with-preflight-and-SecurityHeadersPolicy

- SecurityHeadersPolicy
- SimpleCORS

## CORS-and-SecurityHeadersPolicy

CloudFront 콘솔에서 이 정책 보기

이 관리형 정책을 사용하여 모든 원본의 간단한 CORS 요청을 허용합니다. 또한 이 정책은 CloudFront가 뷰어에게 보내는 모든 응답에 보안 헤더 집합을 추가합니다. 이 정책은 the section called "SimpleCORS"과 the section called "SecurityHeadersPolicy" 정책을 하나로 묶습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`e61eb60c-9c35-4d20-a928-2b84e02af89c`

정책 설정

| | 헤더 이름 | 헤더 값 | 원본을 재정의할까요? |
|---|---|---|---|
| CORS 헤더: | `Access-Control-Allow-Origin` | `*` | 아니요 |
| 보안 헤더: | `Referrer-Policy` | `strict-origin-when-cross-origin` | 아니요 |
| | `Strict-Transport-Security` | `max-age=31536000` | 아니요 |
| | `X-Content-Type-Options` | `nosniff` | 예 |
| | `X-Frame-Options` | `SAMEORIGIN` | 아니요 |
| | `X-XSS-Protection` | `1; mode=block` | 아니요 |

## CORS-With-Preflight

CloudFront 콘솔에서 이 정책 보기

이 관리형 정책을 사용하여 사전 요청을 포함하여 모든 원본의 CORS 요청을 허용합니다. 사전 요청(HTTP `OPTIONS` 메서드 사용)의 경우 CloudFront는 다음 세 가지 헤더를 모두 응답에 추가합니다. 간단한 CORS 요청의 경우 CloudFront는 `Access-Control-Allow-Origin` 헤더만 추가합니다.

CloudFront가 원본에서 수신하는 응답에 이러한 헤더가 포함된 경우 CloudFront는 뷰어에 대한 응답에서 수신된 헤더(및 해당 값)를 사용합니다. 이 경우 CloudFront는 이 정책의 헤더를 사용하지 않습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`5cc3b908-e619-4b99-88e5-2cf7f45965bd`

정책 설정

| | 헤더 이름 | 헤더 값 | 원본을 재정의할까요? |
|---|---|---|---|
| CORS 헤더: | `Access-Control-Allow-Methods` | `DELETE, GET, HEAD, OPTIONS, PATCH, POST, PUT` | 아니요 |
| | `Access-Control-Allow-Origin` | `*` | |
| | `Access-Control-Expose-Headers` | `*` | |

## CORS-with-preflight-and-SecurityHeadersPolicy

CloudFront 콘솔에서 이 정책 보기

이 관리형 정책을 사용하여 모든 원본의 CORS 요청을 허용합니다. 여기에는 사전 요청이 포함됩니다. 또한 이 정책은 CloudFront가 뷰어에게 보내는 모든 응답에 보안 헤더 집합을 추가합니다. 이 정책은 the section called “CORS-With-Preflight”과 the section called “SecurityHeadersPolicy” 정책을 하나로 묶습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`eaab4381-ed33-4a86-88ca-d9558dc6cd63`

정책 설정

| | 헤더 이름 | 헤더 값 | 원본을 재정의 할까요? |
|---|---|---|---|
| CORS 헤더: | `Access-Control-Allow-Methods` | `DELETE, GET, HEAD, OPTIONS, PATCH, POST, PUT` | 아니요 |
| | `Access-Control-Allow-Origin` | `*` | |
| | `Access-Control-Expose-Headers` | `*` | |
| 보안 헤더: | `Referrer-Policy` | `strict-origin-when-cross-origin` | 아니요 |
| | `Strict-Transport-Security` | `max-age=31536000` | 아니요 |
| | `X-Content-Type-Options` | `nosniff` | 예 |
| | `X-Frame-Options` | `SAMEORIGIN` | 아니요 |
| | `X-XSS-Protection` | `1; mode=block` | 아니요 |

## SecurityHeadersPolicy

CloudFront 콘솔에서 이 정책 보기

이 관리형 정책을 사용하여 CloudFront가 뷰어에게 보내는 모든 응답에 보안 헤더 집합을 추가할 수 있습니다. 이러한 보안 헤더에 대한 자세한 내용은 Mozilla의 웹 보안 가이드라인을 참조하십시오.

이 응답 헤더 정책에서 CloudFront는 모든 응답에 `X-Content-Type-Options: nosniff`를 추가합니다. CloudFront가 원본에서 받은 응답에 이 헤더가 포함되어 있는지 여부와 상관 없습니다.
이 정책의 다른 모든 헤더에 대해 CloudFront가 원본에서 수신하는 응답에 해당 헤더가 포함된 경우 CloudFront는 뷰어에 대한 응답에서 수신된 헤더(및 해당 값)를 사용합니다. 이 경우 CloudFront는 이 정책의 헤더를 사용하지 않습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`67f7725c-6f97-4210-82d7-5512b31e9d03`

정책 설정

| | 헤더 이름 | 헤더 값 | 원본을 재정의할까요? |
|---|---|---|---|
| 보안 헤더: | `Referrer-Policy` | `strict-origin-when-cross-origin` | 아니요 |
| | `Strict-Transport-Security` | `max-age=31536000` | 아니요 |
| | `X-Content-Type-Options` | `nosniff` | 예 |
| | `X-Frame-Options` | `SAMEORIGIN` | 아니요 |
| | `X-XSS-Protection` | `1; mode=block` | 아니요 |

## SimpleCORS

CloudFront 콘솔에서 이 정책 보기

이 관리형 정책을 사용하여 모든 원본의 간단한 CORS 요청을 허용합니다. 이 정책을 사용하면 CloudFront에서 단순 CORS 요청에 대한 모든 응답에 헤더 `Access-Control-Allow-Origin: *`를 추가합니다.

CloudFront가 원본에서 수신하는 응답에 `Access-Control-Allow-Origin` 헤더가 포함된 경우 CloudFront는 뷰어에 대한 응답에서 해당 헤더(및 해당 값)를 사용합니다. 이 경우 CloudFront는 이 정책의 헤더를 사용하지 않습니다.

CloudFormation, AWS CLI 또는 CloudFront API를 사용할 때 이 정책의 ID는 다음과 같습니다.

`60669652-455b-4ae9-85a4-c4c02393f86c`

정책 설정

| | 헤더 이름 | 헤더 값 | 원본을 재정의할까요? |
|---|---|---|---|
| CORS 헤더: | `Access-Control-Allow-Origin` | `*` | 아니요 |

# 요청 및 응답 동작

다음 주제에서는 CloudFront에서 요청 및 응답을 처리하는 방법을 설명합니다.

CloudFront가 Amazon S3 또는 사용자 지정 오리진과 상호 작용하고, 다양한 HTTP 메서드 및 헤더를 취급하고, 상태 코드를 처리하고, 캐싱 및 오류 응답을 관리하는 방법에 대해 알아볼 수 있습니다.

주제


- CloudFront에서 HTTP 및 HTTPS 요청을 처리하는 방법
- Amazon S3 오리진에 대한 요청 및 응답 동작
- 사용자 지정 오리진에 대한 요청 및 응답 동작
- 오리진 그룹에 대한 요청 및 응답 동작
- 사용자 지정 헤더를 오리진 요청에 추가
- CloudFront에서 객체에 대한 부분적인 요청을 처리하는 방법(범위 GET)
- CloudFront에서 오리진의 HTTP 3xx 상태 코드를 처리하는 방법
- CloudFront에서 오리진의 HTTP 4xx 및 5xx 상태 코드를 처리하는 방법
- 사용자 지정 오류 응답 생성


## CloudFront에서 HTTP 및 HTTPS 요청을 처리하는 방법

Amazon S3 오리진의 경우, CloudFront에서는 기본적으로 CloudFront 배포의 객체에 대한 HTTP 및 HTTPS 프로토콜 모두의 요청을 수락합니다. 그런 뒤 CloudFront에서는 요청이 전달된 프로토콜과 같은 프로토콜을 사용하여 이 요청을 Amazon S3 버킷에 전달합니다.

사용자 지정 오리진의 경우 배포를 만들 때 HTTP 전용 또는 최종 사용자가 사용한 프로토콜과 일치 중에 CloudFront에서 오리진에 액세스하는 방법을 지정할 수 있습니다. CloudFront에서 사용자 지정 오리진에 대한 HTTP 및 HTTPS 요청을 처리하는 방법에 대한 자세한 내용은 프로토콜 단원을 참조하십시오.

최종 사용자가 HTTPS만 사용하여 객체에 액세스할 수 있도록 배포를 제한하는 방법에 대한 자세한 내용은 CloudFront에서 HTTPS 사용 섹션을 참조하세요.

> **Note**
>
> HTTPS 요청에 대한 요금은 HTTP 요청에 대한 요금에 비해 높습니다. 부과되는 요율에 대한 자세한 내용은 CloudFront 요금을 참조하세요.

# Amazon S3 오리진에 대한 요청 및 응답 동작

오리진으로 Amazon S3를 사용할 때 CloudFront에서 요청과 응답을 처리하는 방법을 이해하려면 다음 섹션을 참조하세요.

주제

- CloudFront에서 요청을 처리하고 Amazon S3 오리진에 요청을 전달하는 방법
- CloudFront에서 Amazon S3 오리진의 요청을 처리하는 방법

## CloudFront에서 요청을 처리하고 Amazon S3 오리진에 요청을 전달하는 방법

CloudFront에서 뷰어 요청을 처리하고 이 요청을 Amazon S3 오리진에 전달하는 방법을 알아봅니다.

목차


- 캐싱 시간 및 최소 TTL
- 클라이언트 IP 주소
- 조건부 GET 요청
- Cookies
- 교차 오리진 리소스 공유(CORS)
- 본문이 포함되는 GET 요청
- HTTP 메소드
- CloudFront에서 제거하거나 업데이트하는 HTTP 요청 헤더
- 요청의 최대 길이 및 최대 URL 길이
- OCSP 스테이플링
- 프로토콜
- 쿼리 문자열

- 오리진 연결 제한 시간 및 시도 횟수
- 오리진 응답 제한 시간
- 동일 객체에 대한 동시 요청(요청 축소)

## 캐싱 시간 및 최소 TTL

CloudFront에서 다른 요청을 오리진에 전달하기 전에 객체를 CloudFront 캐시에 보관하는 시간을 제어하려면 다음을 수행합니다.

- `Cache-Control` 또는 `Expires` 헤더 파일을 각 객체에 추가하도록 오리진을 구성합니다.
- CloudFront 캐시 동작에 최소 TTL 값을 지정합니다.
- 기본값인 24시간을 사용합니다.

자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 단원을 참조하세요.

## 클라이언트 IP 주소

뷰어가 CloudFront에 요청을 보내고 `X-Forwarded-For` 요청 헤더를 포함하지 않는 경우, CloudFront는 TCP 연결에서 뷰어의 IP 주소를 가져오고 IP 주소를 포함하는 `X-Forwarded-For` 헤더를 추가하고 오리진에 요청을 전달합니다. 예를 들어, CloudFront가 TCP 연결에서 IP 주소 `192.0.2.2`를 가져오면 오리진에 다음 헤더를 전달합니다.

```
X-Forwarded-For: 192.0.2.2
```

최종 사용자가 CloudFront에 요청을 보내고 `X-Forwarded-For` 요청 헤더를 포함하는 경우, CloudFront는 TCP 연결에서 최종 사용자의 IP 주소를 가져와 `X-Forwarded-For` 헤더의 끝에 첨부하고 오리진에 요청을 전달합니다. 예를 들어, 최종 사용자 요청에 `X-Forwarded-For: 192.0.2.4,192.0.2.3`이 포함되고 CloudFront가 TCP 연결에서 IP 주소 `192.0.2.2`를 가져오면 오리진에 다음 헤더를 전달합니다.

```
X-Forwarded-For: 192.0.2.4,192.0.2.3,192.0.2.2
```

> **Note**
>
> `X-Forwarded-For` 헤더에는 IPv4 주소(예: 192.0.2.44)와 IPv6 주소(예: 2001:0db8:85a3::8a2e:0370:7334)가 포함됩니다.

## 조건부 GET 요청

CloudFront는 엣지 캐시에서 만료된 객체에 대한 요청을 받으면 이 요청을 Amazon S3 오리진으로 전달하여 최신 버전의 객체를 가져오거나 CloudFront 엣지 캐시에 이미 최신 버전이 있는지 Amazon S3의 확인을 받습니다. Amazon S3에서는 처음에 CloudFront에 객체를 보낼 때 `ETag` 값과 `LastModified` 값을 응답에 포함하여 보냈습니다. CloudFront에서 Amazon S3에 전달하는 새 요청에서 CloudFront는 다음 헤더 중 하나 또는 둘 모두를 추가합니다.

- 만료된 버전의 객체에 대한 `If-Match` 값을 포함하는 `If-None-Match` 또는 `ETag` 헤더
- 만료된 버전의 객체에 대한 `If-Modified-Since` 값을 포함하는 `LastModified` 헤더

Amazon S3에서는 이 정보를 사용하여 객체가 업데이트되는지와, 그에 따라 전체 객체를 CloudFront에 반환할지 아니면 HTTP 304 상태 코드만 반환할지(수정되지 않음) 여부를 결정합니다.

## Cookies

Amazon S3는 쿠키를 처리하지 않습니다. 쿠키를 Amazon S3 오리진에 전달하도록 캐시 동작을 구성하는 경우, CloudFront에서는 이 쿠키를 전달하지만 Amazon S3에서는 이를 무시합니다. 향후 동일한 객체에 대한 모든 요청은 쿠키 변경 여부와 관계없이 캐시에 들어 있는 기존 객체로 처리합니다.

## 교차 오리진 리소스 공유(CORS)

CloudFront에서 Amazon S3 CORS(Cross-Origin Resource Sharing) 설정을 준수하도록 하려는 경우 선택한 헤더를 Amazon S3로 전달하도록 CloudFront를 구성합니다. 자세한 내용은 요청 헤더 기반의 콘텐츠 캐싱 단원을 참조하세요.

## 본문이 포함되는 GET 요청

최종 사용자 `GET` 요청에 본문이 포함되는 경우, CloudFront에서는 HTTP 상태 코드 403(금지됨)을 최종 사용자에게 반환합니다.

## HTTP 메소드

지원되는 모든 HTTP 메소드를 처리하도록 CloudFront를 구성하는 경우, CloudFront에서는 최종 사용자의 다음 요청을 수락하고 이를 Amazon S3 오리진에 전달합니다.

- `DELETE`
- `GET`

- HEAD
- OPTIONS
- PATCH
- POST
- PUT

CloudFront에서는 GET 및 HEAD 요청에 대한 응답을 항상 캐싱합니다. 또한 OPTIONS 요청에 대한 응답을 캐싱하도록 CloudFront를 구성할 수도 있습니다. CloudFront에서는 다른 메서드를 사용하는 요청에 대한 응답을 캐싱하지 않습니다.

다중 파트 업로드를 사용하여 객체를 Amazon S3 버킷에 추가하려는 경우, CloudFront 오리진 액세스 제어(OAC)를 배포에 추가하고 이 OAC에 필요한 사용 권한을 부여해야 합니다. 자세한 내용은 the section called "Amazon S3 오리진에 대한 액세스 제한" 섹션을 참조하세요.

> **Important**
>
> CloudFront에서 지원하는 모든 HTTP 메서드를 허용하고 Amazon S3에 전달하도록 CloudFront를 구성하는 경우, Amazon S3 콘텐츠에 대한 액세스를 제한하는 CloudFront OAC를 만들어 이 OAC에 필수 사용 권한을 부여해야 합니다. 예를 들어, PUT 메서드를 사용할 의도로 이러한 메서드를 허용 및 전달하도록 CloudFront를 구성한 경우, 뷰어에 의해 삭제되는 것을 원치 않는 리소스를 이들이 삭제할 수 없도록 DELETE 요청을 적절히 처리하여 Amazon S3 버킷 정책을 구성해야 합니다. 자세한 내용은 the section called "Amazon S3 오리진에 대한 액세스 제한" 섹션을 참조하세요.

Amazon S3에서 지원되는 동작에 대한 자세한 내용은 Amazon S3 설명서를 참조하십시오.

## CloudFront에서 제거하거나 업데이트하는 HTTP 요청 헤더

CloudFront는 Amazon S3 오리진에 요청을 전달하기 전에 몇몇 헤더를 제거하거나 업데이트합니다. 대부분의 헤더에서 이 동작은 사용자 지정 오리진에서의 동작과 같습니다. HTTP 요청 헤더의 전체 목록과 CloudFront에서 이를 처리하는 방법은 HTTP 요청 헤더 및 CloudFront 동작(사용자 지정 및 Amazon S3 오리진) 단원을 참조하십시오.

## 요청의 최대 길이 및 최대 URL 길이

경로, 쿼리 문자열(있는 경우), 헤더를 모두 포함한 요청의 최대 길이는 20,480바이트입니다.

CloudFront에서는 이 요청으로부터 URL을 구성합니다. 이 URL의 최대 길이는 8,192바이트입니다.

요청 또는 URL이 최대 길이를 초과할 경우 CloudFront에서는 HTTP 요청 코드 413(요청 개체가 너무 큼)을 반환한 후 뷰어와의TCP 연결을 종료합니다.

## OCSP 스테이플링

뷰어가 객체에 대한 HTTPS 요청을 제출할 때 CloudFront 또는 뷰어는 CA(인증 기관)을 통해 도메인의 SSL 인증서가 해지되지 않았는지 확인해야 합니다. OCSP 스테이플링은 CloudFront에서 인증서의 유효성을 검사하고 CA로부터 응답을 캐싱할 수 있도록 함으로써 클라이언트가 CA를 통해 직접 인증서의 유효성을 검사하지 않아도 되므로 인증서 유효성 검사 속도가 향상됩니다.

OSCP 스테이플링의 성능 개선은 CloudFront에서 같은 도메인 내의 객체에 대해 많은 HTTPS 요청을 받은 경우 더욱 확연히 드러납니다. CloudFront 엣지 로케이션의 각 서버는 개별적인 유효성 검사 요청을 제출해야 합니다. CloudFront에서 같은 도메인에 대해 다수의 HTTPS 요청을 받은 경우, 엣지 로케이션의 각 서버에서는 곧 CA로부터 SSL 핸드쉐이크의 패킷에 스테이플할 수 있다는 응답을 받습니다. 뷰어가 인증서가 유효하다는 조건을 충족하면 CloudFront에서는 요청된 객체를 제공할 수 있습니다. 배포가 CloudFront 엣지 로케이션에서 많은 양의 트래픽을 받지 않는 경우, 새로운 요청은 아직 CA를 통해 인증서의 유효성을 검사하지 않은 서버로 리디렉션될 가능성이 높습니다. 그러한 경우, 최종 사용자는 유효성 검사 단계를 개별적으로 수행하고 CloudFront 서버에서는 객체를 제공합니다. 해당 CloudFront 서버에서는 또한 유효성 검사 요청을 CA에 제출하고, 다음에 동일한 도메인 이름을 포함한 요청을 수신하면 CA로부터 유효성 검사 응답을 받습니다.

## 프로토콜

CloudFront에서는 HTTP 또는 HTTPS 요청을 최종 사용자 요청의 프로토콜(HTTP 또는 HTTPS)을 바탕으로 오리진 서버에 전달합니다.

> **Important**
>
> 웹 사이트 엔드포인트로 Amazon S3 버킷이 구성되어 있는 경우, Amazon S3에서 해당 구성으로 HTTPS 연결을 지원하지 않으므로 HTTPS를 사용하여 오리진과 통신하도록 CloudFront를 구성할 수 없습니다.

## 쿼리 문자열

CloudFront에서 쿼리 문자열 파라미터를 Amazon S3 오리진에 전달할지 여부를 구성할 수 있습니다. 자세한 내용은 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱 단원을 참조하세요.

## 오리진 연결 제한 시간 및 시도 횟수

오리진 연결 제한 시간은 CloudFront가 오리진에 대한 연결을 설정하려고 할 때 기다리는 시간(초)입니다.

오리진 연결 시도는 CloudFront가 오리진에 대한 연결을 시도하는 횟수입니다.

이러한 설정은 모두 CloudFront가 보조 오리진에 대해 장애 조치하거나(오리진 그룹의 경우) 뷰어에 대한 오류 응답을 반환하기 전에 오리진에 대한 연결을 시도하는 시간을 결정합니다. 기본적으로 CloudFront는 보조 오리진에 연결하거나 오류 응답을 반환하기 전에 30초(각각 10초 동안 3회 시도)까지 기다립니다. 더 짧은 연결 제한 시간, 더 적은 시도 횟수 중 하나 또는 둘 다를 지정하여 이 시간을 줄일 수 있습니다.

자세한 내용은 오리진 제한 시간 및 시도 횟수 제어 단원을 참조하세요.

## 오리진 응답 제한 시간

오리진 응답 제한 시간(오리진 읽기 제한 시간 또는 오리진 요청 제한 시간이라고도 함)은 다음 두 값에 모두 적용됩니다.

- CloudFront가 오리진에 요청을 전달한 후 응답을 기다리는 시간(초).
- CloudFront가 오리진으로부터 응답 패킷을 수신한 후 다음 패킷을 수신할 때까지 대기하는 시간(초).

CloudFront 동작은 최종 사용자 요청의 HTTP 메서드에 따라 달라집니다.

- `GET` 및 `HEAD` 요청 – 오리진에서 30초 내에 응답하지 않거나 30초 동안 응답이 중지된 경우, CloudFront에서는 연결을 끊습니다. 지정된 오리진 연결 시도 횟수가 1보다 많으면 CloudFront가 다시 연결을 설정하려고 시도합니다. CloudFront는 오리진 연결 시도 설정 값에 따라 최대 3회까지 시도합니다. 오리진이 마지막 시도에 응답하지 않는 경우, CloudFront에서는 동일한 오리진의 콘텐츠에 대해 다른 요청을 받을 때까지 다시 시도하지 않습니다.
- `DELETE`, `OPTIONS`, `PATCH`, `PUT` 및 `POST` 요청 – 오리진에서 30초 내에 응답하지 않는 경우, CloudFront에서는 연결을 끊고 오리진에 다시 연결을 시도하지 않습니다. 필요한 경우 클라이언트는 요청을 다시 제출할 수 있습니다.

Amazon S3 오리진(정적 웹 사이트 호스팅으로 구성되지 않은 S3 버킷)에 대한 응답 제한 시간을 변경할 수 없습니다.

## 동일 객체에 대한 동시 요청(요청 축소)

CloudFront 엣지 로케이션에서 객체에 대한 요청을 받을 때 객체가 캐시에 있지 않거나 캐시된 객체가 만료된 경우, CloudFront에서는 즉시 요청을 오리진으로 보냅니다. 하지만 동일 객체에 대한 동시 요청이 있는 경우, 즉 CloudFront가 첫 번째 요청에 대한 응답을 수신하기 전에 동일 객체에 대한 추가 요청(동일 캐시 키 포함)이 엣지 로케이션에 도착하는 경우 CloudFront는 추가 요청을 오리진에 전달하기 전에 작업을 일시 중지합니다. 이러한 일시 중지는 오리진에 부하가 걸리는 것을 줄여 줍니다. CloudFront는 원래 요청의 응답을 일시 중지된 동안 수신한 모든 요청에 보냅니다. 이를 요청 축소라고 합니다. CloudFront 로그에서, 첫 번째 요청은 `x-edge-result-type` 필드에서 `Miss`로 식별되고 축소된 요청은 `Hit`로 식별됩니다. CloudFront 로그에 대한 자세한 내용은 the section called “CloudFront 및 엣지 함수 로깅” 단원을 참조하세요.

CloudFront는 캐시 키를 공유하는 요청만 축소합니다. 요청 헤더 또는 쿠키 또는 쿼리 문자열에 따라 캐시하도록 CloudFront를 구성하는 등의 이유로 추가 요청이 동일한 캐시 키를 공유하지 않는 경우, CloudFront에서는 고유한 캐시 키가 있는 모든 요청을 오리진에 전달합니다.

모든 요청이 축소되는 것을 방지하려면 캐싱을 방지하는 관리형 캐시 정책 `CachingDisabled`를 사용하면 됩니다. 자세한 내용은 관리형 캐시 정책 사용 섹션을 참조하세요.

특정 객체에 대한 요청 축소를 방지하려면 캐시 동작의 최소 TTL을 0으로 설정하고, 그 후 `Cache-Control: private`, `Cache-Control: no-store`, `Cache-Control: no-cache`, `Cache-Control: max-age=0` 또는 `Cache-Control: s-maxage=0`을 보내도록 오리진을 구성합니다. 이 구성을 사용하면 오리진에 걸리는 부하가 가중되고 CloudFront가 첫 번째 요청에 대한 응답을 기다리는 동안 일시 중지된 동시 요청으로 인한 추가 지연 시간이 발생합니다.

> **Important**
>
> 현재 CloudFront는 캐시 정책, 오리진 요청 정책 또는 레거시 캐시 설정에서 쿠키 전달을 활성화한 경우 요청 축소를 지원하지 않습니다.

# CloudFront에서 Amazon S3 오리진의 요청을 처리하는 방법

CloudFront에서 Amazon S3 오리진의 응답을 처리하는 방법을 알아봅니다.

목차


- 취소된 요청
- CloudFront에서 제거하거나 업데이트하는 HTTP 응답 헤더

- 최대 캐시 가능 파일 크기
- 리디렉션

## 취소된 요청

객체가 엣지 캐시에 있지 않은 경우 최종 사용자가 CloudFront가 오리진에서 객체를 가져온 뒤 요청된 객체를 제공하기 전에 브라우저 닫기 등으로 세션을 종료하면, CloudFront에서는 엣지 로케이션의 객체를 캐싱하지 않습니다.

## CloudFront에서 제거하거나 업데이트하는 HTTP 응답 헤더

CloudFront는 Amazon S3 오리진에서 최종 사용자에게 응답을 전달하기 전에 다음 헤더 필드를 제거하거나 업데이트합니다.

- `X-Amz-Id-2`
- `X-Amz-Request-Id`
- `Set-Cookie` – 쿠키를 전달하도록 CloudFront를 구성하는 경우, `Set-Cookie` 헤더 필드가 클라이언트에 전달됩니다. 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하세요.
- `Trailer`
- `Transfer-Encoding` – Amazon S3 오리진에서 이 헤더 필드를 반환하는 경우, CloudFront에서는 최종 사용자에게 응답을 반환하기 전에 `chunked`에 값을 설정합니다.
- `Upgrade`
- `Via` – CloudFront는 최종 사용자에 대한 응답으로 값을 다음과 같이 설정합니다.

  ```
  Via: http-version alphanumeric-string.cloudfront.net (CloudFront)
  ```

  예를 들어, 값은 다음과 같습니다.

  ```
  Via: 1.1 1026589cc7887e7a0dc7827b4example.cloudfront.net (CloudFront)
  ```

## 최대 캐시 가능 파일 크기

CloudFront에서 캐시에 저장하는 응답 본문의 최대 크기는 50GB입니다. 여기에는 `Content-Length` 헤더 값으로 지정하지 않은 조각난 전송 응답이 포함됩니다.

CloudFront를 사용하면 범위 요청을 사용하여 각각 50GB 이하인 부분으로 객체를 요청하여 이보다 큰 객체를 캐시할 수 있습니다. CloudFront는 각 부분이 50GB 이하이기 때문에 이러한 부분을 캐시합니

다. 뷰어는 객체의 모든 부분을 검색한 후 원래의 더 큰 객체를 재구성할 수 있습니다. 자세한 내용은 범위 요청을 사용하여 대형 객체 캐시 섹션을 참조하세요.

## 리디렉션

모든 요청을 다른 호스트 이름으로 리디렉션하도록 Amazon S3 버킷을 구성할 수 있습니다. 이 대상은 다른 Amazon S3 버킷 또는 HTTP 서버가 될 수 있습니다. 모든 요청을 리디렉션하도록 버킷을 구성하는 경우와 버킷이 CloudFront 배포에 대한 오리진인 경우, 배포의 도메인 이름(예: d111111abcdef8.cloudfront.net) 또는 배포와 연결된 대체 도메인 이름(CNAME)(예: example.com)을 사용하여 모든 요청을 CloudFront 배포에 리디렉션하도록 버킷을 구성하는 것이 좋습니다. 그렇지 않은 경우 최종 사용자는 CloudFront를 우회하도록 요청하고 객체는 새 오리진에서 직접 제공됩니다.

> **Note**
>
> 요청을 대체 도메인 이름으로 리디렉션하는 경우, CNAME 레코드를 추가하여 도메인에 대한 DNS 서비스 역시 업데이트해야 합니다. 자세한 내용은 대체 도메인 이름(CNAME)을 추가하여 사용자 지정 URL 사용 단원을 참조하세요.

모든 요청을 리디렉션하도록 버킷을 구성했을 경우 다음과 같은 상황이 발생합니다.

1. 브라우저 등의 최종 사용자가 CloudFront에서 객체를 요청합니다.
2. CloudFront에서는 이 요청을 배포에 대한 오리진인 Amazon S3 버킷에 전달합니다.
3. Amazon S3에서는 HTTP 상태 코드 301(영구적으로 옮겨짐)과 함께 새 위치를 반환합니다.
4. CloudFront에서는 리디렉션 상태 코드와 새 위치를 캐싱하고 최종 사용자에게 값을 반환합니다. CloudFront는 이 리디렉션을 따라가지 않고 새 위치에서 객체를 가져옵니다.
5. 최종 사용자는 객체에 대한 다른 요청을 보내지만, 이때 최종 사용자는 CloudFront에서 가져온 새 위치를 지정합니다.

   - Amazon S3 버킷에서 모든 요청을 CloudFront 배포로 리디렉션하는 경우, CloudFront에서는 배포의 도메인 이름 또는 대체 도메인 이름을 사용하여 새 위치의 Amazon S3 버킷 또는 HTTP 서버에서 객체를 요청합니다. 새 위치에서 객체를 반환하는 경우, CloudFront에서는 최종 사용자에게 이를 반환하고 엣지 로케이션에 이를 캐싱합니다.
   - Amazon S3 버킷에서 요청을 다른 위치로 리디렉션하는 경우, 두 번째 요청은 CloudFront를 우회합니다. 새 위치의 Amazon S3 버킷 또는 HTTP 서버에서 최종 사용자에게 직접 객체를 반환하므로 객체는 CloudFront 엣지 캐시에서 캐싱되지 않습니다.

# 사용자 지정 오리진에 대한 요청 및 응답 동작

사용자 지정 오리진을 사용할 때 CloudFront에서 요청과 응답을 처리하는 방법을 이해하려면 다음 섹션을 참조하세요.

주제


- CloudFront에서 요청을 처리하고 사용자 지정 오리진에 요청을 전달하는 방법
- CloudFront에서 사용자 지정 오리진 서버의 요청을 처리하는 방법


## CloudFront에서 요청을 처리하고 사용자 지정 오리진에 요청을 전달하는 방법

CloudFront에서 뷰어 요청을 처리하고 이 요청을 사용자 지정 오리진에 전달하는 방법을 알아봅니다.

목차


- 인증
- 캐싱 시간 및 최소 TTL
- 클라이언트 IP 주소
- 클라이언트측 SSL 인증
- 압축
- 조건부 요청
- Cookies
- 교차 오리진 리소스 공유(CORS)
- 암호화
- 본문이 포함되는 GET 요청
- HTTP 메소드
- HTTP 요청 헤더 및 CloudFront 동작(사용자 지정 및 Amazon S3 오리진)
- HTTP 버전
- 요청의 최대 길이 및 최대 URL 길이
- OCSP 스테이플링
- 지속적인 연결

- 프로토콜
- 쿼리 문자열
- 오리진 연결 제한 시간 및 시도 횟수
- 오리진 응답 제한 시간
- 동일 객체에 대한 동시 요청(요청 축소)
- User-Agent 헤더

## 인증

`Authorization` 헤더를 오리진에 전달하는 경우, 다음 요청 유형에 대해 클라이언트 인증을 요청하도록 오리진 서버를 구성할 수 있습니다.

- `DELETE`
- `GET`
- `HEAD`
- `PATCH`
- `PUT`
- `POST`

`OPTIONS` 요청의 경우 클라이언트 인증은 다음 CloudFront 설정을 사용하는 경우에만 구성할 수 있습니다:

- CloudFront는 `Authorization` 헤더를 오리진으로 전달하도록 구성됩니다.
- CloudFront는 `OPTIONS` 요청에 대한 응답을 캐시하지 않도록 구성됩니다.

자세한 내용은 `Authorization` 헤더를 전달하도록 CloudFront 구성 섹션을 참조하세요.

HTTP 또는 HTTPS를 사용하여 오리진 서버에 요청을 전달할 수 있습니다. 자세한 내용은 CloudFront에서 HTTPS 사용 섹션을 참조하세요.

## 캐싱 시간 및 최소 TTL

CloudFront에서 다른 요청을 오리진에 전달하기 전에 객체를 CloudFront 캐시에 보관하는 시간을 제어하려면 다음을 수행합니다.

- `Cache-Control` 또는 `Expires` 헤더 파일을 각 객체에 추가하도록 오리진을 구성합니다.
- CloudFront 캐시 동작에 최소 TTL 값을 지정합니다.
- 기본값인 24시간을 사용합니다.

자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 단원을 참조하세요.

## 클라이언트 IP 주소

최종 사용자가 CloudFront에 요청을 보내고 `X-Forwarded-For` 요청 헤더를 포함하지 않는 경우, CloudFront는 TCP 연결에서 최종 사용자의 IP 주소를 가져오고 IP 주소를 포함하는 `X-Forwarded-For` 헤더를 추가하고 오리진에 요청을 전달합니다. 예를 들어, CloudFront가 TCP 연결에서 IP 주소 `192.0.2.2`를 가져오면 오리진에 다음 헤더를 전달합니다.

```
X-Forwarded-For: 192.0.2.2
```

최종 사용자가 CloudFront에 요청을 보내고 `X-Forwarded-For` 요청 헤더를 포함하는 경우, CloudFront는 TCP 연결에서 최종 사용자의 IP 주소를 가져와 `X-Forwarded-For` 헤더의 끝에 첨부하고 오리진에 요청을 전달합니다. 예를 들어, 최종 사용자 요청에 `X-Forwarded-For: 192.0.2.4,192.0.2.3`이 포함되고 CloudFront가 TCP 연결에서 IP 주소 `192.0.2.2`를 가져오면 오리진에 다음 헤더를 전달합니다.

```
X-Forwarded-For: 192.0.2.4,192.0.2.3,192.0.2.2
```

로드 밸런서(Elastic Load Balancing 포함), 웹 애플리케이션 방화벽, 역방향 프록시, 침입 방지 시스템 및 API Gateway와 같은 일부 애플리케이션들은 해당 요청을 전달한 CloudFront 엣지 서버의 IP 주소를 `X-Forwarded-For` 헤더의 끝에 추가합니다. 예를 들어, CloudFront가 ELB에 전달하는 요청에 `X-Forwarded-For: 192.0.2.2`를 포함시키고 CloudFront 엣지 서버의 IP 주소가 192.0.2.199인 경우, EC2 인스턴스가 수신하는 요청에는 다음과 같은 헤더가 들어 있습니다.

```
X-Forwarded-For: 192.0.2.2,192.0.2.199
```

> **Note**
>
> `X-Forwarded-For` 헤더에는 IPv4 주소(예: 192.0.2.44)와 IPv6 주소(예: 2001:0db8:85a3::8a2e:0370:7334)가 포함됩니다.
> 또한 `X-Forwarded-For` 헤더는 현재 서버(CloudFront) 경로에 있는 모든 노드에서 수정할 수 있습니다. 자세한 내용은 RFC 7239의 섹션 8.1을 참조하세요. CloudFront 엣지 컴퓨팅 함수를 사용하여 헤더를 수정할 수도 있습니다.

## 클라이언트측 SSL 인증

CloudFront는 클라이언트와 서버가 인증서를 사용하여 서로 인증하는 상호 TLS(mTLS) 인증을 지원합니다. mTLS가 구성된 경우 CloudFront는 TLS 핸드셰이크 중에 클라이언트 인증서를 검증하고 선택적으로 CloudFront Functions를 실행하여 사용자 지정 검증 로직을 구현할 수 있습니다.

mTLS가 구성되지 않은 상태에서 오리진이 클라이언트 측 인증서를 요청하는 경우 CloudFront는 요청을 삭제합니다.

mTLS 구성에 대한 자세한 내용은 ??? 섹션을 참조하세요.

CloudFront는 클라이언트측 SSL 인증서와 클라이언트 인증을 지원하지 않습니다. 오리진이 클라이언트측 인증서를 요청하는 경우 CloudFront에서는 이 요청을 삭제합니다.

## 압축

자세한 내용은 압축된 파일 제공 섹션을 참조하세요.

## 조건부 요청

CloudFront는 엣지 캐시에서 만료된 객체에 대한 요청을 받으면 이 요청을 오리진으로 전달하여 최신 버전의 객체를 가져오거나 CloudFront 엣지 캐시에 이미 최신 버전이 있는지 오리진의 확인을 받습니다. 대개 오리진에서 마지막에 CloudFront에 객체를 보낼 때는 `ETag` 값, `LastModified` 값 또는 두 값 모두를 응답에 포함하여 보냅니다. CloudFront에서 오리진에 전달하는 새 요청에서 CloudFront는 다음 중 하나 또는 둘 모두를 추가합니다.

- 만료된 버전의 객체에 대한 `If-Match` 값을 포함하는 `If-None-Match` 또는 `ETag` 헤더
- 만료된 버전의 객체에 대한 `If-Modified-Since` 값을 포함하는 `LastModified` 헤더

오리진에서는 이 정보를 사용하여 객체가 업데이트되는지와, 그에 따라 전체 객체를 CloudFront에 반환할지 아니면 HTTP 304 상태 코드만 반환할지(수정되지 않음) 여부를 결정합니다.

> **Note**
>
> CloudFront가 쿠키(전체 또는 일부)를 전달하도록 구성된 경우 `If-Modified-Since` 및 `If-None-Match` 조건부 요청이 지원되지 않습니다.
> 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 섹션을 참조하세요.

## Cookies

쿠키를 오리진에 전달하도록 CloudFront를 구성할 수 있습니다. 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하세요.

## 교차 오리진 리소스 공유(CORS)

CloudFront에서 CORS(Cross-Origin Resource Sharing) 설정을 준수하도록 하려는 경우 `Origin` 헤더를 오리진으로 전달하도록 CloudFront를 구성합니다. 자세한 내용은 요청 헤더 기반의 콘텐츠 캐싱 섹션을 참조하세요.

## 암호화

최종 사용자는 HTTPS를 사용하여 CloudFront에 요청을 전송하도록 하고 CloudFront에는 최종 사용자에 의해 사용된 프로토콜을 사용하여 사용자 지정 오리진에 요청을 전달하도록 할 수 있습니다. 자세한 내용은 다음 배포 설정을 참조하십시오.

- 뷰어 프로토콜 정책
- 프로토콜(사용자 지정 오리진만 해당)

CloudFront에서는 SSLv3, TLSv1.0, TLSv1.1, TLSv1.2 및 TLSv1.3 프로토콜을 사용하여 HTTPS 요청을 오리진 서버에 전달합니다. 사용자 지정 오리진의 경우, 오리진과 통신할 때 CloudFront에서 사용하려는 SSL 프로토콜을 선택할 수 있습니다.

- CloudFront 콘솔을 사용하는 경우, 오리진 SSL 프로토콜 확인란을 사용하여 프로토콜을 선택합니다. 자세한 내용은 배포 생성 단원을 참조하세요.
- CloudFront API를 사용하는 경우, `OriginSslProtocols` 요소를 사용하여 프로토콜을 지정합니다. 자세한 내용은 Amazon CloudFront API 참조의 OriginSslProtocols 및 DistributionConfig를 참조하십시오.

오리진이 Amazon S3 버킷인 경우 CloudFront는 기본적으로 TLSv1.3을 사용합니다.

> **Important**
>
> 다른 버전의 SSL 및 TLS는 지원되지 않습니다.

CloudFront에서 HTTPS를 사용하는 방법은 CloudFront에서 HTTPS 사용 단원을 참조하십시오. 최종 사용자와 CloudFront 간의 HTTPS 통신 및 CloudFront와 오리진 간의 HTTPS 통신에 대해 CloudFront가 지원하는 암호 목록은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호 단원을 참조하십시오.

## 본문이 포함되는 GET 요청

최종 사용자 GET 요청에 본문이 포함되는 경우, CloudFront에서는 HTTP 상태 코드 403(금지됨)을 최종 사용자에게 반환합니다.

## HTTP 메소드

지원되는 모든 HTTP 메서드를 처리하도록 CloudFront를 구성하는 경우, CloudFront에서는 최종 사용자의 다음 요청을 수락하고 이를 사용자 지정 오리진에 전달합니다.

- DELETE
- GET
- HEAD
- OPTIONS
- PATCH
- POST
- PUT

CloudFront에서는 GET 및 HEAD 요청에 대한 응답을 항상 캐싱합니다. 또한 OPTIONS 요청에 대한 응답을 캐싱하도록 CloudFront를 구성할 수도 있습니다. CloudFront에서는 다른 메서드를 사용하는 요청에 대한 응답을 캐싱하지 않습니다.

사용자 지정 오리진에서 이러한 메소드를 처리할지 여부를 구성하는 방법에 대한 자세한 내용은 오리진에 대한 설명서를 참조하십시오.

> **Important**
>
> CloudFront에서 지원되는 모든 HTTP 메서드를 허용하고 오리진에 전달하도록 CloudFront를 구성한 경우, 모든 메서드를 처리하도록 오리진 서버를 구성합니다. 예를 들어, POST를 사용할 의도로 이러한 메서드를 허용 및 전달하도록 CloudFront를 구성한 경우, 최종 사용자에 의해 삭제되는 것을 원치 않는 리소스를 이들이 삭제할 수 없도록 DELETE 요청을 적절히 처리하여 오리진 서버를 구성해야 합니다. 자세한 내용은 HTTP 서버에 대한 설명서를 참조하십시오.

## HTTP 요청 헤더 및 CloudFront 동작(사용자 지정 및 Amazon S3 오리진)

다음 테이블에는 사용자 지정 오리진과 Amazon S3 오리진 모두에 전달할 수 있는 HTTP 요청 헤더가 나열되어 있습니다(예외 참조). 각 헤더의 테이블에는 다음에 대한 정보가 포함되어 있습니다.

- 헤더를 오리진에 전달하도록 CloudFront를 구성하지 않은 경우의 CloudFront 동작으로, CloudFront에서 헤더 값에 따라 객체를 캐싱하는 원인이 됩니다.
- 해당 헤더에 대해 헤더 값에 따라 객체를 캐싱하도록 CloudFront를 구성할 수 있는지 여부.

  `Date` 및 `User-Agent` 헤더의 값에 따라 객체를 캐싱하도록 CloudFront를 구성할 수 있지만, 권장되지는 않습니다. 이러한 헤더는 여러 가지 값을 가질 수 있으며, 헤더의 값에 따른 캐싱으로 인해 CloudFront에서 오리진에 전달되는 요청이 눈에 띄게 증가할 수 있습니다.

헤더 값에 따른 캐싱에 대한 자세한 내용은 요청 헤더 기반의 콘텐츠 캐싱 단원을 참조하십시오.

| 헤더 | 헤더 값에 따라 캐싱하도록 CloudFront를 구성하지 않은 경우의 동작 | 헤더 값에 따른 캐싱이 지원되는지 여부 |
| --- | --- | --- |
| 기타 정의 헤더 | 레거지 캐시 설정 - CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `Accept` | CloudFront에서 헤더를 제거합니다. | 예 |
| `Accept-Charset` | CloudFront에서 헤더를 제거합니다. | 예 |
| `Accept-Encoding` | 값에 `gzip` 또는 `br`이 포함된 경우 CloudFront는 정규화된 `Accept-Encoding` 헤더를 오리진으로 전달합니다.<br><br>자세한 내용은 압축 지원 및 압축된 파일 제공 단원을 참조하십시오. | 예 |
| `Accept-Language` | CloudFront에서 헤더를 제거합니다. | 예 |

| 헤더 | 헤더 값에 따라 캐싱하도록 CloudFront를 구성하지 않은 경우의 동작 | 헤더 값에 따른 캐싱이 지원되는지 여부 |
| --- | --- | --- |
| `Authorization` | • `GET` 및 `HEAD` 요청 – CloudFront에서는 요청을 오리진에 전달하기 전에 `Authorization` 헤더 필드를 제거합니다.<br>• `OPTIONS` 요청 – `Authorization` 요청에 대한 응답을 캐싱하도록 CloudFront를 구성한 경우, CloudFront에서는 요청을 오리진에 전달하기 전에 `OPTIONS` 헤더 필드를 제거합니다.<br>OPTIONS 요청에 대한 응답을 캐싱하도록 CloudFront를 구성하지 않은 경우, CloudFront에서는 `Authorization` 헤더 필드를 오리진에 전달합니다.<br>• `DELETE`, `PATCH`, `POST`, `PUT` 요청 – CloudFront에서는 요청을 오리진에 전달하기 전에 헤더 필드를 제거하지 않습니다. | 예 |
| `Cache-Control` | CloudFront에서 오리진에 헤더를 전달합니다. | 아니요 |
| `CloudFront-Forwarded-Proto` | 요청을 오리진에 전달하기 전에는 CloudFront에서 헤더를 추가하지 않습니다.<br>자세한 내용은 요청의 프로토콜을 기반으로 캐싱하도록 구성 단원을 참조하세요. | 예 |

| 헤더 | 헤더 값에 따라 캐싱하도록 CloudFront를 구성하지 않은 경우의 동작 | 헤더 값에 따른 캐싱이 지원되는지 여부 |
| --- | --- | --- |
| `CloudFront-Is-Desktop-Viewer` | 요청을 오리진에 전달하기 전에는 CloudFront에서 헤더를 추가하지 않습니다.<br><br>자세한 내용은 디바이스 유형을 기반으로 캐싱하도록 구성 단원을 참조하세요. | 예 |
| `CloudFront-Is-Mobile-Viewer` | 요청을 오리진에 전달하기 전에는 CloudFront에서 헤더를 추가하지 않습니다.<br><br>자세한 내용은 디바이스 유형을 기반으로 캐싱하도록 구성 단원을 참조하세요. | 예 |
| `CloudFront-Is-Tablet-Viewer` | 요청을 오리진에 전달하기 전에는 CloudFront에서 헤더를 추가하지 않습니다.<br><br>자세한 내용은 디바이스 유형을 기반으로 캐싱하도록 구성 단원을 참조하세요. | 예 |
| `CloudFront-Viewer-Country` | 요청을 오리진에 전달하기 전에는 CloudFront에서 헤더를 추가하지 않습니다. | 예 |
| `Connection` | CloudFront 에서는 요청을 오리진에 전달하기 전에 `Connection: Keep-Alive` 로 이 헤더를 대체합니다. | 아니요 |
| `Content-Length` | CloudFront에서 오리진에 헤더를 전달합니다. | 아니요 |
| `Content-MD5` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |

| 헤더 | 헤더 값에 따라 캐싱하도록 CloudFront를 구성하지 않은 경우의 동작 | 헤더 값에 따른 캐싱이 지원되는지 여부 |
|---|---|---|
| `Content-Type` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `Cookie` | 쿠키를 전달하도록 CloudFront를 구성한 경우, `Cookie` 헤더 필드가 오리진에 전달됩니다. 그렇지 않으면 CloudFront가 `Cookie` 헤더 필드를 제거합니다. 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하세요. | 아니요 |
| `Date` | CloudFront에서 오리진에 헤더를 전달합니다. | 지원되나 권장되지 않음 |
| `Expect` | CloudFront에서 헤더를 제거합니다. | 예 |
| `From` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `Host` | CloudFront에서 값을 요청된 객체와 연결된 오리진의 도메인 이름에 설정합니다.<br><br>Amazon S3 또는 MediaStore 오리진에 대한 호스트 헤더를 기반으로 캐싱을 할 수 없습니다. | 예(사용자 지정)<br><br>아니요 (S3 및 MediaStore) |
| `If-Match` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `If-Modified-Since` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `If-None-Match` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |

| 헤더 | 헤더 값에 따라 캐싱하도록 CloudFront를 구성하지 않은 경우의 동작 | 헤더 값에 따른 캐싱이 지원되는지 여부 |
| --- | --- | --- |
| `If-Range` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `If-Unmodified-Since` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `Max-Forwards` | CloudFront에서 오리진에 헤더를 전달합니다. | 아니요 |
| `Origin` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `Pragma` | CloudFront에서 오리진에 헤더를 전달합니다. | 아니요 |
| `Proxy-Authenticate` | CloudFront에서 헤더를 제거합니다. | 아니요 |
| `Proxy-Authorization` | CloudFront에서 헤더를 제거합니다. | 아니요 |
| `Proxy-Connection` | CloudFront에서 헤더를 제거합니다. | 아니요 |
| `Range` | CloudFront에서 오리진에 헤더를 전달합니다. 자세한 내용은 CloudFront에서 객체에 대한 부분적인 요청을 처리하는 방법(범위 GET) 단원을 참조하세요. | 기본적으로 지원됨 |
| `Referer` | CloudFront에서 헤더를 제거합니다. | 예 |
| `Request-Range` | CloudFront에서 오리진에 헤더를 전달합니다. | 아니요 |

| 헤더 | 헤더 값에 따라 캐싱하도록 CloudFront를 구성하지 않은 경우의 동작 | 헤더 값에 따른 캐싱이 지원되는지 여부 |
| --- | --- | --- |
| `TE` | CloudFront에서 헤더를 제거합니다. | 아니요 |
| `Trailer` | CloudFront에서 헤더를 제거합니다. | 아니요 |
| `Transfer-Encoding` | CloudFront에서 오리진에 헤더를 전달합니다. | 아니요 |
| `Upgrade` | CloudFront에서는 WebSocket 연결이 설정되지 않았을 경우 헤더를 제거합니다. | 아니요 (WebSocket 연결 제외) |
| `User-Agent` | CloudFront에서 이 헤더의 값을 `Amazon CloudFront` 로 대체합니다. CloudFront에서 사용자가 이용 중인 디바이스에 따라 콘텐츠를 캐싱하도록 하려는 경우, 디바이스 유형을 기반으로 캐싱하도록 구성 단원을 참조하십시오. | 지원되나 권장되지 않음 |
| `Via` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `Warning` | CloudFront에서 오리진에 헤더를 전달합니다. | 예 |
| `X-Amz-Cf-Id` | CloudFront에서는 요청을 오리진에 전달하기 전에 최종 사용자 요청에 헤더를 추가합니다. 헤더 값에는 요청을 고유하게 식별하는 암호화된 문자열이 포함됩니다. | 아니요 |
| `X-Edge-*` | CloudFront는 모든 `X-Edge-*` 헤더를 제거합니다. | 아니요 |

| 헤더 | 헤더 값에 따라 캐싱하도록 CloudFront를 구성하지 않은 경우의 동작 | 헤더 값에 따른 캐싱이 지원되는지 여부 |
| --- | --- | --- |
| `X-Forwarded-For` | CloudFront에서 오리진에 헤더를 전달합니다. 자세한 내용은 클라이언트 IP 주소 단원을 참조하세요. | 예 |
| `X-Forwarded-Proto` | CloudFront에서 헤더를 제거합니다. | 아니요 |
| `X-HTTP-Method-Override` | CloudFront에서 헤더를 제거합니다. | 예 |
| `X-Real-IP` | CloudFront에서 헤더를 제거합니다. | 아니요 |

## HTTP 버전

CloudFront에서는 HTTP/1.1을 사용하여 사용자 지정 오리진에 요청을 전달합니다.

## 요청의 최대 길이 및 최대 URL 길이

경로, 쿼리 문자열(있는 경우), 헤더를 모두 포함한 요청의 최대 길이는 20,480바이트입니다.

CloudFront에서는 이 요청으로부터 URL을 구성합니다. 이 URL의 최대 길이는 8,192바이트입니다.

요청 또는 URL이 이 최대값을 초과할 경우 CloudFront에서는 HTTP 요청 코드 413(요청 개체가 너무 큼)을 반환한 후 최종 사용자와의TCP 연결을 종료합니다.

## OCSP 스테이플링

최종 사용자가 객체에 대한 HTTPS 요청을 제출할 때 CloudFront 또는 최종 사용자는 CA(인증 기관)을 통해 도메인의 SSL 인증서가 해지되지 않았는지 확인해야 합니다. OCSP 스테이플링은 CloudFront에서 인증서의 유효성을 검사하고 CA로부터 응답을 캐싱할 수 있도록 함으로써 클라이언트가 CA를 통해 직접 인증서의 유효성을 검사하지 않아도 되므로 인증서 유효성 검사 속도가 향상됩니다.

OSCP 스테이플링의 성능 개선은 CloudFront에서 같은 도메인 내의 객체에 대해 많은 HTTPS 요청을 받은 경우 더욱 확연히 드러납니다. CloudFront 엣지 로케이션의 각 서버는 개별적인 유효성 검사 요청

을 제출해야 합니다. CloudFront에서 같은 도메인에 대해 다수의 HTTPS 요청을 받은 경우, 엣지 로케이션의 각 서버에서는 곧 CA로부터 SSL 핸드쉐이크의 패킷에 "스테이플"할 수 있다는 응답을 받습니다. 최종 사용자가 인증서가 유효하다는 조건을 충족하면 CloudFront에서는 요청된 객체를 제공할 수 있습니다. 배포가 CloudFront 엣지 로케이션에서 많은 양의 트래픽을 받지 않는 경우, 새로운 요청은 아직 CA를 통해 인증서의 유효성을 검사하지 않은 서버로 리디렉션될 가능성이 높습니다. 그러한 경우, 최종 사용자는 유효성 검사 단계를 개별적으로 수행하고 CloudFront 서버에서는 객체를 제공합니다. 해당 CloudFront 서버에서는 또한 유효성 검사 요청을 CA에 제출하고, 다음에 동일한 도메인 이름을 포함한 요청을 수신하면 CA로부터 유효성 검사 응답을 받습니다.

## 지속적인 연결

CloudFront가 오리진으로부터 응답을 받는 경우, 그 시간 동안 다른 요청이 도착하면 몇 초 정도 연결을 유지하려고 합니다. 지속적인 연결을 유지하면 TCP 연결 재설정 및 이후 요청에 대한 별도의 TLS 핸드쉐이크 수행에 필요한 시간이 절약됩니다.

지속적 연결의 지속 시간을 구성하는 방법 등 자세한 내용은 연결 유지 제한 시간(사용자 지정 및 VPC 오리진만 해당) 단원의 모든 배포 설정 참조을 참조하십시오.

## 프로토콜

CloudFront에서는 다음 사항을 바탕으로 HTTP 또는 HTTPS 요청을 오리진 서버에 전달합니다.

- 최종 사용자가 CloudFront에 보낸 요청의 프로토콜(HTTP 또는 HTTPS)
- CloudFront 콘솔에서 오리진 프로토콜 정책 필드의 값 또는 CloudFront API를 사용 중인 경우 `OriginProtocolPolicy` 복합 형식의 `DistributionConfig` 요소. CloudFront 콘솔에는 HTTP만 해당, HTTPS만 해당 및 최종 사용자 일치 옵션이 있습니다.

HTTP만 해당 또는 HTTPS만 해당을 지정하는 경우, 최종 사용자 요청의 프로토콜과 관계없이 CloudFront에서는 지정된 프로토콜을 사용하여 오리진 서버로 요청을 전달합니다.

최종 사용자 일치를 지정하는 경우, CloudFront에서는 최종 사용자 요청의 프로토콜을 사용하여 오리진 서버에 요청을 전달합니다. 최종 사용자가 HTTP 및 HTTPS 프로토콜 모두를 사용하여 요청하더라도 CloudFront에서는 한 번만 객체를 캐싱합니다.

> **Important**
>
> CloudFront가 HTTPS 프로토콜을 사용하여 원본으로 요청을 전달하고 오리진 서버가 잘못된 인증서 또는 자체 서명한 인증서를 반환하는 경우, CloudFront에서는 TCP 연결을 끊습니다.

CloudFront 콘솔을 사용하여 배포를 업데이트하는 방법에 대한 자세한 내용은 배포 업데이트 단원을 참조하십시오. CloudFront API를 사용하여 배포를 업데이트하는 방법에 대한 자세한 내용은 Amazon CloudFront API 참조의 UpdateDistribution으로 이동하십시오.

## 쿼리 문자열

CloudFront에서 쿼리 문자열 파라미터를 오리진에 전달할지 여부를 구성할 수 있습니다. 자세한 내용은 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱 단원을 참조하세요.

## 오리진 연결 제한 시간 및 시도 횟수

오리진 연결 제한 시간은 CloudFront가 오리진에 대한 연결을 설정하려고 할 때 기다리는 시간(초)입니다.

오리진 연결 시도는 CloudFront가 오리진에 대한 연결을 시도하는 횟수입니다.

이러한 설정은 모두 CloudFront가 보조 오리진에 대해 장애 조치하거나(오리진 그룹의 경우) 뷰어에 대한 오류 응답을 반환하기 전에 오리진에 대한 연결을 시도하는 시간을 결정합니다. 기본적으로 CloudFront는 보조 오리진에 연결하거나 오류 응답을 반환하기 전에 30초(각각 10초 동안 3회 시도)까지 기다립니다. 더 짧은 연결 제한 시간, 더 적은 시도 횟수 중 하나 또는 둘 다를 지정하여 이 시간을 줄일 수 있습니다.

자세한 내용은 오리진 제한 시간 및 시도 횟수 제어 단원을 참조하세요.

## 오리진 응답 제한 시간

오리진 응답 제한 시간(오리진 읽기 제한 시간 또는 오리진 요청 제한 시간이라고도 함)은 다음 두 값에 모두 적용됩니다.

- CloudFront가 오리진에 요청을 전달한 후 응답을 기다리는 시간(초).
- CloudFront가 오리진으로부터 응답 패킷을 수신한 후 다음 패킷을 수신할 때까지 대기하는 시간(초).

CloudFront 동작은 최종 사용자 요청의 HTTP 메서드에 따라 달라집니다.

- `GET` 및 `HEAD` 요청 - 오리진에서 응답 제한 시간 내에 응답하지 않거나 응답이 중지된 경우, CloudFront에서는 연결을 끊습니다. 지정된 오리진 연결 시도 횟수가 1보다 많으면 CloudFront가 다시 연결을 설정하려고 시도합니다. CloudFront는 오리진 연결 시도 설정 값에 따라 최대 3회까지 시도합니다. 오리진이 마지막 시도에 응답하지 않는 경우, CloudFront에서는 동일한 오리진의 콘텐츠에 대해 다른 요청을 받을 때까지 다시 시도하지 않습니다.

- DELETE, OPTIONS, PATCH, PUT 및 POST 요청 - 오리진이 읽기 제한 시간 동안 응답하지 않는 경우, CloudFront는 연결을 끊고 오리진에 다시 연결을 시도하지 않습니다. 필요한 경우 클라이언트는 요청을 다시 제출할 수 있습니다.

오리진 응답 제한 시간을 구성하는 방법 등 자세한 내용은 응답 제한 시간 단원을 참조하십시오.

## 동일 객체에 대한 동시 요청(요청 축소)

CloudFront 엣지 로케이션에서 객체에 대한 요청을 받을 때 객체가 캐시에 있지 않거나 캐시된 객체가 만료된 경우, CloudFront에서는 즉시 요청을 오리진으로 보냅니다. 하지만 동일 객체에 대한 동시 요청이 있는 경우, 즉 CloudFront가 첫 번째 요청에 대한 응답을 수신하기 전에 동일 객체에 대한 추가 요청(동일 캐시 키 포함)이 엣지 로케이션에 도착하는 경우 CloudFront는 추가 요청을 오리진에 전달하기 전에 작업을 일시 중지합니다. 이러한 일시 중지는 오리진에 부하가 걸리는 것을 줄여 줍니다. CloudFront는 원래 요청의 응답을 일시 중지된 동안 수신한 모든 요청에 보냅니다. 이를 요청 축소라고 합니다. CloudFront 로그에서, 첫 번째 요청은 `x-edge-result-type` 필드에서 `Miss`로 식별되고 축소된 요청은 `Hit`로 식별됩니다. CloudFront 로그에 대한 자세한 내용은 the section called “CloudFront 및 엣지 함수 로깅” 단원을 참조하세요.

CloudFront는 캐시 키를 공유하는 요청만 축소합니다. 요청 헤더 또는 쿠키 또는 쿼리 문자열에 따라 캐시하도록 CloudFront를 구성하는 등의 이유로 추가 요청이 동일한 캐시 키를 공유하지 않는 경우, CloudFront에서는 고유한 캐시 키가 있는 모든 요청을 오리진에 전달합니다.

모든 요청이 축소되는 것을 방지하려면 캐싱을 방지하는 관리형 캐시 정책 `CachingDisabled`를 사용하면 됩니다. 자세한 내용은 관리형 캐시 정책 사용 섹션을 참조하세요.

특정 객체에 대한 요청 축소를 방지하려면 캐시 동작의 최소 TTL을 0으로 설정하고, 그 후 `Cache-Control: private`, `Cache-Control: no-store`, `Cache-Control: no-cache`, `Cache-Control: max-age=0` 또는 `Cache-Control: s-maxage=0`을 보내도록 오리진을 구성합니다. 이 구성을 사용하면 오리진에 걸리는 부하가 가중되고 CloudFront가 첫 번째 요청에 대한 응답을 기다리는 동안 일시 중지된 동시 요청으로 인한 추가 지연 시간이 발생합니다.

> **Important**
>
> 현재 CloudFront는 캐시 정책, 오리진 요청 정책 또는 레거시 캐시 설정에서 쿠키 전달을 활성화한 경우 요청 축소를 지원하지 않습니다.

### User-Agent 헤더

CloudFront에서 사용자가 콘텐츠를 보기 위해 사용 중인 디바이스에 따라 여러 버전의 객체를 캐싱하도록 하려는 경우, 다음과 같이 하나 이상의 헤더를 사용자 지정 오리진에 전달하도록 CloudFront를 구성하는 것이 좋습니다.

- `CloudFront-Is-Desktop-Viewer`
- `CloudFront-Is-Mobile-Viewer`
- `CloudFront-Is-SmartTV-Viewer`
- `CloudFront-Is-Tablet-Viewer`

`User-Agent` 헤더의 값에 따라 CloudFront에서는 요청을 오리진에 전달하기 전에 이러한 헤더의 값을 `true` 또는 `false`로 설정합니다. 디바이스가 둘 이상의 범주에 해당하는 경우 둘 이상의 값이 `true`일 수 있습니다. 예를 들어 일부 태블릿 디바이스의 경우 CloudFront에서는 `CloudFront-Is-Mobile-Viewer`와 `CloudFront-Is-Tablet-Viewer`를 모두 `true`로 설정할 수 있습니다. 요청 헤더에 따라 캐싱하도록 CloudFront를 구성하는 방법에 대한 자세한 내용은 요청 헤더 기반의 콘텐츠 캐싱 단원을 참조하십시오.

`User-Agent` 헤더의 값에 따라 객체를 캐싱하도록 CloudFront를 구성할 수 있지만, 권장되지는 않습니다. `User-Agent` 헤더는 여러 가지 값을 가질 수 있으며, 그러한 값에 따른 캐싱으로 인해 CloudFront에서 오리진에 전달되는 요청이 눈에 띄게 증가할 수 있습니다.

`User-Agent` 헤더의 값에 따라 객체를 캐싱하도록 CloudFront를 구성하지 않는 경우, CloudFront에서는 오리진에 요청을 전달하기 전에 다음 값으로 `User-Agent` 헤더를 추가합니다.

```
User-Agent = Amazon CloudFront
```

CloudFront에서는 최종 사용자의 요청에 `User-Agent` 헤더가 포함되어 있는지와 관계없이 이 헤더를 추가합니다. 최종 사용자의 요청에 `User-Agent` 헤더가 포함되어 있는 경우, CloudFront에서는 이를 제거합니다.

## CloudFront에서 사용자 지정 오리진 서버의 요청을 처리하는 방법

CloudFront에서 사용자 지정 오리진 서버의 요청을 처리하는 방법을 알아봅니다.

목차

- 100 Continue 응답

- 캐싱
- 취소된 요청
- 콘텐츠 협상
- Cookies
- TCP 연결 끊김
- CloudFront에서 제거하거나 교체하는 HTTP 응답 헤더
- 최대 캐시 가능 파일 크기
- 오리진 사용 불가
- 리디렉션
- Transfer-Encoding 헤더

## `100 Continue` 응답

오리진은 CloudFront에 하나 이상의 100-continue 응답을 전송할 수 없습니다. 첫 번째 100-continue 응답 후에 CloudFront는 HTTP 200 OK 응답을 예상합니다. 오리진이 첫 번째 응답 후 또 다른 100-continue 응답을 전송하면 CloudFront는 오류를 반환합니다.

## 캐싱

- 오리진 서버에서 `Date` 및 `Last-Modified` 헤더 필드에 유효하고 정확한 값을 설정했는지 확인합니다.
- CloudFront에서는 오리진의 응답으로 보통 `Cache-Control: no-cache` 헤더를 신뢰합니다. 예외 사항은 동일 객체에 대한 동시 요청(요청 축소) 단원을 참조하십시오.

## 취소된 요청

객체가 엣지 캐시에 있지 않은 경우 최종 사용자가 CloudFront가 오리진에서 객체를 가져온 뒤 요청된 객체를 제공하기 전에 브라우저 닫기 등으로 세션을 종료하면, CloudFront에서는 엣지 로케이션의 객체를 캐싱하지 않습니다.

## 콘텐츠 협상

오리진이 응답에서 `Vary:*`를 반환하는 경우와 해당 캐시 동작의 Minimum TTL 값이 0인 경우, CloudFront에서는 객체를 캐싱하지만 오리진에 객체의 모든 후속 요청을 전송하여 캐시에 객체의 최신 버전이 포함되어 있음을 확인합니다. CloudFront에는 `If-None-Match` 또는 `If-Modified-`

`Since`와 같은 조건부 헤더가 포함되지 않습니다. 결과적으로 오리진은 모든 요청에 응답하여 객체를 CloudFront에 반환합니다.

오리진이 응답에서 `Vary:*`를 반환하는 경우와 해당 캐시 동작의 Minimum TTL 값이 다른 값인 경우, CloudFront에서는 `Vary`에 설명된 CloudFront에서 제거하거나 교체하는 HTTP 응답 헤더 헤더를 처리합니다.

## Cookies

캐시 동작에 대한 쿠키를 활성화한 경우 오리진에서 객체를 통해 쿠키를 반환하면 CloudFront에서는 객체와 쿠키를 모두 캐싱합니다. 이를 통해 객체에 대한 캐싱 가능성이 줄어듭니다. 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하세요.

## TCP 연결 끊김

오리진이 객체를 CloudFront에 반환하는 동안 CloudFront와 오리진 간의 TCP 연결이 끊어진 경우, CloudFront 동작은 오리진이 응답에 `Content-Length` 헤더를 포함했는지 여부에 따라 달라집니다.

- Content-Length 헤더 – CloudFront에서는 오리진에서 객체를 가져오는 동안 최종 사용자에게 객체를 반환합니다. 그러나 `Content-Length` 헤더의 값이 객체의 크기와 일치하지 않는 경우, CloudFront에서는 객체를 캐싱하지 않습니다.
- Transfer-Encoding: Chunked – CloudFront에서는 오리진에서 객체를 가져오는 동안 최종 사용자에게 객체를 반환합니다. 그러나 조각난 응답이 완전하지 않은 경우, CloudFront에서는 객체를 캐싱하지 않습니다.
- No Content-Length 헤더 – CloudFront에서는 최종 사용자에게 객체를 반환하고 이를 캐싱하지만, 이 객체는 불완전할 수 있습니다. CloudFront에서는 `Content-Length` 헤더 없이는 TCP 연결이 실수로 또는 고의로 끊어졌는지 여부를 판단할 수 없습니다.

`Content-Length` 헤더를 추가하여 CloudFront에서 부분적인 객체를 캐싱하지 못하도록 HTTP 서버를 구성하는 것이 좋습니다.

## CloudFront에서 제거하거나 교체하는 HTTP 응답 헤더

CloudFront는 오리진에서 최종 사용자에게 응답을 전달하기 전에 다음 헤더 필드를 제거하거나 업데이트합니다.

- `Set-Cookie` – 쿠키를 전달하도록 CloudFront를 구성하는 경우, `Set-Cookie` 헤더 필드가 클라이언트에 전달됩니다. 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하세요.

- `Trailer`
- `Transfer-Encoding` – 오리진에서 이 헤더 필드를 반환하는 경우, CloudFront에서는 최종 사용자에게 응답을 반환하기 전에 `chunked`에 값을 설정합니다.
- `Upgrade`
- `Vary` – 다음을 참조하십시오.
  - 디바이스별 헤더 중 하나를 오리진(`CloudFront-Is-Desktop-Viewer`, `CloudFront-Is-Mobile-Viewer`, `CloudFront-Is-SmartTV-Viewer`, `CloudFront-Is-Tablet-Viewer`)에 전달하도록 CloudFront를 구성하고 `Vary:User-Agent`를 CloudFront에 반환하도록 오리진을 구성하는 경우, CloudFront에서는 최종 사용자에게 `Vary:User-Agent`를 반환합니다. 자세한 내용은 디바이스 유형을 기반으로 캐싱하도록 구성 단원을 참조하세요.
  - `Accept-Encoding` 헤더에서 `Cookie` 또는 `Vary`를 포함하도록 오리진을 구성하는 경우, CloudFront는 최종 사용자에 대한 응답에 대한 값을 포함합니다.
  - 헤더를 오리진에 전달하도록 CloudFront를 구성하는 경우와 `Vary` 헤더(예: `Vary:Accept-Charset,Accept-Language`)의 CloudFront에 헤더 이름을 반환하도록 오리진을 구성하는 경우, CloudFront에서는 최종 사용자에게 `Vary` 헤더를 그 값과 함께 반환합니다.
  - CloudFront가 `*` 헤더의 `Vary` 값을 처리하는 방법에 대한 자세한 내용은 콘텐츠 협상 단원을 참조하십시오.
  - `Vary` 헤더에 그 밖의 값을 포함하도록 오리진을 구성하는 경우, CloudFront는 최종 사용자에게 응답을 반환하기 전에 값을 제거합니다.
- `Via` – CloudFront는 최종 사용자에 대한 응답으로 값을 다음과 같이 설정합니다.

  ```
  Via: http-version alphanumeric-string.cloudfront.net (CloudFront)
  ```

  예를 들어, 값은 다음과 같습니다.

  ```
  Via: 1.1 1026589cc7887e7a0dc7827b4example.cloudfront.net (CloudFront)
  ```

## 최대 캐시 가능 파일 크기

CloudFront에서 캐시에 저장하는 응답 본문의 최대 크기는 50GB입니다. 여기에는 `Content-Length` 헤더 값으로 지정하지 않은 조각난 전송 응답이 포함됩니다.

CloudFront를 사용하면 범위 요청을 사용하여 각각 50GB 이하인 부분으로 객체를 요청하여 이보다 큰 객체를 캐시할 수 있습니다. CloudFront는 각 부분이 50GB 이하이기 때문에 이러한 부분을 캐시합니다. 뷰어는 객체의 모든 부분을 검색한 후 원래의 더 큰 객체를 재구성할 수 있습니다. 자세한 내용은 범위 요청을 사용하여 대형 객체 캐시 섹션을 참조하세요.

## 오리진 사용 불가

오리진 서버를 사용할 수 없고 CloudFront에서 엣지 캐시에 있지만 만료된(`Cache-Control max-age` 명령에 지정된 기간이 지났다는 이유 등으로) 객체에 대한 요청을 받은 경우, CloudFront에서는 만료된 버전의 객체를 제공하거나 사용자 지정 오류 페이지를 표시합니다. 사용자 지정 오류 페이지를 구성한 경우 CloudFront 동작에 대한 자세한 내용은 사용자 지정 오류 페이지를 구성했을 때 CloudFront에서 오류를 처리하는 방법 단원을 참조하십시오.

경우에 따라 자주 요청되지는 않는 객체는 제거되고 엣지 캐시에서 더 이상 사용할 수 없게 됩니다. CloudFront에서는 제거된 객체를 제공할 수 없습니다.

## 리디렉션

오리진 서버에 있는 객체의 위치를 변경하는 경우 요청을 새 위치로 리디렉션하도록 웹 서버를 구성할 수 있습니다. 리디렉션을 구성한 뒤 처음으로 뷰어가 객체에 대한 요청을 제출할 때 CloudFront에서는 요청을 오리진에 전송하고 오리진에서는 리디렉션으로 응답합니다(예: `302 Moved Temporarily`). CloudFront에서는 이 리디렉션을 캐싱하고 최종 사용자에게 이를 반환합니다. CloudFront에서는 이 리디렉션을 따라가지 않습니다.

다음 위치 중 하나로 요청을 리디렉션하도록 웹 서버를 구성할 수 있습니다.

- 오리진 서버에 있는 객체의 새 URL입니다. 최종 사용자가 새 URL에 대한 리디렉션을 따라갈 때 최종 사용자는 CloudFront를 우회하고 오리진으로 직행합니다. 따라서 오리진에 있는 객체의 새 URL로 요청을 리디렉션하지 않는 것이 좋습니다.
- 객체에 대한 새 CloudFront URL입니다. 최종 사용자가 새 CloudFront URL을 포함한 요청을 제출할 때, CloudFront는 새 위치에서 객체를 가져오고 이를 엣지 로케이션에서 캐싱하여 최종 사용자에게 반환합니다. 이 객체에 대한 후속 요청은 엣지 로케이션에서 제공됩니다. 이를 통해 최종 사용자의 오리진에서의 객체 요청과 관련된 시간 지연과 로드 발생을 피할 수 있습니다. 그러나 객체에 대한 새로운 요청이 발생할 때마다 CloudFront에 대한 두 개의 요청에 대해 요금이 부과됩니다.

## `Transfer-Encoding` 헤더

CloudFront에서는 `chunked` 헤더에 대해 `Transfer-Encoding` 값만 지원합니다. 오리진에서 `Transfer-Encoding: chunked`를 반환하는 경우, CloudFront에서는 객체를 엣지 로케이션에서 수신한 객체로 클라이언트에 반환하고 후속 요청에 대해 조각난 형식의 객체를 캐싱합니다.

최종 사용자가 `Range GET` 요청을 하고 오리진이 `Transfer-Encoding: chunked`를 반환하는 경우, CloudFront에서는 요청한 범위 대신에 최종 사용자에 전체 객체를 반환합니다.

응답의 콘텐츠 길이를 사전에 결정할 수 없는 경우 chunked 인코딩을 사용하는 것이 좋습니다. 자세한 내용은 TCP 연결 끊김 단원을 참조하세요.

## 오리진 그룹에 대한 요청 및 응답 동작

오리진 그룹에 대한 요청은, 오리진 장애 조치가 있는 경우를 제외하면, 오리진 그룹으로 설정되지 않은 오리진에 대한 요청과 동일하게 작동합니다. 다른 모든 오리진에서와 마찬가지로, CloudFront가 요청을 수신하고 콘텐츠가 이미 엣지 로케이션에 캐싱되는 경우 콘텐츠는 캐시로부터 최종 사용자에게 제공됩니다. 캐시 누락이 있고 오리진이 오리진 그룹인 경우, 최종 사용자의 요청은 오리진 그룹의 기본 오리진으로 전달됩니다.

기본 오리진에 대한 요청 및 응답 동작은 오리진 그룹에 없는 오리진에 대한 경우와 동일합니다. 자세한 내용은 Amazon S3 오리진에 대한 요청 및 응답 동작 및 사용자 지정 오리진에 대한 요청 및 응답 동작 단원을 참조하십시오.

다음은 기본 오리진이 특정 HTTP 상태 코드를 반환할 때의 오리진 장애 조치에 대한 동작에 대한 설명입니다.

- HTTP 2xx 상태 코드(성공): CloudFront가 파일을 캐싱하여 최종 사용자에게 반환합니다.
- HTTP 3xx 상태 코드(리디렉션): CloudFront가 상태 코드를 최종 사용자에게 반환합니다.
- HTTP 4xx 또는 5xx 상태 코드(클라이언트/서버 오류): 반환된 상태 코드가 장애 조치에 대해 구성된 경우, CloudFront가 오리진 그룹의 보조 오리진에 동일한 요청을 전송합니다.
- HTTP 4xx 또는 5xx 상태 코드(클라이언트/서버 오류): 반환된 상태 코드가 장애 조치에 대하여 구성된 경우, CloudFront가 오류를 최종 사용자에게 반환합니다.

CloudFront는 최종 사용자 요청의 HTTP 메서드가 `GET`, `HEAD` 또는 `OPTIONS`인 경우에만 보조 오리진으로 장애 조치합니다. 최종 사용자가 다른 HTTP 메서드(예: `POST`, `PUT` 등)를 보내면 CloudFront는 장애 조치를 수행하지 않습니다.

CloudFront에서 요청을 보조 오리진에 전송하는 경우, 응답 동작은 오리진 그룹에 없는 CloudFront 오리진에 대한 동작과 동일합니다.

오리진 그룹에 대한 자세한 내용은 CloudFront 오리진 장애 조치를 통한 고가용성 최적화 단원을 참조하십시오.

# 사용자 지정 헤더를 오리진 요청에 추가

오리진으로 보내는 요청에 사용자 지정 헤더를 추가하도록 CloudFront를 구성할 수 있습니다. 사용자 지정 헤더를 사용하면 일반적인 뷰어 요청을 통해 얻을 수 없는 정보를 오리진에서 보내고 수집할 수 있습니다. 각 오리진에 대해 이러한 헤더를 사용자 지정할 수도 있습니다. CloudFront는 사용자 지정 오리진 및 Amazon S3 오리진에 대해 사용자 지정 헤더를 지원합니다.

목차


- 사용 사례
- 사용자 지정 헤더를 오리진 요청에 추가하도록 CloudFront 구성
- CloudFront에서 오리진 요청에 추가할 수 없는 사용자 지정 헤더
- Authorization 헤더를 전달하도록 CloudFront 구성


## 사용 사례

다음 예와 같이 사용자 지정 헤더를 사용할 수 있습니다.

CloudFront에서 요청 식별

오리진이 CloudFront에서 수신한 요청을 식별할 수 있습니다. 사용자가 CloudFront를 우회하는지 알고 싶은 경우나 2개 이상의 CDN을 사용하고 있는데 어떤 요청이 각 CDN에서 오는지에 대한 정보를 원하는 경우 유용합니다.

> **Note**
>
> Amazon S3 오리진을 사용하고 Amazon S3 서버 액세스 로깅을 활성화하는 경우, 로그에는 헤더 정보가 포함되지 않습니다.

어떤 요청이 특정 배포에서 오는지 확인

동일한 오리진을 사용하도록 둘 이상의 CloudFront 배포를 구성하는 경우, 각 배포에 각각 다른 사용자 지정 헤더를 추가할 수 있습니다. 그러면 오리진의 로그를 사용하여 어떤 요청이 어떤 CloudFront 배포에서 왔는지 확인할 수 있습니다.

Cross-Origin 리소스 공유(CORS) 활성화

일부 최종 사용자가 Cross-Origin 리소스 공유(CORS)를 지원하지 않는 경우, 오리진으로 보내는 요청에 `Origin` 헤더를 항상 추가하도록 CloudFront를 구성할 수 있습니다. 그러면 모든 요청에

대해 `Access-Control-Allow-Origin` 헤더를 반환하도록 오리진을 구성할 수 있습니다. 또한 CORS 설정을 준수하도록 CloudFront를 구성해야 합니다.

콘텐츠에 대한 액세스 제어

사용자 지정 헤더를 사용하여 콘텐츠에 대한 액세스를 제어할 수 있습니다. CloudFront에 의해 추가된 사용자 지정 헤더가 포함된 경우에만 요청에 응답하도록 오리진을 구성하여 사용자가 CloudFront를 우회하는 것과 오리진에서 직접 콘텐츠에 액세스하는 것을 방지할 수 있습니다. 자세한 내용은 사용자 지정 오리진의 파일에 대한 액세스 제한 단원을 참조하십시오.

## 사용자 지정 헤더를 오리진 요청에 추가하도록 CloudFront 구성

오리진으로 보내는 요청에 사용자 지정 헤더를 추가하도록 배포를 구성하려면 다음 방법 중 하나를 사용하여 오리진의 구성을 업데이트합니다.

- CloudFront 콘솔 – 배포를 생성 또는 업데이트할 경우, 사용자 지정 헤더 추가 설정에 헤더 이름과 값을 지정합니다. 자세한 내용은 사용자 지정 헤더 추가 단원을 참조하십시오.
- CloudFront API – 사용자 지정 헤더를 추가하려는 각 오리진에 대해 `Origin` 필드 내부 `CustomHeaders`에 헤더 이름과 값을 지정합니다. 자세한 내용은 Amazon CloudFront API 참조의 CreateDistribution 또는 UpdateDistribution를 참조하세요.

지정하는 헤더 이름과 값이 아직 최종 사용자 요청에 있지 않은 경우, CloudFront는 이를 오리진 요청에 추가합니다. 헤더가 있는 경우, CloudFront는 오리진에 요청을 전달하기 전에 헤더 값을 덮어씁니다.

오리진 사용자 지정 헤더에 적용되는 할당량에 대한 자세한 내용은 헤더에 대한 할당량 섹션을 참조하세요.

## CloudFront에서 오리진 요청에 추가할 수 없는 사용자 지정 헤더

오리진으로 보내는 요청에 다음 헤더 중 하나라도 추가하도록 CloudFront를 구성할 수 없습니다.

- `Cache-Control`
- `Connection`
- `Content-Length`
- `Cookie`
- `Host`
- `If-Match`

- `If-Modified-Since`
- `If-None-Match`
- `If-Range`
- `If-Unmodified-Since`
- `Max-Forwards`
- `Pragma`
- `Proxy-Authenticate`
- `Proxy-Authorization`
- `Proxy-Connection`
- `Range`
- `Request-Range`
- `TE`
- `Trailer`
- `Transfer-Encoding`
- `Upgrade`
- `Via`
- `X-Amz-`로 시작되는 헤더
- `X-Edge-`로 시작되는 헤더
- `X-Real-Ip`

## `Authorization` 헤더를 전달하도록 CloudFront 구성

CloudFront에서 최종 사용자 요청을 오리진에 전달할 때 `Authorization` 헤더를 포함한 일부 최종 사용자 헤더는 기본적으로 제거됩니다. 오리진 요청의 `Authorization` 헤더가 오리진에 항상 수신되도록 하려면 다음 옵션을 선택할 수 있습니다.

- 캐시 정책을 사용하여 캐시 키에 `Authorization` 헤더를 추가합니다. 캐시 키의 모든 헤더가 오리진 요청에 자동으로 포함됩니다. 자세한 내용은 정책으로 캐시 키 제어 단원을 참조하십시오.
- 모든 최종 사용자 헤더를 오리진으로 전달하는 오리진 요청 정책을 사용합니다. 오리진 요청 정책에서 `Authorization` 헤더를 개별적으로 전달할 수는 없지만 모든 최종 사용자 헤더를 전달하면 CloudFront에서 최종 사용자 요청에 `Authorization` 헤더를 포함합니다. CloudFront는 이 사용 사례를 위해 Managed-AllViewer라는 관리형 오리진 요청 정책을 제공합니다. 자세한 내용은 관리형 오리진 요청 정책 사용 단원을 참조하십시오.

# CloudFront에서 객체에 대한 부분적인 요청을 처리하는 방법(범위 GET)

대용량 객체의 경우 뷰어(웹 브라우저 또는 기타 클라이언트)는 여러 GET 요청을 하고 Range 요청 헤더를 사용하여 객체를 더 작은 부분으로 다운로드할 수 있습니다. Range GET 요청이라고도 하는 이러한 바이트 범위의 요청을 통해 부분 다운로드의 효율을 높이고 부분적으로 실패한 전송을 쉽게 복구할 수 있습니다.

CloudFront에서 Range GET 요청을 받은 경우, 요청을 받은 엣지 로케이션에서 캐시를 확인합니다. 엣지 로케이션의 캐시에 이미 전체 객체 또는 객체의 요청된 부분이 포함되어 있는 경우, CloudFront에서는 캐시로부터 요청된 범위를 즉시 제공합니다.

캐시에 요청된 범위가 포함되어 있지 않은 경우, CloudFront에서는 요청을 원본에 전달합니다. (성능 최적화를 위해 CloudFront에서는 클라이언트가 Range GET에서 요청한 것보다 더 큰 범위를 요청할 수 있습니다.) 오리진에서 Range GET 요청을 지원하는지 여부에 따라 다음 상황이 달라집니다.

- 오리진에서 **Range GET** 요청을 지원하는 경우 - 요청된 범위를 반환합니다. CloudFront에서는 요청된 범위를 제공하고 이를 이후 요청을 위해 캐싱합니다. (Amazon S3는 많은 HTTP 서버와 마찬가지로 Range GET 요청을 지원합니다.)
- 오리진에서 **Range GET** 요청을 지원하지 않는 경우 - 전체 객체를 반환합니다. CloudFront에서는 전체 객체를 전송하고 향후 요청을 위해 캐싱하여 현재 요청을 처리합니다. CloudFront에서 엣지 캐시에 전체 객체를 캐싱한 후에는 요청된 범위를 제공하여 새 Range GET 요청에 응답합니다.

두 경우 모두 오리진에서 첫 번째 바이트가 도착하면 CloudFront에서는 요청된 범위 또는 객체를 최종 사용자에게 제공하기 시작합니다.

> **Note**
>
> 최종 사용자가 Range GET 요청을 하고 오리진이 `Transfer-Encoding: chunked`를 반환하는 경우, CloudFront에서는 요청한 범위 대신에 최종 사용자에 전체 객체를 반환합니다.

CloudFront에서는 일반적으로 Range 헤더에 대한 RFC 사양을 준수합니다. 그러나 Range 헤더가 다음 요구 사항을 준수하지 않는 경우, CloudFront에서는 지정된 범위와 함께 상태 코드 200을 반환하는 대신 전체 객체와 함께 HTTP 상태 코드 206을 반환합니다.

- 범위는 오름차순으로 나열되어야 합니다. 예를 들어, `100-200,300-400`은 유효하지만 `300-400,100-200`은 유효하지 않습니다.
- 범위는 중복되어서는 안 됩니다. 예를 들어, `100-200,150-250`은 유효하지 않습니다.
- 모든 범위 사양은 유효해야 합니다. 예를 들어, 범위의 일부로 음수 값을 지정할 수 없습니다.

Range 요청 헤더에 대한 자세한 내용은 RFC 7233의 요청 범위 또는 MDN 웹 문서의 범위를 참조하십시오.

## 범위 요청을 사용하여 대형 객체 캐시

캐싱이 활성화되면 CloudFront는 50GB보다 큰 객체를 검색하거나 캐시하지 않습니다. 원본이 객체가 이보다 크다는 것을 나타내는 경우(`Content-Length` 응답 헤더에서), CloudFront는 원본에 대한 연결을 닫고 뷰어에게 오류를 반환합니다. (캐싱을 비활성화하면 CloudFront는 원본에서 이보다 큰 객체를 검색하여 뷰어에 전달할 수 있습니다. 그러나 CloudFront는 객체를 캐시하지 않습니다.)

그러나 범위 요청의 경우 CloudFront를 사용하여 최대 캐시 가능 파일 크기보다 큰 객체를 캐시할 수 있습니다.

Example예제

1. 예를 들어 100GB 객체가 있는 오리진을 가정해 보겠습니다. 캐싱이 활성화되면 CloudFront는 이렇게 큰 객체를 검색하거나 캐시하지 않습니다. 그러나 뷰어는 이 객체를 50GB보다 작은 각 부분으로 검색하기 위해 여러 범위 요청을 보낼 수 있습니다.
2. 뷰어는 첫 번째 부분을 검색하기 위해 헤더 `Range: bytes=0-21474836480`가 포함된 요청, 다음 부분을 검색하기 위해 헤더 `Range: bytes=21474836481-42949672960`가 포함된 다른 요청을 전송하는 등의 방식으로 20GB 부분의 객체를 요청할 수 있습니다.
3. 뷰어가 모든 부분을 받으면 이를 결합하여 원래 100GB 객체를 구성할 수 있습니다.
4. 이 경우 CloudFront는 객체의 20GB 부분을 캐시하고 캐시에서 동일한 부분에 대한 후속 요청에 응답할 수 있습니다.

압축된 객체에 대한 범위 요청의 경우 바이트 범위 요청은 객체의 원래 크기가 아닌 압축된 크기를 기반으로 합니다. 압축에 대한 자세한 내용은 압축된 파일 제공 섹션을 참조하시기 바랍니다.

# CloudFront에서 오리진의 HTTP 3xx 상태 코드를 처리하는 방법

CloudFront가 Amazon S3 버킷 또는 사용자 지정 오리진 서버에서 객체를 요청하는 경우 오리진에서 HTTP S3 상태 코드를 반환하는 경우가 있습니다. 이 메시지는 일반적으로 다음 중 하나를 나타냅니다.

- 객체의 URL이 변경되었습니다(예: 상태 코드 301, 302, 307 또는 308).
- 마지막으로 CloudFront가 요청한 이후 개체가 변경되지 않았습니다(상태 코드 304).

CloudFront는 CloudFront 배포의 설정 및 응답의 헤더에 따라 3xx 응답을 캐싱합니다. CloudFront는 원본의 응답에 `Cache-Control` 헤더를 포함하는 경우에만 307 및 308 응답을 캐시합니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 섹션을 참조하세요.

오리진에서 리디렉션 상태 코드(예: 301 또는 307)를 반환하는 경우 CloudFront는 리디렉션을 따르지 않습니다. CloudFront는 최종 사용자에게 301 또는 307 응답을 전달하며, 최종 사용자는 새 요청을 전송하여 리디렉션을 따를 수 있습니다.

# CloudFront에서 오리진의 HTTP 4xx 및 5xx 상태 코드를 처리하는 방법

CloudFront가 Amazon S3 버킷 또는 사용자 지정 오리진 서버에서 객체를 요청할 때, 오리진에서 오류가 발생했음을 나타내는 HTTP 4xx 또는 5xx 상태 코드를 반환하는 경우가 있습니다. CloudFront 동작은 다음에 따라 달라집니다.

- 사용자 지정 오류 페이지를 구성했는지 여부
- CloudFront에서 오리진으로부터 오류 응답을 캐싱하려는 시간(오류 캐싱 최소 TTL)을 구성했는지 여부
- 상태 코드
- 5xx 상태 코드의 경우 요청된 객체가 현재 CloudFront 엣지 캐시에 있는지 여부
- 일부 4xx 상태 코드의 경우 오리진에서 `Cache-Control max-age` 또는 `Cache-Control s-maxage` 헤더를 반환하는지 여부

CloudFront에서는 `GET` 및 `HEAD` 요청에 대한 응답을 항상 캐싱합니다. 또한 `OPTIONS` 요청에 대한 응답을 캐싱하도록 CloudFront를 구성할 수도 있습니다. CloudFront에서는 다른 메서드를 사용하는 요청에 대한 응답을 캐싱하지 않습니다.

오리진이 응답하지 않는 경우 오리진에 대한 CloudFront 요청은 시간 초과됩니다. 이 상태는 오리진이 해당 오류로 응답하지 않았더라도 오리진의 HTTP 5xx 오류로 간주됩니다. 해당 시나리오에서 CloudFront는 캐싱된 콘텐츠를 계속 제공합니다. 자세한 내용은 오리진 사용 불가 단원을 참조하세요.

로깅을 활성화한 경우 CloudFront에서는 HTTP 상태 코드에 관계없이 결과를 로그에 작성합니다.

CloudFront에서 반환되는 오류 메시지에 관련된 기능 및 옵션에 대한 자세한 내용은 다음을 참조하십시오.

- CloudFront 콘솔에서 사용자 지정 오류 페이지를 설정하는 방법에 대한 자세한 내용은 사용자 지정 오류 페이지 및 오류 캐싱 단원을 참조하십시오.
- CloudFront 콘솔의 오류 캐싱 최소 TTL에 관한 자세한 내용은 오류 캐싱 최소 TTL(초) 단원을 참조하십시오.
- CloudFront에서 캐싱하는 HTTP 상태 코드의 목록은 CloudFront가 캐싱하는 HTTP 4xx 및 5xx 상태 코드 단원을 참조하십시오.

주제


- 사용자 지정 오류 페이지를 구성했을 때 CloudFront에서 오류를 처리하는 방법
- 사용자 지정 오류 페이지를 구성하지 않은 경우 CloudFront에서 오류를 처리하는 방법
- CloudFront가 캐싱하는 HTTP 4xx 및 5xx 상태 코드


# 사용자 지정 오류 페이지를 구성했을 때 CloudFront에서 오류를 처리하는 방법

사용자 지정 오류 페이지를 구성한 경우 CloudFront의 동작은 요청된 객체가 엣지 캐시에 있는지 여부에 따라 달라집니다.

## 요청된 객체가 엣지 캐시에 없는 경우

CloudFront는 다음이 모두 참이면 오리진에서 요청한 객체를 계속 가져오려 합니다.

- 최종 사용자가 객체를 요청하는 경우
- 객체가 엣지 캐시에 있지 않는 경우
- 오리진이 HTTP 4xx 또는 5xx 상태 코드를 반환하며 다음 중 하나가 true인 경우
  - 오리진이 304 상태 코드(수정되지 않음) 또는 객체의 업데이트된 버전을 반환하는 대신에 HTTP 5xx 상태 코드를 반환하는 경우
  - 오리진이 캐시 제어 헤더에 의해 제한되지 않으며 상태 코드 목록(CloudFront가 캐싱하는 HTTP 4xx 및 5xx 상태 코드)에 포함된 HTTP 4xx 상태 코드를 반환하는 경우
  - 오리진이 `Cache-Control max-age` 헤더 또는 `Cache-Control s-maxage` 헤더와 함께 HTTP 4xx 상태 코드를 반환하며 상태 코드가 상태 코드 목록(Control `Cache-Control` 헤더에 따라 CloudFront에서 캐싱하는 HTTP 4xx 상태 코드)에 포함됩니다.

CloudFront에서는 다음을 수행합니다.

1. 최종 사용자 요청을 받은 CloudFront 엣지 캐시에서 CloudFront는 배포 구성을 확인하여 오리진 서버에서 반환한 상태 코드에 해당하는 사용자 지정 오류 페이지의 경로를 가져옵니다.
2. CloudFront에서는 사용자 지정 오류 페이지의 경로와 일치하는 경로 패턴을 가진 배포의 첫 번째 캐시 동작을 확인합니다.
3. CloudFront 엣지 로케이션에서는 캐시 동작에 지정된 오리진에 사용자 지정 오류 페이지에 대한 요청을 전송합니다.
4. 오리진에서는 엣지 로케이션에 사용자 지정 오류 페이지를 반환합니다.
5. CloudFront가 요청한 최종 사용자에게 사용자 지정 오류 페이지를 반환하며, 다음의 최대값에 대해 사용자 지정 오류 페이지도 캐싱합니다.
   - 오류 캐싱 최소 TTL(기본값: 10초)에 의해 지정된 시간
   - 첫 번째 요청이 오류를 생성할 때 오리진에서 반환된 `Cache-Control max-age` 헤더 또는 `Cache-Control s-maxage` 헤더에 의해 지정된 시간
6. 캐싱 시간(5단계에서 결정)이 경과한 후 CloudFront는 오리진에 다른 요청을 전달하여 요청된 객체를 다시 가져오려고 시도합니다. CloudFront는 오류 캐싱 최소 TTL에 의해 지정된 간격으로 계속 다시 시도합니다.

> **Note**
>
> 동일한 사용자 지정 오류 페이지에 대한 캐시 동작도 구성한 경우 CloudFront는 대신 캐시 동작 TTL을 사용합니다. 이 경우 CloudFront는 5단계와 6단계에서 다음을 수행합니다.
>
> - CloudFront가 요청을 수행한 뷰어에 사용자 지정 오류 페이지를 반환하면 CloudFront는 캐시 동작 TTL(예: 기본 TTL을 5초로 설정)을 확인합니다. 그런 다음 CloudFront는 사용자 지정 오류 페이지를 해당 최대값까지 캐싱합니다.
> - 5초가 지나면 CloudFront는 오리진에서 사용자 지정 오류 페이지를 다시 가져옵니다. CloudFront는 캐시 동작 TTL에 의해 지정된 간격으로 계속 재시도합니다.
>
> 자세한 내용은 캐시 동작 TTL 설정을 참조하세요.

## 요청된 객체가 엣지 캐시에 있는 경우

CloudFront는 다음이 모두 참이면 현재 엣지 캐시에 있는 객체를 계속 제공하려 합니다.

- 최종 사용자가 객체를 요청하는 경우
- 객체가 엣지 캐시에 있지만 만료된 경우
- 오리진이 304 상태 코드(수정되지 않음) 또는 객체의 업데이트된 버전을 반환하는 대신에 HTTP 5xx 상태 코드를 반환하는 경우

CloudFront에서는 다음을 수행합니다.

1. 오리진이 5xx 상태 코드를 반환할 경우 CloudFront에서는 객체가 만료되었더라도 이를 제공합니다. 오류 캐싱 최소 TTL 동안 CloudFront는 엣지 캐시에서 객체를 제공함으로써 최종 사용자 요청에 지속적으로 응답합니다.

   오리진이 4xx 상태 코드를 반환할 경우 CloudFront는 최종 사용자에게 요청된 객체가 아니라 상태 코드를 반환합니다.
2. 오류 캐싱 최소 TTL이 경과한 후, CloudFront에서는 다른 요청을 오리진에 전달함으로써 요청된 객체를 다시 가져오려고 시도합니다. 객체가 자주 요청되지 않는 경우, CloudFront에서는 오리진 서버가 아직 5xx 응답을 반환하는 중이라도 이를 엣지 캐시에서 제거할 수 있습니다. 객체가 CloudFront 엣지 캐시에 보관되는 시간에 대한 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 단원을 참조하십시오.

## 사용자 지정 오류 페이지를 구성하지 않은 경우 CloudFront에서 오류를 처리하는 방법

사용자 지정 오류 페이지를 구성하지 않은 경우 CloudFront 동작은 요청된 객체가 엣지 캐시에 있는지 여부에 따라 달라집니다.

주제

- 요청된 객체가 엣지 캐시에 없는 경우
- 요청된 객체가 엣지 캐시에 있는 경우

### 요청된 객체가 엣지 캐시에 없는 경우

CloudFront는 다음이 모두 참이면 오리진에서 요청한 객체를 계속 가져오려 합니다.

- 최종 사용자가 객체를 요청하는 경우
- 객체가 엣지 캐시에 있지 않는 경우
- 오리진이 HTTP 4xx 또는 5xx 상태 코드를 반환하며 다음 중 하나가 true인 경우

- 오리진이 304 상태 코드(수정되지 않음) 또는 객체의 업데이트된 버전을 반환하는 대신에 HTTP 5xx 상태 코드를 반환하는 경우
- 오리진이 캐시 제어 헤더에 의해 제한되지 않으며 상태 코드 목록(CloudFront가 캐싱하는 HTTP 4xx 및 5xx 상태 코드)에 포함된 HTTP 4xx 상태 코드를 반환하는 경우
- 오리진이 `Cache-Control max-age` 헤더 또는 `Cache-Control s-maxage` 헤더와 함께 HTTP 4xx 상태 코드를 반환하며 상태 코드가 상태 코드 목록(Control `Cache-Control` 헤더에 따라 CloudFront에서 캐싱하는 HTTP 4xx 상태 코드)에 포함됩니다.

CloudFront에서는 다음을 수행합니다.

1. CloudFront는 최종 사용자에게 4xx 또는 5xx 상태 코드를 반환하며, 다음의 최대값에 대해 요청을 수신하는 엣지 캐시에 상태 코드도 캐싱합니다.
   - 오류 캐싱 최소 TTL(기본값: 10초)에 의해 지정된 시간
   - 첫 번째 요청이 오류를 생성할 때 오리진에서 반환된 `Cache-Control max-age` 헤더 또는 `Cache-Control s-maxage` 헤더에 의해 지정된 시간
2. 캐싱 시간(1단계에서 결정됨) 동안 CloudFront는 동일 객체에 대한 후속 최종 사용자 요청에 캐싱된 4xx 또는 5xx 상태 코드로 응답합니다.
3. 캐싱 시간(1단계에서 결정)이 경과한 후 CloudFront는 오리진에 다른 요청을 전달하여 요청된 객체를 다시 가져오려고 시도합니다. CloudFront는 오류 캐싱 최소 TTL에 의해 지정된 간격으로 계속 다시 시도합니다.

## 요청된 객체가 엣지 캐시에 있는 경우

CloudFront는 다음이 모두 참이면 현재 엣지 캐시에 있는 객체를 계속 제공하려 합니다.

- 최종 사용자가 객체를 요청하는 경우
- 객체가 엣지 캐시에 있지만 만료된 경우 즉, 기한 경과 객체입니다.
- 오리진이 304 상태 코드(수정되지 않음) 또는 객체의 업데이트된 버전을 반환하는 대신에 HTTP 5xx 상태 코드를 반환하는 경우

CloudFront에서는 다음을 수행합니다.

1. 오리진이 5xx 오류 코드를 반환할 경우 CloudFront에서는 객체가 만료되었더라도 이를 제공합니다. 오류 캐싱 최소 TTL(기본값: 10초) 동안 CloudFront는 엣지 캐시에서 객체를 제공함으로써 최종 사용자 요청에 지속적으로 응답합니다.

오리진이 4xx 상태 코드를 반환할 경우 CloudFront는 최종 사용자에게 요청된 객체가 아니라 상태 코드를 반환합니다.

2. 오류 캐싱 최소 TTL이 경과한 후, CloudFront에서는 다른 요청을 오리진에 전달함으로써 요청된 객체를 다시 가져오려고 시도합니다. 객체가 자주 요청되지 않는 경우, CloudFront에서는 오리진 서버가 아직 5xx 응답을 반환하는 중이라도 이를 엣지 캐시에서 제거할 수 있습니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 섹션을 참조하세요.

> **Tip**
>
> - `stale-if-error` 또는 `Stale-While-Revalidate` 지시문을 구성하는 경우 엣지 캐시에서 기한 경과한 객체를 사용할 수 있는 기간을 지정할 수 있습니다. 이렇게 하면 오리진을 사용할 수 없는 경우에도 뷰어에게 콘텐츠를 계속 제공할 수 있습니다. 자세한 내용은 오래된(만료된) 콘텐츠 제공 섹션을 참조하세요.
> - CloudFront는 지정된 최대 TTL 값까지 기한 경과 객체만 제공합니다. 이 기간이 지나면 엣지 캐시에서 객체를 사용할 수 없습니다.

## CloudFront가 캐싱하는 HTTP 4xx 및 5xx 상태 코드

CloudFront에서는 반환된 특정 상태 코드 및 오리진이 응답으로 특정 헤더를 반환하는지 여부에 따라 오리진에서 반환된 HTTP 4xx 및 5xx 상태 코드를 캐싱합니다.

CloudFront는 오리진에서 반환한 다음 HTTP 4xx 및 5xx 상태 코드를 캐싱합니다. HTTP 상태 코드에 대해 사용자 지정 오류 페이지를 구성한 경우, CloudFront에서는 사용자 지정 오류 페이지를 캐싱합니다.

> **Note**
>
> CachingDisabled 관리형 캐시 정책을 사용하는 경우 CloudFront는 이러한 상태 코드나 사용자 지정 오류 페이지를 캐싱하지 않습니다.

| | |
|---|---|
| 404 | 찾을 수 없음 |

| | |
|---|---|
| 414 | 요청-URI가 너무 큼 |
| 500 | 내부 서버 오류 |
| 501 | 구현되지 않음 |
| 502 | 잘못된 게이트웨이 |
| 503 | 서비스 사용 불가 |
| 504 | 게이트웨이 시간 초과 |

## `Cache-Control` 헤더에 따라 CloudFront에서 캐싱하는 HTTP 4xx 상태 코드

CloudFront에서는 오리진이 `Cache-Control max-age` 또는 `Cache-Control s-maxage` 헤더를 반환할 경우 오리진에서 반환된 다음 HTTP 4xx 상태 코드만 캐싱합니다. 이러한 HTTP 상태 코드 중 하나에 대해 사용자 지정 오류 페이지를 구성한 경우, 오리진이 캐시 제어 헤더 중 하나를 반환하면 CloudFront는 사용자 지정 오류 페이지를 캐싱합니다.

| | |
|---|---|
| 400 | 잘못된 요청 |
| 403 | 금지됨 |
| 405 | 허용되지 않은 메소드 |
| 412[1] | 사전 조건 실패 |
| 415[1] | 지원되지 않는 미디어 유형 |

[1]CloudFront는 이러한 HTTP 상태 코드에 대한 사용자 지정 오류 페이지 생성을 지원하지 않습니다.

# 사용자 지정 오류 응답 생성

CloudFront를 통해 제공하는 객체를 특정 이유로 인해 사용할 수 없는 경우, 일반적으로 웹 서버에서는 해당하는 HTTP 상태 코드를 CloudFront에 반환하여 이를 나타냅니다. 예를 들어 뷰어가 잘못된 URL을 요청한 경우 웹 서버에서는 HTTP 404(찾을 수 없음) 상태 코드를 CloudFront에 반환하고 CloudFront에서는 다시 뷰어에게 이 상태 코드를 반환합니다. 이 기본 오류 응답을 사용하는 대신 CloudFront가 뷰어에게 반환하는 사용자 지정 오류 응답을 생성할 수 있습니다.

HTTP 상태 코드에 대해 사용자 지정 오류 페이지를 반환하도록 CloudFront를 구성했으나 사용자 지정 오류 페이지를 사용할 수 없는 경우, CloudFront에서는 사용자 지정 오류 페이지가 포함된 오리진에서 CloudFront로 보낸 상태 코드를 최종 사용자에게 반환합니다. 예를 들어, 사용자 지정 오리진에서 500 상태 코드를 반환하고 500 상태 코드에 대한 사용자 지정 오류 페이지를 Amazon S3 버킷에서 가져오도록 CloudFront를 구성했다고 가정합니다. 하지만 누군가 실수로 사용자 지정 오류 페이지를 Amazon S3 버킷에서 삭제했고, CloudFront에서는 HTTP 404 상태 코드(찾을 수 없음)를 객체를 요청한 최종 사용자에게 반환합니다.

CloudFront에서 사용자 지정 오류 페이지를 최종 사용자에게 반환하는 경우, 사용자 지정 오류 페이지에 대해 표준 CloudFront 요금을 지불하며 요청된 객체에 대해서는 요금이 부과되지 않습니다. CloudFront 요금에 대한 자세한 내용은 Amazon CloudFront 요금을 참조하세요.

주제


- 오류 응답 동작 구성
- 특정 HTTP 상태 코드에 대한 사용자 지정 오류 페이지 생성
- 다른 위치에 객체 및 사용자 지정 오류 페이지 저장
- CloudFront에서 반환하는 응답 코드 변경
- CloudFront에서 오류를 캐싱하는 기간 제어


## 오류 응답 동작 구성

오류가 발생할 경우 CloudFront가 응답하는 방식을 관리할 수 있는 몇 가지 옵션이 제공됩니다. 사용자 지정 오류 응답을 구성하려면 CloudFront 콘솔, CloudFront API 또는 을 사용할 수 있습니다 CloudFormation 구성 업데이트에 선택한 방법과 관계없이 다음 팁과 권장 사항을 고려하세요.

- CloudFront에서 액세스 가능한 위치에 사용자 지정 오류 페이지를 저장합니다. Amazon S3 버킷에 저장하는 것이 좋으며 나머지 웹 사이트 또는 애플리케이션 콘텐츠와 같은 위치에 저장하지 않는 것이 좋습니다. 웹 사이트 또는 애플리케이션과 동일한 오리진에 사용자 지정 오류 페이지를 저장할

경우 오리진이 5xx 오류를 반환하기 시작하면 오리진 서버를 사용할 수 없기 때문에 CloudFront가 사용자 지정 오류 페이지를 가져올 수 없습니다. 자세한 내용은 다른 위치에 객체 및 사용자 지정 오류 페이지 저장 섹션을 참조하세요.

- CloudFront에 사용자 지정 오류 페이지를 가져올 권한이 있는지 확인합니다. 사용자 지정 오류 페이지가 Amazon S3에 저장되어 있는 경우 페이지에 공개적으로 액세스할 수 있거나 CloudFront 오리진 액세스 제어(OAC)를 구성해야 합니다. 사용자 지정 오류 페이지가 사용자 지정 오리진에 저장되어 있는 경우 페이지에 공개적으로 액세스할 수 있어야 합니다.
- (선택 사항) 원하는 경우 사용자 지정 오류 페이지와 함께 `Cache-Control` 또는 `Expires` 헤더를 추가하도록 오리진을 구성합니다. [오류 캐싱 최소 TTL(Error Caching Minimum TTL)] 설정을 사용하여 CloudFront에서 사용자 지정 오류 페이지를 캐싱할 기간을 제어할 수도 있습니다. 자세한 내용은 CloudFront에서 오류를 캐싱하는 기간 제어 섹션을 참조하세요.

## 사용자 지정 오류 응답 구성

CloudFront 콘솔에서 사용자 지정 오류 응답을 구성하려면 CloudFront 배포가 있어야 합니다. 콘솔에서 사용자 지정 오류 응답에 대한 구성 설정은 기존 배포에만 사용할 수 있습니다. 배포를 생성하는 방법에 대한 자세한 내용은 CloudFront 표준 배포 시작 섹션을 참조하세요.

Console

사용자 지정 오류 응답을 구성하려면(콘솔)

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudfront/v4/home#distributions에 있는 CloudFront 콘솔에서 배포 페이지를 엽니다.
2. 배포 목록에서 업데이트할 배포를 선택합니다.
3. [오류 페이지(Error Pages)] 탭을 선택한 다음 [사용자 지정 오류 응답 생성(Create Custom Error Response)]을 선택합니다.
4. 관련 값들을 입력합니다. 자세한 내용은 사용자 지정 오류 페이지 및 오류 캐싱 섹션을 참조하세요.
5. 원하는 값을 입력한 후 [생성(Create)]을 선택합니다.

CloudFront API or CloudFormation

CloudFront API 또는 CloudFormation을(를) 사용하여 사용자 지정 오류 응답을 구성하려면 배포에서 `CustomErrorResponse` 유형을 사용합니다. 자세한 내용은 다음 자료를 참조하십시오.

- AWS CloudFormation 사용자 설명서의 AWS::CloudFront::Distribution CustomErrorResponse

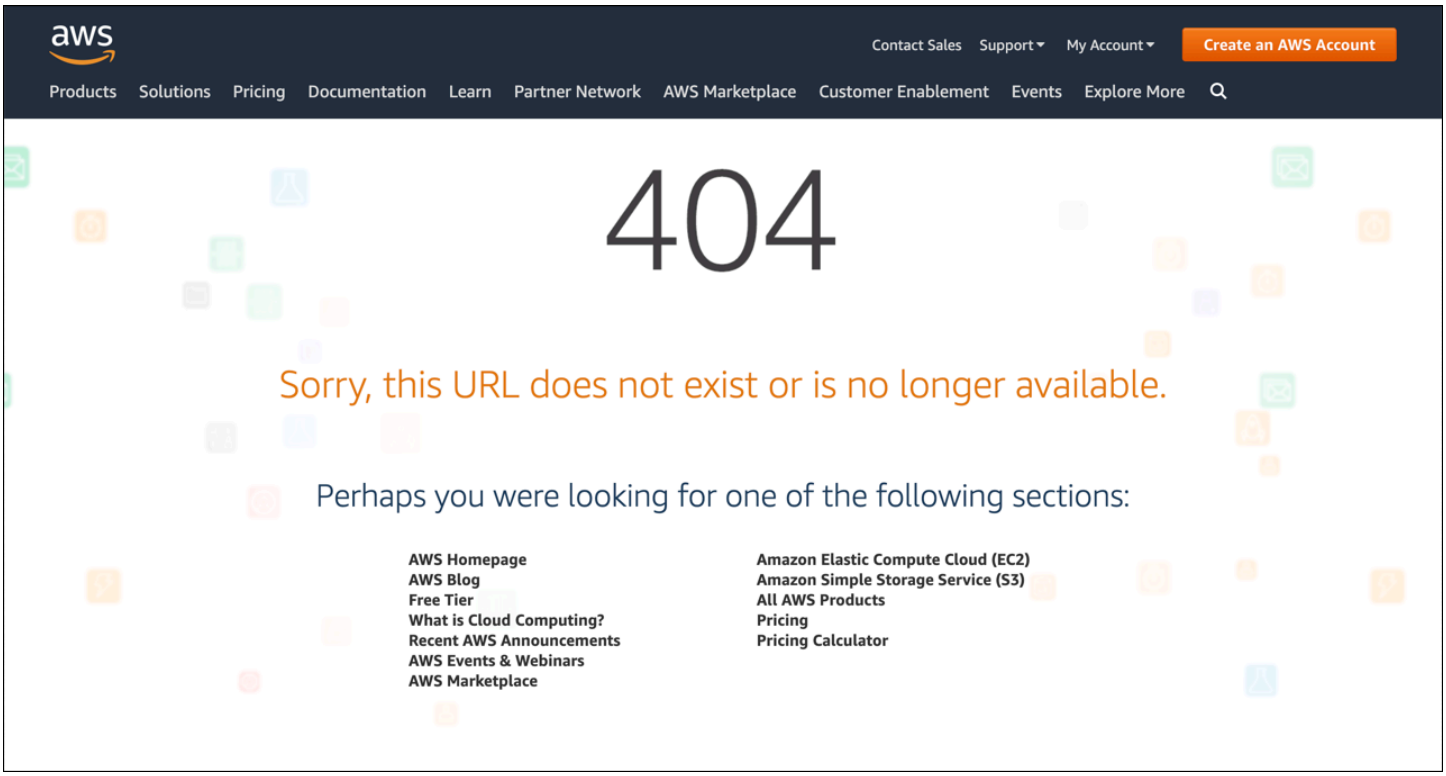
- Amazon CloudFront API 참조의 CustomErrorResponse

## 특정 HTTP 상태 코드에 대한 사용자 지정 오류 페이지 생성

기본 메시지 대신 나머지 웹 사이트와 동일한 형식을 사용하는 페이지와 같은 사용자 지정 오류 메시지를 표시하려는 경우, CloudFront가 최종 사용자에게 사용자 지정 오류 메시지가 포함된 HTML 파일 등의 객체를 반환하게 하면 됩니다.

반환하려는 파일과 이 파일을 반환해야 하는 오류를 지정하려면 CloudFront 배포를 업데이트하여 해당 값을 지정합니다. 자세한 내용은 오류 응답 동작 구성 섹션을 참조하세요.

예를 들어 다음은 사용자 지정 오류 페이지입니다.




지원되는 HTTP 상태 코드 각각에 서로 다른 객체를 지정하거나 지원되는 상태 코드 전체에 동일한 객체를 사용할 수 있습니다. 일부 상태 코드에 사용자 지정 오류 페이지를 지정하고 다른 상태 코드에는 사용자 지정 오류 페이지를 지정하지 않도록 선택할 수 있습니다.

CloudFront를 통해 제공하는 객체는 다양한 이유로 사용 불가능할 수 있습니다. 이러한 이유는 넓게 보면 다음 두 가지 범주로 나뉩니다.

- 클라이언트 오류: 요청에 문제가 있음을 나타냅니다. 지정된 이름의 객체를 사용할 수 없거나, 사용자에게 Amazon S3 버킷의 객체를 가져오기 위해 필요한 권한이 없는 경우를 예로 들 수 있습니다.

클라이언트 오류가 발생하면 오리진에서는 4xx 범위의 HTTP 상태 코드를 CloudFront에 반환합니다.

- 서버 오류: 오리진 서버에 문제가 있음을 나타냅니다. HTTP 서버가 사용 중이거나 사용 불가능한 상태인 경우를 예로 들 수 있습니다. 서버 오류가 발생하면 오리진 서버에서 5xx 범위의 HTTP 상태 코드를 CloudFront에 반환합니다. 또는 특정 기간 동안 CloudFront에서 오리진 서버로부터 응답을 가져오지 않고 504 상태 코드(게이트웨이 시간 초과)를 적용합니다.

CloudFront에서 사용자 지정 오류 페이지를 반환할 수 있는 HTTP 상태 코드에는 다음이 포함됩니다.

- 400, 403, 404, 405, 414, 416
- 500, 501, 502, 503, 504

> **참고**
>
> - CloudFront에서 요청이 안전하지 않을 수 있음을 감지하면 CloudFront는 사용자 지정 오류 페이지 대신 400(잘못된 요청) 오류를 반환합니다.
> - HTTP 상태 코드 416(요청된 범위를 충족할 수 없음)에 대한 사용자 지정 오류 페이지를 만들 수 있으며 오리진에서 CloudFront로 상태 코드 416을 반환할 경우 CloudFront에서 최종 사용자로 반환하는 HTTP 상태 코드를 변경할 수 있습니다. 자세한 내용은 CloudFront에서 반환하는 응답 코드 변경 섹션을 참조하세요. 그러나 CloudFront에서는 상태 코드 416 응답을 캐싱하지 않습니다. 따라서 상태 코드 416에 대해 [오류 캐싱 최소 TTL(Error Caching Minimum TTL)]의 값을 지정하더라도 CloudFront에 이 값이 사용되지 않습니다.
> - 경우에 따라서는 HTTP 503 상태 코드에 대한 사용자 지정 오류 페이지를 반환하도록 CloudFront를 구성하더라도 이 페이지가 반환되지 않습니다. CloudFront 오류 코드가 `Capacity Exceeded` 또는 `Limit Exceeded`인 경우 CloudFront는 사용자 지정 오류 페이지를 사용하지 않고 최종 사용자에게 503 상태 코드를 반환합니다.
> - 사용자 지정 오류 페이지를 생성한 경우 CloudFront는 다음 응답 코드에 대해 `Connection: close` 또는 `Connection: keep-alive`를 반환합니다.
>  - CloudFront는 400, 405, 414, 416, 500, 501 상태 코드에 대해 `Connection: close`를 반환합니다.
>  - CloudFront는 403, 404, 502, 503, 504 상태 코드에 대해 `Connection: keep-alive`를 반환합니다.

CloudFront에서 오리진에서 보낸 오류 응답을 처리하는 방법에 대한 자세한 설명을 보려면 CloudFront에서 오리진의 HTTP 4xx 및 5xx 상태 코드를 처리하는 방법 단원을 참조하세요.

## 다른 위치에 객체 및 사용자 지정 오류 페이지 저장

객체와 사용자 지정 오류 페이지를 서로 다른 위치에 저장하려는 경우, 배포에는 다음을 충족하는 캐시 동작이 포함되어야 합니다.

- 경로 패턴의 값이 사용자 지정 오류 메시지와 일치합니다. `/4xx-errors`라는 이름의 디렉터리에 있는 Amazon S3 버킷에 4xx 오류에 대한 사용자 지정 오류 페이지를 저장한 경우를 예로 들어 보겠습니다. 배포에는 사용자 지정 오류 페이지에 대한 요청을 해당 위치로 라우팅하는 경로 패턴(예: `/4xx-errors/*`)의 캐시 동작이 포함되어야 합니다.
- Origin(오리진)의 값은 사용자 지정 오류 페이지를 포함한 오리진에 대한 Origin ID(오리진 ID)의 값을 지정합니다.

자세한 내용은 캐시 동작 설정 섹션을 참조하세요.

## CloudFront에서 반환하는 응답 코드 변경

오리진에서 받은 것과 다른 HTTP 상태 코드를 최종 사용자에게 반환하도록 CloudFront를 구성할 수 있습니다. 예를 들어 오리진에서 CloudFront에 500 상태 코드를 반환하는 경우, CloudFront에서 사용자 지정 오류 페이지와 200 상태 코드(OK)를 최종 사용자에게 반환하려 할 수 있습니다. CloudFront에서 오리진에서 최종 사용자에게 반환한 것과 다른 상태 코드를 CloudFront로 반환하려는 이유는 다음과 같이 다양합니다.

- 일부 인터넷 디바이스(예: 일부 방화벽 및 기업 프록시)에서는 HTTP 4xx 및 5xx 상태 코드를 가로채서 응답이 최종 사용자에게 반환되는 것을 막습니다. 이 시나리오에서 `200`을 대체하면 응답을 가로채지 않습니다.
- 여러 가지 클라이언트 오류 또는 서버 오류 중에 구별할 필요가 없는 경우, CloudFront에서 전체 4xx 또는 5xx 상태 코드에 대해 반환하는 값으로 `400` 또는 `500`을 지정할 수 있습니다.
- 고객이 웹 사이트가 다운된 것을 모르도록 `200` 상태 코드(OK)와 정적 웹 사이트를 반환하려 할 수 있습니다.

CloudFront 표준 로그를 활성화하고 응답에서 HTTP 상태 코드를 변경하도록 CloudFront를 구성하는 경우 로그의 `sc-status` 열 값에 지정한 상태 코드가 포함됩니다. 그러나 `x-edge-result-type` 열의 값은 영향을 받지 않습니다. 여기에는 오리진 응답의 결과 유형이 포함됩니다. 예를 들어 오리진이 `200`(찾을 수 없음)을 CloudFront로 반환할 때 최종 사용자에게 `404`의 상태 코드를 반환하도록

CloudFront를 구성한다고 가정합니다. 오리진이 요청에 `404` 상태 코드로 응답하는 경우 로그의 `sc-status` 열 값은 `200`이 되지만 `x-edge-result-type` 열의 값은 `Error`가 됩니다.

다음 HTTP 상태 코드와 사용자 지정 오류 페이지를 반환하도록 CloudFront를 구성할 수 있습니다.

- 200
- 400, 403, 404, 405, 414, 416
- 500, 501, 502, 503, 504

## CloudFront에서 오류를 캐싱하는 기간 제어

CloudFront는 오류 응답을 기본 기간인 10초 동안 캐싱합니다. 그런 다음 CloudFront는 객체에 대한 다음 요청을 오리진에 제출하여 오류를 일으킨 문제가 해결되었고 요청된 객체를 사용할 수 있는지 확인합니다.

CloudFront에서 캐싱하는 4xx 및 5xx 상태 코드 각각에 대해 오류 캐싱 기간(오류 캐싱 최소 TTL(Error Caching Minimum TTL))을 지정할 수 있습니다. (자세한 내용은 CloudFront가 캐싱하는 HTTP 4xx 및 5xx 상태 코드 섹션을 참조하세요.) 기간을 지정할 때는 다음 사항에 유의하십시오.

- 짧은 오류 캐싱 기간을 지정하는 경우, CloudFront에서는 더 긴 기간을 지정하는 경우에 비해 오리진에 더 많은 요청을 전달합니다. 5xx 오류의 경우, 이는 원래 오리진에서 오류를 반환하게 만든 문제를 악화시킬 수 있습니다.
- 오리진에서 객체에 대한 오류를 반환할 경우, CloudFront에서는 오류 캐싱 기간이 경과할 때까지 객체에 대한 요청에 오류 응답이나 사용자 지정 오류 페이지로 응답합니다. 오류 캐싱 기간을 길게 지정한 경우, CloudFront에서는 객체가 다시 사용 가능한 상태가 된 후에도 장시간 오류 응답이나 사용자 지정 오류 페이지로 요청에 계속해서 응답할 수 있습니다.

> **Note**
>
> HTTP 상태 코드 416(요청된 범위를 충족할 수 없음)에 대한 사용자 지정 오류 페이지를 만들 수 있으며 오리진에서 CloudFront로 상태 코드 416을 반환할 경우 CloudFront에서 최종 사용자로 반환하는 HTTP 상태 코드를 변경할 수 있습니다. (자세한 내용은 CloudFront에서 반환하는 응답 코드 변경 섹션을 참조하세요.) 그러나 CloudFront에서는 상태 코드 416 응답을 캐싱하지 않습니다. 따라서 상태 코드 416에 대해 [오류 캐싱 최소 TTL(Error Caching Minimum TTL)]의 값을 지정하더라도 CloudFront에 이 값이 사용되지 않습니다.

CloudFront에서 개별 객체에 대한 오류를 캐싱하는 시간을 제어하려는 경우, 다음과 같이 관련 헤더를 해당 객체에 대한 오류 응답에 추가하도록 오리진 서버를 구성할 수 있습니다.

오리진에서 `Cache-Control: max-age` 또는 `Cache-Control: s-maxage` 명령이나 `Expires` 헤더를 추가하는 경우: CloudFront에서는 헤더의 값과 오류 캐싱 최소 TTL(Error Caching Minimum TTL)의 값 중 더 큰 값 동안 오류 응답을 캐싱합니다.

> **Note**
>
> `Cache-Control: max-age` 값과 `Cache-Control: s-maxage` 값은 오류 페이지를 가져오는 캐시 동작에 대해 설정된 최대 TTL 값보다 클 수 없습니다.

오리진이 오류 코드 404, 410, 414 또는 501에 대한 `Cache-Control: no-store`, `Cache-Control: no-cache` 또는 `Cache-Control: private` 지시문을 추가하는 경우 CloudFront는 오류 응답을 캐싱하지 않습니다. 다른 모든 오류 코드의 경우 CloudFront는 `no-store`, `no-cache` 및 `private` 지시문을 무시하고 오류 캐싱 최소 TTL 값에 대한 오류 응답을 캐싱합니다.

오리진에서 다른 `Cache-Control` 명령을 추가하거나 헤더를 추가하지 않는 경우: CloudFront에서는 오류 캐싱 최소 TTL(Error Caching Minimum TTL)의 값 동안 오류 응답을 캐싱합니다.

객체에 대한 4xx 또는 5xx 상태 코드의 만료 시간이 원하는 시간보다 더 긴 경우 객체를 다시 사용할 수 있게 될 때 요청된 객체의 URL을 사용하여 캐싱된 오류 코드를 무효화할 수 있습니다. 오리진에서 여러 객체에 대한 오류 응답을 반환하려는 경우, 각 객체를 개별적으로 무효화해야 할 수 있습니다. 객체 무효화에 대한 자세한 내용은 파일을 무효화하여 콘텐츠 제거 단원을 참조하십시오.

S3 버킷 오리진에 대해 캐싱을 사용 설정하고 CloudFront 배포에서 최소 TTL을 0초로 캐싱하는 오류를 구성한 경우에도 S3 오리진 오류에 대해 최소 TTL을 1초로 캐싱하는 오류는 계속 표시됩니다. 이렇게 하여 CloudFront는 DDoS 공격으로부터 오리진을 보호합니다. 다른 유형의 오리진에는 적용되지 않습니다.

# CloudFront가 배포하는 콘텐츠 추가, 제거 또는 교체

이 단원에서는 최종 사용자에게 제공하려는 콘텐츠를 CloudFront가 액세스할 수 있게 하는 방법, 웹 사이트 또는 애플리케이션에서 객체를 지정하는 방법, 콘텐츠를 제거하거나 바꾸는 방법을 대해 설명합니다.

주제


- CloudFront가 배포하는 콘텐츠 추가 및 액세스
- 파일 버전 관리를 사용하여 CloudFront 배포에서 콘텐츠 업데이트 또는 제거
- CloudFront에서 파일에 대한 URL 형식 사용자 지정
- 기본 루트 객체 지정
- 파일을 무효화하여 콘텐츠 제거
- 압축된 파일 제공


## CloudFront가 배포하는 콘텐츠 추가 및 액세스

CloudFront가 콘텐츠(객체)를 배포하게 하려면, 배포를 위해 지정한 오리진 중 하나에 파일을 추가하고 파일에 대한 CloudFront 링크를 공개합니다. CloudFront 엣지 로케이션은 파일에 대한 최종 사용자 요청을 받기 전까지는 오리진으로부터 새 파일을 가져오지 않습니다. 자세한 내용은 CloudFront에서 콘텐츠를 제공하는 방법 단원을 참조하세요.

CloudFront에서 배포하려는 파일을 추가하는 경우, 배포에 지정된 Amazon S3 버킷 중 하나에 이를 추가해야 하고 사용자 오리진의 경우 지정된 도메인의 디렉터리에 이를 추가해야 합니다. 또한 해당 캐시 동작의 경로 패턴에서 정확한 오리진에 요청을 전송하는지 확인합니다.

예를 들어, 캐시 동작의 경로 패턴이 `*.html`이라고 가정합니다. 이 오리진으로 요청을 전송하도록 구성된 다른 캐시 동작이 없는 경우 CloudFront는 `*.html` 파일만 전송합니다. 예를 들어, 이 시나리오에서 CloudFront는 오리진에 업로드한 .jpg 파일을 배포하지 않습니다. .jpg 파일을 포함하는 캐시 동작을 생성하지 않았기 때문입니다.

CloudFront 서버는 제공하는 객체의 MIME 형식을 확인하지 않습니다. 오리진에 파일을 업로드할 경우 이 파일에 대한 `Content-Type` 헤더 필드를 설정하는 것이 좋습니다.

# 파일 버전 관리를 사용하여 CloudFront 배포에서 콘텐츠 업데이트 또는 제거

CloudFront에서 배포하도록 설정된 기존 콘텐츠를 업데이트하려면 파일 이름 또는 폴더 이름에 버전 식별자를 사용하는 것이 좋습니다. 이를 통해 CloudFront에서 제공하는 콘텐츠 관리에 대한 제어를 강화할 수 있습니다.

## 버전이 지정된 파일 이름을 사용하여 기존 파일 업데이트

CloudFront 배포의 기존 파일을 업데이트할 경우, 콘텐츠를 보다 잘 제어할 수 있도록 파일 이름 또는 디렉터리 이름에 일종의 버전 식별자를 포함하는 것이 좋습니다. 이 식별자는 날짜-타임스탬프, 일련번호 또는 동일 객체의 두 버전을 식별하는 기타 방법이 될 수 있습니다.

예를 들어, 그래픽 파일의 이름을 image.jpg라고 지정하는 대신 image_1.jpg라고 할 수 있습니다. 이 파일의 새 버전 제공을 시작하려는 경우, 새 파일의 이름을 image_2.jpg라고 지정하고 image_2.jpg 파일을 가리키도록 웹 애플리케이션 또는 웹 사이트의 링크를 업데이트할 수 있습니다. 또는, images_v1 디렉터리에 모든 그래픽 파일을 넣을 수 있으며 그래픽 파일 하나 이상의 새 버전 제공을 시작하려는 경우 새 images_v2 디렉터리를 만들고 이 디렉터리를 가리키도록 링크를 업데이트할 수 있습니다. 버전 관리를 사용하면 CloudFront에서 새 버전의 객체 제공을 시작하기 전에 객체가 만료될 때까지 기다릴 필요가 없으며, 객체 무효화에 비용을 지불할 필요도 없습니다.

파일의 버전을 관리하는 경우라도 만료 날짜는 설정하는 것이 좋습니다. 자세한 내용은 콘텐츠가 캐시에 유지되는 기간(만료) 관리 단원을 참조하세요.

> **Note**
>
> 버전이 지정된 파일 이름 또는 디렉터리 이름을 지정하는 것은 Amazon S3 객체 버전 관리와는 관련이 없습니다.

## CloudFront가 콘텐츠를 배포하지 않도록 콘텐츠 제거

CloudFront 배포에 더 이상 포함시키지 않으려는 오리진에서 파일을 제거할 수 있습니다. 하지만 CloudFront는 파일이 만료될 때까지 엣지 캐시의 콘텐츠를 최종 사용자에게 계속 표시합니다.

파일을 즉시 제거하려면 다음 방법 중 하나를 사용해야 합니다.

- 파일 버전 관리를 사용합니다. 버전 관리를 사용하는 경우 한 파일의 여러 버전은 서로 다른 이름을 사용합니다. CloudFront 배포에 이 이름을 사용하여 최종 사용자에게 반환되는 파일을 변경할 수 있습니다. 자세한 내용은 버전이 지정된 파일 이름을 사용하여 기존 파일 업데이트 섹션을 참조하세요.
- 파일을 무효화합니다. 자세한 내용은 파일을 무효화하여 콘텐츠 제거 섹션을 참조하세요.

# CloudFront에서 파일에 대한 URL 형식 사용자 지정

CloudFront가 최종 사용자에게 제공하게 하려는 객체(콘텐츠)와 함께 오리진을 설정한 후, 올바른 URL을 사용하여 웹 사이트 또는 애플리케이션 코드에 해당 객체를 참조해야 CloudFront가 객체를 제공할 수 있습니다.

웹 페이지 또는 웹 애플리케이션에서 객체의 URL에 사용하는 도메인 이름은 다음 중 하나일 수 있습니다.

- 배포를 생성할 때 CloudFront에서 자동으로 할당하는 도메인 이름(예: `d111111abcdef8.cloudfront.net`)
- 고유의 도메인 이름(예: `example.com`)

예를 들어, 다음 URL 중 하나를 사용하여 `image.jpg` 파일을 반환할 수 있습니다.

```
https://d111111abcdef8.cloudfront.net/images/image.jpg
```

```
https://example.com/images/image.jpg
```

콘텐츠를 Amazon S3 버킷에 저장하든 사용자 지정 오리진(예: 자체 웹 서버 중 하나)에 저장하든 상관없이 동일한 URL 형식을 사용합니다.

> **Note**
>
> URL 형식은 배포에서 Origin Path(오리진 경로)에 지정하는 값에 따라 부분적으로 달라집니다. 이 값은 CloudFront에 객체의 최상위 디렉터리 경로를 제공합니다. 배포를 생성할 때 오리진 경로를 설정하는 방법에 대한 자세한 내용은 오리진 경로 섹션을 참조하세요.

URL 형식에 대한 자세한 내용은 다음 단원을 참조하십시오.

## 고유의 도메인 이름 사용(example.com)

배포를 생성할 때 CloudFront에서 할당하는 기본 도메인 이름을 사용하는 대신, `example.com`과 같은 대체 도메인 이름을 추가할 수 있습니다. CloudFront를 사용하여 고유의 도메인 이름을 설정하면 배포에서 객체에 이와 같은 URL을 사용할 수 있습니다.

```
https://example.com/images/image.jpg
```

최종 사용자와 CloudFront 간에 HTTPS를 사용하려는 경우 대체 도메인 이름과 HTTPS 사용 단원을 참조하세요.

## URL에서 후행 슬래시(/) 사용

CloudFront 배포에서 디렉터리에 대한 URL을 지정할 때 후행 슬래시를 항상 사용하거나 후행 슬래시를 사용하지 않도록 선택합니다. 예를 들어, 모든 URL에 대해 다음 형식 중 하나만 선택합니다.

```
https://d111111abcdef8.cloudfront.net/images/
```

```
https://d111111abcdef8.cloudfront.net/images
```

이 선택이 중요한 이유

두 형식 모두 CloudFront 객체에 연결할 수 있지만, 형식을 일치시키면 나중에 디렉터리를 무효화하려는 경우 문제를 방지할 수 있습니다. CloudFront는 후행 슬래시를 포함하여 지정된 그대로 URL을 저장합니다. 따라서 형식이 일치하지 않는 경우 CloudFront에서 디렉터리가 확실히 제거되도록 슬래시가 있는 디렉터리 URL과 슬래시가 없는 디렉터리 URL을 모두 무효화해야 합니다.

두 URL 형식을 모두 무효화하는 작업은 불편하며 추가 비용이 발생할 수 있습니다. 두 가지 유형의 URL을 모두 처리하기 위해 무효화를 중복해서 수행해야 하는 경우 해당 월에 허용되는 무료 무효화의 최대 수를 초과할 수 있기 때문입니다. 그리고 이러한 경우 CloudFront에 각 디렉터리 URL의 형식이 하나만 있더라도 모든 무효화 요금을 지불해야 합니다.

## 제한된 콘텐츠에 대한 서명된 URL 생성

액세스를 제한하려는 콘텐츠가 있는 경우 서명된 URL을 생성할 수 있습니다. 예를 들어, 인증한 사용자에게만 콘텐츠를 배포하려는 경우 지정된 기간 동안에만 유효하거나 지정된 IP 주소에서만 사용할 수 있는 URL을 생성할 수 있습니다. 자세한 내용은 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공 섹션을 참조하세요.

# 기본 루트 객체 지정

사용자(뷰어)가 배포의 객체가 아닌 배포의 루트 URL을 요청할 경우, 특정 객체(기본 루트 객체)를 반환하도록 CloudFront를 구성할 수 있습니다. 기본 루트 객체를 사용하면 배포 콘텐츠가 노출되지 않게 할 수 있습니다.

목차


- 기본 루트 객체를 지정하는 방법
- 기본 루트 객체 작동 방식
- 루트 객체를 정의하지 않을 경우 CloudFront의 작동 방식


## 기본 루트 객체를 지정하는 방법

배포의 콘텐츠가 노출되거나 오류가 반환되는 것을 방지하려면 배포에 대한 기본 루트 객체를 지정합니다. 정확한 파일 이름 또는 파일의 경로를 지정할 수 있습니다. 예를 들어 루트 객체가 `index.html` 파일인 경우 해당 파일 이름을 지정할 수 있습니다. `index.html` 파일이 다른 폴더에 있는 경우 대신 경로를 지정합니다(예: *`exampleFolderName`*`/index.html`). 경로를 기본 루트 객체로 설정하면 배포의 루트 URL에 대한 뷰어 요청은 해당 경로에서 지정된 파일을 반환합니다. 기본 루트 객체가 루트 수준 대신 폴더에 있을 수 있으므로 파일 경로를 사용하여 오리진에서 콘텐츠를 보다 유연하게 구성할 수 있습니다.

배포에 대한 기본 루트 객체를 지정하려면

1. 배포에서 가리키는 오리진에 기본 루트 객체를 업로드합니다.

   파일은 CloudFront에서 지원하는 모든 유형이 가능합니다. 파일 이름에 대한 제한 사항 목록을 보려면 Amazon CloudFront API 참조에 나와 있는 DistributionConfig 섹션에서 `DefaultRootObject` 요소를 참조하시기 바랍니다.

   > **Note**
   >
   > 기본 루트 객체의 파일 이름이 너무 길거나 유효하지 않은 문자가 포함된 경우, CloudFront에서는 `HTTP 400 Bad Request - InvalidDefaultRootObject` 오류를 반환합니다. 또한 CloudFront에서는 10초(기본값) 동안 코드를 캐싱하고 결과를 액세스 로그에 작성합니다.

2. 객체에 대한 권한이 CloudFront에 최소 읽기 액세스 권한을 부여하는지 확인합니다.

Amazon S3 권한에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 Amazon S3의 Identity and Access Management를 참조하세요.

3. CloudFront 콘솔 또는 CloudFront API를 사용하여 기본 루트 객체를 참조하는 배포를 업데이트합니다.

   CloudFront 콘솔을 사용하여 기본 루트 객체를 지정하려면 다음과 같이 합니다.

   a. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
   b. 상단 창의 배포 목록에서 업데이트하려는 배포를 선택합니다.
   c. 설정 창의 일반 탭에서 편집을 선택하세요.
   d. 설정 편집 대화 상자의 기본 루트 객체 필드에 기본 루트 객체의 파일 이름을 입력합니다.

   > **Tip**
   > 문자열은 슬래시(/)로 시작할 수 없습니다. 객체 이름 또는 객체의 경로만 지정합니다. 예는 `index.html` 또는 *`exampleFolderName`*`/index.html` 섹션을 참조하세요. **/***`exampleFolderName`*`/index.html` 또는 **/**`index.html`을 지정하면 403 액세스 거부됨 오류가 발생할 수 있습니다.

   e. 변경 사항 저장을 선택합니다.

   CloudFront API를 사용하여 구성을 업데이트하려면 배포에서 `DefaultRootObject` 요소의 값을 지정합니다. CloudFront API를 사용한 기본 루트 객체 지정에 대한 자세한 내용은 Amazon CloudFront API 참조의 DistributionConfig를 참조하세요.

4. 루트 URL을 요청하여 기본 루트 객체를 활성화했는지 확인합니다. 브라우저에 기본 루트 객체가 표시되지 않는 경우 다음 단계를 수행합니다.

   a. CloudFront 콘솔에서 배포의 상태를 확인하여 배포가 모두 완료되었는지 확인합니다.
   b. 2단계와 3단계를 반복하여 올바른 권한을 부여하고 기본 루트 객체를 지정하는 배포 구성을 올바르게 업데이트했는지 확인합니다.

## 기본 루트 객체 작동 방식

예를 들어 다음 요청은 `image.jpg` 객체를 가리킵니다.

`https://d111111abcdef8.cloudfront.net/image.jpg`

반면에, 다음 요청은 특정 객체 대신에 앞 예제와 동일한 배포의 루트 URL을 가리킵니다.

`https://d111111abcdef8.cloudfront.net/`

기본 루트 객체를 정의한 경우 배포의 루트를 호출하는 최종 사용자 요청은 기본 루트 객체를 반환합니다. 예를 들어, `index.html` 파일을 기본 루트 객체로 지정하는 경우, 그에 대한 요청은 다음과 같습니다.

`https://d111111abcdef8.cloudfront.net/`

반환 형식:

`https://d111111abcdef8.cloudfront.net/index.html`

> **Note**
>
> CloudFront는 여러 개의 후행 슬래시가 있는 URL(`https://d111111abcdef8.cloudfront.net///`)이 `https://d111111abcdef8.cloudfront.net/`와 동일한지 여부를 확인하지 않습니다. 오리진 서버가 이러한 비교를 수행합니다.

기본 루트 객체를 정의한 경우 배포의 하위 디렉터리에 대한 최종 사용자 요청은 기본 루트 객체를 반환하지 않습니다. 예를 들어, `index.html`이 기본 루트 객체이고 CloudFront에서 CloudFront 배포 아래의 `install` 디렉터리에 대한 최종 사용자 요청을 받았다고 가정합니다.

`https://d111111abcdef8.cloudfront.net/install/`

`index.html`의 사본이 `install` 디렉터리에 나타나더라도 CloudFront에서는 기본 루트 객체를 반환하지 않습니다. 하지만 기본 루트 객체에 대한 경로를 지정한 경우 (`install/index.html`)CloudFront는 `install` 디렉터리의 최종 사용자 요청에 대해 기본 루트 객체를 반환합니다.

CloudFront에서 지원되는 모든 HTTP 메서드를 허용하도록 배포를 구성한 경우, 기본 루트 객체는 모든 메서드에 적용됩니다. 예를 들어 기본 루트 객체가 index.php이고 POST 요청을 도메인(https://example.com)의 루트에 제출하도록 애플리케이션을 작성한 경우, CloudFront에서는 요청을 https://example.com/index.php로 보냅니다.

CloudFront 기본 루트 객체의 동작은 Amazon S3 인덱스 문서의 동작과는 다릅니다. Amazon S3 버킷을 웹 사이트로 구성하고 인덱스 문서를 지정한 경우, 사용자가 버킷의 하위 디렉터리를 요청하더라도

Amazon S3에서는 인덱스 문서를 반환합니다. (인덱스 문서의 사본은 모든 하위 디렉터리에 나타나야 합니다.) Amazon S3 버킷을 웹 사이트로 구성하는 방법과 인덱스 문서에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 Amazon S3에서 웹 사이트 호스팅 단원을 참조하세요.

> **Important**
>
> 기본 루트 객체는 CloudFront 배포에만 적용된다는 점에 유의하세요. 이 경우에도 오리진의 보안은 관리해야 합니다. 예를 들어, Amazon S3 오리진을 사용하는 경우, Amazon S3 버킷 ACL을 적절히 설정하여 버킷에 대해 원하는 액세스 수준을 보장해야 합니다.

## 루트 객체를 정의하지 않을 경우 CloudFront의 작동 방식

기본 루트 객체를 정의하지 않은 경우, 배포의 루트에 대한 요청은 오리진 서버로 전달됩니다. Amazon S3 오리진을 사용하는 경우 다음 중 하나가 반환될 수 있습니다.

- Amazon S3 버킷의 콘텐츠 목록 - 다음 조건 중 하나에 해당하는 경우 오리진의 콘텐츠가 CloudFront를 사용하여 배포에 액세스하는 사람에게 표시됩니다.
  - 버킷이 적절하게 구성되지 않았습니다.
  - 배포에 연결된 버킷과 이 버킷에 있는 객체에 대한 Amazon S3 권한은 모두에 대한 액세스 권한을 부여합니다.
  - 최종 사용자가 오리진 루트 URL을 사용하여 오리진에 액세스합니다.
- 오리진의 비공개 콘텐츠 목록 - 비공개 배포(사용자 및 CloudFront에만 액세스 권한이 있음)로 오리진을 구성하는 경우, 배포에 연결된 Amazon S3 버킷의 콘텐츠는 CloudFront를 통해 배포에 액세스하는 자격 증명을 가진 모두에게 표시됩니다. 이러한 경우 사용자는 오리진 루트 URL을 통해 콘텐츠에 액세스할 수 없습니다. 비공개 콘텐츠를 배포하는 방법에 대한 자세한 내용은 the section called "서명된 URL과 서명된 쿠키를 사용하여 콘텐츠 제한"를 참조하십시오.
- `Error 403 Forbidden` - 배포에 연결된 Amazon S3 버킷에 대한 권한이나 이 버킷에 있는 객체에 대한 권한이 CloudFront 및 모두에 대한 액세스를 거부하는 경우, CloudFront에서는 이 오류를 반환합니다.

# 파일을 무효화하여 콘텐츠 제거

CloudFront 엣지 캐시에서 파일이 만료되기 전에 파일을 제거해야 할 경우, 다음 방법 중 하나를 사용하면 됩니다.

- 엣지 캐시에서 파일을 무효화합니다. 다음에 최종 사용자가 파일을 요청하면 CloudFront는 최신 버전의 파일을 가져오도록 오리진에 반환합니다.
- 파일 버전 관리를 사용하여 서로 다른 이름을 가진 여러 버전의 파일을 제공합니다. 자세한 내용은 버전이 지정된 파일 이름을 사용하여 기존 파일 업데이트 섹션을 참조하세요.

주제


- 파일을 무효화하는 방법과 버전이 지정된 파일 이름을 사용하는 방법 중에 선택
- 무효화할 파일 결정
- 파일 무효화 시 알아야 할 사항
- 파일 무효화
- 동시 무효화 요청 최대값
- 파일 무효화에 대한 요금 결제


## 파일을 무효화하는 방법과 버전이 지정된 파일 이름을 사용하는 방법 중에 선택

배포가 제공하는 파일의 버전을 제어하는 방법에는 파일을 무효화하는 방법과, 버전 지정된 파일 이름을 제공하는 방법이 있습니다. 파일을 자주 업데이트하려는 경우, 기본적으로 파일 버전 관리를 사용하는 것이 좋습니다. 그 이유는 다음과 같습니다.

- 버전 관리를 사용하면 사용자가 로컬에 또는 회사 캐싱 프록시 뒤에 캐싱된 버전을 가지고 있더라도 요청이 반환하는 파일을 제어할 수 있습니다. 파일을 무효화하면 파일이 해당 캐시에서 만료될 때까지는 사용자에게 이전 버전이 계속 표시될 수 있습니다.
- 버전 관리를 통해 파일 변경의 결과를 더 쉽게 분석할 수 있도록 CloudFront 액세스 로그에는 파일의 이름이 포함됩니다.
- 버전 관리를 사용하면 서로 다른 사용자에게 여러 버전의 파일을 제공할 수 있습니다.
- 버전 관리를 통해 파일 개정 버전을 손쉽게 앞 버전이나 뒤 버전으로 이동할 수 있습니다.
- 버전 관리는 비용이 저렴합니다. CloudFront가 새 버전의 파일을 엣지 로케이션에 전송하게 하려면 요금이 발생하지만, 파일 무효화에 대해서는 요금을 지불할 필요가 없습니다.

파일 버전 관리에 대한 자세한 내용은 버전이 지정된 파일 이름을 사용하여 기존 파일 업데이트 단원을 참조하십시오.

## 무효화할 파일 결정

특정 디렉터리에 있는 모든 파일이나 같은 문자로 시작하는 이름을 가진 모든 파일 등 여러 파일을 무효화하려는 경우, 무효화 경로의 끝에 * 와일드카드를 포함시키면 됩니다. * 와일드카드 사용에 대한 자세한 내용은 Invalidation paths 단원을 참조하십시오.

파일을 무효화하려면 개별 파일에 대한 경로나 * 와일드카드로 끝나는 경로 중에 지정할 수 있습니다. 이는 다음 예에 나와 있듯 각각 하나의 파일 또는 여러 파일에 적용됩니다.

- `/images/image1.jpg`
- `/images/image*`
- `/images/*`

선택한 파일을 무효화하려 하지만 사용자가 오리진의 모든 파일에 반드시 액세스할 필요가 없는 경우, CloudFront에서 최종 사용자가 요청한 파일을 확인하여 그러한 파일만 무효화할 수 있습니다. 최종 사용자가 요청한 파일을 확인하려면 CloudFront 액세스 로깅을 활성화합니다. 액세스 로그에 대한 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하십시오.

## 파일 무효화 시 알아야 할 사항

무효화할 파일을 지정할 때 다음 정보를 참조합니다.

대소문자 구분

무효화 경로는 대/소문자를 구분합니다. 예를 들어 `/images/image.jpg` 및 `/images/Image.jpg`는 서로 다른 두 파일을 지정합니다.

Lambda 함수를 사용하여 URI 변경

CloudFront 배포가 최종 사용자 요청 이벤트에서 Lambda 함수를 트리거하고 해당 함수가 요청된 파일의 URI를 변경하는 경우 CloudFront 엣지 캐시에서 파일을 제거하려면 두 URI를 모두 무효화하는 것이 좋습니다.

- 최종 사용자 요청의 URI
- 함수가 변경한 후의 URI

Example예제

예를 들어 Lambda 함수가 파일의 URI를 다음에서

`https://d111111abcdef8.cloudfront.net/index.html`

언어 디렉터리가 포함된 다음 URI로 변경한다고 가정합니다.

```
https://d111111abcdef8.cloudfront.net/en/index.html
```

파일을 무효화하려면 다음 경로를 지정해야 합니다.

- `/index.html`
- `/en/index.html`

자세한 내용은 Invalidation paths 단원을 참조하세요.

기본 루트 객체

기본 루트 객체(파일)를 무효화하려면 다른 파일에 대한 경로를 지정할 때와 같은 방법으로 경로를 지정합니다. 자세한 내용은 기본 루트 객체 작동 방식 섹션을 참조하세요.

쿠키 전달

오리진에 쿠키를 전달하도록 CloudFront를 구성하는 경우, CloudFront 엣지 캐시에는 여러 버전의 파일이 포함되어 있을 수 있습니다. 파일을 무효화할 경우 CloudFront에서는 연결된 쿠키와 상관없이 각 캐싱된 버전의 파일을 무효화합니다. 연결된 쿠키에 따라 일부 버전만 선택적으로 무효화하고 나머지는 무효화하지 않을 수는 없습니다. 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하세요.

헤더 전달

오리진에 헤더 목록을 전달하고 헤더 값에 따라 캐싱하도록 CloudFront를 구성한 경우, CloudFront 엣지 캐시에는 여러 버전의 파일이 포함되어 있을 수 있습니다. 파일을 무효화할 경우 CloudFront에서는 헤더 값과 상관없이 각 캐싱된 버전의 파일을 무효화합니다. 헤더 값에 따라 일부 버전만 선택적으로 무효화하고 나머지는 무효화하지 않을 수는 없습니다. 모든 헤더를 오리진에 전달하도록 CloudFront를 구성한 경우 CloudFront는 파일을 캐싱하지 않습니다. 자세한 내용은 요청 헤더 기반의 콘텐츠 캐싱 단원을 참조하세요.

쿼리 문자열 전달

CloudFront에서 쿼리 문자열을 오리진에 전달하도록 구성한 경우, 다음 예제에서와 같이 파일 무효화 시에는 쿼리 문자열을 포함해야 합니다.

- `/images/image.jpg?parameter1=a`
- `/images/image.jpg?parameter1=b`

클라이언트 요청에 동일 파일에 대한 5가지 쿼리 문자열을 포함하는 경우, 파일을 5번 무효화하거나 각 쿼리 문자열에 대해 한 번만 무효화할 수 있습니다. 또는 다음 예제에서와 같이 * 와일드카드를 무효화 경로에 사용할 수 있습니다.

`/images/image.jpg*`

무효화 경로에 와일드카드를 사용하는 방법에 대한 자세한 내용은 Invalidation paths 단원을 참조하십시오.

쿼리 문자열에 대한 자세한 내용은 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱 단원을 참조하십시오.

사용 중인 쿼리 문자열을 확인하기 위해서는 CloudFront 로깅을 활성화하면 됩니다. 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하세요.

최대 허용

허용되는 최대 무효화 수에 대한 자세한 내용은 동시 무효화 요청 최대값 섹션을 참조하세요.

Microsoft Smooth Streaming 파일

해당 캐시 동작에 대해 Smooth Streaming을 활성화한 경우 Microsoft Smooth Streaming 형식의 미디어 파일을 무효화할 수 없습니다.

ASCII가 아니거나 경로에서 안전하지 않은 문자

경로에 비 ASCII 문자나 RFC 1738에 정의된 안전하지 않은 문자가 포함된 경우, 그러한 문자를 URL로 인코딩합니다. 경로에 있는 그 밖의 다른 문자는 URL로 인코딩하지 마세요. 이렇게 하면 CloudFront에서 기존 버전의 업데이트된 파일을 무효화하지 않습니다.

> **Important**
>
> 경로에 ~ 문자를 사용하면 안 됩니다. CloudFront는 URL 인코딩 여부에 관계없이 무효화에 대해 이 문자를 지원하지 않습니다.

무효화 경로

이 경로는 배포에 상대적입니다. 예를 들어, `https://d111111abcdef8.cloudfront.net/images/image2.jpg`의 파일을 무효화하려면 `/images/image2.jpg`를 지정할 수 있습니다.

> **Note**
>
> CloudFront 콘솔에서 `images/image2.jpg`와 같이 경로의 선행 슬래시를 생략할 수 있습니다. CloudFront API를 직접 사용하는 경우 무효화 경로는 선행 슬래시로 시작해야 합니다.

또한 * 와일드카드를 사용하여 여러 파일을 동시에 무효화할 수도 있습니다. *는 0개 이상의 문자로 대체되며 무효화 경로의 마지막 문자여야 합니다.

> **Important**
>
> 무효화에 와일드카드(*)를 사용하려면 경로 끝에 와일드카드를 입력해야 합니다. 다른 곳에 별표(*)를 삽입하면 와일드카드 무효화 대신 문자 그대로 일치하는 것을 찾도록 처리됩니다.

AWS Command Line Interface(AWS CLI)를 사용하여 파일을 무효화하고 * 와일드카드를 포함하는 경로를 지정하는 경우에는 경로를 따옴표("")로 묶어야 합니다(예: "/*").

경로의 최대 길이는 4,000자입니다.

Example예: 무효화 경로

- 특정 디렉터리의 모든 파일을 무효화하려면:

  /*directory-path*/*

- 특정 디렉터리와 그 하위 디렉터리 전체, 디렉터리와 하위 디렉터리에 들어 있는 모든 파일을 무효화하려면:

  /*directory-path**

- logo.jpg, logo.png, log.gif와 같이 같은 이름에 다른 파일 이름 확장명을 지닌 모든 파일을 무효화하려면

  /*directory-path*/*file-name*.*

- 특정 디렉터리의 전체 파일 중 파일 이름이 같은 문자로 시작하는 파일(예: HLS 포맷 비디오용 파일 전체)을 무효화하려면

  /*directory-path*/*initial-characters-in-file-name**

- 쿼리 문자열 파라미터에 따라 캐싱하도록 CloudFront를 구성할 때 각 버전의 파일을 무효화하려는 경우

  /*directory-path*/*file-name*.*file-name-extension**

- 특정 배포의 모든 파일을 무효화하려면:

  /*

Lambda 함수를 사용하여 URI를 변경하는 경우 파일을 무효화하는 방법에 대한 자세한 내용은 Changing the URI Using a Lambda Function 단원을 참조하세요.

무효화 경로가 디렉터리이고 후행 슬래시(/) 포함 여부 등 디렉터리를 지정하는 방식을 표준화하지 않은 경우, 후행 슬래시(/)를 포함한 디렉터리와 포함하지 않은 디렉터리 모두를 무효화하는 것이 좋습니다(예: `/images` 및 `/images/`).

서명된 URL

서명된 URL을 사용하는 경우 물음표(?) 앞의 URL 부분만 포함하여 파일을 무효화합니다.

# 파일 무효화

CloudFront 콘솔을 사용하여 무효화를 생성 및 실행하고 이전에 제출한 무효화의 목록과 개별 무효화에 관한 세부 정보를 표시할 수 있습니다. 또한 기존 무효화를 복사하고 파일 경로의 목록을 편집하며 편집된 무효화를 실행할 수 있습니다. 목록에서 무효화를 제거할 수 없습니다.

목차


- 파일 무효화
- 기존 무효화를 복사, 편집, 재실행
- 무효화 취소
- 무효화 목록 작성
- 무효화에 대한 정보를 표시하려면


## 파일 무효화

CloudFront 콘솔을 사용하여 파일을 무효화하려면 다음과 같이 합니다.

Console

파일을 무효화하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 파일을 무효화하려는 배포를 선택합니다.
3. Invalidations(무효화) 탭을 선택합니다.
4. 무효화 생성을 선택합니다.

5. 무효화하려는 파일에 대해 한 줄에 하나의 무효화 경로를 입력합니다. 무효화 경로 지정에 대한 자세한 내용은 파일 무효화 시 알아야 할 사항 단원을 참조하십시오.

> **Important**
>
> 파일 경로를 신중하게 지정합니다. 무효화 요청을 시작한 후에는 취소할 수 없습니다.

6. 무효화 생성을 선택합니다.

CloudFront API

객체를 무효화하고 무효화 관련 정보를 표시하는 방법을 알아보려면 Amazon CloudFront API 참조에서 다음 주제를 참조하세요.

- CreateInvalidation
- ListInvalidations
- GetInvalidation

> **Note**
>
> AWS Command Line Interface(AWS CLI)를 사용하여 파일을 무효화하고 * 와일드카드를 포함하는 경로를 지정하는 경우에는 다음 예시와 같이 경로를 따옴표(")로 묶어야 합니다.
>
> ```
> aws cloudfront create-invalidation --distribution-id distribution_ID --paths
> "/*"
> ```

## 기존 무효화를 복사, 편집, 재실행

이전에 생성한 무효화를 복사하고 무효화 경로의 목록을 업데이트하여 업데이트된 무효화를 실행할 수 있습니다. 기존 무효화를 복사하고 무효화 경로를 업데이트하여 업데이트된 무효화를 실행하지 않고 저장할 수는 없습니다.

> **Important**
>
> 아직 진행 중인 무효화를 복사하고 무효화 경로의 목록을 업데이트하여 업데이트된 무효화를 실행하는 경우, CloudFront에서는 복사한 무효화를 중단하거나 삭제하지 않습니다. 무효화 경

로가 원본 및 복사본에 표시되는 경우 CloudFront에서는 파일에 대해 두 번 무효화를 시도하며, 두 무효화 모두 월별 무료 무효화 최대 횟수로 산정됩니다. 이미 무료 무효화 최대 횟수에 도달한 경우 각 파일의 무효화 두 건에 대해 요금이 청구됩니다. 자세한 내용은 동시 무효화 요청 최대값 섹션을 참조하세요.

기존 무효화를 복사, 편집, 재실행하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 복사하려는 무효화가 포함된 배포를 선택합니다.
3. Invalidations(무효화) 탭을 선택합니다.
4. 복사하려는 무효화를 선택합니다.

   복사하려는 무효화가 확실치 않은 경우, 무효화를 하나 선택하고 세부 정보 보기를 선택하여 해당 무효화에 대한 세부 정보를 표시할 수 있습니다.
5. 신규로 복사를 선택합니다.
6. 해당되는 경우 무효화 경로의 목록을 업데이트합니다.
7. 무효화 생성을 선택합니다.

## 무효화 취소

CloudFront에 무효화 요청을 제출하면 CloudFront는 몇 초 이내에 모든 엣지 로케이션으로 요청을 전달하며, 각 엣지 로케이션은 즉시 무효화를 처리하기 시작합니다. 따라서 무효화를 제출한 후에는 취소할 수 없습니다.

## 무효화 목록 작성

CloudFront 콘솔을 사용하여 배포에 대해 생성하고 실행한 최근 100건의 무효화 목록을 표시할 수 있습니다. 100건이 넘는 무효화 목록을 가져오려는 경우 `ListInvalidations` API 작업을 사용합니다. 자세한 내용은 Amazon CloudFront API 참조의 ListInvalidations를 참조하세요.

무효화 목록을 작성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 무효화 목록을 표시하려는 배포를 선택합니다.

3. Invalidations(무효화) 탭을 선택합니다.

> **Note**
> 목록에서 무효화를 제거할 수 없습니다.

### 무효화에 대한 정보를 표시하려면

배포 ID, 무효화 ID, 무효화의 상태, 무효화가 생성된 날짜 및 시간, 무효화 경로의 전체 목록을 포함한 무효화에 관한 세부 정보를 표시할 수 있습니다.

무효화에 대한 정보를 표시하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 세부 정보를 표시하려는 무효화가 포함된 배포를 선택합니다.
3. Invalidations(무효화) 탭을 선택합니다.
4. 해당하는 무효화 ID를 선택하거나 무효화 ID를 선택한 다음 세부 정보 보기를 선택합니다.

## 동시 무효화 요청 최대값

파일을 개별적으로 무효화하는 경우, 배포당 한 번에 최대 3,000개의 파일에 대해 진행 중인 무효화 요청을 보유할 수 있습니다. 이것은 최대 3,000개의 파일에 대해 한 번의 무효화 요청을 하거나, 각 파일에 대해 최대 3,000건의 요청을 하거나, 3,000개의 파일을 초과하지 않는 기타 모든 조합이 가능함을 의미합니다. 예를 들어, 각각 100개의 파일을 무효화하는 30건의 무효화 요청을 제출할 수 있습니다. 30건의 무효화 요청 모두가 아직 진행 중인 경우 더 이상의 무효화 요청은 제출할 수 없습니다. 이 최대값을 초과하면 CloudFront에서 오류 메시지를 반환합니다.

* 와일드카드를 사용하는 경우 한 번에 최대 15개의 무효화 경로에 대해 진행 중인 요청을 보유할 수 있습니다. 또한 배포당 최대 3,000개의 개별 파일에 대해 진행 중인 무효화 요청을 동시에 보유할 수 있습니다. 허용되는 와일드카드 무효화 요청에 대한 최대값은 개별적으로 파일을 무효화하는 데 적용되는 최대값과는 독립적입니다.

## 파일 무효화에 대한 요금 결제

월별로 제출한 최초 1,000개의 무효화 경로에 대해서는 무료입니다. 월별로 1,000개를 초과하는 각 무효화 경로에 대해서 요금을 지불합니다. 무효화 경로는 단일 파일(예:`/images/logo.jpg`) 또는 여러 파일(예: `/images/*`)에 적용될 수 있습니다. CloudFront가 수천 개의 파일을 무효화하더라도 `*` 와일드카드 수가 하나의 경로로 포함되는 경로입니다.

월별 1,000개의 무료 무효화 경로 최대값은 하나의 AWS 계정으로 생성한 전체 배포에서 무효화 경로의 총 수에 적용됩니다. 예를 들어, AWS 계정 `john@example.com`을 사용하여 세 개의 배포를 만든 경우 지정된 월에 배포별로 600개의 무효화 경로를 제출했다면(총 1,800개의 무효화 경로에 해당), AWS에서는 총 무효화 경로와 1,000개의 무료 한도 간의 차이에 대해 요금을 부과합니다. 이 예시에서 AWS는 해당 월에 800개의 무효화 경로에 대해 요금을 부과합니다.

무효화 경로를 제출하는 데 부과되는 요금은 무효화하는 파일 수와 관계없이 단일 객체(`/images/logo.jpg`) 또는 배포와 연결된 모든 파일(`/*`)에 대해 동일합니다. 무효화 요청에서는 경로별로 요금이 부과되므로 여러 경로를 단일 요청으로 묶더라도 각 경로는 여전히 청구 시 개별적으로 계산됩니다.

무효화 요금에 대한 자세한 내용은 Amazon CloudFront 요금을 참조하세요. 무효화 경로에 대한 자세한 내용은 Invalidation paths 단원을 참조합니다.

# 압축된 파일 제공

객체가 압축되면 객체 크기가 더 작아지므로 다운로드 속도가 빨라집니다. 원래 크기의 1/4 이하로 줄어드는 경우도 있습니다. 특히 JavaScript 및 CSS 파일의 경우 다운로드 속도가 빨라지면 뷰어에게 표시되는 웹 페이지의 렌더링 속도가 빨라집니다. 또한 CloudFront 데이터 전송 비용은 제공된 총 데이터 양을 기준으로 합니다. 압축된 객체를 제공하면 압축되지 않은 객체를 제공하는 것보다 비용이 저렴할 수 있습니다.

주제


- 객체를 압축하도록 CloudFront 구성
- CloudFront 압축의 작동 원리
- 압축 조건
- CloudFront가 압축하는 파일 형식
- ETag 헤더 변환

# 객체를 압축하도록 CloudFront 구성

CloudFront가 객체를 압축하도록 구성하려면 압축된 객체를 제공하려는 캐시 동작을 업데이트합니다.

콘솔에서 객체를 압축하도록 CloudFront를 구성하려면 다음과 같이 합니다.

1. CloudFront 콘솔에 로그인합니다.
2. 배포를 선택한 다음 편집할 동작을 선택합니다.
3. 자동으로 객체 압축 설정에서 예를 선택합니다.
4. 캐시 정책을 사용하여 캐싱 설정을 지정하고 Gzip과 Brotli 압축 형식 모두 사용 설정합니다.

> **Notes**
>
> - Brotli 압축을 사용하려면 캐시 정책을 사용해야 합니다. Brotli는 레거시 캐시 설정을 지원하지 않습니다.
> - CloudFormation 또는 CloudFront API를 사용하여 압축을 사용 설정하려면 `Compress`, `EnableAcceptEncodingGzip`, `EnableAcceptEncodingBrotli` 파라미터를 `true`로 설정합니다.

CloudFront가 객체를 압축하는 방법을 이해하려면 다음 섹션을 참조하시기 바랍니다.

## CloudFront 압축의 작동 원리

1. 뷰어가 객체를 요청하는 경우 뷰어는 요청에 `Accept-Encoding` HTTP 헤더를 포함하며 헤더 값에는 `gzip`, `br` 또는 둘 다 포함됩니다. 이는 뷰어가 압축된 객체를 지원함을 의미합니다. 뷰어가 Gzip과 Brotli 모두 지원하는 경우 CloudFront는 Brotli를 사용합니다.

   > **Note**
   >
   > Chrome 및 Firefox 웹 브라우저는 HTTPS를 사용하여 요청이 전송될 경우에만 Brotli 압축을 지원합니다. HTTP 요청에서는 Brotli를 지원하지 않습니다.

2. 엣지 로케이션에서 CloudFront는 요청 객체의 압축된 사본 캐시를 확인합니다.
3. 압축된 객체가 캐시에 있는지 여부에 따라 CloudFront는 다음 중 하나를 수행합니다.

- 압축된 객체가 이미 캐시에 있으면 CloudFront는 뷰어에게 객체를 전송하고 나머지 단계를 건너뜁니다.
- 압축된 객체가 캐시에 없으면 CloudFront는 요청을 오리진에 전달합니다.

> **Note**
>
> 캐시에 이미 압축되지 않은 객체 사본이 있는 경우 CloudFront는 요청을 오리진에 전달하지 않고 뷰어에게 전송할 수 있습니다. 예를 들어 CloudFront가 이전에 압축을 건너뛴 경우 이 문제가 발생할 수 있습니다. 이 경우 CloudFront는 압축되지 않은 객체를 캐싱하고 객체가 만료, 제거 또는 무효화될 때까지 계속 처리합니다.

4. HTTP 응답의 `Content-Encoding` 헤더에 표시되는 것처럼 오리진에서 압축 객체를 반환하는 경우, CloudFront는 압축된 객체를 뷰어에게 전송하고 캐시에 추가한 다음, 나머지 단계는 건너뜁니다. CloudFront는 객체를 다시 압축하지 않습니다.
5. 오리진이 압축되지 않은 객체를 CloudFront에 반환하는 경우(HTTP 응답의 `Content-Encoding` 헤더에 없는 경우) CloudFront는 객체의 압축 가능 여부를 결정합니다. 자세한 내용은 압축 조건 섹션을 참조하세요.
6. 객체를 압축할 수 있으면 CloudFront는 객체를 압축하고 뷰어에게 압축된 객체를 전송한 다음, 이를 캐시에 추가합니다.
7. 동일한 객체에 대한 뷰어 요청이 잇따르는 경우 CloudFront는 첫 번째 캐싱된 버전을 반환합니다. 예를 들어 뷰어가 Gzip 압축을 사용하는 특정 캐시 객체를 요청하고 뷰어가 Gzip 형식을 수락하면 뷰어가 Brotli와 Gzip 모두를 수락하더라도 동일한 객체에 대한 후속 요청은 항상 Gzip 버전을 반환합니다.

일부 사용자 지정 오리진도 객체를 압축할 수 있습니다. CloudFront에서 압축하지 않는 객체를 오리진에서 압축할 수 있습니다. 자세한 내용은 CloudFront가 압축하는 파일 형식 섹션을 참조하세요.

## 압축 조건

다음 목록은 CloudFront가 객체를 압축하지 않는 시나리오에 대한 자세한 정보를 제공합니다.

요청에는 HTTP 1.0을 사용합니다

CloudFront에 대한 요청이 HTTP 1.0을 사용하는 경우 CloudFront는 `Accept-Encoding` 헤더를 제거하고 응답에서 객체를 압축하지 않습니다.

**`Accept-Encoding` 요청 헤더**

뷰어 요청에서 `Accept-Encoding` 헤더가 누락되었거나 `gzip` 또는 `br`을 값으로 포함하지 않은 경우 CloudFront는 응답에서 객체를 압축하지 않습니다. `Accept-Encoding` 헤더에 `deflate`와 같은 추가 값이 포함된 경우, CloudFront에서는 오리진 서버에 요청을 전달하기 전에 이를 제거합니다.

CloudFront가 객체를 압축하도록 구성된 경우, 자동으로 캐시 키와 오리진 요청에 `Accept-Encoding` 헤더를 추가합니다.

CloudFront가 객체를 압축하도록 구성한 경우 콘텐츠는 이미 캐싱됩니다.

CloudFront는 오리진에서 객체를 가져올 때 객체를 압축합니다. CloudFront가 객체를 압축하도록 구성한 경우 CloudFront는 이미 엣지 로케이션에 캐싱된 객체는 압축하지 않습니다. 또한 엣지 로케이션에서 캐시 객체가 만료되고 CloudFront가 오리진에 객체에 대한 다른 요청을 전달하면 CloudFront는 오리진이 HTTP 상태 코드 304를 반환하는 경우 객체를 압축하지 않습니다. 즉, 엣지 로케이션에 이미 최신 버전의 객체가 있습니다. CloudFront로 이미 엣지 로케이션에 캐싱된 객체를 압축하려는 경우 해당 객체를 무효화해야 합니다. 자세한 내용은 파일을 무효화하여 콘텐츠 제거 섹션을 참조하세요.

오리진이 이미 객체를 압축하도록 구성된 경우

CloudFront에서 객체를 압축하도록 구성하고 또한 오리진에서 객체를 압축한 경우 오리진에는 `Content-Encoding` 헤더가 포함되어야 합니다. 이 헤더는 CloudFront에 객체가 이미 압축되었음을 나타냅니다. 오리진에서의 응답에 `Content-Encoding` 헤더가 포함되어 있는 경우, CloudFront는 헤더 값과 관계없이 객체를 압축하지 않습니다. CloudFront가 뷰어로 응답을 보내고 엣지 로케이션에서 객체를 캐싱합니다.

CloudFront가 압축하는 파일 형식

전체 목록은 CloudFront가 압축하는 파일 형식 섹션을 참조하세요.

CloudFront가 압축하는 객체 크기

CloudFront는 1,000바이트~10,000,000바이트 크기의 객체를 압축합니다.

**`Content-Length` 헤더**

오리진에는 응답의 `Content-Length` 헤더를 포함하 여 객체의 크기가 CloudFront가 압축하는 범위에 있는지 CloudFront에서 확인할 수 있도록 해야 합니다. 만약 `Content-Length` 헤더가 누락되거나, 잘못된 값을 포함하거나, CloudFront Front가 압축하는 크기 범위를 벗어나는 값을 포함하는 경우 CloudFront는 객체를 압축하지 않습니다. CloudFront가 크기 범위를 초과할 수 있는 대형

객체를 처리하는 방법에 대한 자세한 내용은 CloudFront에서 객체에 대한 부분적인 요청을 처리하는 방법(범위 GET) 섹션을 참조하시기 바랍니다.

응답의 HTTP 상태 코드

CloudFront는 응답의 HTTP 상태 코드가 `200`, `403` 또는`404`인 경우에만 객체를 압축합니다.

응답에 본문 없음

오리진의 HTTP 응답에 본문이 없으면 CloudFront가 압축할 수 있는 것은 없습니다.

**`ETag`** 헤더

CloudFront는 때때로 객체를 압축 할 때 HTTP 응답에 `ETag` 헤더를 수정합니다. 자세한 내용은 the section called "`ETag` 헤더 변환" 섹션을 참조하세요.

CloudFront의 압축 건너뛰기

CloudFront는 가능한 한 객체를 압축합니다. 드문 경우지만 CloudFront에서 트래픽 부하가 높으면 CloudFront는 객체 압축을 건너뜁니다. CloudFront는 호스트 용량을 비롯한 다양한 요인을 기반으로 이러한 결정을 내립니다. CloudFront는 객체에 대한 압축을 건너뛴 경우, 압축되지 않은 객체를 캐시하고 객체가 만료되거나 제거되거나 무효화될 때까지 뷰어에 계속 제공합니다.

## CloudFront가 압축하는 파일 형식

CloudFront가 콘텐츠를 압축하도록 구성하는 경우, CloudFront는 `Content-Type` 응답 헤더에 다음 값이 포함된 객체를 압축합니다.

- `application/dash+xml`
- `application/eot`
- `application/font`
- `application/font-sfnt`
- `application/javascript`
- `application/json`
- `application/opentype`
- `application/otf`
- `application/pdf`
- `application/pkcs7-mime`
- `application/protobuf`

- application/rss+xml
- application/truetype
- application/ttf
- application/vnd.apple.mpegurl
- application/vnd.mapbox-vector-tile
- application/vnd.ms-fontobject
- application/wasm
- application/xhtml+xml
- application/xml
- application/x-font-opentype
- application/x-font-truetype
- application/x-font-ttf
- application/x-httpd-cgi
- application/x-javascript
- application/x-mpegurl
- application/x-opentype
- application/x-otf
- application/x-perl
- application/x-ttf
- font/eot
- font/opentype
- font/otf
- font/ttf
- image/svg+xml
- text/css
- text/csv
- text/html
- text/javascript
- text/js
- text/plain

- text/richtext
- text/tab-separated-values
- text/xml
- text/x-component
- text/x-java-source
- text/x-script
- vnd.apple.mpegurl

## ETag 헤더 변환

오리진의 압축되지 않은 객체에 유효하고 강력한 ETag HTTP 헤더가 포함되어 있고 CloudFront가 객체를 압축한 경우, CloudFront는 강력한 ETag 헤더 값을 취약한 ETag로 변환하고, 취약한 ETag 값을 뷰어에게 반환합니다. 최종 사용자는 취약한 ETag 값을 저장하고 이를 사용하여 If-None-Match HTTP 헤더가 있는 조건부 요청을 보낼 수 있습니다. 이를 통해 최종 사용자, CloudFront 및 오리진은 객체의 압축 버전과 압축되지 않은 버전을 의미상 동일하게 취급할 수 있으므로 불필요한 데이터 전송이 줄어듭니다.

유효하고 강력한 ETag 헤더 값은 큰따옴표("")로 시작하고 끝납니다. 강력한 ETag 값을 취약한 값으로 변환하려면 CloudFront는 강력한 W/ 값의 시작 부분에 ETag 문자를 추가합니다.

오리진 객체에 취약한 ETag 헤더 값(W/ 문자로 시작하는 값)이 포함된 경우, CloudFront에서는 이 값을 수정하지 않고 오리진에서 받은 최종 사용자에게 반환합니다.

오리진 객체에 잘못된 ETag 헤더 값(값이 " 또는 W/로 시작하지 않음)이 포함된 경우, CloudFront에서는 ETag 헤더를 제거하고 ETag 응답 헤더 없이 최종 사용자에게 객체를 반환합니다.

자세한 내용은 MDN 웹 문서의 다음 페이지를 참조하세요.

- 지시문(ETag HTTP 헤더)
- 취약한 검증(HTTP 조건부 요청)
- If-None-Match HTTP 헤더

# AWS WAF 보호 사용

AWS WAF를 사용하여 CloudFront 배포와 오리진 서버를 보호할 수 있습니다. AWS WAF는 요청이 서버에 도달하기 전에 요청을 차단하여 웹 애플리케이션 및 API를 보호하는 데 도움이 되는 웹 애플리케이션 방화벽입니다. 자세한 내용은 CloudFront 및 AWS WAF를 사용한 웹 사이트 가속화 및 보호 및 AWS WAF 구현 지침을 참조하세요.

다음과 같은 방법으로 AWS WAF 보호를 활성화할 수 있습니다.

- CloudFront 콘솔에서 원클릭 보호 기능을 사용합니다. 원클릭 보호 기능은 AWS WAF 웹 액세스 제어 목록(웹 ACL)을 생성하고, 일반적인 웹 위협으로부터 서버를 보호하는 규칙을 구성하고, 웹 ACL을 CloudFront 배포에 자동으로 연결합니다. 이 섹션의 항목에서는 원클릭 보호 기능을 사용한다고 가정합니다.
- AWS WAF 콘솔 또는 AWS WAF API를 사용하여 생성한 미리 구성된 웹 액세스 제어 목록(ACL)을 사용합니다. 자세한 내용은 AWS WAF 개발자 안내서의 웹 액세스 제어 목록(ACL) 및 AWS WAF API 참조의 AssociateWebACL을 참조하세요.

다음과 같은 경우에 AWS WAF를 활성화할 수 있습니다.

- 배포 생성
- 보안 대시보드를 사용하여 기존 배포의 보안 설정을 편집합니다.

원클릭 보호를 사용하는 경우 CloudFront는 다음과 같은 일련의 AWS 권장 보호를 적용합니다.

- Amazon 내부 위협 인텔리전스를 기반으로 잠재적 위협으로부터 IP 주소를 차단합니다.
- OWASP Top 10에 나온 웹 애플리케이션에서 발견되는 가장 일반적인 취약성으로부터 보호합니다.
- 애플리케이션 취약성을 발견하는 악의적인 행위자로부터 보호합니다.

> **Important**
>
> CloudFront 보안 대시보드에서 보안 지표를 보려면 AWS WAF를 활성화해야 합니다. AWS WAF가 활성화되지 않은 경우 보안 대시보드를 사용하여 AWS WAF를 활성화하거나 CloudFront 지리적 제한을 구성할 수만 있습니다. 대시보드에 대한 자세한 내용은 이 섹션의 후반부에 나오는 CloudFront 보안 대시보드에서 AWS WAF 보안 보호 관리 항목을 참조하세요.

주제


- 배포에 AWS WAF 활성화
- CloudFront 보안 대시보드에서 AWS WAF 보안 보호 관리
- 속도 제한 설정
- AWS WAF 보안 보호 비활성화


# 배포에 AWS WAF 활성화

배포를 생성할 때 AWS WAF를 활성화하거나, 기존 액세스 제어 목록(ACL)에 대한 보안 보호를 활성화할 수 있습니다.

CloudFront 배포에 AWS WAF를 활성화하면 Bot Control을 활성화하고 봇 범주별로 보안 보호를 구성할 수도 있습니다.

주제


- 새 배포에 AWS WAF 활성화
- 기존 웹 ACL 사용
- Bot Control 활성화
- 봇 범주별 보호 구성


## 새 배포에 AWS WAF 활성화

다음 절차에서는 CloudFront 배포를 생성할 때 AWS WAF를 활성화하는 방법을 보여줍니다.

새 배포에 AWS WAF를 활성화하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 배포를 선택한 후 배포 생성을 선택합니다.
3. 필요한 경우 배포 생성의 단계를 따릅니다.
4. 웹 애플리케이션 방화벽 섹션에서 편집을 선택한 후 보안 보호 활성화를 선택합니다.
5. 다음 작업을 완료합니다.
    - 모니터링 모드 사용 – 보호 기능이 어떻게 작동하는지 테스트하기 위해 먼저 데이터를 수집하려는 경우 모니터링 모드를 활성화합니다. 모니터링 모드를 활성화하면 보호 기능이 활성화되어

있어도 요청이 차단되지 않습니다. 대신 모니터링 모드는 보호 기능이 활성화된 경우 차단될 요청에 대한 데이터를 수집합니다. 차단을 시작할 준비가 되면 보안 페이지에서 차단을 활성화할 수 있습니다.

- 추가 보호 – 활성화하려는 옵션을 선택합니다. 속도 제한을 사용하도록 설정하는 경우 자세한 내용은 the section called "속도 제한 설정" 단원을 참조하세요.
- 예상 가격 – 섹션을 열면 표시되는 필드에 다른 요청/월 수를 입력하여 새 예상 가격을 확인할 수 있습니다.

6. 나머지 배포 설정을 검토한 다음 배포 생성을 선택합니다.

배포를 생성한 후 CloudFront는 보안 대시보드를 생성합니다. 이 대시보드를 사용하여 AWS WAF를 비활성화하거나 활성화할 수 있습니다. AWS WAF를 아직 활성화하지 않은 경우 대시보드의 차트와 그래프는 비어 있습니다.

## 기존 웹 ACL 사용

기존 웹 ACL이 있는 경우 AWS WAF가 제공하는 보호 기능 대신 웹 ACL을 사용할 수 있습니다.

기존 AWS WAF 구성을 사용하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다
2. 다음 중 하나를 수행합니다.
   a. 배포 생성을 선택하고 배포 생성의 단계를 수행한 다음 이 항목으로 돌아옵니다.
   b. 기존 구성을 선택한 다음 보안 탭을 선택합니다.
3. 웹 애플리케이션 방화벽(WAF) 섹션에서 편집을 선택한 후 보안 보호 활성화를 선택합니다.
4. 기존 WAF 구성 사용을 선택합니다. 이 옵션은 웹 ACL을 구성한 경우에만 표시됩니다.
5. 웹 ACL 선택 테이블에서 기존 웹 ACL을 선택합니다.
6. 나머지 배포 설정을 검토한 다음 배포 생성을 선택합니다.

## Bot Control 활성화

CloudFront 배포에 AWS WAF를 활성화하면 CloudFront 콘솔의 보안 대시보드에서 지정된 시간 범위에 대한 봇 요청을 볼 수 있습니다. 또한 여기서 Bot Control을 활성화하거나 비활성화할 수 있습니다.

Bot Control을 활성화하면 요금이 발생합니다. 보안 대시보드에서는 비용 추정을 제공합니다.

Bot Control을 활성화하면 보안 대시보드에 각 봇 유형 및 범주별로 봇 트래픽이 표시됩니다. Bot Control을 비활성화하면 요청 샘플링을 기반으로 한 봇 트래픽이 표시됩니다.

Bot Control을 활성화하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 배포를 선택한 후 변경하려는 배포를 선택합니다.
3. 보안 탭을 선택합니다.
4. 지정된 시간 범위의 봇 요청 섹션까지 아래로 스크롤하고 Bot Control 활성화를 선택합니다.
5. Bot Control 대화 상자의 구성에서 일반 봇에 대해 Bot Control 활성화 확인란을 선택합니다.
6. Save changes(변경 사항 저장)를 선택합니다.

## 봇 범주별 보호 구성

Bot Control을 활성화하면 검증되지 않은 각 봇이 봇 범주별로 처리되는 방법을 구성할 수 있습니다. 예를 들어, HTTP 라이브러리 봇을 모니터링 모드로 설정하고 링크 검사기에 챌린지를 할당할 수 있습니다.

> **Note**
> 알려진 검색 엔진 크롤러와 같이 AWS에서 일반적이고 검증 가능한 것으로 알려진 봇에는 여기에서 설정한 작업이 적용되지 않습니다. Bot Control은 검증된 봇으로 표시하기 전에 검증된 봇이 스스로 주장하는 소스에서 나온 것인지 확인합니다.

봇 범주에 대한 보호를 구성하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 배포를 선택한 후 변경하려는 배포를 선택합니다.
3. 보안 탭을 선택합니다.
4. 봇 범주별 요청 차트에서 확인되지 않은 봇 작업 열의 항목 중 하나를 가리키고 연필 아이콘을 선택하여 편집합니다.
5. 결과 목록을 열고 다음 중 하나를 선택합니다.
   - 차단

- 허용
- 모니터링 모드
- CAPTCHA
- 챌린지

6. 목록 옆의 체크 표시를 선택하여 변경 사항을 저장합니다.

# CloudFront 보안 대시보드에서 AWS WAF 보안 보호 관리

CloudFront는 각 배포에 대한 보안 대시보드를 생성합니다. CloudFront 콘솔에서 대시보드를 사용할 수 있습니다. 대시보드를 사용하면 CloudFront와 AWS WAF를 단일 위치에서 함께 사용하여 웹 애플리케이션에 대한 일반적인 보안 보호를 모니터링하고 관리할 수 있습니다. 대시보드는 다음과 같은 작업과 데이터를 제공합니다.

- 보안 구성 - AWS WAF 보호를 활성화 및 비활성화하고 WordPress 보호와 같은 앱별 보호를 확인할 수 있습니다.
- 보안 추세 - 허용 및 차단된 요청, 챌린지 및 CAPTCHA 요청, 주요 공격 유형을 포함한 보안 트렌드가 포함됩니다. 트래픽 비율과 시간에 따른 트래픽 비율의 변화를 확인할 수 있습니다. 예를 들어 모든 요청이 3% 증가했지만 허용된 요청이 14% 증가했다면 현재 기간에 더 많은 양의 트래픽을 허용했음을 의미합니다.
- 봇 요청 - 봇에서 발생하는 트래픽의 양, 봇 유형(검증된 봇과 검증되지 않은 봇), 시간 경과에 따른 봇 유형(검증된 봇과 검증되지 않은 봇)의 할당 비율 변화를 확인할 수 있습니다. Bot Control 활성화에 대한 자세한 내용은 Bot Control 활성화 섹션을 참조하세요.
- 요청 로그 - 로그 데이터는 보안 트렌드 또는 봇 요청에 대한 질문에 답하는 데 도움이 될 수 있습니다. 쿼리를 작성하지 않고도 로그를 검색할 수 있으며, 집계된 차트를 통해 필터링된 로그 집합이 주로 일부 HTTP 메서드, IP 주소, URI 경로 또는 국가에서 발생하는지 확인할 수 있습니다. 차트의 값에 마우스 포인터를 올려놓고 IP 주소 및 국가를 차단할 수 있습니다. 자세한 내용은 AWS WAF 로그 활성화 단원을 참조하십시오.
- 지리적 제한 관리 - CloudFront와 AWS WAF는 지리적 제한 기능을 제공합니다. CloudFront는 무료로 지리적 제한 기능을 제공하지만 CloudFront의 지리적 제한에 대한 지표는 보안 대시보드에 표시되지 않습니다. 차단된 국가 요청에 대한 요청 지표를 보려면 AWS WAF의 지리적 제한 기능을 사용해야 합니다. 이렇게 하려면 보안 대시보드의 국가 표시줄에 마우스를 갖다 대고 국가를 차단하면 됩니다. 자세한 내용은 CloudFront 지리적 제한 사용 단원을 참조하십시오.
  - 이전에 CloudFront 콘솔 외부에서 국가를 차단하는 사용자 지정 AWS WAF 규칙을 생성한 경우 차단 옵션을 사용하지 못할 수 있습니다.

주제


- 사전 조건
- AWS WAF 로그 활성화


## 사전 조건

CloudFront 보안 대시보드에서 보안 지표를 보려면 AWS WAF를 활성화해야 합니다. AWS WAF를 활성화하지 않는 경우 보안 대시보드를 사용하여 AWS WAF를 활성화하거나 CloudFront 지리적 제한을 구성할 수만 있습니다.

AWS WAF 활성화에 대한 자세한 내용은 배포에 AWS WAF 활성화 단원을 참조하십시오.

## AWS WAF 로그 활성화

AWS WAF 로그 데이터는 특정 트래픽 패턴을 분리하는 데 도움이 될 수 있습니다. 예를 들어, 로그를 통해 특정 트래픽이 어디서 오는지 또는 어떤 작업을 수행하는지를 알 수 있습니다.

CloudWatch에 대한 AWS WAF 로깅을 활성화하면 CloudFront 보안 대시보드가 CloudWatch 로그를 쿼리하고 집계하여 인사이트를 표시합니다. 보안 대시보드 사용에 대해서는 요금이 부과되지 않지만, 대시보드를 통해 쿼리된 로그에는 CloudWatch 요금이 적용됩니다. 자세한 내용은 Amazon CloudWatch 요금을 참조하세요.

로그를 활성화하려면

1. 요청 횟수/월 상자에 예상 요청량을 입력하여 로그 활성화 비용을 추정합니다.
2. AWS WAF 로그 활성 확인란을 선택합니다.
3. 활성화를 선택합니다.

CloudFront는 CloudWatch 로그 그룹을 생성하고 AWS WAF 구성을 업데이트하여 CloudWatch에 로깅을 시작합니다. 처음 사용 시 로그 데이터가 표시되는 데 몇 분 정도 걸릴 수 있습니다. 차트의 요청 섹션에는 각 요청이 나열됩니다. 개별 요청 아래의 막대 차트에는 HTTP 메서드, 상위 URI 경로, 상위 IP 주소 및 상위 국가별로 데이터가 집계됩니다. 차트를 통해 패턴을 찾을 수 있습니다. 예를 들어, 단일 IP 주소에서 요청의 양이 너무 많거나 이전에 로그에서 볼 수 없었던 국가의 데이터가 표시될 수 있습니다. 국가, 호스트 헤더 및 기타 속성을 기준으로 요청을 필터링하여 원치 않는 트래픽을 찾을 수 있습니다. 트래픽을 식별한 후 개별 요청 또는 차트 항목 위에 마우스 포인터를 놓아 IP 주소 또는 국가를 차단합니다.

> **Note**
> 표시되는 지표는 웹 ACL을 기반으로 합니다. 따라서 동일한 웹 ACL을 여러 배포에 연결하면 해당 배포에 대해 처리된 AWS WAF 요청뿐 아니라 웹 ACL에 대한 모든 지표가 표시됩니다.

# 속도 제한 설정

속도 제한은 보안 보호를 구성할 때 받을 수 있는 권장 사항 중 하나입니다.

CloudFront는 모니터링 모드에서 항상 속도 제한을 사용합니다. 모니터링 모드를 사용하도록 설정하면 CloudFront는 속도 제한 필드에서 구성한 속도가 초과되었는지, 얼마나 자주, 얼마나 초과되었는지를 알려주는 지표를 캡처합니다.

배포를 저장하면, CloudFront는 속도 제한 필드에 있는 숫자에 따라 데이터를 수집하기 시작합니다.

CloudFront 배포의 보안 탭에 있는 보안 - 웹 애플리케이션 방화벽(WAF) 섹션에서 속도 제한 설정을 사용 또는 관리할 수 있습니다.

> **Note**
> 배포에 S3가 아닌 사용자 지정 오리진을 지정한 경우에만 CloudFront 콘솔에 속도 제한 옵션이 나타납니다. 그렇지 않으면 배포에 대해 사용 설정된 코어 보호만 표시됩니다. 오리진에 대한 자세한 내용은 CloudFront 배포에 다양한 오리진 사용 섹션을 참조하시기 바랍니다.

속도 제한 설정

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 배포를 선택한 후 변경하려는 배포를 선택합니다.
3. 보안 탭을 선택합니다.
4. 보안 - 웹 애플리케이션 방화벽(WAF) 섹션에서 편집을 선택합니다.
5. 추가 보호에서 속도 제한을 선택합니다. 선택적으로 속도 제한을 변경할 수 있습니다. 속도를 미세 조정한 경우 변경 사항 저장을 선택합니다.
6. 보안 - 웹 애플리케이션 방화벽(WAF) 섹션의 속도 제한 옆에서 모니터링 모드를 선택한 다음, 차단 활성화를 선택하여 모니터링 모드를 사용 해제할 수 있습니다. CloudFront가 지정된 속도 제한을 초과하는 요청을 차단하기 시작합니다.

AWS WAF 사용 설정 및 속도 제한에 대한 자세한 내용은 Introducing CloudFront Security Dashboard, a Unified CDN and Security Experience 블로그 게시물을 참조하시기 바랍니다.

# AWS WAF 보안 보호 비활성화

배포에 AWS WAF 보안 보호가 필요하지 않은 경우 CloudFront 콘솔을 사용하여 이 기능을 비활성화할 수 있습니다.

이전에 AWS WAF 보호를 활성화하고 기존 WAF 구성(원클릭 보호라고도 함)을 선택하지 않은 경우 CloudFront는 자동으로 웹 ACL을 생성합니다. 이러한 방식으로 생성된 웹 ACL의 경우 CloudFront 콘솔은 리소스 연결을 끊고 웹 ACL을 삭제합니다.

웹 ACL의 연결을 해제하는 것은 웹 ACL을 삭제하는 것과 다릅니다. 연결을 해제하면 배포에서 웹 ACL이 제거되지만 AWS 계정에서 삭제되지는 않습니다. 자세한 내용은 AWS WAF, AWS Firewall Manager, AWS Shield Advanced 개발자 안내서에서 웹 ACL과 AWS 리소스 연결 또는 연결 해제하기를 참조합니다.

AWS WAF 보호를 비활성화하고 배포에서 웹 ACL의 연결을 끊으려면 다음 절차를 참조합니다.

CloudFront에서 AWS WAF 보안 보호를 비활성화하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다
2. 탐색 창에서 배포를 선택한 후 변경하려는 배포를 선택합니다.
3. 보안탭을 선택한 다음 편집 탭을 선택합니다.
4. 웹 애플리케이션 방화벽(WAF) 섹션에서 AWS WAF 보호 비활성화를 선택합니다.
5. Save changes(변경 사항 저장)를 선택합니다.

> **Notes**
>
> - AWS WAF 보안 보호를 비활성화했는데도 AWS 계정에서 웹 ACL을 삭제하려는 경우 수동으로 삭제할 수 있습니다. 절차에 따라 웹 ACL을 삭제합니다. AWS WAF & Shield 콘솔에서 웹 ACL 페이지의 경우 글로벌(CloudFront) 목록을 선택하여 웹 ACL을 찾아야 합니다
> - CloudFront 콘솔에서 배포를 삭제하면 원클릭 보호를 선택한 경우 CloudFront는 웹 ACL도 삭제하려고 시도합니다. 이는 최선의 노력이며 보장되지 않습니다. 자세한 내용은 배포 삭제 단원을 참조하십시오.

# 보안 액세스 구성 및 콘텐츠에 대한 액세스 제한

CloudFront에서 제공되는 콘텐츠의 보안을 위해 몇 가지 옵션을 사용할 수 있습니다. 다음은 CloudFront에서 콘텐츠에 대한 액세스를 보호하고 제한하는 데 사용할 수 있는 몇 가지 방법입니다.

- HTTPS 연결 구성
- 특정 지리적 위치의 사용자가 콘텐츠에 액세스하지 못하도록 차단
- 사용자가 CloudFront 서명된 URL 또는 서명된 쿠키를 사용하여 콘텐츠에 액세스하도록 요구
- 특정 콘텐츠 필드에 대한 필드 수준 암호화 설정
- AWS WAF를 사용하여 콘텐츠에 대한 액세스 제어

또한 인프라 및 애플리케이션을 위한 DDoS 복원력이 있는 아키텍처를 구현해야 합니다. 자세한 내용은 DDoS 복원력에 대한 AWS 모범 사례를 참조하세요.

추가 정보는 다음을 참조하세요.

- CloudFront를 사용하여 콘텐츠 전송 보호
- Amazon OpenSearch Service의 SIEM

주제


- CloudFront에서 HTTPS 사용
- 대체 도메인 이름과 HTTPS 사용
- CloudFront에 대한 상호 TLS 인증(뷰어 mTLS)
- CloudFront에 대한 상호 TLS(오리진)
- 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공
- AWS 오리진에 대한 액세스 제한
- Application Load Balancer에 대한 액세스 제한
- 콘텐츠의 지리적 배포 제한
- 필드 수준 암호화를 사용하여 민감한 데이터 보호

# CloudFront에서 HTTPS 사용

뷰어가 HTTPS를 사용할 것을 요청하도록 CloudFront를 구성할 수 있습니다. 이렇게 하면 CloudFront가 뷰어와 통신할 때 연결이 암호화됩니다. 또한 CloudFront가 오리진과 HTTPS를 사용하도록 구성할 수 있습니다. 이렇게 하면 CloudFront가 오리진과 통신할 때 연결이 암호화됩니다.

CloudFront가 뷰어와 통신할 때 그리고 오리진과 통신할 때 모두 HTTPS를 요구하도록 구성할 경우, CloudFront에서 요청을 받을 때 다음과 같은 프로세스가 수행됩니다.

1. 최종 사용자가 HTTPS 요청을 CloudFront에 제출합니다. 뷰어와 CloudFront 간에 SSL/TLS 협상이 수행됩니다. 결국에는 최종 사용자가 암호화된 형식을 요청을 제출합니다.
2. CloudFront 엣지 로케이션에 캐시 응답을 포함하는 경우 CloudFront가 응답을 암호화하여 뷰어에게 반환하면 뷰어는 이를 해독합니다.
3. CloudFront 엣지 로케이션에 캐시 응답을 포함하지 않는 경우 CloudFrontnt는 오리진과 SSL/TLS 협상을 수행하며 협상이 완료되면 요청을 암호화된 형식으로 오리진에 전달합니다.
4. 오리진은 요청을 복호화한 후 요청을 처리하여 응답을 생성 및 암호화한 후 CloudFront에 반환합니다.
5. CloudFront는 응답을 복호화한 후 다시 암호화하여 뷰어에게 전달합니다. 또한 CloudFront는 엣지 로케이션에 응답을 캐시하여 다음에 요청할 때 사용할 수 있도록 합니다.
6. 최종 사용자가 응답을 해독합니다.

이 프로세스는 오리진 서버가 Amazon S3 버킷이든, MediaStore이든, HTTP/S 서버와 같은 사용자 지정 오리진이든 상관없이 기본적으로 동일하게 작동합니다.

> **Note**
>
> SSL 재협상 유형 공격을 차단하기 위해 CloudFront는 최종 사용자와 오리진의 요청에 대한 재협상을 지원하지 않습니다.

또는 CloudFront 배포에 대해 상호 인증을 켤 수 있습니다. 자세한 내용은 CloudFront에 대한 상호 TLS 인증(뷰어 mTLS) 섹션을 참조하세요.

최종 사용자와 CloudFront 간에 그리고 CloudFront와 오리진 간에 HTTPS를 요청하는 방법을 알아보려면 다음 주제를 참조하세요.

주제

- 뷰어와 CloudFront 간의 통신에 HTTPS 요구
- CloudFront와 사용자 지정 오리진 간의 통신에 HTTPS 요구
- CloudFront와 Amazon S3 오리진 간의 통신에 HTTPS 요구
- 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호
- CloudFront와 오리진 간에 지원되는 프로토콜 및 암호

## 뷰어와 CloudFront 간의 통신에 HTTPS 요구

CloudFront 배포에 하나 이상의 캐시 동작을 구성하여 최종 사용자와 CloudFront 간의 통신에 HTTPS를 요구할 수 있습니다. 또한 CloudFront가 일부 객체에 대해서만 HTTPS를 요구하도록 하기 위해 HTTP 및 HTTPS 모두를 허용하는 캐시 동작을 한 개 이상 구성할 수 있습니다. 구성 단계는 객체 URL에 사용하는 도메인 이름에 따라 다릅니다.

- CloudFront가 배포에 할당한 도메인 이름(예: d111111abcdef8.cloudfront.net)을 사용할 경우 한 개 이상의 캐시 동작에 대해 최종 사용자 프로토콜 정책(Viewer Protocol Policy) 설정을 변경하여 HTTPS 통신을 요구하도록 합니다. 이 구성에서 CloudFront는 SSL/TLS 인증서를 제공합니다.

  CloudFront 콘솔을 사용하여 최종 사용자 프로토콜 정책(Viewer Protocol Policy) 값을 변경하려면 이 단원의 절차를 참조하세요.

  CloudFront API를 사용하여 `ViewerProtocolPolicy` 요소의 값을 변경하는 방법을 보려면 Amazon CloudFront API 참조의 UpdateDistribution을 참조하세요.
- example.com 등의 자체 도메인 이름을 사용할 경우 몇 가지 CloudFront 설정을 변경해야 합니다. 또한 AWS Certificate Manager(ACM)에서 제공한 SSL/TLS 인증서를 사용하거나, 다른 인증 기관의 인증서를 또는 IAM 인증서 스토어로 가져와야 합니다. 자세한 내용은 대체 도메인 이름과 HTTPS 사용 단원을 참조하세요.

> **Note**
>
> CloudFront가 오리진에서 객체를 받을 때 최종 사용자가 CloudFront로부터 받는 객체가 암호화되도록 하려면 CloudFront와 오리진 간에 항상 HTTPS를 사용하세요. 최근에 CloudFront와 오리진 간의 연결을 HTTP에서 HTTPS로 변경한 경우 CloudFront 엣지 로케이션의 객체를 무효화할 것을 권장합니다. CloudFront는 최종 사용자가 사용한 프로토콜(HTTP 또는 HTTPS)이, CloudFront가 객체를 받는 데 사용한 프로토콜과 일치하는지와 상관없이 최종 사용자에게

객체를 반환합니다. 배포에서 객체 제거 또는 대체에 대한 자세한 내용은 CloudFront가 배포하는 콘텐츠 추가, 제거 또는 교체 단원을 참조하십시오.

## 뷰어에 HTTPS 요구

하나 이상의 캐시 동작에 대해 최종 사용자와 CloudFront 간에 HTTPS를 요구하려면 다음과 같이 하세요.

CloudFront에서 최종 사용자와 CloudFront 간에 HTTPS를 요구하도록 구성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. CloudFront 콘솔의 상단 창에서 업데이트할 배포의 ID를 선택합니다.
3. 동작 탭에서 업데이트할 캐시 동작을 선택한 후 편집을 선택합니다.
4. 최종 사용자 프로토콜 정책에 다음 값 중 하나를 지정합니다.

   Redirect HTTP to HTTPS

   최종 사용자가 두 프로토콜을 모두 사용할 수 있습니다. HTTP `GET` 및 `HEAD` 요청은 HTTPS 요청으로 자동 리디렉션됩니다. CloudFront는 HTTP 상태 코드 301(영구 이동됨)을 새로운 HTTPS URL과 함께 반환합니다. 그러면 최종 사용자는 HTTPS URL을 사용하여 이 요청을 CloudFront에 다시 제출합니다.

   > **Important**
   >
   > HTTP를 통해 `POST`, `PUT`, `DELETE`, `OPTIONS` 또는 `PATCH`를 HTTP - HTTPS 캐시 동작 및 HTTP 1.1 이상의 요청 프로토콜 버전으로 보내면, CloudFront는 HTTP 상태 코드 307(임시 리디렉션)과 함께 HTTPS 위치로 요청을 리디렉션합니다. 이렇게 하면 요청이 동일한 메서드와 본문 페이로드를 사용하여 새 위치로 다시 전송됩니다.
   > `POST`, `PUT`, `DELETE`, `OPTIONS` 또는 `PATCH` 요청을 HTTP - HTTPS 캐시 동작을 통해 HTTP 1.1 미만인 요청 프로토콜 버전으로 보내면, CloudFront는 HTTP 상태 코드 403(사용할 수 없음)을 반환합니다.

   최종 사용자가 HTTPS 요청으로 리디렉션되는 HTTP 요청을 만들 경우 CloudFront에서 두 요청 모두에 대해 요금을 부과합니다. HTTP 요청의 경우에는 CloudFront에서 최종 사용자에게

반환되는 요청 및 헤더에만 요금이 부과됩니다. HTTPS 요청의 경우에는 요청, 헤더, 그리고 오리진에 의해 반환된 객체에 대해 요금이 부과됩니다.

HTTPS Only

최종 사용자가 HTTPS를 사용할 경우에만 콘텐츠에 액세스할 수 있습니다. 최종 사용자가 HTTPS 요청 대신에 HTTP 요청을 보내면 CloudFront는 HTTP 상태 코드 403(금지됨)을 반환하고 객체는 반환하지 않습니다.

5. 변경 사항 저장을 선택합니다.
6. 최종 사용자와 CloudFront 간에 HTTPS를 요구하려는 캐시 동작에 대해 3~5단계를 반복합니다.
7. 업데이트한 구성을 프로덕션 환경에서 사용하기 전에 다음 사항을 확인합니다.
   - 각 캐시 동작의 경로 패턴이 최종 사용자에 HTTPS를 사용하도록 지정한 요청에만 적용되는가.
   - 캐시 동작이 CloudFront에서 평가하도록 할 순서대로 나열되었는가. 자세한 내용은 경로 패턴 단원을 참조하세요.
   - 캐시 동작이 올바른 오리진에 요청을 라우팅하는가.

## CloudFront와 사용자 지정 오리진 간의 통신에 HTTPS 요구

CloudFront와 오리진 간의 통신에 HTTPS를 요청할 수 있습니다.

> **Note**
> 오리진이 웹 사이트 엔드포인트로 구성된 Amazon S3 버킷인 경우, Amazon S3에서 웹 사이트 엔드포인트로 HTTPS를 지원하지 않으므로 오리진에 HTTPS를 사용하도록 CloudFront를 구성할 수 없습니다.

CloudFront와 오리진 간에 HTTPS를 요청하려면 이 항목의 절차에 따라 다음을 수행하세요.

1. 배포에서 오리진에 대한 오리진 프로토콜 정책 설정을 변경합니다.
2. 오리진 서버에 SSL/TLS 인증서를 설치합니다(Amazon S3 오리진 또는 기타 특정 AWS 오리진을 사용하는 경우 필요 없음).

주제


- 사용자 지정 오리진에 HTTPS 요구
- 사용자 지정 오리진에 SSL/TLS 인증서 설치

## 사용자 지정 오리진에 HTTPS 요구

다음 절차는 CloudFront에서 HTTPS를 사용하여 Elastic Load Balancing 로드 밸런서, Amazon EC2 인스턴스 또는 다른 사용자 지정 오리진과 통신하도록 구성하는 방법을 보여 줍니다. CloudFront API를 사용한 배포 업데이트에 대한 자세한 내용은 Amazon CloudFront API 참조의 배포 업데이트를 참조하십시오.

CloudFront에서 CloudFront와 사용자 지정 오리진 간에 HTTPS를 요구하도록 구성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. CloudFront 콘솔의 상단 창에서 업데이트할 배포의 ID를 선택합니다.
3. 동작 탭에서 업데이트할 오리진을 선택하고 편집을 선택합니다.
4. 다음 설정을 업데이트합니다.

   오리진 프로토콜 정책

   배포의 해당 오리진에 대해 오리진 프로토콜 정책을 변경합니다.

   - HTTPS만(HTTPS Only) - CloudFront에서 사용자 지정 오리진과 통신할 때 HTTPS만 사용합니다.
   - 최종 사용자와 일치(Match Viewer) - CloudFront에서 최종 사용자 요청의 프로토콜에 따라 HTTP 또는 HTTPS를 사용하여 사용자 지정 오리진과 통신합니다. 예를 들면 오리진 프로토콜 정책(Origin Protocol Policy)에 최종 사용자와 일치(Match Viewer)를 선택하고 최종 사용자가 HTTPS를 사용하여 CloudFront에서 객체를 요청하는 경우, CloudFront에서도 HTTPS를 사용하여 해당 요청을 오리진으로 전달합니다.

     Viewer Protocol Policy(최종 사용자 프로토콜 정책)에서 Redirect HTTP to HTTPS(HTTP를 HTTPS로 재지정) 또는 HTTPS Only(HTTPS만 해당)를 지정한 경우에만 Match Viewer를 선택합니다.

     최종 사용자가 HTTP 및 HTTPS 프로토콜 모두를 사용하여 요청하더라도 CloudFront에서는 한 번만 객체를 캐싱합니다.

   오리진 SSL 프로토콜

   배포의 해당 오리진에 대해 오리진 SSL 프로토콜을 선택합니다. SSLv3 프로토콜이 덜 안전하므로 사용자가 TLSv1 이상을 지원하지 않는 경우에만 SSLv3을 선택하는 것이 좋습니다. TLSv1 핸드쉐이크는 SSLv3의 이전 버전 및 이후 버전과 모두 호환되지만, TLSv1.1 이상은 호환되지 않습니다. SSLv3을 선택하면 CloudFront는 SSLv3 핸드셰이크 요청만 보냅니다.

5. 변경 사항 저장을 선택합니다.
6. CloudFront와 사용자 지정 오리진 간에 HTTPS를 요구하려는 다른 오리진에 대해 3~5단계를 반복합니다.
7. 업데이트한 구성을 프로덕션 환경에서 사용하기 전에 다음 사항을 확인합니다.
   - 각 캐시 동작의 경로 패턴이 최종 사용자에 HTTPS를 사용하도록 지정한 요청에만 적용되는가.
   - 캐시 동작이 CloudFront에서 평가하도록 할 순서대로 나열되었는가. 자세한 내용은 경로 패턴 단원을 참조하세요.
   - 캐시 동작은 변경한 오리진 프로토콜 정책에 따라 요청을 오리진으로 라우팅합니다.

## 사용자 지정 오리진에 SSL/TLS 인증서 설치

사용자 지정 오리진에서 다음 소스의 SSL/TLS 인증서를 사용할 수 있습니다.

- 오리진이 Elastic Load Balancing 로드 밸런서인 경우 AWS Certificate Manager(ACM)에서 제공하는 인증서를 사용할 수 있습니다. 또한 신뢰할 수 있는 다른 인증 기관에서 서명한 인증서 및 ACM으로 가져온 인증서를 사용할 수도 있습니다.
- Elastic Load Balancing 로드 밸런서 외의 오리진에서는 Comodo, DigiCert, Symantec 같이 신뢰할 수 있는 서드 파티 인증 기관(CA)에서 서명한 인증서를 사용해야 합니다.

오리진에서 반환되는 인증서는 다음 도메인 이름 중 하나를 포함하고 있어야 합니다.

- 오리진의 오리진 도메인(Origin domain) 필드(CloudFront API의 `DomainName` 필드)에 있는 도메인 이름.
- 캐시 동작이 `Host` 헤더를 오리진으로 전달하도록 구성된 경우, `Host` 헤더의 도메인 이름.

CloudFront가 HTTPS를 사용하여 오리진과 통신하면 CloudFront는 인증서가 신뢰할 수 있는 인증 기관에 의해 발급되었는지 확인합니다. CloudFront는 Mozilla가 지원하는 것과 동일한 인증 기관을 지원합니다. 최신 목록은 Mozilla에 포함된 CA 인증서 목록을 참조하십시오. CloudFront와 오리진 간의 HTTPS 통신에는 자체 서명한 인증서를 사용할 수 없습니다.

> **Important**
>
> 오리진 서버에서 만료된 인증서, 잘못된 인증서 또는 자체 서명된 인증서를 반환하는 경우 또는 오리진 서버에서 잘못된 순서로 인증서 체인을 반환하는 경우에는 CloudFront에서 TCP 연결을 끊고 HTTP 상태 코드 502(잘못된 게이트웨이)를 뷰어로 반환하며 `X-Cache` 헤더를

`Error from cloudfront`로 설정합니다. 또한 중간 인증서를 포함하여 인증서의 전체 체인이 없는 경우, CloudFront는 TCP 연결을 끊습니다.

## CloudFront와 Amazon S3 오리진 간의 통신에 HTTPS 요구

오리진이 Amazon S3 버킷인 경우 CloudFront와 통신을 위해 HTTPS를 사용하는 옵션은 버킷을 사용하는 방식에 따라 달라집니다. 웹 사이트 엔드포인트로 Amazon S3 버킷이 구성되어 있는 경우, Amazon S3에서 해당 구성으로 HTTPS 연결을 지원하지 않으므로 HTTPS를 사용하여 오리진과 통신하도록 CloudFront를 구성할 수 없습니다.

오리진이 HTTPS 통신을 지원하는 Amazon S3 버킷일 경우 CloudFront는 뷰어가 요청을 전송하는 데 사용된 프로토콜을 사용하여 S3로 요청을 전달합니다. 프로토콜(사용자 지정 오리진만 해당) 설정의 기본 설정은 최종 사용자와 일치(Match Viewer)이며 변경할 수 없습니다. 그러나 Amazon S3 오리진에 대해 오리진 액세스 제어(OAC)를 사용 설정하면 CloudFront와 Amazon S3 간에 사용되는 통신은 사용자가 어떻게 설정하는지에 따라 달라집니다. 자세한 내용은 새 오리진 액세스 제어 생성 섹션을 참조하세요.

CloudFront와 Amazon S3 간의 통신에 HTTPS를 요구하려면, 최종 사용자 프로토콜 정책(Viewer Protocol Policy) 값을 Redirect HTTP to HTTPS(HTTP를 HTTPS로 재지정) 또는 HTTPS Only(HTTPS만)로 변경해야 합니다. 이 단원의 나머지 절차에서는 CloudFront 콘솔을 사용하여 최종 사용자 프로토콜 정책(Viewer Protocol Policy) 값을 변경하는 방법을 설명합니다. CloudFront API를 사용하여 배포용 `ViewerProtocolPolicy` 요소를 업데이트하는 자세한 내용은 Amazon CloudFront API 참조의 UpdateDistribution을 참조하세요.

HTTPS 통신을 지원하는 Amazon S3 버킷과 함께 HTTPS를 사용하면 Amazon S3가 SSL/TLS 인증서를 제공하므로 사용자가 제공할 필요가 없습니다.

### Amazon S3 오리진에 대해 HTTPS 요구

다음 절차에서는 CloudFront에서 Amazon S3 오리진에 대해 HTTPS를 요구하도록 구성하는 방법을 알아봅니다.

CloudFront에서 Amazon S3 오리진에 대해 HTTPS를 요구하도록 구성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. CloudFront 콘솔의 상단 창에서 업데이트할 배포의 ID를 선택합니다.

3. 동작 탭에서 업데이트할 캐시 동작을 선택한 후 편집을 선택합니다.
4. 최종 사용자 프로토콜 정책에 다음 값 중 하나를 지정합니다.

   Redirect HTTP to HTTPS

   최종 사용자가 두 프로토콜 모두 사용할 수 있지만, HTTP 요청은 자동으로 HTTPS 요청으로 리디렉션됩니다. CloudFront는 HTTP 상태 코드 301(영구 이동됨)을 새로운 HTTPS URL과 함께 반환합니다. 그러면 최종 사용자는 HTTPS URL을 사용하여 이 요청을 CloudFront에 다시 제출합니다.

   > **Important**
   >
   > CloudFront는 HTTP에서 HTTPS로 `DELETE`, `OPTIONS`, `PATCH`, `POST` 또는 `PUT` 요청을 리디렉션하지 않습니다. HTTPS에 리디렉션할 캐시 동작을 구성하는 경우, CloudFront는 해당 캐시 동작에 대한 HTTP `DELETE`, `OPTIONS`, `PATCH`, `POST`, 또는 `PUT` 요청에 HTTP 상태 코드 403(금지됨)으로 응답합니다.

   최종 사용자가 HTTPS 요청으로 리디렉션되는 HTTP 요청을 만들 경우 CloudFront에서 두 요청 모두에 대해 요금을 부과합니다. HTTP 요청의 경우에는 CloudFront에서 최종 사용자에게 반환되는 요청 및 헤더에만 요금이 부과됩니다. HTTPS 요청의 경우에는 요청, 헤더, 그리고 오리진에 의해 반환된 객체 모두에 요금이 부과됩니다.

   HTTPS Only

   최종 사용자가 HTTPS를 사용할 경우에만 콘텐츠에 액세스할 수 있습니다. 최종 사용자가 HTTPS 요청 대신에 HTTP 요청을 보내면 CloudFront는 HTTP 상태 코드 403(금지됨)을 반환하고 객체는 반환하지 않습니다.
5. 예, 편집합니다를 선택합니다.
6. 최종 사용자와 CloudFront 간에 그리고 CloudFront와 S3 간에 HTTPS를 요구하려는 캐시 동작에 대해 3~5단계를 반복합니다.
7. 업데이트한 구성을 프로덕션 환경에서 사용하기 전에 다음 사항을 확인합니다.
   - 각 캐시 동작의 경로 패턴이 최종 사용자에 HTTPS를 사용하도록 지정한 요청에만 적용되는가.
   - 캐시 동작이 CloudFront에서 평가하도록 할 순서대로 나열되었는가. 자세한 내용은 경로 패턴 단원을 참조하세요.
   - 캐시 동작이 올바른 오리진에 요청을 라우팅하는가.

# 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호

최종 사용자와 CloudFront 배포 간에 HTTPS가 필요한 경우 보안 정책을 선택해야 하며 이는 다음 설정을 결정합니다.

- CloudFront가 최종 사용자와 통신하는 데 사용하는 최소 SSL/TLS 프로토콜.
- CloudFront가 최종 사용자와의 통신을 암호화할 때 사용할 수 있는 암호

보안 정책을 선택하려면 보안 정책(최소 SSL/TLS 버전)에 해당 값을 지정합니다. 다음 표에는 CloudFront가 각 보안 정책에 사용할 수 있는 프로토콜 및 암호가 나와 있습니다.

최종 사용자는 지원되는 암호 중 하나 이상을 지원하여 CloudFront와의 HTTPS 연결을 설정해야 합니다. CloudFront에서는 최종 사용자가 지원하는 암호 중에서 나열된 순서대로 암호를 선택합니다. 또한 OpenSSL, s2n 및 RFC 암호 이름 단원도 참조하십시오.

| | 보안 정책 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | SSLv3 | TLSv1 | TLSv1_6 | TLSv1.016 | TLSv1.018 | TLSv1.019 | TLSv1.021 | TLSv1.025 | TLSv1.3_2025 |
| 지원되는 SSL/TLS 프로토콜 | | | | | | | | | |
| TLSv1.3 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLSv1.2 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | |
| TLSv1.1 | ◆ | ◆ | ◆ | ◆ | | | | | |
| TLSv1 | ◆ | ◆ | ◆ | | | | | | |
| SSLv3 | ◆ | | | | | | | | |
| 지원되는 TLSv1.3 암호 | | | | | | | | | |
| TLS_AES_128_GCM_SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_AES_256_GCM_SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |

| | 보안 정책 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | SSLv3 | TLSv1 | TLSv1_6 | TLSv1.016 | TLSv1.018 | TLSv1.019 | TLSv1.021 | TLSv1.025 | TLSv1.3_2025 |
| TLS_CHACHA20_POLY1305_SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | | ◆ |
| 지원되는 ECDSA 암호 | | | | | | | | | |
| ECDHE-ECDSA-AES128-GCM-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | |
| ECDHE-ECDSA-AES128-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | | | |
| ECDHE-ECDSA-AES128-SHA | ◆ | ◆ | ◆ | ◆ | | | | | |
| ECDHE-ECDSA-AES256-GCM-SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | |
| ECDHE-ECDSA-CHACHA20-POLY1305 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | | |
| ECDHE-ECDSA-AES256-SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | | | |
| ECDHE-ECDSA-AES256-SHA | ◆ | ◆ | ◆ | ◆ | | | | | |
| 지원되는 RSA 암호 | | | | | | | | | |

| | 보안 정책 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | SSLv3 | TLSv1 | TLSv1_6 | TLSv1.016 | TLSv1.018 | TLSv1.019 | TLSv1.021 | TLSv1.025 | TLSv1.3_2025 |
| ECDHE-RSA-AES128-GCM-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | |
| ECDHE-RSA-AES128-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | | | |
| ECDHE-RSA-AES128-SHA | ◆ | ◆ | ◆ | ◆ | | | | | |
| ECDHE-RSA-AES256-GCM-SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | |
| ECDHE-RSA-CHACHA20-POLY1305 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | | |
| ECDHE-RSA-AES256-SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | | | |
| ECDHE-RSA-AES256-SHA | ◆ | ◆ | ◆ | ◆ | | | | | |
| AES128-GCM-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | | | | |
| AES256-GCM-SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | | | | |
| AES128-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | | | | |
| AES256-SHA | ◆ | ◆ | ◆ | ◆ | | | | | |
| AES128-SHA | ◆ | ◆ | ◆ | ◆ | | | | | |

| | 보안 정책 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | SSLv3 | TLSv1 | TLSv1_6 | TLSv1.016 | TLSv1.018 | TLSv1.019 | TLSv1.021 | TLSv1.025 | TLSv1.3_2025 |
| DES-CBC3-SHA | ♦ | ♦ | | | | | | | |
| RC4-MD5 | ♦ | | | | | | | | |

## OpenSSL, s2n 및 RFC 암호 이름

OpenSSL 및 s2n은 TLS 표준에서 사용하는 암호 이름과 다른 이름을 사용합니다(RFC 2246, RFC 4346, RFC 5246 및 RFC 8446). 다음 표에는 각 암호의 RFC 이름에 OpenSSL 및 s2n 이름이 매핑되어 있습니다.

CloudFront는 클래식 키 교환과 양자 보안 키 교환을 모두 지원합니다. 타원 곡선을 사용하는 클래식 키 교환의 경우 CloudFront는 다음을 지원합니다.

- `prime256v1`
- `X25519`
- `secp384r1`

양자 보안 키 교환의 경우 CloudFront는 다음을 지원합니다.

- `X25519MLKEM768`
- `SecP256r1MLKEM768`

> **Note**
>
> 양자 보안 키 교환은 TLS 1.3에서만 지원됩니다. TLS 1.2 및 이전 버전은 양자 보안 키 교환을 지원하지 않습니다.

자세한 내용은 다음 항목을 참조하세요.

- 양자 내성 암호
- 암호화 알고리즘 및 AWS 서비스
- TLS 1.3의 하이브리드 키 교환

CloudFront의 인증서 요구 사항에 대한 자세한 내용은 CloudFront에서 SSL/TLS 인증서를 사용하기 위한 요구 사항 섹션을 참조하시기 바랍니다.

| OpenSSL 및 s2n 암호 이름 | RFC 암호화 이름 |
| --- | --- |
| 지원되는 TLSv1.3 암호 | |
| TLS_AES_128_GCM_SHA256 | TLS_AES_128_GCM_SHA256 |
| TLS_AES_256_GCM_SHA384 | TLS_AES_256_GCM_SHA384 |
| TLS_CHACHA20_POLY1305_SHA256 | TLS_CHACHA20_POLY1305_SHA256 |
| 지원되는 ECDSA 암호 | |
| ECDHE-ECDSA-AES128-GCM-SHA256 | TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 |
| ECDHE-ECDSA-AES128-SHA256 | TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 |
| ECDHE-ECDSA-AES128-SHA | TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA |
| ECDHE-ECDSA-AES256-GCM-SHA384 | TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 |
| ECDHE-ECDSA-CHACHA20-POLY1305 | TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 |
| ECDHE-ECDSA-AES256-SHA384 | TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 |
| ECDHE-ECDSA-AES256-SHA | TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA |
| 지원되는 RSA 암호 | |
| ECDHE-RSA-AES128-GCM-SHA256 | TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 |

| OpenSSL 및 s2n 암호 이름 | RFC 암호화 이름 |
| --- | --- |
| ECDHE-RSA-AES128-SHA256 | TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 |
| ECDHE-RSA-AES128-SHA | TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA |
| ECDHE-RSA-AES256-GCM-SHA384 | TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 |
| ECDHE-RSA-CHACHA20-POLY1305 | TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 |
| ECDHE-RSA-AES256-SHA384 | TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 |
| ECDHE-RSA-AES256-SHA | TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA |
| AES128-GCM-SHA256 | TLS_RSA_WITH_AES_128_GCM_SHA256 |
| AES256-GCM-SHA384 | TLS_RSA_WITH_AES_256_GCM_SHA384 |
| AES128-SHA256 | TLS_RSA_WITH_AES_128_CBC_SHA256 |
| AES256-SHA | TLS_RSA_WITH_AES_256_CBC_SHA |
| AES128-SHA | TLS_RSA_WITH_AES_128_CBC_SHA |
| DES-CBC3-SHA | TLS_RSA_WITH_3DES_EDE_CBC_SHA |
| RC4-MD5 | TLS_RSA_WITH_RC4_128_MD5 |

## 최종 사용자와 CloudFront 간에 지원되는 서명 체계

CloudFront에서는 최종 사용자와 CloudFront 간의 연결을 위해 다음과 같은 서명 체계를 지원합니다.

| | 보안 정책 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 서명 스키마 | SSLv3 | TLSv1 | TLSv1_6 | TLSv1.016 | TLSv1.018 | TLSv1.019 | TLSv1.021 | TLSv1.025 | TLSv1.3_2025 |
| TLS_SIGNATURE_SCHEME_RSA_PSS_PSS_SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_RSA_PSS_PSS_SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_RSA_PSS_PSS_SHA512 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_RSA_PSS_RSAE_SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_RSA_PSS_RSAE_SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_RSA_PSS_RSAE_SHA512 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHE | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |

| | 보안 정책 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 서명 스키마 | SSLv3 | TLSv1 | TLSv1_6 | TLSv1.016 | TLSv1.018 | TLSv1.019 | TLSv1.021 | TLSv1.025 | TLSv1.3_2025 |
| ME_RSA_PKCS1_SHA256 | | | | | | | | | |
| TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA512 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA224 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | | |
| TLS_SIGNATURE_SCHEME_ECDSA_SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_ECDSA_SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_ECDSA_SHA512 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_ECDSA_SHA224 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | | |

| | 보안 정책 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 서명 스키마 | SSLv3 | TLSv1 | TLSv1_6 | TLSv1.016 | TLSv1.018 | TLSv1.019 | TLSv1.021 | TLSv1.025 | TLSv1.3_2025 |
| TLS_SIGNATURE_SCHEME_ECDSA_SECP256R1_SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_ECDSA_SECP384R1_SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA1 | ◆ | ◆ | ◆ | ◆ | | | | | |
| TLS_SIGNATURE_SCHEME_ECDSA_SHA1 | ◆ | ◆ | ◆ | ◆ | | | | | |

## CloudFront와 오리진 간에 지원되는 프로토콜 및 암호

CloudFront와 오리진 간에 HTTPS 필요(require HTTPS between CloudFront and your origin)를 선택하면 보안 연결을 허용하기 위한 SSL/TLS 프로토콜을 결정할 수 있으며 CloudFront가 아래 표에 나열된 ECDSA 또는 RSA 암호를 사용하여 오리진에 연결할 수 있습니다. 오리진에 대해 HTTPS 연결을 설정하려면 오리진이 CloudFront에 대해 이들 암호 중 하나 이상을 지원해야 합니다.

OpenSSL 및 s2n은 TLS 표준에서 사용하는 암호 이름과 다른 이름을 사용합니다(RFC 2246, RFC 4346, RFC 5246 및 RFC 8446). 다음 표에는 각 암호의 OpenSSL 및 s2n 이름, RFC 이름이 포함되어 있습니다.

타원 곡선 키 교환 알고리즘이 포함된 암호의 경우 CloudFront는 다음과 같은 타원 곡선을 지원합니다.

- prime256v1

- secp384r1
- X25519

| OpenSSL 및 s2n 암호 이름 | RFC 암호화 이름 |
| --- | --- |
| 지원되는 ECDSA 암호 | |
| ECDHE-ECDSA-AES256-GCM-SHA384 | TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 |
| ECDHE-ECDSA-AES256-SHA384 | TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 |
| ECDHE-ECDSA-AES256-SHA | TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA |
| ECDHE-ECDSA-AES128-GCM-SHA256 | TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 |
| ECDHE-ECDSA-AES128-SHA256 | TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 |
| ECDHE-ECDSA-AES128-SHA | TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA |
| 지원되는 RSA 암호 | |
| ECDHE-RSA-AES256-GCM-SHA384 | TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 |
| ECDHE-RSA-AES256-SHA384 | TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 |
| ECDHE-RSA-AES256-SHA | TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA |
| ECDHE-RSA-AES128-GCM-SHA256 | TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 |

| OpenSSL 및 s2n 암호 이름 | RFC 암호화 이름 |
| --- | --- |
| ECDHE-RSA-AES128-SHA256 | TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 |
| ECDHE-RSA-AES128-SHA | TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA |
| AES256-SHA | TLS_RSA_WITH_AES_256_CBC_SHA |
| AES128-SHA | TLS_RSA_WITH_AES_128_CBC_SHA |
| DES-CBC3-SHA | TLS_RSA_WITH_3DES_EDE_CBC_SHA |
| RC4-MD5 | TLS_RSA_WITH_RC4_128_MD5 |

CloudFront와 오리진 간에 지원되는 서명 체계

CloudFront에서는 CloudFront와 오리진 간의 연결을 위해 다음과 같은 서명 체계를 지원합니다.

- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA256
- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA384
- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA512
- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA224
- TLS_SIGNATURE_SCHEME_ECDSA_SHA256
- TLS_SIGNATURE_SCHEME_ECDSA_SHA384
- TLS_SIGNATURE_SCHEME_ECDSA_SHA512
- TLS_SIGNATURE_SCHEME_ECDSA_SHA224
- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA1
- TLS_SIGNATURE_SCHEME_ECDSA_SHA1

# 대체 도메인 이름과 HTTPS 사용

파일의 URL에 자체 도메인 이름(예: `https://www.example.com/image.jpg`)을 사용하고 최종 사용자가 HTTPS를 사용하게 하려면 다음 주제에 나온 단계를 수행해야 합니다. (기본 CloudFront 배

포 도메인 이름을 URL에 사용하는 경우에는(예: `https://d111111abcdef8.cloudfront.net/image.jpg`) 뷰어와 CloudFront 간의 통신에 HTTPS 요구 주제의 지침을 따르세요.)

> **Important**
>
> 인증서를 배포에 추가하는 경우 CloudFront는 인증서를 모든 엣지 로케이션으로 즉시 전파합니다. 새 엣지 로케이션을 사용할 수 있게 되면 CloudFront에서는 인증서를 그러한 새 위치로도 전파합니다. CloudFront에서 인증서를 전파하는 엣지 로케이션을 제한할 수는 없습니다.

주제


- CloudFront에서 HTTPS 요청을 제공하는 방식 선택
- CloudFront에서 SSL/TLS 인증서를 사용하기 위한 요구 사항
- CloudFront에서 SSL/TLS 인증서 사용 시의 할당량(뷰어와 CloudFront 간의 HTTPS만 해당)
- 대체 도메인 이름과 HTTPS 구성
- SSL/TLS RSA 인증서에서 퍼블릭 키 크기 확인
- SSL/TLS 인증서에 대한 할당량 증가
- SSL/TLS 인증서 교체
- 사용자 지정 SSL/TLS 인증서에서 기본 CloudFront 인증서로 되돌리기
- 사용자 지정 SSL/TLS 인증서를 전용 IP 주소 사용에서 SNI 사용으로 전환


## CloudFront에서 HTTPS 요청을 제공하는 방식 선택

최종 사용자가 HTTPS를 사용하고 파일에 대체 도메인 이름을 사용하게 하려면 CloudFront에서 HTTPS 요청을 제공하는 방식에 대해 다음 옵션 중 하나를 선택합니다.

- 서버 이름 표시(SNI) 사용 – 권장
- 각 엣지 로케이션에서 전용 IP 주소 사용

이 단원에서는 각 옵션이 어떻게 작동하는지를 설명합니다.

### SNI를 사용하여 HTTPS 요청 제공(대부분 클라이언트에 적용)

서버 이름 표시(SNI)는 2010년 이후 출시된 브라우저와 클라이언트에서 지원되는 TLS 프로토콜의 확장 기능입니다. SNI를 사용하여 HTTPS 요청을 제공하도록 CloudFront를 구성한 경우, CloudFront에서는 대체 도메인 이름을 각 엣지 로케이션의 IP 주소와 연결합니다. 최종 사용자가 HTTPS 콘텐츠 요

청을 제출하면 DNS는 이 요청을 올바른 엣지 로케이션의 IP 주소로 라우팅합니다. 도메인 이름의 IP 주소는 SSL/TLS 핸드셰이크 협상 중에 결정됩니다(IP 주소는 배포 전용이 아님).

HTTPS 연결 설정 과정 초기에 SSL/TLS 협상이 발생합니다. CloudFront에서 요청과 관련된 도메인을 즉시 확인할 수 없으면 연결이 끊어집니다. SNI를 지원하는 최종 사용자가 콘텐츠에 대해 HTTPS 요청을 전송할 경우 다음과 같은 작업이 수행됩니다.

1. 최종 사용자는 요청 URL에서 도메인 이름을 자동으로 가져와서 SNI 확장 또는 TLS 클라이언트 hello 메시지에 추가합니다.
2. CloudFront가 TLS 클라이언트 hello를 수신하면 SNI 확장의 도메인 이름을 사용하여 일치하는 CloudFront 배포를 찾고 연결된 TLS 인증서를 다시 보냅니다.
3. 최종 사용자와 CloudFront는 SSL/TLS 협상을 수행합니다.
4. CloudFront;가 요청된 콘텐츠를 최종 사용자에 반환합니다.

SNI를 지원하는 최신 브라우저 목록은 Wikipedia 항목인 Server Name Indication을 참조하세요.

SNI를 사용하고자 하지만 사용자 브라우저 중 일부에서 SNI가 지원되지 않는 경우 몇 가지 옵션 중 하나를 사용할 수 있습니다.

- SNI 대신에 전용 IP 주소를 사용하여 HTTPS 요청을 제공하도록 CloudFront;를 구성합니다. 자세한 내용은 전용 IP 주소를 사용하여 HTTPS 요청 제공(모든 클라이언트에 적용) 단원을 참조하세요.
- 사용자 지정 인증서 대신에 CloudFront SSL/TLS 인증서를 사용합니다. 그러기 위해서는 파일에 대한 URL에서 배포의 CloudFront 도메인 이름을 사용해야 합니다. 예: `https://d111111abcdef8.cloudfront.net/logo.png`.

  기본 CloudFront 인증서를 사용할 경우 최종 사용자가 SSL 프로토콜 TLSv1 또는 그 이상 버전을 지원해야 합니다. CloudFront는 기본 CloudFront 인증서로 SSLv3를 지원하지 않습니다.

  또한 CloudFront에서 사용 중인 SSL/TLS 인증서를 사용자 지정 인증서에서 기본 CloudFront 인증서로 다음과 같이 변경해야 합니다.

  - 배포를 사용하여 콘텐츠를 배포하지 않은 경우에는 구성을 변경하면 됩니다. 자세한 내용은 배포 업데이트 단원을 참조하세요.
  - 배포를 사용하여 콘텐츠를 배포한 경우에는 새 CloudFront 배포를 생성하고 파일에 대한 URL을 변경하여 콘텐츠를 사용할 수 없는 시간을 줄이거나 없애야 합니다. 자세한 내용은 사용자 지정 SSL/TLS 인증서에서 기본 CloudFront 인증서로 되돌리기 단원을 참조하세요.
- 사용자가 사용하는 브라우저를 개발자가 제어할 수 있는 경우 사용자가 해당 브라우저를 SNI가 지원되는 브라우저로 업그레이드하게 합니다.

- HTTPS 대신에 HTTP를 사용합니다.

## 전용 IP 주소를 사용하여 HTTPS 요청 제공(모든 클라이언트에 적용)

서버 이름 표시(SNI)는 요청을 도메인과 연결하는 한 가지 방법입니다. 또 다른 방법으로는 전용 IP 주소 사용을 들 수 있습니다. 2010년 이후 출시 브라우저나 클라이언트로 업그레이드가 불가한 사용자가 있다면 전용 IP 주소를 사용하여 HTTPS 요청을 제공할 수 있습니다. SNI를 지원하는 최신 브라우저 목록은 Wikipedia 항목인 Server Name Indication을 참조하세요.

> **Important**
>
> 전용 IP 주소를 사용하여 HTTPS 요청을 제공하도록 CloudFront를 구성하면 추가 월별 요금이 발생합니다. 요금 발생은 SSL/TLS 인증서를 배포와 연결하고 배포를 활성화할 때 시작됩니다. CloudFront 요금에 대한 자세한 내용은 Amazon CloudFront 요금을 참조하세요. 또한 Using the Same Certificate for Multiple CloudFront Distributions 단원을 참조하세요.

전용 IP 주소를 사용하여 HTTPS 요청을 제공하도록 CloudFront를 구성한 경우 CloudFront에서는 인증서를 각 CloudFront 엣지 로케이션의 전용 IP 주소와 연결합니다. 최종 사용자가 콘텐츠에 대해 HTTPS 요청을 전송할 경우 다음과 같은 작업이 수행됩니다.

1. DNS에서 배포의 해당 엣지 로케이션에 대한 IP 주소로 요청을 라우팅합니다.
2. 클라이언트 요청이 `ClientHello` 메시지에 SNI 확장을 제공하는 경우 CloudFront는 해당 SNI와 연결된 배포를 검색합니다.
   - 일치하는 항목이 있는 경우 CloudFront는 SSL/TLS 인증서를 사용하여 요청에 응답합니다.
   - 일치하는 항목이 없는 경우, CloudFront에서는 대신 이 IP 주소를 사용하여 배포를 식별하고 최종 사용자에게 반환할 SSL/TLS 인증서를 결정합니다.
3. 최종 사용자와 CloudFront가 SSL/TLS 인증서를 사용하여 SSL/TLS 협상을 수행합니다.
4. CloudFront;가 요청된 콘텐츠를 최종 사용자에 반환합니다.

이 방법은 사용자가 사용 중인 브라우저나 기타 최종 사용자와 상관없이 모든 HTTPS 요청에 작동합니다.

> **Note**
>
> 전용 IP는 고정 IP가 아니며 시간이 지남에 따라 변경될 수 있습니다. 엣지 로케이션에 대해 반환되는 IP 주소는 CloudFront 엣지 서버 목록의 IP 주소 범위에서 동적으로 할당됩니다.

CloudFront 엣지 서버의 IP 주소 범위는 변경될 수 있습니다. IP 주소 변경 사항에 대한 알림을 받으려면 Amazon SNS를 통해 AWS 퍼블릭 IP 주소 변경 사항을 구독하세요.

### 3개 이상의 전용 IP SSL/TLS 인증서를 사용할 수 있는 권한 요청

CloudFront에 3개 이상의 SSL/TLS 전용 IP 인증서를 영구적으로 연결할 수 있는 권한이 필요할 경우 다음과 같이 합니다. HTTPS 요청에 대한 자세한 내용은 CloudFront에서 HTTPS 요청을 제공하는 방식 선택 단원을 참조하세요.

> **Note**
>
> 이 절차는 CloudFront 배포에 3개 이상의 전용 IP 인증서를 사용하는 경우에 사용됩니다. 기본값은 2입니다. 배포에 2개 이상의 SSL 인증서를 바인딩할 수 없다는 점에 유의하세요.
> 한 번에 하나의 SSL/TLS 인증서만 CloudFront 배포에 연결할 수 있습니다. 이 수치는 모든 CloudFront 배포에서 사용할 수 있는 전용 IP SSL 인증서의 총 개수에 해당됩니다.

CloudFront 배포에 3개 이상의 인증서를 사용할 수 있는 권한을 요청하려면

1. 지원 센터로 이동하여 사례를 생성합니다.
2. 사용 권한이 필요한 인증서 수를 지정하고 요청 상황을 설명하세요. 그러고 나면 귀하의 계정을 가능한 빨리 업데이트하겠습니다.
3. 다음 절차로 진행합니다.

## CloudFront에서 SSL/TLS 인증서를 사용하기 위한 요구 사항

SSL/TLS 인증서에 대한 요구 사항은 이 주제에 설명되어 있습니다. 별도로 명시되지 않는 한, 다음 두 가지에 모두 적용됩니다.

- 최종 사용자와 CloudFront 간에 HTTPS를 사용하기 위한 인증서
- CloudFront와 오리진 간에 HTTPS를 사용하기 위한 인증서

주제

- 인증서 발행자
- AWS 리전 for AWS Certificate Manager

- 인증서 형식
- 중간 인증서
- 키 유형
- 프라이빗 키
- 권한
- 인증서 키의 크기
- 지원되는 인증서 유형
- 인증서 만료 날짜 및 갱신
- CloudFront 배포 및 인증서의 도메인 이름
- 최소 SSL/TLS 프로토콜 버전
- 지원되는 HTTP 버전

## 인증서 발행자

AWS Certificate Manager(ACM)에서 발급된 퍼블릭 인증서를 사용하는 것이 좋습니다. ACM에서 인증서를 받는 방법에 대한 자세한 내용은 AWS Certificate Manager 사용 설명서를 참조하세요. CloudFront 배포에서 ACM 인증서를 사용하려면 미국 동부(버지니아 북부) 리전(`us-east-1`)에서 인증서를 요청하거나 가져와야 합니다.

CloudFront는 Mozilla와 동일한 인증 기관(CA)를 지원하므로 ACM을 사용하지 않는 경우Mozilla Included CA Certificate List에 등록된 CA에서 발행한 인증서를 사용하세요.

CloudFront 배포에 지정한 오리진에서 사용하는 TLS 인증서도 Mozilla 포함 CA 인증서 목록의 CA에서 발급해야 합니다.

인증서 가져오기 및 설치에 대한 자세한 내용은 HTTP 서버 소프트웨어 설명서와 CA 설명서를 참조하세요.

## AWS 리전 for AWS Certificate Manager

최종 사용자와 CloudFront 간에 HTTPS를 요구하기 위해 AWS Certificate Manager(ACM) 인증서를 사용하려면 미국 동부(버지니아 북부) 리전(`us-east-1`)에서 인증서를 요청하거나 가져와야 합니다.

CloudFront와 오리진 간에 HTTPS가 필요하고 오리진으로 Elastic Load Balancing 로드 밸런서를 사용하는 경우 모든 AWS 리전에서 인증서를 요청하거나 가져올 수 있습니다.

## 인증서 형식

인증서는 X.509 PEM 형식이어야 합니다. AWS Certificate Manager를 사용할 경우 이 형식이 기본 형식입니다.

## 중간 인증서

서드 파티 인증 기관(CA)을 사용할 경우, 도메인 인증서에 서명한 CA 관련 인증서부터, `.pem` 파일에 있는 인증서 체인의 모든 중간 인증서를 나열합니다. 일반적으로 올바른 체인 순서대로 중간 및 루트 인증서를 나열하는 파일이 CA 웹 사이트에 있습니다.

> **Important**
>
> 루트 인증서, 신뢰 경로에 없는 중간 인증서 또는 CA의 퍼블릭 키 인증서는 포함하지 마세요.

다음은 그 예입니다:

```
-----BEGIN CERTIFICATE-----
Intermediate certificate 2
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Intermediate certificate 1
-----END CERTIFICATE-----
```

## 키 유형

CloudFront는 RSA 및 ECDSA 퍼블릭/프라이빗 키 페어를 지원합니다.

CloudFront는 RSA 및 ECDSA 인증서를 사용하여 최종 사용자와 오리진 모두에 대한 HTTPS 연결을 지원합니다. AWS Certificate Manager(ACM)을 사용하면 RSA 인증서 및 ECDSA 인증서를 요청하고 가져올 수 있으며 그 다음 CloudFront 배포와 연결할 수 있습니다.

HTTPS 연결에서 협상할 수 있는 RSA 및 ECDSA 암호 목록(CloudFront 지원)은 the section called “최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호” 및 the section called “CloudFront와 오리진 간에 지원되는 프로토콜 및 암호” 단원을 참조하세요.

## 프라이빗 키

다른 인증 기관(CA)의 인증서를 사용할 경우 다음 사항에 유의하세요.

- 프라이빗 키는 인증서에 있는 퍼블릭 키와 일치해야 합니다.
- 프라이빗 키는 PEM 형식이어야 합니다.
- 프라이빗 키는 암호로 암호화할 수 없습니다.

AWS Certificate Manager(ACM)에서 인증서를 제공할 경우 ACM은 프라이빗 키를 공개하지 않습니다. 이 프라이빗 키는 ACM과 통합된 AWS 서비스에서 사용하기 위해 ACM에 저장됩니다.

### 권한

SSL/TLS 인증서를 사용하고 가져올 수 있는 권한이 있어야 합니다. AWS Certificate Manager(ACM)를 사용하는 경우, AWS Identity and Access Management 권한을 사용하여 인증서에 대한 액세스를 제한하는 것이 좋습니다. 자세한 내용은 AWS Certificate Manager 사용 설명서에서 자격 증명 및 액세스 관리를 참조하세요.

## 인증서 키의 크기

CloudFront에서 지원하는 인증서 키 크기는 키 및 인증서 유형에 따라 다릅니다.

RSA 인증서의 경우:

> CloudFront가 1,024비트, 2,048비트, 3,072비트, 4,096비트 RSA 키를 지원합니다. CloudFront에서 사용하는 RSA 인증서의 최대 키 길이는 4,096비트입니다.
>
> 참고로 ACM은 최대 2048비트 키로 RSA 인증서를 발급합니다. 3,072비트 또는 4,096비트 RSA 인증서를 사용하려면 인증서를 외부에서 얻어 ACM으로 가져와야 합니다. 그러면 CloudFront에서 사용할 수 있습니다.
>
> RSA 키의 크기를 확인하는 방법에 대한 자세한 내용은 SSL/TLS RSA 인증서에서 퍼블릭 키 크기 확인 단원을 참조하세요.

ECDSA 인증서의 경우:

> CloudFront는 256비트 키를 지원합니다. 최종 사용자와 CloudFront 간에 HTTPS를 요구하기 위해 ACM의 ECDSA 인증서를 사용하려면 prime256v1 타원 곡선을 사용합니다.

## 지원되는 인증서 유형

CloudFront는 신뢰할 수 있는 인증 기관에서 발급한 모든 유형의 인증서를 지원합니다.

## 인증서 만료 날짜 및 갱신

서드 파티 인증 기관(CA)에서 받은 인증서를 사용할 경우, 인증서 만료 날짜를 직접 모니터링하여 인증서가 만료되기 전에 AWS Certificate Manager(ACM)에 가져온 인증서를 갱신하거나 AWS Identity and Access Management 인증서 스토어에 업로드해야 합니다.

> **Important**
>
> 인증서 만료 문제를 방지하려면 현재 인증서의 `NotAfter` 값보다 최소 24시간 전에 인증서를 갱신하거나 다시 가져옵니다. 인증서가 24시간 이내에 만료되는 경우 ACM에서 새 인증서를 요청하거나 ACM으로 새 인증서를 가져옵니다. 그런 다음 새 인증서를 CloudFront 배포에 연결합니다.
>
> 인증서 갱신 또는 다시 가져오기가 진행되는 동안 CloudFront에서 이전 인증서를 계속 사용할 수 있습니다. 이는 CloudFront에서 변경 사항을 표시하기까지 최대 24시간이 걸릴 수 있는 비동기 프로세스입니다.

ACM에서 제공한 인증서를 사용할 경우 ACM에서 인증서 갱신을 관리해 줍니다. 자세한 내용은 AWS Certificate Manager 사용 설명서에서 관리형 갱신을 참조하세요.

## CloudFront 배포 및 인증서의 도메인 이름

사용자 지정 오리진을 사용하는 경우 오리진의 SSL/TLS 인증서에는 일반 이름(Common Name) 필드에 도메인 이름이 포함되며 대체 도메인 이름(Subject Alternative Names) 필드에도 몇 번 더 포함될 수 있습니다. CloudFront는 인증서 도메인 이름 내 와일드카드 문자 사용을 지원합니다.

인증서의 도메인 이름 중 하나는 오리진 도메인 이름으로 지정하는 도메인 이름과 일치해야 합니다. 일치하는 도메인 이름이 없는 경우 CloudFront는 최종 사용자에게 HTTP 상태 코드 `502 (Bad Gateway)`를 반환합니다.

> **Important**
>
> 배포에 대체 도메인 이름을 추가하면, CloudFront는 해당 대체 도메인 이름이 연결한 인증서에 포함되어 있는지 확인합니다. 인증서의 SAN(주체 대체 이름) 필드에 대체 도메인 이름이 있어야 합니다. 즉, SAN 필드에는 대체 도메인 이름과 정확히 일치하는 항목이 포함되거나 추가할 대체 도메인 이름과 동일한 수준에서 와일드카드가 포함되어야 합니다.
>
> 자세한 내용은 대체 도메인 이름 사용과 관련된 요구 사항 섹션을 참조하세요.

### 최소 SSL/TLS 프로토콜 버전

전용 IP 주소를 사용할 경우 보안 정책을 선택하여 최종 사용자와 CloudFront 간의 연결을 위한 최소 SSL/TLS 프로토콜 버전을 설정합니다.

자세한 내용은 모든 배포 설정 참조 주제에서 보안 정책(최소 SSL/TLS 버전) 단원을 참조하세요.

### 지원되는 HTTP 버전

한 인증서를 둘 이상의 CloudFront 배포와 연결하는 경우, 인증서와 연결된 모든 배포에서 지원되는 HTTP 버전에 대해 동일한 옵션을 사용해야 합니다. CloudFront 배포를 생성 또는 업데이트할 때 이 옵션을 지정합니다.

## CloudFront에서 SSL/TLS 인증서 사용 시의 할당량(뷰어와 CloudFront 간의 HTTPS만 해당)

CloudFront에서 SSL/TLS 인증서 사용 시 다음 할당량에 유의하시기 바랍니다. 이러한 할당량은 AWS Certificate Manager(ACM)를 사용하여 프로비저닝하거나, ACM으로 가져오거나, 뷰어와 CloudFront 간의 HTTPS 통신을 위해 IAM 인증서 스토어에 업로드하는 SSL/TLS 인증서에만 적용됩니다.

자세한 내용은 SSL/TLS 인증서에 대한 할당량 증가 섹션을 참조하세요.

CloudFront 배포당 최대 인증서 수

각 CloudFront 배포에 최대 한 개의 SSL/TLS 인증서를 연결할 수 있습니다.

ACM으로 가져오거나 IAM 인증서 스토어로 업로드할 수 있는 인증서의 최대 수

서드 파티 인증 기관에서 SSL/TLS 인증서를 구입한 경우 다음 위치 중 하나에 인증서를 저장해야 합니다.

- AWS Certificate Manager – ACM 인증서 수의 현재 할당량은 AWS Certificate Manager 사용 설명서의 할당량을 참조하세요. 명시된 할당량은 ACM을 사용하여 프로비저닝한 인증서와 ACM으로 가져온 인증서를 모두 포함한 총 개수입니다.
- IAM 인증서 스토어 – 하나의 AWS 계정에서 IAM 인증서 스토어에 업로드할 수 있는 인증서 수의 현재 할당량(이전에는 제한이라고 함)은 IAM 사용 설명서의 IAM 및 STS 제한을 참조하세요. Service Quotas 콘솔을 사용하여 더 높은 할당량을 요청할 수 있습니다.

AWS 계정당 인증서 최대 수(전용 IP 주소만 해당)

전용 IP 주소를 사용하여 HTTPS 요청을 처리하려는 경우 다음 사항에 유의하세요.

- 기본적으로 CloudFront는 AWS 계정에 두 인증서를 사용할 수 있는 권한(일일 사용을 위한 인증서와 여러 배포용으로 인증서를 교체해야 할 때 사용할 인증서)을 부여합니다.
- AWS 계정에 사용자 지정 SSL/TLS 인증서가 두 개 이상 필요한 경우 Service Quotas 콘솔에서 더 높은 할당량을 요청할 수 있습니다.

다른 AWS 계정을 사용하여 생성된 CloudFront 배포에 동일한 인증서 사용

다른 인증 기관을 사용할 경우 그리고 서로 다른 AWS 계정으로 생성된 여러 CloudFront 배포에서 동일한 인증서를 사용하려면 각 AWS 계정마다 한 번씩 해당 인증서를 ACM으로 가져오거나 IAM 인증서 스토어로 업로드해야 합니다.

ACM에서 제공한 인증서를 사용하는 경우, 다른 AWS 계정이 생성한 인증서를 사용하도록 CloudFront를 구성할 수 없습니다.

CloudFront 및 다른 AWS 서비스에 동일한 인증서 사용

신뢰할 수 있는 인증 기관(예: Comodo, DigiCert, Symantec 등)에서 인증서를 구입한 경우, CloudFront 및 기타 AWS 서비스에서 동일한 인증서를 사용할 수 있습니다. ACM으로 인증서를 가져올 경우 한 번만 가져와서 여러 AWS 서비스에 사용해야 합니다.

ACM에서 제공한 인증서를 사용할 경우 인증서가 ACM에 저장됩니다.

다중 CloudFront 배포에 동일한 인증서 사용

HTTPS 요청을 처리하는 데 사용하는 임의의 또는 모든 CloudFront 배포에서 동일한 인증서를 사용할 수 있습니다. 다음을 참조하세요.

- 전용 IP 주소를 사용한 요청 제공과 SNI를 사용한 요청 제공에 동일한 인증서를 사용할 수 있습니다.
- 단 한 개의 인증서를 각각의 배포와 연결할 수 있습니다.
- 각각의 배포에 하나 이상의 대체 도메인 이름이 포함되어야 하며, 이 이름은 인증서의 Common Name(일반 이름) 필드나 Subject Alternative Names(주체 대체 이름) 필드에도 표시됩니다.
- 전용 IP 주소를 사용하여 HTTPS 요청을 제공하고 동일한 AWS 계정을 사용해 모든 배포를 생성한 경우, 모든 배포에 동일한 인증서를 사용함으로써 비용을 크게 절감할 수 있습니다. CloudFront는 배포가 아닌 인증서별로 요금을 청구합니다.

  예를 들어, 동일한 AWS 계정을 사용하여 세 개의 배포를 생성하고, 이 세 개의 배포에 모두 동일한 인증서를 사용한다고 가정하겠습니다. 이 경우, 전용 IP 주소 사용에 대한 단 한 건의 수수료만 청구됩니다.

  하지만 전용 IP 주소를 사용하여 HTTPS 요청을 제공하고 서로 다른 AWS 계정에서 동일한 인증서를 사용해 CloudFront 배포를 생성할 경우, 각각의 계정에 전용 IP 주소 사용에 대한 요금이 청

구됩니다. 예를 들어, 세 개의 AWS 계정을 사용하여 세 개의 배포를 생성하고 이 세 배포에 동일한 인증서를 사용할 경우, 각 계정에 전용 IP 주소 사용에 대한 수수료 전액이 청구됩니다.

# 대체 도메인 이름과 HTTPS 구성

파일의 URL에 대체 도메인 이름을 사용하고, 최종 사용자와 CloudFront 간에 HTTPS를 사용하려면 해당 절차를 수행합니다.

주제

- SSL/TLS 인증서 받기
- SSL/TLS 인증서 가져오기
- CloudFront 배포 업데이트

## SSL/TLS 인증서 받기

SSL/TLS 인증서를 받습니다(아직 없는 경우). 자세한 내용은 다음과 같이 해당 문서를 참조하세요.

- AWS Certificate Manager(ACM)에서 제공하는 인증서를 사용하려면 AWS Certificate Manager 사용 설명서를 참조하세요. CloudFront 배포 업데이트 단원을 참조하세요.

  > **Note**
  >
  > AWS 관리형 리소스에 대해 SSL/TLS 인증서를 프로비저닝 및 관리하고 배포할 때 ACM을 사용하는 것이 좋습니다. 미국 동부(버지니아 북부) 리전에서 ACM 인증서를 요청해야 합니다.

- 다른 인증 기관(CA)의 인증서를 받으려면 인증 기관에서 제공한 설명서를 참조하세요. 인증서가 있으면 다음 절차로 진행합니다.

## SSL/TLS 인증서 가져오기

다른 인증 기관의 인증서를 받았을 경우 ACM으로 가져오거나 IAM 인증서 스토어에 업로드합니다.

ACM(권장)

ACM에서 ACM 콘솔을 사용하거나 프로그래밍 방식으로 타사 인증서를 가져올 수 있습니다. 인증서를 ACM으로 가져오는 방법에 대한 자세한 내용은 AWS Certificate Manager 사용 설명서의

AWS Certificate Manager로 인증서 가져오기 섹션을 참조하세요. 미국 동부(버지니아 북부) 리전에서 인증서를 가져와야 합니다.

IAM 인증서 스토어

(권장하지 않음) 다음 AWS CLI 명령을 사용하여 서드 파티 인증서를 IAM 인증서 스토어에 업로드합니다.

```
aws iam upload-server-certificate \
        --server-certificate-name CertificateName \
        --certificate-body file://public_key_certificate_file \
        --private-key file://privatekey.pem \
        --certificate-chain file://certificate_chain_file \
        --path /cloudfront/path/
```

다음을 참조하세요.

- AWS 계정 – CloudFront 배포를 생성하는 데 사용한 것과 동일한 AWS 계정을 사용하여 인증서를 IAM 인증서 스토어로 업로드해야 합니다.
- --path 파라미터 – 인증서를 IAM으로 업로드할 때 `--path` 파라미터(인증서 경로) 값은 `/cloudfront/`로 시작해야 합니다(예: `/cloudfront/production/` 또는 `/cloudfront/test/`). 경로는 /로 끝나야 합니다.
- 기존 인증서 – `--server-certificate-name` 및 `--path` 파라미터에, 기존 인증서와 연결되어 있는 값과 다른 값을 지정해야 합니다.
- CloudFront 콘솔 사용 – AWS CLI에서 `--server-certificate-name` 파라미터에 지정하는 값(예: `myServerCertificate`)은 CloudFront 콘솔의 SSL 인증서 목록에 나타납니다.
- CloudFront API 사용 – AWS CLI에서 반환하는 영숫자 문자열(예:`AS1A2M3P4L5E67SIIXR3J`)을 메모해 둡니다. 이 문자열은 `IAMCertificateId` 요소에서 지정하는 값입니다. IAM ARN은 필요하지 않습니다. CLI에서도 반환되기 때문입니다.

AWS CLI에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서 및 AWS CLI 명령 참조를 참조하세요.

## CloudFront 배포 업데이트

배포의 설정을 업데이트하려면 다음과 같이 합니다.

대체 도메인 이름을 사용하도록 CloudFront 배포를 구성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 업데이트하려는 배포의 ID를 선택합니다.
3. [General] 탭에서 [Edit]를 선택합니다.
4. 다음 값을 업데이트합니다.

   대체 도메인 이름(CNAME)(Alternate domain names(CNAME))

   항목 추가를 선택하여 해당 대체 도메인 이름을 추가합니다. 도메인 이름을 쉼표로 구분하거나 각각의 이름을 새 줄에 입력합니다.

   Custom SSL Certificate(사용자 정의 SSL 인증서)

   드롭다운 목록에서 인증서를 선택합니다.

   최대 100개의 인증서가 여기에 나열됩니다. 100개 이상의 인증서가 있고 추가할 인증서가 보이지 않는 경우에는 해당 필드에 인증서 ARN을 입력하여 선택할 수 있습니다.

   IAM 인증서 스토어에 인증서를 업로드했는데 목록에 나타나지 않아서 필드에 이름을 입력하는 방법으로 선택을 할 수 없는 경우에는 SSL/TLS 인증서 가져오기 절차를 검토하여 인증서를 올바르게 업로드했는지 확인합니다.

   > **Important**
   >
   > SSL/TLS 인증서를 CloudFront 배포와 연결한 후에 모든 배포에서 인증서를 제거하고 모든 배포판이 배포될 때까지 ACM 또는 IAM 인증서 스토어에서 인증서를 삭제하지 않습니다.

5. 변경 사항 저장을 선택합니다.
6. CloudFront에서 최종 사용자와 CloudFront 간에 HTTPS를 요구하도록 구성:

   a. 동작 탭에서 업데이트할 캐시 동작을 선택하고 편집을 선택합니다.

   b. Viewer Protocol Policy(최종 사용자 프로토콜 정책)에 다음 값 중 하나를 지정합니다.

      Redirect HTTP to HTTPS

      최종 사용자가 두 프로토콜 모두 사용할 수 있지만, HTTP 요청은 자동으로 HTTPS 요청으로 리디렉션됩니다. CloudFront는 HTTP 상태 코드 `301 (Moved Permanently)`을 새

로운 HTTPS URL과 함께 반환합니다. 그러면 최종 사용자는 HTTPS URL을 사용하여 이 요청을 CloudFront에 다시 제출합니다.

> **Important**
>
> CloudFront는 HTTP에서 HTTPS로 `DELETE`, `OPTIONS`, `PATCH`, `POST` 또는 `PUT` 요청을 리디렉션하지 않습니다. HTTPS에 리디렉션할 캐시 동작을 구성하는 경우, CloudFront는 해당 캐시 동작에 대한 HTTP `DELETE`, `OPTIONS`, `PATCH`, `POST`, 또는 `PUT` 요청에 HTTP 상태 코드 `403 (Forbidden)`으로 응답합니다.

최종 사용자가 HTTPS 요청으로 리디렉션되는 HTTP 요청을 만들 경우 CloudFront에서 두 요청 모두에 대해 요금을 부과합니다. HTTP 요청의 경우에는 CloudFront에서 최종 사용자에게 반환되는 요청 및 헤더에만 요금이 부과됩니다. HTTPS 요청의 경우에는 요청, 헤더, 그리고 오리진에 의해 반환된 파일을 모두 처리합니다.

HTTPS Only

최종 사용자가 HTTPS를 사용할 경우에만 콘텐츠에 액세스할 수 있습니다. 최종 사용자가 HTTPS 요청 대신에 HTTP 요청을 보내면 CloudFront는 HTTP 상태 코드 `403 (Forbidden)`을 반환하고 파일은 반환하지 않습니다.

c. 예, 편집합니다를 선택합니다.

d. 최종 사용자와 CloudFront 간에 HTTPS를 요구하려는 추가적인 캐시 동작 각각에 대해 단계 a부터 c까지 반복합니다.

7. 업데이트한 구성을 프로덕션 환경에서 사용하기 전에 다음 사항을 확인합니다.

- 각 캐시 동작의 경로 패턴이 최종 사용자에 HTTPS를 사용하도록 지정한 요청에만 적용되는가.
- 캐시 동작이 CloudFront에서 평가하도록 할 순서대로 나열되었는가. 자세한 내용은 경로 패턴 단원을 참조하세요.
- 캐시 동작이 올바른 오리진에 요청을 라우팅하는가.

## SSL/TLS RSA 인증서에서 퍼블릭 키 크기 확인

CloudFront 대체 도메인 이름 및 HTTPS를 사용하는 경우, SSL/TLS RSA 인증서 퍼블릭 키의 최대 크기는 4,096비트입니다. (이는 퍼블릭 키의 문자 수가 아니라 키 크기입니다.) 인증서에서 AWS Certificate Manager를 사용하는 경우에는 ACM이 더 큰 RSA 키를 지원하더라도 CloudFront에서 더 큰 키를 사용할 수 없습니다.

다음 OpenSSL 명령을 실행하여 RSA 퍼블릭 키의 크기를 확인할 수 있습니다.

```
openssl x509 -in path and filename of SSL/TLS certificate -text -noout
```

여기서 각 항목은 다음과 같습니다.

- `-in`은 SSL/TLS RSA 인증서의 경로 및 파일 이름을 지정합니다.
- `-text`를 지정하면 OpenSSL에서 RSA 퍼블릭 키 길이를 비트로 표시합니다.
- `-noout`를 지정하면 OpenSSL에서 퍼블릭 키가 표시되지 않습니다.

출력 예:

```
Public-Key: (2048 bit)
```

# SSL/TLS 인증서에 대한 할당량 증가

AWS Certificate Manager(ACM)로 가져오거나 AWS Identity and Access Management(IAM)에 업로드할 수 있는 SSL/TLS 인증서 수에는 할당량이 있습니다. 또한 전용 IP 주소를 사용하여 HTTPS 요청을 제공하도록 CloudFront를 구성할 경우 AWS 계정에서 사용할 수 있는 SSL/TLS 인증서 개수에도 할당량이 있습니다. 하지만 더 높은 할당량을 요청할 수 있습니다.

주제


- ACM으로 가져오는 인증서의 할당량 증가
- IAM에 업로드하는 인증서의 할당량 늘리기
- 전용 IP 주소와 함께 사용되는 인증서의 할당량 증가


## ACM으로 가져오는 인증서의 할당량 증가

ACM으로 가져올 수 있는 인증서 개수에 대한 할당량은 AWS Certificate Manager 사용 설명서의 할당량을 참조하세요.

더 높은 할당량을 요청하려면 Service Quotas 콘솔을 사용하세요. 자세한 내용은 Service Quotas 사용 설명서의 할당량 증가 요청을 참조하세요.

## IAM에 업로드하는 인증서의 할당량 늘리기

IAM에 업로드할 수 있는 인증서 개수에 대한 할당량(이전에는 제한이라고 함)은 IAM 사용 설명서의 IAM 및 STS 제한을 참조하세요.

더 높은 할당량을 요청하려면 Service Quotas 콘솔을 사용하세요. 자세한 내용은 Service Quotas 사용 설명서의 할당량 증가 요청을 참조하세요.

### 전용 IP 주소와 함께 사용되는 인증서의 할당량 증가

전용 IP 주소를 사용하여 HTTPS 요청을 제공할 경우 각 AWS 계정에서 사용할 수 있는 SSL 인증서 개수에 대한 할당량은 SSL 인증서의 할당량 섹션을 참조하세요.

더 높은 할당량을 요청하려면 Service Quotas 콘솔을 사용하세요. 자세한 내용은 Service Quotas 사용 설명서의 할당량 증가 요청을 참조하세요.

## SSL/TLS 인증서 교체

SSL/TLS 인증서가 만료될 때 배포의 보안을 보장하고 뷰어의 서비스 중단을 방지하려면 인증서를 교체해야 합니다. 다음과 같은 방법으로 인증서를 교체할 수 있습니다.

- AWS Certificate Manager(ACM)에서 제공하는 SSL/TLS 인증서의 경우 인증서를 교체할 필요가 없습니다. ACM에서 자동으로 인증서 갱신을 관리합니다. 자세한 내용은 AWS Certificate Manager 사용 설명서의 Managed certificate renewal을 참조하시기 바랍니다.
- 서드 파티 인증 기관을 사용하여 인증서를 ACM으로 가져오거나(권장) IAM 인증서 스토어에 업로드한 경우 필요에 따라 인증서를 교체해야 합니다.

> **Important**
>
> - ACM은 서드 파티 인증 기관에서 확보하여 ACM으로 가져온 인증서에 대해서는 갱신을 관리하지 않습니다.
> - 전용 IP 주소를 사용하여 HTTPS 요청을 제공하도록 CloudFront를 구성한 경우 인증서를 교체하는 동안은 하나 이상의 추가 인증서 사용에 대한 비례 할당으로 계산된 추가 요금이 발생할 수 있습니다. 추가 요금을 최소화하기 위해 배포를 업데이트할 것을 권장합니다.

### SSL/TLS 인증서 교체

인증서를 교체하려면 다음 절차를 수행합니다. 최종 사용자는 인증서를 교체하는 동안은 물론, 프로세스가 완료된 후에도 콘텐츠에 계속 액세스할 수 있습니다.

SSL/TLS 인증서를 교체하려면

1. SSL/TLS 인증서에 대한 할당량 증가는 추가 SSL 인증서를 사용하기 위한 권한이 필요한지 여부를 확인합니다. 권한이 필요한 경우 권한을 요청하고 기다렸다 권한이 부여되면 2단계로 넘어갑니다.
2. 새 인증서를 ACM으로 가져오거나 IAM에 업로드합니다. 자세한 내용은 Amazon CloudFront 개발자 안내서의 SSL/TLS 인증서 가져오기를 참조하세요.
3. (IAM 인증서만 해당) 새 인증서를 사용하도록 한 번에 하나씩 배포를 업데이트합니다. 자세한 내용은 배포 업데이트 섹션을 참조하세요.
4. (선택 사항) ACM 또는 IAM에서 이전 인증서를 삭제합니다.

> **Important**
>
> 모든 배포에서 해당 인증서가 제거되고 업데이트된 배포의 상태가 `Deployed`로 변경될 때까지는 SSL/TLS 인증서를 삭제하지 않습니다.

## 사용자 지정 SSL/TLS 인증서에서 기본 CloudFront 인증서로 되돌리기

CloudFront에서 최종 사용자와 CloudFront 간에 HTTPS를 사용하도록 구성했으며 CloudFront에서 사용자 지정 SSL/TLS 인증서를 사용하도록 구성한 경우, 기본 CloudFront SSL/TLS 인증서를 사용하도록 구성을 변경할 수 있습니다. 이 절차는 콘텐츠를 배포하는 데 배포를 사용했는지 여부에 따라 다릅니다.

- 배포를 사용하여 콘텐츠를 배포하지 않은 경우에는 구성을 변경하면 됩니다. 자세한 내용은 배포 업데이트 단원을 참조하세요.
- 배포를 사용하여 콘텐츠를 배포한 경우에는 새 CloudFront 배포를 생성하고 파일에 대한 URL을 변경하여 콘텐츠를 사용할 수 없는 시간을 줄이거나 없애야 합니다. 이렇게 하려면 다음과 같이 합니다.

### 기본 CloudFront 인증서로 되돌리기

다음 절차에서는 사용자 지정 SSL/TLS 인증서에서 기본 CloudFront 인증서로 되돌리는 방법을 보여줍니다.

기본 CloudFront 인증서로 되돌리려면

1. 원하는 구성을 사용하여 새 CloudFront 배포를 생성합니다. SSL 인증서(SSL Certificate)에서 기본 CloudFront 인증서(Default CloudFront Certificate)(*.cloudfront.net)를 선택합니다.

   자세한 내용은 배포 생성 섹션을 참조하세요.

2. CloudFront를 사용하여 배포하고 있는 파일의 경우에는 CloudFront에서 새 배포에 할당한 도메인 이름이 사용되도록 애플리케이션에서 URL을 업데이트합니다. 예를 들면 `https://www.example.com/images/logo.png`를 `https://d111111abcdef8.cloudfront.net/images/logo.png`로 변경합니다.

3. 사용자 지정 SSL/TLS 인증서와 연결된 배포를 삭제하거나, SSL 인증서(SSL Certificate) 값이 기본 CloudFront 인증서(Default CloudFront Certificate)(*.cloudfront.net)로 변경되도록 배포를 업데이트합니다. 자세한 내용은 배포 업데이트 단원을 참조하세요.

   > **Important**
   >
   > 이 단계를 완료하지 않으면 AWS에서 사용자 지정 SSL/TLS 인증서 사용에 요금을 계속 부과합니다.

4. (선택 사항) 사용자 지정 SSL/TLS 인증서를 삭제합니다.

   a. AWS CLI 명령 `list-server-certificates`를 실행하여 삭제할 인증서의 인증서 ID를 확인합니다. 자세한 내용은 AWS CLI 명령 참조의 list-server-certificates를 참조하세요.

   b. AWS CLI 명령 `delete-server-certificate`를 실행하여 인증서를 삭제합니다. 자세한 내용은 AWS CLI 명령 참조의 delete-server-certificate를 참조합니다.

## 사용자 지정 SSL/TLS 인증서를 전용 IP 주소 사용에서 SNI 사용으로 전환

사용자 지정 SSL/TLS 인증서에 전용 IP 주소를 사용하도록 CloudFront를 구성한 경우 SSL/TLS 인증서에 SNI를 대신 사용하고 전용 IP 주소와 연결된 요금이 청구되지 않도록 전환할 수 있습니다.

> **Important**
>
> 이 CloudFront 구성 업데이트는 SNI를 지원하는 최종 사용자에는 영향을 미치지 않습니다. 최종 사용자는 변경 전후뿐만 아니라 변경 사항이 CloudFront 엣지 로케이션에 전파되는 동안에도 콘텐츠에 액세스할 수 있습니다. SNI를 지원하지 않는 뷰어는 변경 후에 콘텐츠에 액세스할

수 없습니다. 자세한 내용은 CloudFront에서 HTTPS 요청을 제공하는 방식 선택 섹션을 참조하세요.

### 사용자 지정 인증서에서 SNI로 전환

다음 절차에서는 사용자 지정 SSL/TLS 인증서를 전용 IP 주소 사용에서 SNI 사용으로 전환하는 방법을 보여줍니다.

사용자 지정 SSL/TLS 인증서를 전용 IP 주소 사용에서 SNI 사용으로 전환하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 보거나 업데이트하려는 배포의 ID를 선택합니다.
3. Distribution Settings(배포 설정)를 선택합니다.
4. [General] 탭에서 [Edit]를 선택합니다.
5. 사용자 지정 SSL 인증 - 선택 사항에서 레거시 클라이언트 지원을 선택 해제합니다.
6. 예, 편집합니다를 선택합니다.

## CloudFront에 대한 상호 TLS 인증(뷰어 mTLS)

상호 TLS 인증(상호 전송 계층 보안 인증 - mTLS)은 양방향 인증서 기반 인증을 요구하여 표준 TLS 인증을 확장하는 보안 프로토콜로, 클라이언트와 서버 모두 보안 연결을 설정하기 전에 자격 증명을 증명해야 합니다. 상호 TLS를 사용하면 신뢰할 수 있는 TLS 인증서를 제공하는 클라이언트만 CloudFront 배포에 액세스할 수 있습니다.

### 작동 방식

표준 TLS 핸드셰이크에서는 서버만 클라이언트에 자격 증명을 증명하는 인증서를 제공합니다. 상호 TLS를 사용하면 인증 프로세스가 양방향이 됩니다. 클라이언트가 CloudFront 배포에 연결을 시도하면 CloudFront는 TLS 핸드셰이크 중에 클라이언트 인증서를 요청합니다. 클라이언트는 보안 연결을 설정하기 전에 CloudFront가 구성된 트러스트 스토어에 대해 검증하는 유효한 X.509 인증서를 제시해야 합니다.

CloudFront는 AWS 엣지 로케이션에서 이 인증서 검증을 수행하여 오리진 서버에서 인증 복잡성을 오프로드하는 동시에 CloudFront의 글로벌 성능 이점을 유지합니다. 확인 모드(모든 클라이언트가 유효

한 인증서를 제시해야 함) 또는 선택적 모드(제공 시 인증서를 검증하지만 인증서 없는 연결도 허용함)의 두 가지 모드로 mTLS를 구성할 수 있습니다.

## 사용 사례

CloudFront를 사용한 상호 TLS 인증은 기존 인증 방법이 충분하지 않은 몇 가지 중요한 보안 시나리오를 해결합니다.

- 콘텐츠 캐싱을 사용한 디바이스 인증 - 펌웨어 업데이트, 게임 다운로드 또는 내부 리소스에 대한 액세스를 허용하기 전에 게임 콘솔, IoT 디바이스 또는 회사 하드웨어를 인증할 수 있습니다. 각 디바이스에는 CloudFront의 캐싱 기능을 활용하면서 신뢰성을 입증하는 고유한 인증서가 포함되어 있습니다.
- API-to-API 인증 - 신뢰할 수 있는 비즈니스 파트너, 결제 시스템 또는 마이크로 서비스 간의 시스템 대 시스템 통신을 보호할 수 있습니다. 인증서 기반 인증을 사용하면 공유 보안 암호 또는 API 키를 사용할 필요 없이 자동 데이터 교환을 위한 강력한 자격 증명 확인을 제공합니다.

주제


- 트러스트 스토어 및 인증서 관리
- CloudFront 배포에 상호 TLS 활성화
- CloudFront 연결 함수 연결
- 추가 설정 구성
- 오리진으로 전달된 캐시 정책에 대한 뷰어 mTLS 헤더
- CloudFront 연결 함수 및 KVS를 사용한 해지
- 연결 로그를 사용한 관찰성


## 트러스트 스토어 및 인증서 관리

트러스트 스토어를 생성하고 구성하는 것은 CloudFront와의 상호 TLS 인증을 구현하기 위한 필수 요구 사항입니다. 트러스트 스토어에는 CloudFront가 인증 프로세스 중에 클라이언트 인증서를 검증하는 데 사용하는 인증 기관(CA) 인증서가 포함되어 있습니다.

### 트러스트 스토어란 무엇인가요?

트러스트 스토어는 CloudFront가 상호 TLS 인증 중에 클라이언트 인증서를 검증하는 데 사용하는 CA 인증서의 리포지토리입니다. 트러스트 스토어에는 클라이언트 인증서를 인증하기 위한 신뢰 체인을 구성하는 루트 및 중간 CA 인증서가 포함되어 있습니다.

CloudFront와 상호 TLS를 구현하면 트러스트 스토어는 유효한 클라이언트 인증서를 발급하기 위해 신뢰할 수 있는 인증 기관을 정의합니다. CloudFront는 TLS 핸드셰이크 중에 트러스트 스토어에 대해 각 클라이언트 인증서를 검증합니다. 트러스트 스토어의 CA 중 하나에 연결된 인증서를 제공하는 클라이언트만 성공적으로 인증됩니다.

CloudFront의 트러스트 스토어는 여러 배포와 연결할 수 있는 계정 수준 리소스입니다. 이를 통해 CA 인증서 관리를 간소화하면서 전체 CloudFront 배포에서 일관된 인증서 검증 정책을 유지할 수 있습니다.

## 인증 기관 지원

CloudFront는 AWS 프라이빗 인증 기관과 타사 프라이빗 인증 기관에서 발급한 인증서를 지원합니다. 이러한 유연성을 통해 기존 인증서 인프라를 사용하거나 조직 요구 사항에 따라 AWS 관리형 인증서 서비스를 활용할 수 있습니다.

- AWS 프라이빗 인증 기관: 관리형 프라이빗 인증 기관 서비스를 제공하는 AWS Private CA에서 발급한 인증서를 사용할 수 있습니다. 이 통합은 인증서 수명 주기 관리를 간소화하고 다른 AWS 서비스와의 원활한 통합을 제공합니다.
- 타사 프라이빗 인증 기관: 엔터프라이즈 CA 또는 기타 타사 인증서 공급자를 포함하여 기존 프라이빗 인증 기관 인프라의 인증서를 사용할 수도 있습니다. 이렇게 하면 CloudFront의 mTLS 기능을 추가하면서 현재 인증서 관리 프로세스를 유지할 수 있습니다.

## 인증서 요구 사항 및 사양

트러스트 스토어에는 포함된 CA 인증서에 대한 특정 요구 사항이 있습니다.

### CA 인증서 형식 요구 사항

- 형식: PEM(Privacy Enhanced Mail) 형식
- 콘텐츠 경계: 인증서는 -----BEGIN CERTIFICATE----- 및 -----END CERTIFICATE----- 경계 내에 묶여야 합니다.
- 주석: 앞에는 # 문자를 붙여야 하며 - 문자를 포함할 수 없습니다.
- 줄 바꿈: 인증서 사이에 빈 줄이 허용되지 않습니다.

### 지원되는 인증서 사양

- 인증서 유형: X.509v3
- 퍼블릭 키 유형:

  - RSA 2048, RSA 4096
  - ECDSA: secp256r1, secp384r1
- 서명 알고리즘:
  - RSA를 사용하는 SHA256, SHA384, SHA512
  - EC를 사용하는 SHA256, SHA384, SHA512
  - MGF1 지원 RSASSA-PSS를 사용하는 SHA256, SHA384, SHA512

예제 인증서 번들 형식

여러 인증서(PEM 인코딩):


```
# Root CA Certificate
-----BEGIN CERTIFICATE-----
MIIDXTCCAkWgAwIBAgIJAKoK/OvD/XqiMA0GCSqGSIb3DQEBCwUAMEUxCzAJBgNV
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTcwNzEyMTU0NzQ4WhcNMjcwNzEwMTU0NzQ4WjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAuuExKvY1xzHFylsHiuowqpmzs7rEcuuylOuEszpFp+BtXh0ZuEtts9LP
-----END CERTIFICATE-----
# Intermediate CA Certificate
-----BEGIN CERTIFICATE-----
MIIDXTCCAkWgAwIBAgIJAKoK/OvD/XqjMA0GCSqGSIb3DQEBCwUAMEUxCzAJBgNV
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTcwNzEyMTU0NzQ4WhcNMjcwNzEwMTU0NzQ4WjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAuuExKvY1xzHFylsHiuowqpmzs7rEcuuylOuEszpFp+BtXh0ZuEtts9LP
-----END CERTIFICATE-----
```


## 트러스트 스토어 생성

트러스트 스토어를 생성하기 전에 PEM 형식의 CA 인증서 번들을 Amazon S3 버킷에 업로드해야 합니다. 인증서 번들에는 클라이언트 인증서를 검증하는 데 필요한 신뢰할 수 있는 모든 루트 및 중간 CA 인증서가 포함되어야 합니다.

CA 인증서 번들은 트러스트 스토어를 생성할 때 S3에서 한 번만 읽습니다. CA 인증서 번들을 나중에 변경하는 경우 트러스트 스토어를 수동으로 업데이트해야 합니다. 트러스트 스토어와 S3 CA 인증서 번들 간에 동기화가 유지되지 않습니다.

사전 조건

- Amazon S3 버킷에 업로드된 인증 기관(CA)의 인증서 번들
- CloudFront 리소스를 생성하는 데 필요한 권한

트러스트 스토어를 생성하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어 생성을 선택합니다.
4. 트러스트 스토어 이름에 트러스트 스토어의 이름을 입력합니다.
5. 인증 기관(CA) 번들의 경우 PEM 형식 CA 인증서 번들에 Amazon S3 경로를 입력합니다.
6. 트러스트 스토어 생성을 선택합니다.

트러스트 스토어를 생성하려면(AWS CLI)

```
aws cloudfront create-trust-store \
  --name MyTrustStore \
  --certificate-authority-bundle-s3-location Bucket=my-bucket,Key=ca-bundle.pem \
  --tags Items=[{Key=Environment,Value=Production}]
```

## 트러스트 스토어를 배포와 연결

트러스트 스토어를 생성한 후에는 이를 CloudFront 배포와 연결하여 상호 TLS 인증을 활성화해야 합니다.

사전 조건

- HTTPS 전용 뷰어 프로토콜 정책이 활성화되고 HTTP3 지원이 비활성화된 기존 CloudFront 배포입니다.

트러스트 스토어를 연결하려면(콘솔)

CloudFront 콘솔 내에서 트러스트 스토어 세부 정보 페이지 또는 배포 설정 페이지를 통해 트러스트 스토어를 연결할 수 있는 두 가지 방법이 있습니다.

트러스트 스토어 세부 정보 페이지를 통해 트러스트 스토어 연결:

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 연결할 트러스트 스토어의 이름을 선택합니다.
4. 배포에 연결을 선택합니다.
5. 사용 가능한 뷰어 mTLS 옵션을 구성합니다.
   - 클라이언트 인증서 검증 모드: 필수 모드와 선택 모드 중에서 선택합니다. 필수 모드에서는 모든 클라이언트가 인증서를 제시해야 합니다. 선택적 모드에서는 인증서를 제공하는 클라이언트가 검증되는 반면, 인증서를 제공하지 않는 클라이언트는 액세스가 허용됩니다.
   - 트러스트 스토어 CA 이름 알림: TLS 핸드셰이크 중에 트러스트 스토어의 CA 이름을 클라이언트에 알릴지 여부를 선택합니다.
   - 인증서 만료 날짜 무시: 만료된 인증서와의 연결을 허용할지 여부를 선택합니다(다른 검증 기준은 계속 적용됨).
   - 연결 함수: 선택적 연결 함수를 연결하여 다른 사용자 지정 기준에 따라 연결을 허용/거부할 수 있습니다.
6. 트러스트 스토어와 연결할 배포를 하나 이상 선택합니다. HTTP3가 비활성화되고 HTTPS 전용 캐시 동작이 있는 배포만 뷰어 mTLS를 지원할 수 있습니다.
7. 연결을 선택합니다.

배포 설정 페이지를 통해 트러스트 스토어 연결:

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 연결하려는 배포를 선택합니다.
3. 일반 탭의 설정 컨테이너에서 오른쪽 상단 모서리에 있는 편집을 선택합니다.
4. 페이지 하단까지 아래로 스크롤하여 연결 컨테이너 내에서 뷰어 mTLS 스위치를 켭니다.
5. 사용 가능한 뷰어 mTLS 옵션을 구성합니다.
   - 클라이언트 인증서 검증 모드: 필수 모드와 선택 모드 중에서 선택합니다. 필수 모드에서는 모든 클라이언트가 인증서를 제시해야 합니다. 선택적 모드에서는 인증서를 제공하는 클라이언트가 검증되는 반면, 인증서를 제공하지 않는 클라이언트는 액세스가 허용됩니다.
   - 트러스트 스토어 CA 이름 알림: TLS 핸드셰이크 중에 트러스트 스토어의 CA 이름을 클라이언트에 알릴지 여부를 선택합니다.

- 인증서 만료 날짜 무시: 만료된 인증서와의 연결을 허용할지 여부를 선택합니다(다른 검증 기준은 계속 적용됨).
- 연결 함수: 선택적 연결 함수를 연결하여 다른 사용자 지정 기준에 따라 연결을 허용/거부할 수 있습니다.

6. 오른쪽 하단 모서리에서 변경 사항 저장을 선택합니다.

### 트러스트 스토어를 연결하려면(AWS CLI)

트러스트 스토어는 DistributionConfig.ViewerMtlsConfig 속성을 통해 배포에 연결할 수 있습니다. 즉, 먼저 배포 구성을 가져온 다음 후속 UpdateDistribution 요청에서 ViewerMtlsConfig를 제공해야 합니다.

```
// First fetch the distribution
aws cloudfront get-distribution {DISTRIBUTION_ID}

// Update the distribution config, for example:
Distribution config, file://distConf.json:
{
  ...other fields,
  ViewerMtlsConfig: {
    Mode: 'required',
    TrustStoreConfig: {
        AdvertiseTrustStoreCaNames: false,
        IgnoreCertificateExpiry: true,
        TrustStoreId: {TRUST_STORE_ID}
    }
  }
}

aws cloudfront update-distribution \
   --id {DISTRIBUTION_ID} \
   --if-match {ETAG} \
   --distribution-config file://distConf.json
```

## 트러스트 스토어 관리

### 트러스트 스토어 세부 정보 보기

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.

2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어 이름을 선택하여 세부 정보 페이지를 봅니다.

세부 정보 페이지에는 다음이 표시됩니다.

- 트러스트 스토어 이름 및 ID
- CA 인증서 수
- 생성 날짜 및 마지막 수정 날짜
- 연결된 배포
- Tags

트러스트 스토어 수정

CA 인증서 번들을 교체하려면:

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어의 이름을 선택합니다.
4. 작업을 선택한 다음 편집을 선택합니다.
5. 인증 기관(CA) 번들에 업데이트된 CA 번들 PEM 파일의 Amazon S3 위치를 입력합니다.
6. 트러스트 스토어 업데이트를 선택합니다.

트러스트 스토어 삭제

사전 조건: 먼저 모든 CloudFront 배포에서 트러스트 스토어의 연결을 해제해야 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 트러스트 스토어를 선택합니다.
3. 트러스트 스토어의 이름을 선택합니다.
4. 트러스트 스토어 삭제를 선택합니다.
5. [삭제]를 선택하여 확인합니다.

### 다음 단계

트러스트 스토어를 생성하고 CloudFront 배포와 연결한 후 배포에서 상호 TLS 인증을 활성화하고 인증서 헤더를 오리진에 전달하는 등의 추가 설정을 구성할 수 있습니다. 배포에서 mTLS를 활성화하는 방법에 대한 자세한 지침은 CloudFront 배포에 상호 TLS 활성화 섹션을 참조하세요.

# CloudFront 배포에 상호 TLS 활성화

## 사전 조건 및 요구 사항

CloudFront의 상호 TLS 확인 모드에서는 모든 클라이언트가 TLS 핸드셰이크 중에 유효한 인증서를 제시하고 유효한 인증서가 없는 연결을 거부해야 합니다. CloudFront 배포에서 상호 TLS를 활성화하기 전에 다음을 확인해야 합니다.

- 인증 기관 인증서로 트러스트 스토어를 생성함
- 트러스트 스토어가 CloudFront 배포와 연결됨
- 모든 배포 캐시 동작이 HTTPS 전용 뷰어 프로토콜 정책을 사용함
- 배포에서 HTTP/2를 사용하고 있음(기본 설정, HTTP/3에서는 뷰어 mTLS가 지원되지 않음)

> **Note**
>
> 상호 TLS 인증에는 뷰어와 CloudFront 간의 HTTPS 연결이 필요합니다. HTTP 연결을 지원하는 캐시 동작이 있는 배포에서는 mTLS를 활성화할 수 없습니다.

## 상호 TLS 활성화(콘솔)

### 새 배포의 경우

CloudFront 콘솔에서 새 배포를 생성하는 과정에서는 뷰어 mTLS를 구성할 수 없습니다. 먼저 모든 방법(콘솔, CLI, API)으로 배포를 생성한 다음 배포 설정을 편집하여 아래 기존 배포 지침에 따라 뷰어 mTLS를 활성화합니다.

### 기존 배포의 경우

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포 목록에서 수정하려는 배포를 선택합니다.

3. 모든 캐시 동작의 뷰어 프로토콜 정책을 HTTP를 HTTPS로 리디렉션 또는 HTTPS 전용으로 설정해야 합니다. (캐시 동작 탭을 선택하여 HTTP 프로토콜 정책으로 캐시 동작을 보고 업데이트할 수 있습니다.)
4. 일반 탭을 선택합니다.
5. 설정 섹션에서 편집을 선택합니다.
6. 연결성 섹션에서 뷰어 상호 인증(mTLS)을 찾습니다.
7. 상호 인증 활성화를 켜기로 전환합니다.
8. 클라이언트 인증서 검증 모드에서 필수(모든 클라이언트가 인증서를 제시해야 함) 또는 선택 사항(클라이언트가 선택적으로 인증서를 제공할 수 있음)을 선택합니다.
9. 트러스트 스토어에서 이전에 생성한 트러스트 스토어를 선택합니다.
10. (선택 사항) TLS 핸드셰이크 중에 CloudFront가 클라이언트에 CA 이름을 보내도록 하려면 트러스트 스토어 CA 이름 알리기를 전환합니다.
11. (선택 사항) 만료된 인증서와의 연결을 허용하려면 인증서 만료 날짜 무시를 전환합니다.
12. 변경 사항 저장을 선택합니다.

## 상호 TLS 활성화(AWS CLI)

### 새 배포의 경우

다음 예제에서는 mTLS 설정을 포함하는 배포 구성 파일(distribution-config.json)을 생성하는 방법을 보여줍니다.

```
{
  "CallerReference": "cli-example-1",
  "Origins": {
    "Quantity": 1,
    "Items": [
      {
        "Id": "my-origin",
        "DomainName": "example.com",
        "CustomOriginConfig": {
          "HTTPPort": 80,
          "HTTPSPort": 443,
          "OriginProtocolPolicy": "https-only"
        }
      }
    ]
  },
```

```
  "DefaultCacheBehavior": {
    "TargetOriginId": "my-origin",
    "ViewerProtocolPolicy": "https-only",
    "MinTTL": 0,
    "ForwardedValues": {
      "QueryString": false,
      "Cookies": {
        "Forward": "none"
      }
    }
  },
  "ViewerCertificate": {
    "CloudFrontDefaultCertificate": true
  },
  "ViewerMtlsConfig": {
    "Mode": "required",
    "TrustStoreConfig": {
        "TrustStoreId": {TRUST_STORE_ID},
        "AdvertiseTrustStoreCaNames": true,
        "IgnoreCertificateExpiry": true
    }
  },
  "Enabled": true
}
```

다음 예제 명령을 사용하여 mTLS가 활성화된 배포를 생성합니다.

```
aws cloudfront create-distribution --distribution-config file://distribution-
config.json
```

기존 배포의 경우

다음 예제 명령을 사용하여 현재 배포 구성을 가져옵니다.

```
aws cloudfront get-distribution-config --id E1A2B3C4D5E6F7 --output json > dist-
config.json
```

파일을 편집하여 mTLS 설정을 추가합니다. 배포 구성에 다음 예제 섹션을 추가합니다.

```
"ViewerMtlsConfig": {
    "Mode": "required",
    "TrustStoreConfig": {
        "TrustStoreId": {TRUST_STORE_ID},
```

```
        "AdvertiseTrustStoreCaNames": true,
        "IgnoreCertificateExpiry": true
    }
}
```

파일에서 ETag 필드를 제거하되 값을 별도로 저장합니다.

다음 예제 명령을 사용하여 배포를 새 구성으로 업데이트합니다.

```
aws cloudfront update-distribution \
    --id E1A2B3C4D5E6F7 \
    --if-match YOUR-ETAG-VALUE \
    --distribution-config file://dist-config.json
```

### 뷰어 프로토콜 정책

상호 TLS를 사용하는 경우 모든 배포 캐시 동작을 HTTPS 전용 뷰어 프로토콜 정책으로 구성해야 합니다.

- HTTP를 HTTPS로 리디렉션 - 인증서 검증을 수행하기 전에 HTTP 요청을 HTTPS로 리디렉션합니다.
- HTTPS 전용 - HTTPS 요청만 수락하고 인증서 검증을 수행합니다.

> **Note**
>
> HTTP 연결은 인증서 검증을 수행할 수 없으므로 HTTP 및 HTTPS 뷰어 프로토콜 정책은 상호 TLS에서 지원되지 않습니다.

### 다음 단계

CloudFront 배포에서 뷰어 TLS를 활성화한 후 연결 함수를 연결하여 사용자 지정 인증서 검증 로직을 구현할 수 있습니다. 연결 함수를 사용하면 사용자 지정 검증 규칙, 인증서 해지 확인 및 로깅을 통해 기본 제공 mTLS 인증 기능을 확장할 수 있습니다. 연결 함수 생성 및 연결에 대한 자세한 내용은 CloudFront 연결 함수 연결 섹션을 참조하세요.

## CloudFront 연결 함수 연결

CloudFront 연결 함수를 사용하면 TLS 핸드셰이크 중에 사용자 지정 인증서 검증 로직을 구현하여 내장 mTLS 인증 기능을 확장할 수 있습니다.

## 연결 함수란 무엇입니까?

연결 함수는 클라이언트 인증서가 검증된 후 TLS 핸드셰이크 중에 실행되는 JavaScript 함수입니다. 검증된 클라이언트 인증서는 연결 함수로 전달되며, 이 시점에서 연결 함수는 액세스 권한 부여 여부를 추가로 결정할 수 있습니다. 연결 함수에 대한 자세한 내용은 CloudFront Functions를 사용하여 엣지에서 사용자 지정 섹션을 참조하세요.

## 연결 함수가 mTLS에서 작동하는 방식

클라이언트가 CloudFront 배포에 대한 mTLS 연결을 설정하려고 하면 다음 시퀀스가 발생합니다.

1. 클라이언트가 CloudFront 엣지 로케이션으로 TLS 핸드셰이크를 시작합니다.
2. CloudFront가 클라이언트 인증서를 요청하고 수신합니다.
3. CloudFront가 트러스트 스토어에 대해 표준 인증서 검증을 수행합니다.
4. 인증서가 표준 검증을 통과하면 CloudFront는 연결 함수를 간접적으로 호출합니다. ViewerMtlsConfig 내에서 IgnoreCertificateExpiry가 활성화된 경우 만료되었지만 유효한 인증서도 연결 함수로 전달됩니다. 클라이언트 인증서가 유효하지 않으면 연결 함수가 간접 호출되지 않습니다.
5. 연결 함수는 구문 분석된 인증서 정보와 연결 세부 정보를 수신합니다.
6. 함수는 사용자 지정 로직을 기반으로 허용/거부 결정을 내립니다.
7. CloudFront는 사용자의 결정에 따라 TLS 연결을 완료하거나 종료합니다.

연결 함수는 확인 모드와 선택적 모드(클라이언트가 인증서를 제공하는 경우) 모두에 대해 간접적으로 호출됩니다.

## 연결 함수 할당량 증가 요청

AWS 계정에 대한 연결 함수 할당량 증가를 요청합니다.

연결 함수 할당량 증가를 요청하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 함수를 선택합니다.
3. 연결 함수 탭을 선택합니다.
4. 요청에서 CloudFront 지원 엔지니어링 팀에 문의할 수 있는 링크를 선택합니다.

5. CloudFront 지원 엔지니어링 팀에서는 요청을 검토합니다. 검토 프로세스는 최대 2일이 걸릴 수 있습니다.

요청이 승인되면 상호 TLS를 사용하는 동안 계정에서 연결 함수를 생성하고 이를 하나 이상의 배포와 연결할 수 있습니다.

## 연결 함수 생성

CloudFront 콘솔 또는 AWS CLI를 사용하여 연결 함수를 생성할 수 있습니다.

연결 함수를 생성하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 함수를 선택합니다.
3. 연결 함수 탭을 선택한 다음 연결 함수 생성을 선택합니다.
4. AWS 계정 내에서 고유한 함수 이름을 입력합니다.
5. 계속을 선택합니다.
6. 함수 편집기에서 인증서 검증을 위한 JavaScript 코드를 작성합니다. 함수 핸들러는 허용 또는 거부를 직접 호출해야 합니다.
7. 선택 사항: KeyValue 스토어를 연결 함수에 연결하여 해지 제어를 구현할 수 있습니다.
8. 변경 사항 저장을 선택합니다.

연결 함수를 생성하려면(AWS CLI)

다음 예제에서는 연결 함수를 생성하는 방법을 보여줍니다.

code.js와 같은 별도의 파일에 함수 코드를 작성합니다.

```
function connectionHandler(connection) {
  connection.allow();
}
```

```
aws cloudfront create-connection-function \
  --name "certificate-validator" \
  --connection-function-config '{
      "Comment": "Client certificate validation function",
```

```
      "Runtime": "cloudfront-js-2.0"
  }' \
  --connection-function-code fileb://code.js
```

## 연결 함수 코드 구조

연결 함수는 인증서 및 연결 정보가 포함된 연결 객체를 수신하는 connectionHandler 함수를 구현합니다. 함수는 `connection.allow()` 또는 `connection.deny()`를 사용하여 연결에 대한 결정을 내려야 합니다.

### 기본 연결 함수 예제

다음 예제는 클라이언트 인증서의 제목 필드를 확인하는 간단한 연결 함수를 보여줍니다.

```
function connectionHandler(connection) {
    // Only process if a certificate was presented
    if (!connection.clientCertificate) {
        console.log("No certificate presented");
        connection.deny();
    }

    // Check the subject field for specific organization
    const subject = connection.clientCertificate.certificates.leaf.subject;
    if (!subject.includes("O=ExampleCorp")) {
        console.log("Certificate not from authorized organization");
       connection.deny();
    } else {
        // All checks passed
        console.log("Certificate validation passed");
        connection.allow();
    }
}
```

연결 객체에서 사용할 수 있는 클라이언트 인증서 속성의 전체 사양은 여기에서 확인할 수 있습니다.

```
{
  "connectionId": "Fdb-Eb7L9gVn2cFakz7wWyBJIDAD4-oNO6g8r3vXDV132BtnIVtqDA==", // Unique
 identifier for this TLS connection
  "clientIp": "203.0.113.42", // IP address of the connecting client (IPv4 or IPv6)
  "clientCertificate": {
    "certificates": {
      "leaf": {
```

```
        "subject": "CN=client.example.com,O=Example Corp,C=US", // Distinguished Name
 (DN) of the certificate holder
        "issuer": "CN=Example Corp Intermediate CA,O=Example Corp,C=US", //
 Distinguished Name (DN) of the certificate authority that issued this certificate
        "serialNumber": "4a:3f:5c:92:d1:e8:7b:6c", // Unique serial number assigned by
 the issuing CA (hexadecimal)
        "validity": {
          "notBefore": "2024-01-15T00:00:00Z", // Certificate validity start date (ISO
 8601 format)
          "notAfter": "2025-01-14T23:59:59Z"   // Certificate expiration date (ISO 8601
 format)
        },
        "sha256Fingerprint": "a1b2c3d4e5f6...abc123def456", // SHA-256 hash of the
 certificate (64 hex characters)
      },
    },
  },
}
```

## 연결 함수 연결

연결 함수를 생성한 후에는 이를 라이브 단계에 게시하고 배포와 연결해야 합니다.

연결 함수를 게시하고 연결하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 함수를 선택합니다.
3. 연결 함수 탭을 선택하고 연결 함수를 선택합니다.
4. 게시를 선택하여 라이브 단계로 이동합니다.
5. 게시 섹션 아래의 연결된 배포 테이블에서 연결 추가를 선택합니다.
6. 연결하려는 뷰어 mTLS가 활성화된 배포를 선택합니다.

또는 배포 세부 정보 페이지에서 게시된 연결 함수를 연결할 수도 있습니다.

1. 모든 배포가 나열된 콘솔 홈 페이지로 이동합니다.
2. 연결하려는 배포를 선택합니다.
3. 일반 탭을 선택합니다.
4. 설정 섹션에서 편집을 선택합니다.

5. 연결성 섹션에서 뷰어 상호 인증(mTLS)을 찾습니다.
6. 연결 함수에서 함수를 선택합니다.
7. 변경 사항 저장을 선택합니다.

연결 함수를 연결하려면(AWS CLI)

다음 예제에서는 연결 함수를 배포와 연결하는 방법을 보여줍니다.

```
// DistributionConfig:
{
   ...other settings,
    "ConnectionFunctionAssociation": {
        "Id": "cf_30c2CV2elHwCoInb3LtcaUJkZeD"
    }
}
```

## 연결 함수 사용 사례

연결 함수를 사용하면 다음과 같은 여러 고급 mTLS 사용 사례를 사용할 수 있습니다.

- 인증서 속성 검증 - 조직 단위 요구 사항 또는 주체 대체 이름 패턴과 같은 클라이언트 인증서의 특정 필드를 확인합니다.
- 인증서 해지 확인 - KeyValueStore를 사용하여 해지된 인증서 일련 번호를 저장하는 사용자 지정 인증서 해지 확인을 구현합니다.
- IP 기반 인증서 정책 - 클라이언트 IP 주소 또는 지리적 제한에 따라 다른 인증서 정책을 적용합니다.
- 다중 테넌트 검증 - 호스트 이름 또는 인증서 속성에 따라 다양한 인증서 요구 사항이 적용되는 테넌트별 검증 규칙을 구현합니다.

> **Note**
>
> 연결 함수는 TLS 핸드셰이크 중에 클라이언트 연결당 한 번 실행됩니다.
> 연결 함수는 HTTP 요청/응답을 수정하지 않고 연결만 허용하거나 거부할 수 있습니다.
> 라이브 단계 함수(게시됨)만 배포와 연결할 수 있습니다.
> 각 배포에는 최대 하나의 연결 함수가 있을 수 있습니다.

## 다음 단계

연결 함수를 CloudFront 배포와 연결한 후 선택적 설정을 구성하여 mTLS 구현의 동작을 사용자 지정할 수 있습니다. 선택적 클라이언트 인증서 검증 모드와 같은 추가 설정을 구성하는 방법에 대한 자세한 지침은 추가 설정 구성 섹션을 참조하세요.

# 추가 설정 구성

기본 상호 TLS 인증을 활성화한 후 특정 사용 사례 및 요구 사항에 맞게 인증 동작을 사용자 지정하도록 추가 설정을 구성할 수 있습니다.

## 클라이언트 인증서 검증 선택적 모드

CloudFront는 제공된 클라이언트 인증서를 검증하지만 인증서를 제시하지 않는 클라이언트에 대한 액세스를 허용하는 선택적 클라이언트 인증서 검증 모드를 제공합니다.

선택적 모드 동작

- 유효한 인증서가 있는 클라이언트에 대한 연결을 부여합니다(잘못된 인증서는 거부됨).
- 인증서 없이 클라이언트에 연결 허용
- 단일 배포를 통해 혼합 클라이언트 인증 시나리오를 허용합니다.

선택적 모드는 mTLS 인증으로 점진적으로 마이그레이션하거나, 인증서가 있는 클라이언트 및 인증서가 없는 클라이언트를 지원하거나, 레거시 클라이언트와의 이전 버전 호환성을 유지하는 데 적합합니다.

> **Note**
>
> 선택적 모드에서는 클라이언트가 인증서를 제시하지 않더라도 연결 함수가 계속 간접 호출됩니다. 이를 통해 클라이언트 IP 주소 로깅 또는 인증서 제시 여부에 따라 다른 정책 적용과 같은 사용자 지정 로직을 구현할 수 있습니다.

선택적 모드를 구성하려면(콘솔)

1. 배포 설정에서 일반 탭으로 이동하여 편집을 선택합니다.
2. 연결성 컨테이너 내의 뷰어 상호 인증(mTLS) 섹션으로 스크롤합니다.
3. 클라이언트 인증서 검증 모드에서 선택 사항을 선택합니다.
4. 변경 내용을 저장합니다.

선택적 모드를 구성하려면(AWS CLI)

다음 예제에서는 선택적 모드를 구성하는 방법을 보여줍니다.

```
"ViewerMtlsConfig": {
   "Mode": "optional",
   ...other settings
}
```

## 인증 기관 광고

AdvertiseTrustStoreCaNames 필드는 TLS 핸드셰이크 중에 CloudFront가 신뢰할 수 있는 CA 이름 목록을 클라이언트에 전송할지 여부를 제어하여 클라이언트가 적절한 인증서를 선택할 수 있도록 지원합니다.

CA 광고를 구성하려면(콘솔)

1. 배포 설정에서 일반 탭으로 이동하여 편집을 선택합니다.
2. 연결성 컨테이너 내의 뷰어 상호 인증(mTLS) 섹션으로 스크롤합니다.
3. 트러스트 스토어 CA 이름 광고 확인란을 선택하거나 선택 취소합니다.
4. 변경 사항 저장을 선택합니다.

CA 광고를 구성하려면(AWS CLI)

다음 예제에서는 CA 광고를 활성화하는 방법을 보여줍니다.

```
"ViewerMtlsConfig": {
   "Mode": "required", // or "optional"
   "TrustStoreConfig": {
      "AdvertiseTrustStoreCaNames": true,
      ...other settings
   }
}
```

## 인증서 만료 처리

IgnoreCertificateExpiry 속성은 CloudFront가 만료된 클라이언트 인증서에 응답하는 방법을 결정합니다. 기본적으로 CloudFront는 만료된 클라이언트 인증서를 거부하지만 필요할 때 수락하도록 구성할

수 있습니다. 이는 일반적으로 즉시 업데이트할 수 없는 인증서가 만료된 디바이스를 대상으로 활성화됩니다.

인증서 만료 처리를 구성하려면(콘솔)

1. 배포 설정에서 일반 탭으로 이동하여 편집을 선택합니다.
2. 연결성 컨테이너의 뷰어 상호 인증(mTLS) 섹션으로 스크롤합니다.
3. 인증서 만료 날짜 무시 확인란을 선택하거나 선택 취소합니다.
4. 변경 사항 저장을 선택합니다.

인증서 만료 처리를 구성하려면(AWS CLI)

다음 예제에서는 인증서 만료를 무시하는 방법을 보여줍니다.

```
"ViewerMtlsConfig": {
  "Mode": "required", // or "optional"
  "TrustStoreConfig": {
     "IgnoreCertificateExpiry": false,
     ...other settings
  }
}
```

> **Note**
>
> IgnoreCertificateExpiry는 인증서 검증 날짜에만 적용됩니다. 다른 모든 인증서 검증 검사(신뢰 체인, 서명 검증)는 계속 적용됩니다.

## 다음 단계

추가 설정을 구성한 후 헤더 전달을 설정하여 인증서 정보를 오리진에 전달하고, 연결 함수 및 KeyValueStore를 사용하여 인증서 해지를 구현하고, 모니터링을 위해 연결 로그를 활성화할 수 있습니다. 오리진에 인증서 정보를 전달하는 방법에 대한 자세한 내용은 오리진에 헤더 전달을 참조하세요.

# 오리진으로 전달된 캐시 정책에 대한 뷰어 mTLS 헤더

상호 TLS 인증을 사용하는 경우, CloudFront는 클라이언트 인증서에서 정보를 추출하여 오리진에 HTTP 헤더로 전달할 수 있습니다. 이렇게 하면 오리진 서버가 인증서 검증 로직을 구현하지 않고도 인증서 세부 정보에 액세스할 수 있습니다.

캐시 동작을 생성하는 데 사용할 수 있는 헤더는 다음과 같습니다.

| 헤더 이름 | 설명 | 예시 값 |
| --- | --- | --- |
| CloudFront-Viewer-Cert-Serial-Number | 인증서 일련 번호의 16진수 표현 | 4a:3f:5c:92:d1:e8:7b:6c |
| CloudFront-Viewer-Cert-Issuer | 발급자 고유 이름(DN)의 RFC2253 문자열 표현 | CN=rootcamtls.com,OU=rootCA,O=mTLS,L=Seattle,ST=Washington,C=US |
| CloudFront-Viewer-Cert-Subject | 주체 고유 이름(DN)의 RFC2253 문자열 표현 | CN=client_.com,OU=client-3,O=mTLS,ST=Washington,C=US |
| CloudFront-Viewer-Cert-Present | 인증서가 있는지 여부를 나타내는 1(존재) 또는 0(존재하지 않음)입니다. 이 값은 필수 모드에서 항상 1입니다. | 1 |
| CloudFront-Viewer-Cert-Sha256 | 클라이언트 인증서의 SHA256 해시 | 01fbf94fef5569753420c349f49adbfd80af5275377816e3ab1fb371b29cb586 |

오리진 요청의 경우 캐시 동작에 사용할 수 있는 위의 헤더 외에도 두 개의 추가 헤더가 제공됩니다.

| 헤더 이름 | 설명 | 예시 값 |
| --- | --- | --- |
| CloudFront-Viewer-Cert-Validity | notBefore 및 notAfter 날짜의 ISO8601 형식 | CloudFront-Viewer-Cert-Validity: NotBefore=2023-09-21T01:50:17Z;NotAfter=2024-09-20T01:50:17Z |
| CloudFront-Viewer-Cert-Pem | 리프 인증서의 URL 인코딩 PEM 형식 | CloudFront-Viewer-Cert-Pem: -----BEGIN%20CERTIFICATE-----%0AMIIG<...reduced...>NmrUlw%0A-----END%20CERTIFICATE-----%0A |

## 헤더 전달 구성

### 콘솔

확인 모드에서 CloudFront는 모든 뷰어 요청에 CloudFront-Viewer-Cert-* 헤더를 자동으로 추가합니다. 이러한 헤더를 오리진에 전달하려면 다음을 수행합니다.

1. 기본 목록 배포 페이지에서 뷰어 mTLS가 활성화된 배포를 선택하고 동작 탭으로 이동합니다.
2. 캐시 동작을 선택한 다음 편집을 선택합니다.
3. 오리진 요청 정책 섹션에서 정책 생성을 선택하거나 기존 정책을 선택합니다.
4. 오리진 요청 정책에 다음 헤더가 포함되어 있는지 확인합니다.
   - CloudFront-Viewer-Cert-Serial-Number
   - CloudFront-Viewer-Cert-Issuer
   - CloudFront-Viewer-Cert-Subject
   - CloudFront-Viewer-Cert-Present
   - Cloudfront-Viewer-Cert-Sha256
   - CloudFront-Viewer-Cert-Validity
   - CloudFront-Viewer-Cert-Pem
5. 생성(새 정책의 경우) 또는 변경 사항 저장(기존 정책의 경우)을 선택합니다.
6. 캐시 동작 내에서 정책을 선택하고 변경 사항을 저장합니다.

### AWS CLI 사용

다음 예제에서는 확인 모드를 위한 mTLS 헤더가 포함된 오리진 요청 정책을 생성하는 방법을 보여줍니다.

```
aws cloudfront create-origin-request-policy \
  --origin-request-policy-config '{
    "Name": "MTLSHeadersPolicy",
    "HeadersConfig": {
      "HeaderBehavior": "whitelist",
      "Headers": {
        "Quantity": 5,
        "Items": [
          "CloudFront-Viewer-Cert-Serial-Number",
          "CloudFront-Viewer-Cert-Issuer",
```

```
          "CloudFront-Viewer-Cert-Subject",
          "CloudFront-Viewer-Cert-Validity",
          "CloudFront-Viewer-Cert-Pem"
        ]
      }
    },
    "CookiesConfig": {
      "CookieBehavior": "none"
    },
    "QueryStringsConfig": {
      "QueryStringBehavior": "none"
    }
  }'
```

## 헤더 처리 고려 사항

인증서 헤더로 작업할 때는 다음 모범 사례를 고려하세요.

- 헤더 검증: 추가 보안 조치로 오리진의 인증서 헤더 값을 확인합니다.
- 헤더 크기 제한: 오리진 서버가 처리할 수 있도록 PEM 인증서 헤더의 크기를 키울 수 있습니다.
- 캐시 고려 사항: 캐시 키에서 인증서 헤더를 사용하면 캐시 조각화가 증가합니다.
- 교차 오리진 요청: 애플리케이션에서 CORS를 사용하는 경우 인증서 헤더를 허용하도록 구성해야 할 수 있습니다.

## 다음 단계

헤더 전달을 구성한 후 CloudFront 연결 함수 및 KeyValueStore를 사용하여 인증서 해지 확인을 구현할 수 있습니다. 해지 확인 구현에 대한 자세한 내용은 CloudFront 연결 함수 및 KVS를 사용한 해지 섹션을 참조하세요.

# CloudFront 연결 함수 및 KVS를 사용한 해지

CloudFront 연결 함수를 KeyValueStore와 결합하여 상호 TLS 인증에 대한 인증서 해지 확인을 구현할 수 있습니다. 이 접근 방식은 CloudFront의 기본 제공 인증서 검증을 보완하는 확장형 실시간 인증서 해지 메커니즘을 제공합니다.

연결 함수는 CloudFront 엣지 로케이션에서 TLS 연결 설정 중에 실행되는 JavaScript 함수로, mTLS 인증을 위한 사용자 지정 인증서 검증 로직을 구현할 수 있습니다. 연결 함수에 대한 자세한 내용은 CloudFront 연결 함수 연결 섹션을 참조하세요.

## 연결 함수에서 인증서 해지 작동 방식

CloudFront의 표준 인증서 검증은 인증서 체인, 서명 및 만료를 확인하지만 내장 인증서 해지 확인은 포함하지 않습니다. 연결 함수를 사용하면 TLS 핸드셰이크 중에 사용자 지정 해지 확인을 구현할 수 있습니다.

인증서 해지 프로세스는 다음과 같이 작동합니다.

1. 해지된 인증서 일련 번호를 CloudFront KeyValueStore에 저장합니다.
2. 클라이언트가 인증서를 제시하면 연결 함수가 간접 호출됩니다.
3. 함수는 KeyValueStore에 대해 인증서의 일련 번호를 확인합니다.
4. 스토어에서 일련 번호를 찾으면 인증서가 해지됩니다.
5. 함수는 해지된 인증서에 대한 연결을 거부합니다.

이 접근 방식은 CloudFront의 글로벌 엣지 네트워크에서 해지 확인을 실시간에 가깝게 제공합니다.

## 해지된 인증서에 대한 KeyValueStore 설정

먼저 KeyValueStore를 생성하여 해지된 인증서의 일련 번호를 저장합니다.

KeyValueStore를 생성하는 방법(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 키 값 스토어를 선택합니다.
3. 키 값 스토어 생성을 선택합니다.
4. 키 값 스토어의 이름(예: revoked-certificates)을 입력합니다.
5. (선택 사항) 설명을 추가합니다.
6. 키 값 스토어 생성을 선택합니다.

KeyValueStore를 생성하려면(AWS CLI)

다음 예제에서는 KeyValueStore를 생성하는 방법을 보여줍니다.

```
aws cloudfront create-key-value-store \
  --name "revoked-certificates" \
  --comment "Store for revoked certificate serial numbers"
```

## 해지된 인증서 일련 번호 가져오기

KeyValueStore를 생성한 후 해지된 인증서의 일련 번호를 가져와야 합니다.

### 해지 데이터 준비

해지된 인증서 일련 번호를 사용하여 JSON 파일을 생성합니다.

```
{
  "data": [
    {
      "key": "ABC123DEF456",
      "value": ""
    },
    {
      "key": "789XYZ012GHI",
      "value": ""
    }
  ]
}
```

### S3에서 가져오기

1. 이 JSON 파일을 S3 버킷에 업로드합니다.
2. 파일을 KeyValueStore로 가져옵니다.

```
aws cloudfront create-key-value-store \
  --name "revoked-certificates" \
  --import-source '{
    "SourceType": "S3",
    "SourceARN": "arn:aws:s3:::amzn-s3-demo-bucket1/revoked-serials.json"
  }'
```

## 해지 확인을 위한 연결 함수 생성

KeyValueStore에 대해 인증서 일련 번호를 확인하는 연결 함수를 생성합니다.

### 연결 함수 코드 예제

다음 예제에서는 인증서 해지 확인을 수행하는 연결 함수를 보여줍니다.

```
import cf from 'cloudfront';

async function connectionHandler(connection) {
    const kvsHandle = cf.kvs();

    // Get client certificate serial number
    const clientSerialNumber =
 connection.clientCertificate.certificates.leaf.serialNumber;

    // Check if the serial number exists in the KeyValueStore
    const isRevoked = await kvsHandle.exists(clientSerialNumber.replaceAll(':', ''));

    if (isRevoked) {
        console.log(`Certificate ${clientSerialNumber} is revoked. Denying
 connection.`);
        connection.logCustomData(`REVOKED:${clientSerialNumber}`);
        connection.deny();
    } else {
        console.log(`Certificate ${clientSerialNumber} is valid. Allowing
 connection.`);
        connection.allow();
    }

}
```

연결 함수를 생성하려면(AWS CLI)

다음 예제에서는 KeyValueStore 연결을 사용하여 연결 함수를 생성하는 방법을 보여줍니다.

```
aws cloudfront create-connection-function \
  --name "revocation-checker" \
  --connection-function-config '{
      "Comment": "Certificate revocation checking function",
      "Runtime": "cloudfront-js-2.0",
      "KeyValueStoreAssociations": {
          "Quantity": 1,
          "Items": [
              {
                  "KeyValueStoreARN": "arn:aws:cloudfront::123456789012:key-value-
store/revoked-certificates"
```

```
                }
            ]
        }
    }' \
    --connection-function-code fileb://revocation-checker.js
```

### 함수를 배포에 연결합니다.

연결 함수를 생성하고 게시한 후 CloudFront 연결 함수 연결 섹션에 설명된 대로 mTLS 지원 CloudFront 배포와 연결합니다.

## 연결 로그를 사용한 관찰성

CloudFront 연결 로그는 상호 TLS 인증 이벤트에 대한 자세한 가시성을 제공하므로 인증서 검증을 모니터링하고, 연결 시도를 추적하고, 인증 문제를 해결할 수 있습니다.

### 연결 로그란 무엇인가요?

연결 로그는 상호 TLS 지원 배포의 TLS 핸드셰이크 및 인증서 검증에 대한 자세한 정보를 캡처합니다. HTTP 요청 정보를 기록하는 표준 액세스 로그와 달리 연결 로그는 특히 다음을 포함한 TLS 연결 설정 단계에 중점을 둡니다.

- 연결 상태(성공/실패)
- 클라이언트 인증서 세부 정보
- TLS 프로토콜 및 암호 정보
- 연결 타이밍 지표
- 연결 함수의 사용자 지정 데이터

이러한 로그는 인증서 기반 인증 이벤트에 대한 포괄적인 가시성을 제공하여 보안을 모니터링하고 문제를 해결하며 규정 준수 요구 사항을 충족하는 데 도움이 됩니다.

### 연결 로그 활성화

연결 로그는 상호 TLS 인증이 활성화된 배포에만 사용할 수 있습니다. CloudWatch Logs, Amazon Data Firehose 및 Amazon S3를 비롯한 여러 대상으로 연결 로그를 전송할 수 있습니다.

#### 사전 조건

연결 로그를 활성화하기 전에:

- CloudFront 배포에 대한 상호 TLS 구성
- CloudFront 배포에 대한 연결 로그 활성화
- 선택한 로깅 대상에 필요한 권한이 있는지 확인합니다.
- 교차 계정 전송의 경우 적절한 IAM 정책을 구성합니다.

연결 로그를 활성화하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포 목록에서 mTLS 지원 배포를 선택합니다.
3. Logging(로깅) 탭을 선택합니다.
4. 추가를 선택합니다.
5. 로그를 수신할 서비스를 선택합니다.
   - CloudWatch 로그
   - Firehose –
   - Amazon S3
6. 대상에서 선택한 서비스의 리소스를 선택합니다.
   - CloudWatch Logs에서 로그 그룹 이름을 입력합니다.
   - Firehose의 경우 Firehose 전송 스트림을 선택합니다.
   - Amazon S3의 경우 버킷 이름을 입력합니다(접두사 포함은 선택 사항).
7. (선택 사항) 추가 설정 구성:
   - 필드 선택: 포함할 특정 로그 필드를 선택합니다.
   - 출력 형식: JSON, 일반, w3c, 원시 또는 Parquet(S3만 해당) 중에서 선택합니다.
   - 필드 구분 기호: 로그 필드를 구분하는 방법을 지정합니다.
8. 변경 사항 저장을 선택합니다

연결 로그를 활성화하려면(AWS CLI)

다음 예제에서는 CloudWatch API를 사용하여 연결 로그를 활성화하는 방법을 보여줍니다.

```
# Step 1: Create a delivery source
aws logs put-delivery-source \
```

```
  --name "cf-mtls-connection-logs" \
  --resource-arn "arn:aws:cloudfront::123456789012:distribution/E1A2B3C4D5E6F7" \
  --log-type CONNECTION_LOGS

# Step 2: Create a delivery destination
aws logs put-delivery-destination \
  --name "s3-destination" \
  --delivery-destination-configuration \
  "destinationResourceArn=arn:aws:s3:::amzn-s3-demo-bucket1"

# Step 3: Create the delivery
aws logs create-delivery \
  --delivery-source-name "cf-mtls-connection-logs" \
  --delivery-destination-arn "arn:aws:logs:us-east-1:123456789012:delivery-
destination:s3-destination"
```

> **Note**
>
> CloudWatch API를 사용하는 경우 로그를 다른 리전으로 전송할 때도 미국 동부(버지니아 북부) 리전(us-east-1)을 지정해야 합니다.

## 연결 로그 필드

연결 로그에는 각 TLS 연결 시도에 대한 세부 정보가 포함됩니다.

| 필드 | 설명 | 예제 |
| --- | --- | --- |
| `eventTimestamp` | 연결이 설정되거나 실패한 경우 ISO 8601 타임스탬프 | 1731620046814 |
| `connectionId` | TLS 연결의 고유 식별자 | `oLHiEKbQSn8lkvJfA3D4gFowK3_iZ0g4i5nMUjE1Akod8TuAzn5nzg==` |
| `connectionStatus` | mTLS 연결 시도의 상태입니다. | `Success` 또는 `Failed` |
| `clientIp` | 연결 클라이언트의 IP 주소 | `2001:0db8:85a3:0000:0000:8a2e:0370:7334` |

| 필드 | 설명 | 예제 |
| --- | --- | --- |
| clientPort | 클라이언트에서 사용하는 포트 | 12137 |
| serverIp | CloudFront 엣지 서버의 IP 주소 | 99.84.71.136 |
| distributionId | CloudFront 배포 ID | E2DX1SLDPK0123 |
| distributionTenantId | CloudFront 배포 테넌트 ID(해당하는 경우) | dt_2te1Ura9X3R2iCGNjW123 |
| tlsProtocol | 사용된 TLS 프로토콜 버전 | TLSv1.3 |
| tlsCipher | 연결에 사용되는 TLS 암호 제품군 | TLS_AES_128_GCM_SHA256 |
| tlsHandshakeDuration | TLS 핸드셰이크 기간(밀리초) | 153 |
| tlsSni | TLS 핸드셰이크의 서버 이름 표시 값 | d111111abcdef8.cloudfront.net |
| clientLeafCertSerialNumber | 클라이언트 인증서의 일련 번호 | 00:b1:43:ed:93:d2:d8:f3:9d |
| clientLeafCertSubject | 클라이언트 인증서의 제목 필드 | C=US, ST=WA, L=Seattle, O=Amazon.com, OU=CloudFront, CN=client.test.mtls.net |
| clientLeafCertIssuer | 클라이언트 인증서의 발급자 필드 | C=US, ST=WA, L=Seattle, O=Amazon.com, OU=CloudFront, CN=test.mtls.net |

| 필드 | 설명 | 예제 |
| --- | --- | --- |
| clientLeafCertValidity | 클라이언트 인증서의 유효 기간 | NotBefore=2025-06-05T23:28:21Z;NotAfter=2125-05-12T23:28:21Z |
| connectionLogCustomData | 연결 함수를 통해 추가된 사용자 지정 데이터 | REVOKED:00:b1:43:ed:93:d2:d8:f3:9d |

## 연결 오류 코드

```
Failed:ClientCertMaxChainDepthExceeded
Failed:ClientCertMaxSizeExceeded
Failed:ClientCertUntrusted
Failed:ClientCertNotYetValid
Failed:ClientCertExpired
Failed:ClientCertTypeUnsupported
Failed:ClientCertInvalid
Failed:ClientCertIntentInvalid
Failed:ClientCertRejected
Failed:ClientCertMissing
Failed:TcpError
Failed:TcpTimeout
Failed:ConnectionFunctionError
Failed:ConnectionFunctionDenied
Failed:Internal
Failed:UnmappedConnectionError
```

연결이 실패하면 CloudFront는 특정 사유 코드를 기록합니다.

| 코드 | 설명 |
| --- | --- |
| ClientCertMaxChainDepthExceeded | 최대 인증서 체인 깊이 초과 |
| ClientCertMaxSizeExceeded | 최대 인증서 크기 초과 |
| ClientCertUntrusted | 클라이언트 인증서를 신뢰할 수 없음 |

| 코드 | 설명 |
| --- | --- |
| ClientCertNotYetValid | 인증서가 아직 유효하지 않음 |
| ClientCertExpired | 인증서가 만료됨 |
| ClientCertTypeUnsupported | 클라이언트 인증서 유형이 지원되지 않음 |
| ClientCertInvalid | 인증서가 잘못됨 |
| ClientCertIntentInvalid | 인증서 의도가 잘못됨 |
| ClientCertRejected | 사용자 지정 검증에서 거부된 인증서 |
| ClientCertMissing | 인증서 누락 |
| TcpError | 연결을 설정하는 동안 오류 발생 |
| TcpTimeout | 제한 시간 내에 연결을 설정할 수 없음 |
| ConnectionFunctionError | 연결 함수 실행 중에 발견되지 않은 예외 발생 |
| 내부 | 내부 서비스 오류 발생 |
| UnmappedConnectionError | 다른 범주에 속하지 않는 오류 발생 |

# CloudFront에 대한 상호 TLS(오리진)

상호 TLS 인증(상호 전송 계층 보안 인증 - mTLS)은 양방향 인증서 기반 인증을 요구하여 표준 TLS 인증을 확장하는 보안 프로토콜로, 클라이언트와 서버 모두 보안 연결을 설정하기 전에 자격 증명을 증명해야 합니다.

## 뷰어 mTLS와 오리진 mTLS 비교

뷰어와 CloudFront 배포(뷰어 mTLS) 간에 또는 CloudFront 배포와 오리진(오리진 mTLS) 간에 상호 인증(mTLS)을 활성화할 수 있습니다. 이 문서는 오리진 mTLS 구성에 관한 것입니다. 뷰어 mTLS 구성은 CloudFront에 대한 상호 TLS 인증(뷰어 mTLS) 섹션을 참조하세요.

상호 TLS(오리진)를 통해 CloudFront가 클라이언트 인증서를 사용하여 오리진 서버에 자체 인증을 수행할 수 있습니다. 상호 TLS(오리진)를 사용하면 승인된 CloudFront 배포만 애플리케이션 서버와의 연결을 설정할 수 있으므로 무단 액세스 시도가 방지됩니다.

> **Note**
>
> 상호 TLS(오리진) 연결에서 CloudFront는 클라이언트 역할을 하며 TLS 핸드셰이크 중에 오리진 서버에 클라이언트 인증서를 제시합니다. CloudFront는 클라이언트 인증서의 유효성 또는 해지 상태에 대한 검증을 수행하지 않습니다. 이는 오리진 서버의 책임입니다. 오리진 인프라는 보안 요구 사항에 따라 트러스트 스토어와 대조하여 클라이언트 인증서를 검증하고, 인증서 만료를 확인하고, 해지 확인(예: CRL 또는 OCSP 검증)을 수행하도록 구성되어야 합니다. CloudFront의 역할은 인증서를 제공하는 것에 국한됩니다. 모든 인증서 검증 로직 및 보안 정책은 오리진 서버에서 적용됩니다.

## 작동 방식

CloudFront와 오리진 사이의 표준 TLS 핸드셰이크에서는 오리진 서버만 CloudFront에 ID를 증명하는 인증서를 제공합니다. 상호 TLS(오리진)를 사용하면 인증 프로세스가 양방향으로 이루어집니다. CloudFront가 오리진 서버에 연결을 시도할 때 CloudFront는 TLS 핸드셰이크 중에 클라이언트 인증서를 요청합니다. 오리진 서버는 보안 연결을 설정하기 전에 트러스트 스토어와 대조하여 이 인증서를 검증합니다.

## 사용 사례

상호 TLS(오리진)는 기존 인증 방법으로 운영 오버헤드가 발생하는 몇 가지 중요한 보안 시나리오를 해결합니다.

- 하이브리드 및 멀티 클라우드 보안 - CloudFront와 AWS 외부에서 호스팅되는 오리진 또는 AWS에 있는 퍼블릭 오리진 간의 연결을 보호할 수 있습니다. 따라서 IP 허용 목록 또는 사용자 지정 헤더 솔루션을 관리할 필요가 없으므로 AWS, 온프레미스 데이터 센터 및 서드파티 공급자 간에 일관된 인증서 기반 인증을 제공할 수 있습니다. 분산 인프라를 운영하는 미디어 회사, 소매업체 및 기업은 전체 인프라에서 표준화된 보안 제어를 활용할 수 있습니다.
- B2B API 및 백엔드 보안 - CloudFront의 성능 이점을 유지하면서 직접 액세스 시도로부터 백엔드 API 및 마이크로서비스를 보호할 수 있습니다. 인증 요구 사항이 엄격한 SaaS 플랫폼, 결제 처리 시스템 및 엔터프라이즈 애플리케이션은 API 요청이 승인된 CloudFront 배포에서만 시작되는지 확인하여 중간자 공격 및 무단 액세스 시도를 방지할 수 있습니다.

# 중요: 오리진 서버 요구 사항

상호 TLS(오리진)를 사용하려면 상호 TLS 인증을 지원하도록 오리진 서버를 구성해야 합니다. 오리진 인프라는 다음을 수행할 수 있어야 합니다.

- TLS 핸드셰이크 중 클라이언트 인증서 요청 및 검증
- CloudFront의 클라이언트 인증서를 발급한 인증 기관 인증서로 트러스트 스토어 유지 관리
- 상호 TLS 연결 이벤트 로깅 및 모니터링
- 인증서 검증 정책 관리 및 인증 실패 처리

CloudFront가 클라이언트 측 인증서 제시를 처리하지만, 이러한 인증서를 검증하고 상호 TLS 연결을 관리할 책임은 오리진 서버에 있습니다. CloudFront에서 상호 TLS(오리진)를 활성화하기 전에 오리진 인프라가 올바르게 구성되어 있는지 확인합니다.

## 시작하기

CloudFront와 상호 TLS(오리진)를 구현하려면 AWS Certificate Manager에서 클라이언트 인증서를 가져오고, 상호 TLS를 요구하도록 오리진 서버를 구성하고, CloudFront 배포에서 상호 TLS(오리진)를 활성화해야 합니다. 다음 섹션에서는 각 구성 작업에 대한 단계별 지침을 제공합니다.

주제

- AWS Certificate Manager를 사용한 인증서 관리
- CloudFront 배포에 상호 TLS(오리진) 활성화
- 상호 TLS(오리진)와 함께 CloudFront Functions 사용

## AWS Certificate Manager를 사용한 인증서 관리

AWS Certificate Manager(ACM)는 상호 TLS(오리진) 인증 중에 CloudFront가 오리진 서버에 제시하는 클라이언트 인증서를 저장합니다.

### 인증 기관 지원

CloudFront 상호 TLS(오리진)는 TLS 클라이언트 인증을 위해 확장 키 사용(EKU)이 포함된 클라이언트 인증서를 필요로 합니다. 이 요구 사항으로 인해 인증 기관에서 인증서를 발급하고 AWS Certificate Manager로 가져와야 합니다. 상호 TLS(오리진) 클라이언트 인증서에는 ACM의 자동 인증서 프로비저닝 및 갱신 기능을 사용할 수 없습니다. CloudFront 상호 TLS(오리진)는 다음 두 소스의 클라이언트 인증서를 지원합니다.

- AWS 프라이빗 인증 기관: 확장 키 사용 필드에 TLS 클라이언트 인증이 포함된 인증서 템플릿(예: EndEntityClientAuthCertificate 템플릿)을 사용하여 AWS Private CA에서 인증서를 발급할 수 있습니다. AWS Private CA에서 인증서를 발급한 후에는 미국 동부(버지니아 북부) 리전(us-east-1)의 ACM으로 인증서를 가져와야 합니다. 이 접근 방식을 사용하면 인증서 수명 주기 관리에 대한 제어력을 유지하면서 AWS Private CA의 보안 이점을 누릴 수 있습니다.
- 서드파티 프라이빗 인증 기관: 기존 프라이빗 인증 기관 인프라에서 인증서를 발급하고 ACM으로 가져올 수도 있습니다. 이렇게 하면 CloudFront의 상호 TLS(오리진) 기능을 활용하면서 현재 인증서 관리 프로세스를 유지할 수 있습니다. 인증서는 확장 키 사용 필드에 TLS 클라이언트 인증을 포함해야 하며 인증서, 프라이빗 키 및 인증서 체인이 포함된 PEM 형식이어야 합니다.

> **Important**
>
> AWS 프라이빗 CA와 서드파티 CA 모두에 대해 인증서 만료 날짜를 모니터링하고 만료 전에 갱신된 인증서를 ACM으로 가져오는 것은 사용자의 책임입니다. ACM의 자동 갱신 기능은 상호 TLS(오리진)에 사용되는 가져온 인증서에는 적용되지 않습니다.

## 인증서 요구 사항 및 사양

### 클라이언트 인증서 요구 사항

- 형식: PEM(Privacy Enhanced Mail) 형식
- 구성 요소: 인증서, 프라이빗 키 및 인증서 체인
- 최대 인증서 체인 깊이: 3(리프 인증서 + 중간 인증서 + 루트 인증서)
- 최대 인증서 체인 크기: 64KB
- 인증서 크기: 96KB를 초과할 수 없음
- 최대 프라이빗 키 크기: 5KB(ACM 제한)
- CloudFront 배포 생성 또는 업데이트 API 직접 호출당 추가하거나 수정할 수 있는 최대 고유 오리진 mTLS 인증서 ARN 수: 5
- 리전: 인증서는 미국 동부(버지니아 북부) 리전(us-east-1)의 ACM에 저장되어야 합니다.

### 지원되는 인증서 사양

- 인증서 유형: X.509v3
- 퍼블릭 키 알고리즘:

  - RSA: 2048비트
  - ECDSA: P-256
- 서명 알고리즘:
  - RSA를 사용하는 SHA256, SHA384, SHA512
  - ECDSA를 사용하는 SHA256, SHA384, SHA512
  - MGF1 지원 RSASSA-PSS를 사용하는 SHA256, SHA384, SHA512
- 확장 키 사용(필수): 인증서는 mTLS 용도로 사용하기 위해 TLS 클라이언트 인증으로 설정된 확장 키 사용(EKU) 확장 기능을 필요로 합니다.

서버 인증서 요구 사항

오리진 서버는 상호 TLS 핸드셰이크 중에 공개적으로 신뢰할 수 있는 인증 기관의 인증서를 제시해야 합니다. 오리진 서버 인증서 요구 사항에 대한 자세한 내용은 CloudFront에서 SSL/TLS 인증서를 사용하기 위한 요구 사항을 참조하세요.

인증서 요청 또는 가져오기

상호 TLS(오리진)를 활성화하기 전에 ACM에서 사용할 수 있는 클라이언트 인증서가 있어야 합니다.

AWS Private CA에서 인증서 요청 및 가져오기

사전 조건:

- 계정에 구성된 AWS Private Certificate Authority
- AWS Private CA에서 인증서를 발급할 수 있는 권한
- ACM으로 인증서를 가져올 수 있는 권한
- 사용 사례에 적합한 `Extended key usage:TLS web client authentication`이 포함된 인증서 템플릿 ARN
- OpenSSL, AWS CLI 및 jq(JSON 구문 분석용)를 설치합니다.

PCA에서 인증서를 요청하고 ACM으로 가져오는 방법(AWS CLI)

1. 쉽게 재사용할 수 있도록 변수에 프라이빗 CA ARN을 설정합니다.

```
PCA_ARN="arn:aws:acm-pca:region:account:certificate-authority/12345678..."
```

2. OpenSSL을 사용하여 ECDSA P-256 프라이빗 키(prime256v1 곡선) 및 인증서 서명 요청(CSR)을 생성하고, ACM 가져오기에 필요한 대로 프라이빗 키를 암호화되지 않은 상태로 유지하기 위해 -node 플래그를 사용합니다.

```
openssl req -new -newkey ec -pkeyopt ec_paramgen_curve:prime256v1 -nodes \
    -keyout private.key \
    -out request.csr \
    -subj "/CN=client.example.com"
```

3. AWS Private CA에 CSR을 제출하여 인증서를 발급하면 새로 발급된 인증서의 ARN이 반환됩니다.

```
CERT_ARN=$(aws acm-pca issue-certificate \
    --certificate-authority-arn "$PCA_ARN" \
    --csr fileb://request.csr \
    --signing-algorithm "SHA256WITHECDSA" \
    --validity Value=365,Type="DAYS" \
    --template-arn arn:aws:acm-pca:::template/EndEntityCertificate/V1 \
    --query 'CertificateArn' --output text)
```

4. get-certificate 명령을 사용하여 AWS PCA에서 인증서 번들을 검색합니다. 이 명령은 리프 인증서와 체인을 모두 반환합니다. 그런 다음 jq를 사용하여 이들을 별도의 파일로 분리합니다.

```
# Retrieve the full certificate bundle in JSON format
aws acm-pca get-certificate \
    --certificate-authority-arn "$PCA_ARN" \
    --certificate-arn "$CERT_ARN" \
    --output json > full_cert.json

# Split into Leaf and Chain
jq -r '.Certificate' full_cert.json > leaf_cert.pem
jq -r '.CertificateChain' full_cert.json > cert_chain.pem
```

5. CLI에서 이진 파일 데이터를 올바르게 처리하도록 fileb:// 프로토콜을 사용하여 암호화되지 않은 프라이빗 키, 리프 인증서 및 인증서 체인을 AWS ACM으로 가져옵니다.

```
aws acm import-certificate \
    --certificate fileb://leaf_cert.pem \
    --private-key fileb://private.key \
    --certificate-chain fileb://cert_chain.pem \
    --region us-east-1 \
    --query 'CertificateArn' \
```

```
    --output text
```

서드파티 CA에서 인증서 가져오기

사전 조건:

- 인증 기관에서 발급한 인증서, 암호화되지 않은 프라이빗 키 및 인증서 체인(PEM 형식)
- 인증서에는 TLS 클라이언트 인증을 위한 확장 키 사용이 포함되어야 함
- ACM으로 인증서를 가져올 수 있는 권한

인증서를 ACM으로 가져오는 방법(AWS CLI)

```
aws acm import-certificate \
  --certificate fileb://certificate.pem \
  --private-key fileb://private-key.pem \
  --certificate-chain fileb://certificate-chain.pem \
  --region us-east-1 \
  --query 'CertificateArn' \
  --output text
```

다음 단계

ACM에서 클라이언트 인증서를 획득하거나 가져온 후에는 오리진 서버에서 상호 TLS 인증을 요구하도록 구성하고 CloudFront 배포에서 상호 TLS(오리진)를 활성화할 수 있습니다. CloudFront에서 상호 TLS(오리진)를 활성화하는 방법에 대한 지침은 다음 섹션인 "CloudFront 배포에 상호 TLS(오리진) 활성화"를 참조하세요.

# CloudFront 배포에 상호 TLS(오리진) 활성화

AWS Certificate Manager를 통해 클라이언트 인증서를 획득하고 상호 TLS를 요구하도록 오리진 서버를 구성한 후 CloudFront 배포에서 상호 TLS(오리진)를 활성화할 수 있습니다.

## 사전 조건 및 요구 사항

CloudFront 배포에서 상호 TLS(오리진)를 활성화하기 전에 다음이 있어야 합니다.

- 미국 동부(버지니아 북부) 리전(us-east-1)의 AWS Certificate Manager에 저장된 클라이언트 인증서
- 상호 TLS 인증을 요구하고 클라이언트 인증서를 검증하도록 구성된 오리진 서버
- 공개적으로 신뢰할 수 있는 인증 기관의 인증서를 제시하는 오리진 서버

- CloudFront 배포를 수정할 수 있는 권한
- 상호 TLS(오리진)는 비즈니스, 프리미엄 또는 종량제 요금제에서만 사용할 수 있습니다.

> **Note**
>
> 사용자 지정 오리진(AWS 외부에서 호스팅되는 오리진 포함)과 Application Load Balancer 및 API Gateway와 같은 상호 TLS를 지원하는 AWS 오리진에 대해 상호 TLS(오리진)를 구성할 수 있습니다.

> **Important**
>
> 다음 CloudFront 기능은 상호 TLS(오리진)에서 지원되지 않습니다.
>
> - gRPC 트래픽: 상호 TLS(오리진)가 활성화된 오리진에는 gRPC 프로토콜이 지원되지 않습니다.
> - WebSocket 연결: 상호 TLS(오리진)가 활성화된 오리진에는 WebSocket 프로토콜이 지원되지 않습니다.
> - VPC 오리진: 상호 TLS(오리진)는 VPC 오리진과 함께 사용할 수 없습니다.
> - Lambda@Edge를 사용한 오리진 요청 및 오리진 응답 트리거: 오리진 요청 및 오리진 응답 위치의 Lambda@Edge 함수는 상호 TLS(오리진)에서 지원되지 않습니다.
> - 임베디드 POP: 임베디드 POP에는 상호 TLS(오리진)가 지원되지 않습니다.

## 상호 TLS(오리진) 활성화

오리진별 구성을 사용하면 동일한 배포 내에서 오리진별로 서로 다른 클라이언트 인증서를 지정할 수 있습니다. 이 접근 방식은 오리진별로 인증 요구 사항이 다를 때 유연성을 극대화합니다.

새 배포의 경우(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포 생성을 선택합니다.
3. 요금제 선택: 비즈니스 또는 프리미엄 또는 종량제를 선택합니다(상호 TLS(오리진)는 무료 요금제로 사용할 수 없음).

4. 오리진 설정 섹션에서 오리진 유형을 기타로 선택합니다.
5. 오리진 설정 섹션에서 오리진 설정 사용자 지정을 선택합니다.
6. 첫 번째 오리진(도메인 이름, 프로토콜 등)을 구성합니다.
7. 오리진 구성에서 상호 TLS(오리진)를 찾습니다.
8. 상호 TLS(오리진) 활성화를 켜기로 전환합니다.
9. 클라이언트 인증서의 경우 Certificate Manager에서 AWS 인증서를 선택합니다.
10. (선택 사항) 자체 상호 TLS(오리진) 구성으로 오리진을 더 추가합니다.
11. 나머지 배포 설정을 완료한 다음 배포 생성을 선택합니다.

기존 배포의 경우(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포 목록에서 수정하려는 배포를 선택합니다. (참고: 해당 배포가 프로 또는 프리미엄 또는 종량제 요금제에 포함되어 있는지 확인하세요. 포함되지 않은 경우 상호 TLS(오리진)를 활성화하기 전에 요금제를 업그레이드해야 합니다.)
3. 오리진 탭을 선택합니다.
4. 구성하려는 오리진을 선택하고 편집을 선택합니다.
5. 오리진 설정에서 상호 TLS(오리진)를 찾습니다.
6. 상호 TLS(오리진) 활성화를 켜기로 전환합니다.
7. 클라이언트 인증서의 경우 AWS Certificate Manager에서 인증서를 선택합니다. (참고: EKU(확장 키 사용) 속성이 "TLS 클라이언트 인증"으로 설정된 클라이언트 인증서만 나열됩니다.)
8. 변경 사항 저장을 선택합니다
9. 필요에 따라 추가 오리진에 대해 이 과정을 반복합니다.

## AWS CLI 사용

오리진별 구성의 경우 각 오리진의 구성 내에서 상호 TLS(오리진) 설정을 지정합니다.

```
{
  "Origins": {
    "Quantity": 2,
    "Items": [
      {
```

```
        "Id": "origin-1",
        "DomainName": "api.example.com",
        "CustomOriginConfig": {
          "HTTPSPort": 443,
          "OriginProtocolPolicy": "https-only"
        },
        "OriginMtlsConfig": {
          "ClientCertificateArn": "arn:aws:acm:us-east-1:123456789012:certificate/
cert-1"
        }
      },
      {
        "Id": "origin-2",
        "DomainName": "backend.example.com",
        "CustomOriginConfig": {
          "HTTPSPort": 443,
          "OriginProtocolPolicy": "https-only"
        },
        "OriginMtlsConfig": {
          "CertificateArn": "arn:aws:acm:us-east-1:123456789012:certificate/cert-2"
        }
      }
    ]
  }
}
```

> **Note**
>
> 서버에서 요청하지 않으면 CloudFront가 클라이언트 인증서를 제공하지 않고 연결이 정상적으로 진행될 수 있습니다.

## 다음 단계

CloudFront 배포에서 상호 TLS(오리진)를 활성화한 후 CloudFront 액세스 로그를 사용하여 인증 이벤트를 모니터링할 수 있습니다.

# 상호 TLS(오리진)와 함께 CloudFront Functions 사용

CloudFront Functions는 콘텐츠 전송을 사용자 지정하도록 엣지에서 경량 서버리스 컴퓨팅을 제공합니다. CloudFront Functions와 함께 상호 TLS(오리진)를 사용하는 경우 오리진 선택 및 조작과 관련하여 알아야 할 특정 동작 및 제한 사항이 있습니다.

## 지원되는 CloudFront Functions 작업

CloudFront Functions는 다음과 같은 방법으로 상호 TLS(오리진)이 활성화된 오리진과 상호 작용할 수 있습니다.

updateRequestOrigin()

updateRequestOrigin() 함수는 상호 TLS(오리진)가 활성화된 오리진을 사용하여 작업할 때 제한적인 수정을 지원합니다.

- 상호 TLS(오리진) 오리진 간 전환: 두 오리진이 동일한 클라이언트 인증서를 사용하는 경우 상호 TLS(오리진)를 사용하는 다른 오리진으로 라우팅하도록 요청을 업데이트할 수 있습니다. 이를 통해 상호 TLS 인증을 유지하면서 사용자 지정 라우팅 로직을 구현할 수 있습니다.
- 상호 TLS(오리진) 비활성화: 함수에서 `mTLSConfig: 'off'`를 설정하여 상호 TLS(오리진)가 활성화된 오리진에서 비상호 TLS 오리진으로 전환할 수 있습니다. 이렇게 하면 요청 특성에 따라 상호 TLS 인증을 조건부로 비활성화할 수 있는 유연성이 제공됩니다.

예: 동일한 인증서를 사용하여 상호 TLS(오리진) 오리진 간 전환

```
function handler(event) {
    var request = event.request;

    // Route to different origin based on request path
    if (request.uri.startsWith('/api/v2')) {
        request.origin = {
            domainName: 'api-v2.example.com',
            customHeaders: {},
            // Both origins must use the same certificate
        };
    }

    return request;
}
```

예: 조건부로 상호 TLS(오리진) 비활성화

```
function handler(event) {
    var request = event.request;

    // Disable mTLS for specific paths
```

```
    if (request.uri.startsWith('/public')) {
        request.origin = {
            domainName: 'public-origin.example.com',
            customHeaders: {},
            mTLSConfig: 'off'
        };
    }

    return request;
}
```

### 지원되지 않는 CloudFront Functions 작업

다음 CloudFront Functions 작업은 정식 출시 시점에 상호 TLS(오리진)가 활성화된 오리진을 지원하지 않습니다.

selectRequestOriginById()

`selectRequestOriginById()` 함수는 상호 TLS(오리진)가 활성화된 오리진을 선택할 수 없습니다. 이 함수를 사용하여 상호 TLS(오리진)가 활성화된 오리진을 선택하려고 하면 검증 오류가 발생합니다.

사용 사례에 따라 상호 TLS(오리진)를 사용한 동적 오리진 선택이 필요한 경우, 모든 대상 오리진이 동일한 클라이언트 인증서를 사용하도록 `updateRequestOrigin()`을 대신 사용하세요.

createRequestOriginGroup()

`createRequestOriginGroup()` 함수는 상호 TLS(오리진)가 활성화된 오리진을 포함하는 오리진 그룹 생성을 지원하지 않습니다. 상호 TLS(오리진) 오리진이 있는 오리진 그룹은 CloudFront Functions를 통해 동적으로 생성할 수 없습니다.

상호 TLS(오리진)를 사용한 오리진 장애 조치 기능이 필요한 경우, 함수에서 동적으로 생성하는 대신 CloudFront 배포 설정에서 오리진 그룹을 직접 구성하세요.

## 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공

인터넷을 통해 콘텐츠를 배포하는 많은 기업에서는 유료 사용자 등 일부 사용자용으로 제작된 각종 문서, 비즈니스 데이터, 미디어 스트림 또는 콘텐츠에 대한 액세스를 제한하고자 합니다. CloudFront를 통해 이러한 프라이빗 콘텐츠를 안전하게 제공하려면 다음과 같이 하세요.

- 사용자가 특별한 CloudFront 서명된 URL 또는 서명된 쿠키를 사용하여 프라이빗 콘텐츠에 액세스하도록 합니다.

- 사용자가 오리진 서버(예: Amazon S3 또는 프라이빗 HTTP 서버)에서 직접 콘텐츠에 액세스하는 URL이 아닌 CloudFront URL을 사용하여 콘텐츠에 액세스해야 합니다. 반드시 CloudFront URL을 사용해야 하는 것은 아니지만, 서명된 URL 또는 서명된 쿠키에 지정한 제약 조건을 사용자가 우회하지 못하도록 하려면 사용하는 것이 좋습니다.

자세한 내용은 파일에 대한 액세스 제한 섹션을 참조하세요.

## 프라이빗 콘텐츠를 제공하는 방법

프라이빗 콘텐츠를 제공하도록 CloudFront를 구성하려면 다음 작업을 수행하세요.

1. (권장 선택 사항) 사용자가 CloudFront를 통해서만 콘텐츠에 액세스하도록 요구합니다. 사용하는 방식은 Amazon S3 또는 사용자 지정 오리진 중 어떤 것을 사용하는지에 따라 다릅니다.

   - Amazon S3 – the section called “Amazon S3 오리진에 대한 액세스 제한” 단원을 참조하세요.
   - 사용자 지정 오리진 – 사용자 지정 오리진의 파일에 대한 액세스 제한 단원을 참조하세요.

   사용자 지정 오리진은 Amazon EC2, 웹 사이트 엔드포인트로 구성된 Amazon S3 버킷, Elastic Load Balancing, 자체 HTTP 웹 서버를 포함합니다.
2. 서명된 URL 또는 서명된 쿠키를 생성할 때 사용할 신뢰할 수 있는 키 그룹 또는 신뢰할 수 있는 서명자를 지정합니다. 신뢰할 수 있는 키 그룹을 사용하는 것이 좋습니다. 자세한 내용은 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정 단원을 참조하세요.
3. 서명된 쿠키를 설정하는 `Set-Cookie` 헤더 또는 서명된 URL을 보유한 권한 있는 사용자의 요청에 응답하는 애플리케이션을 작성합니다. 다음 주제 중 하나에 있는 단계를 따릅니다.

   - 서명된 URL 사용
   - 서명된 쿠키 사용

   어떤 방법을 사용할지 확실하지 않은 경우 서명된 URL 또는 서명된 쿠키 사용 결정 단원을 참조하세요.

주제


- 파일에 대한 액세스 제한
- 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정
- 서명된 URL 또는 서명된 쿠키 사용 결정

- 서명된 URL 사용
- 서명된 쿠키 사용
- base64 인코딩 및 암호화를 위한 Linux 명령 및 OpenSSL
- 서명 URL에 대한 서명을 만드는 코드 예제

## 파일에 대한 액세스 제한

프라이빗 콘텐츠에 대한 사용자 액세스를 두 가지 방법으로 제어할 수 있습니다.

- CloudFront 캐시에 있는 파일에 대한 액세스를 제한합니다.
- 다음 중 하나를 수행하여 오리진의 파일에 대한 액세스를 제한합니다.
  - Amazon S3 버킷의 오리진 액세스 제어(OAC)를 설정합니다.
  - 프라이빗 HTTP 서버(사용자 지정 오리진)에 대한 사용자 지정 헤더를 구성합니다.

### CloudFront 캐시에 있는 파일에 대한 액세스 제한

사용자가 서명된 URL 또는 서명된 쿠키를 사용하여 파일에 액세스할 것을 요구하도록 CloudFront를 구성할 수 있습니다. 그런 다음 서명된 URL을 만들어 인증된 사용자에게 배포하거나 인증된 사용자를 위해 `Set-Cookie` 헤더를 보내 서명된 쿠키를 설정하도록 하는 애플리케이션을 개발합니다. (사용자 몇 명에게 적은 수의 파일에 대한 장기 액세스 권한을 부여하려면 서명된 URL을 수동으로 만드는 방법도 있습니다.)

서명된 URL 또는 서명된 쿠키를 만들어 파일 액세스를 제어할 때 다음과 같은 제약 조건을 지정할 수 있습니다.

- URL의 효력이 사라지는 종료 날짜 및 시간
- (선택 사항) URL의 효력이 발생하는 날짜 및 시간
- (선택 사항) 콘텐츠에 액세스할 때 사용할 수 있는 컴퓨터의 IP 주소 또는 주소 범위

퍼블릭–프라이빗 키 페어의 프라이빗 키를 사용하여 서명된 URL 또는 서명된 쿠키의 일부분을 해시 및 서명합니다. 누군가 서명된 URL 또는 서명된 쿠키를 사용하여 파일에 액세스하면 CloudFront는 URL 또는 쿠키의 서명된 부분과 서명되지 않은 부분을 비교합니다. 양쪽이 일치하지 않으면 CloudFront는 파일을 제공하지 않습니다.

URL 또는 쿠키에 서명하려면 RSA 2048 또는 ECDSA 256 프라이빗 키를 사용해야 합니다.

## Amazon S3 버킷에 있는 파일에 대한 액세스 제한

사용자가 지정된 CloudFront 배포를 통해 액세스할 수는 있지만 Amazon S3 URL을 사용하여 직접 액세스할 수는 없도록 Amazon S3 버킷의 콘텐츠를 선택적으로 보호할 수 있습니다. 이렇게 하면 CloudFront를 우회하고 Amazon S3 URL을 사용하여 액세스 제한 콘텐츠에 접근하는 사람을 막을 수 있습니다. 서명된 URL을 사용하기 위해 꼭 필요한 단계는 아니지만 이렇게 하는 것이 좋습니다.

사용자가 CloudFront URL을 통해 콘텐츠에 액세스하도록 요구하려면 다음 작업을 수행합니다.

- CloudFront에 S3 버킷의 파일을 읽을 수 있는 오리진 액세스 제어 권한을 부여합니다.
- 오리진 액세스 제어를 생성하고 CloudFront 배포와 연결합니다.
- 나머지 모든 사람으로부터 Amazon S3 URL을 사용하여 파일을 읽을 수 있는 권한을 제거합니다.

자세한 내용은 the section called “Amazon S3 오리진에 대한 액세스 제한” 섹션을 참조하세요.

## 사용자 지정 오리진의 파일에 대한 액세스 제한

사용자 지정 오리진을 사용하는 경우 선택적으로 사용자 지정 헤더를 설정하여 액세스를 제한할 수 있습니다. CloudFront가 사용자 지정 오리진에서 파일을 가져오려면 표준 HTTP(또는 HTTPS) 요청을 사용하여 CloudFront에서 해당 파일에 액세스할 수 있어야 합니다. 그러나 사용자 지정 헤더를 사용하면 사용자가 직접 액세스할 수 없고 CloudFront를 통해서만 액세스할 수 있도록 콘텐츠에 대한 액세스를 추가로 제한할 수 있습니다. 서명된 URL을 사용하기 위해 꼭 필요한 단계는 아니지만 이렇게 하는 것이 좋습니다.

사용자가 CloudFront를 통해 콘텐츠에 액세스하도록 하려면 CloudFront 배포의 다음 설정을 변경합니다.

오리진 사용자 지정 헤더

오리진에 사용자 지정 헤더를 전달하도록 CloudFront를 구성합니다. 사용자 지정 헤더를 오리진 요청에 추가하도록 CloudFront 구성을(를) 참조하세요.

Viewer Protocol Policy

최종 사용자가 HTTPS를 사용하여 CloudFront에 액세스하도록 배포를 구성합니다. 뷰어 프로토콜 정책을(를) 참조하세요.

오리진 프로토콜 정책

CloudFront가 최종 사용자와 같은 프로토콜을 사용하여 오리진에 요청을 전달하도록 배포를 구성합니다. 프로토콜(사용자 지정 오리진만 해당)을(를) 참조하세요.

이러한 변경을 수행한 후에는 CloudFront에서 전송하도록 구성한 사용자 지정 헤더를 포함하는 요청만 수락하도록 사용자 지정 오리진에서 애플리케이션을 업데이트합니다.

최종 사용자 프로토콜 정책(Viewer Protocol Policy)과 오리진 프로토콜 정책(Origin Protocol Policy)의 조합을 통해 사용자 지정 헤더가 전송 중에 암호화되도록 합니다. 하지만 주기적으로 다음을 수행하여 CloudFront가 오리진에 전달하는 사용자 지정 헤더를 교체하는 것이 좋습니다.

1. CloudFront 배포를 업데이트하여 사용자 지정 오리진에 새 헤더 전달을 시작합니다.
2. 요청이 CloudFront에서 온다는 확인으로 애플리케이션을 업데이트하여 새 헤더를 허용합니다.
3. 요청에 더 이상 교체하는 헤더가 포함되지 않는 경우, 요청이 CloudFront에서 온다는 확인으로 애플리케이션을 업데이트하여 새 헤더를 허용합니다.

## 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정

주제

- 신뢰할 수 있는 키 그룹(권장)과 AWS 계정 사이에서 선택
- 서명자에 대해 키 페어 생성
- 프라이빗 키 재포맷(.NET 및 Java만 해당)
- 배포에 서명자 추가
- 키 페어 교체

서명된 URL 또는 서명된 쿠키를 만들려면 서명자가 필요합니다. 서명자는 CloudFront에 생성하는 신뢰할 수 있는 키 그룹이거나, CloudFront 키 페어가 포함된 AWS 계정입니다. 서명된 URL 및 서명된 쿠키와 함께 신뢰할 수 있는 키 그룹을 사용하는 것이 좋습니다. 자세한 내용은 신뢰할 수 있는 키 그룹(권장)과 AWS 계정 사이에서 선택 단원을 참조하세요.

서명자에는 두 가지 목적이 있습니다.

- 배포에 서명자를 추가하는 즉시 CloudFront에서는 파일에 액세스할 때 서명된 URL 또는 서명된 쿠키를 사용할 것을 최종 사용자에게 요구하기 시작합니다.
- 서명된 URL 또는 서명된 쿠키를 만들 때는 서명자의 키 페어에서 프라이빗 키를 사용하여 URL 또는 쿠키의 일부분에 서명합니다. 제한된 객체에 대한 요청이 들어오면 CloudFront는 URL 또는 쿠키의 서명을 서명되지 않은 URL 또는 쿠키와 비교하여 URL 또는 쿠키가 변조되지 않았는지 확인합니다. 또한 CloudFront는 URL 또는 쿠키가 유효한지, 의미가 있는지를 확인합니다(예: 만료 날짜 및 시간 이전인지 확인).

서명자를 지정할 때 서명자를 캐시 동작에 추가하여 서명된 URL 또는 서명된 쿠키가 필요한 파일을 간접적으로 지정합니다. 배포에 캐시 동작이 한 개뿐이라면 최종 사용자는 반드시 서명된 URL 또는 서명된 쿠키를 사용하여 해당 배포의 파일에 액세스해야 합니다. 여러 가지 캐시 동작을 만들고 일부 캐시 동작에만 서명자를 추가하는 경우, 일부 파일에 액세스할 때만 서명된 URL 또는 서명된 쿠키를 사용하도록 최종 사용자에게 요구할 수 있습니다.

서명된 URL 또는 서명된 쿠키를 만들고 CloudFront 배포에 서명자를 추가할 수 있는 서명자(프라이빗 키)를 지정하려면 다음 작업을 수행하세요.

1. 신뢰할 수 있는 키 그룹 또는 AWS 계정을 서명자로 사용할지 결정합니다. 신뢰할 수 있는 키 그룹을 사용하는 것이 좋습니다. 자세한 내용은 신뢰할 수 있는 키 그룹(권장)과 AWS 계정 사이에서 선택 단원을 참조하세요.
2. 1단계에서 선택한 서명자에 대해 퍼블릭–프라이빗 키 페어를 만듭니다. 자세한 내용은 서명자에 대해 키 페어 생성 단원을 참조하세요.
3. .NET 또는 Java를 사용하여 서명된 URL 또는 서명된 쿠키를 만드는 경우, 프라이빗 키를 다시 포맷해야 합니다. 자세한 내용은 프라이빗 키 재포맷(.NET 및 Java만 해당) 단원을 참조하세요.
4. 서명된 URL 또는 서명된 쿠키를 만들려는 배포에서 서명자를 지정합니다. 자세한 내용은 배포에 서명자 추가 섹션을 참조하세요.

## 신뢰할 수 있는 키 그룹(권장)과 AWS 계정 사이에서 선택

서명된 URL 또는 서명된 쿠키를 사용하려면 서명자가 필요합니다. 서명자는 CloudFront에 생성하는 신뢰할 수 있는 키 그룹이거나, CloudFront 키 페어가 포함된 AWS 계정입니다. 다음과 같은 이유로 신뢰할 수 있는 키 그룹을 사용하는 것이 좋습니다.

- CloudFront 키 그룹을 사용하면 CloudFront 서명된 URL 및 서명된 쿠키에 대한 퍼블릭 키를 관리하기 위해 AWS 계정 루트 사용자를 사용할 필요가 없습니다. AWS 모범 사례에서는 필요하지 않은 경우 루트 사용자를 사용하지 않는 것을 권장합니다.
- CloudFront 키 그룹을 사용하면 CloudFront API를 사용하여 퍼블릭 키, 키 그룹 및 신뢰할 수 있는 서명자를 관리할 수 있습니다. API를 사용하여 키 생성 및 키 회전을 자동화할 수 있습니다. AWS 루트 사용자를 사용할 때는 AWS Management Console을 사용하여 CloudFront 키 페어를 관리해야 하므로 프로세스를 자동화할 수 없습니다.
- CloudFront API를 사용하여 키 그룹을 관리할 수 있으므로 AWS Identity and Access Management(IAM) 권한 정책을 사용하여 다른 사용자가 수행할 수 있는 작업을 제한할 수도 있습니다. 예를 들어 사용자가 퍼블릭 키를 업로드하되 삭제할 수는 없도록 할 수 있습니다. 또는 멀티 팩터 인증을 사용하거나 특정 네트워크에서 요청을 보내거나 특정 날짜 및 시간 범위 내에서 요청을 보내는 등 특정 조건을 충족하는 경우에만 사용자가 퍼블릭 키를 삭제하도록 허용할 수 있습니다.

- CloudFront 키 그룹을 사용하면 더 많은 수의 퍼블릭 키를 CloudFront 배포와 연결할 수 있으므로 퍼블릭 키를 더 유연하게 사용하고 관리할 수 있습니다. 기본적으로 단일 배포에 최대 4개의 키 그룹을 연결할 수 있으며 키 그룹에 최대 5개의 퍼블릭 키를 포함할 수 있습니다.

  AWS 계정 루트 사용자를 사용하여 CloudFront 키 페어를 관리하는 경우 AWS 계정당 최대 두 개의 활성 CloudFront 키 페어만 가질 수 있습니다.

## 서명자에 대해 키 페어 생성

CloudFront 서명된 URL 또는 서명된 쿠키를 만드는 데 사용하는 각 서명자에게는 퍼블릭–프라이빗 키 페어가 있어야 합니다. 서명자는 프라이빗 키를 사용하여 URL 또는 쿠키에 서명하고 CloudFront는 퍼블릭 키를 사용하여 서명을 확인합니다.

키 페어를 생성하는 방법은 신뢰할 수 있는 키 그룹을 서명자(권장)로 사용하는지 또는 CloudFront 키 페어로 사용하는지에 따라 다릅니다. 자세한 내용은 다음 단원을 참조하세요. 생성하는 키 페어는 다음 요구 사항을 충족해야 합니다.

- SSH-2 RSA 2048 또는 ECDSA 256 키 페어여야 합니다.
- base64로 인코딩된 PEM 형식이어야 합니다.

애플리케이션 보안을 위해 키 페어를 주기적으로 교체하는 것이 좋습니다. 자세한 내용은 키 페어 교체 단원을 참조하세요.

신뢰할 수 있는 키 그룹에 대한 키 페어 생성(권장)

신뢰할 수 있는 키 그룹에 대한 키 페어를 생성하려면 다음 단계를 수행합니다.

1. 퍼블릭–프라이빗 키 페어를 생성합니다.
2. CloudFront에 퍼블릭 키를 업로드합니다.
3. CloudFront 키 그룹에 퍼블릭 키를 추가합니다.

자세한 내용은 다음 절차를 참조하세요.

키 페어를 생성하려면

> **Note**
> 다음 단계에서는 키 페어를 생성하는 방법의 예시로 OpenSSL을 사용합니다. RSA 또는 ECDSA 키 페어를 생성하는 방법에는 여러 가지가 있습니다.

1. 다음 예제 명령 중 하나를 실행합니다.
   - 다음 예제 명령은 OpenSSL을 사용하여 길이가 2048비트인 RSA 키 페어를 생성하고 `private_key.pem`이라는 파일에 저장합니다.

     ```
     openssl genrsa -out private_key.pem 2048
     ```

   - 다음 예제 명령어는 OpenSSL을 사용하여 `prime256v1` 곡선을 가진 ECDSA 키 쌍을 생성하고 `private_key.pem`이라는 이름의 파일에 저장합니다.

     ```
     openssl ecparam -name prime256v1 -genkey -noout -out privatekey.pem
     ```

2. 이렇게 만들어진 파일은 퍼블릭 키와 프라이빗 키를 모두 포함합니다. 다음 예제 명령은 `private_key.pem`이라는 파일에서 퍼블릭 키를 추출합니다.

   ```
   openssl rsa -pubout -in private_key.pem -out public_key.pem
   ```

   나중에 다음 절차에서 `public_key.pem` 파일의 퍼블릭 키를 업로드합니다.

퍼블릭 키를 CloudFront에 업로드하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 메뉴에서 퍼블릭 키를 선택합니다.
3. 퍼블릭 키 생성을 선택합니다.
4. 퍼블릭 키 생성 창에서 다음을 수행합니다.
   a. 키 이름에 퍼블릭 키를 식별할 이름을 입력합니다.

b. 키 값에 퍼블릭 키를 붙여 넣습니다. 이전 절차의 단계를 수행한 경우 퍼블릭 키는 `public_key.pem`이라는 파일에 있습니다. 퍼블릭 키의 내용을 복사하여 붙여 넣으려면 다음과 같이 하면 됩니다.

- 다음과 같이 macOS 또는 Linux 명령줄에서 cat 명령을 사용합니다.

```
cat public_key.pem
```

  해당 명령의 출력을 복사한 다음 키 값(Key value) 필드에 붙여 넣습니다.

- 메모장(Windows) 또는 TextEdit(macOS)와 같은 일반 텍스트 편집기로 `public_key.pem` 파일을 엽니다. 파일의 내용을 복사한 다음 키 값(Key value) 필드에 붙여 넣습니다.

c. (선택 사항) 설명(Comment)에 퍼블릭 키에 대한 설명을 추가합니다.

완료되면 추가(Add)를 선택합니다.

5. 퍼블릭 키 ID를 적어 둡니다. 나중에 서명된 URL 또는 서명된 쿠키를 만들 때 `Key-Pair-Id` 필드 값으로 이를 사용합니다.

키 그룹에 퍼블릭 키를 추가하려면

1. 에서 CloudFront 콘솔을 엽니다https://console.aws.amazon.com/cloudfront/v4/home
2. 탐색 메뉴에서 키 그룹(Key groups)을 선택합니다.
3. 키 그룹 추가(Add key group)를 선택합니다.
4. 키 그룹 만들기(Create key group) 페이지에서 다음을 수행합니다.

   a. 키 그룹 이름(Key group name)에 키 그룹을 식별할 이름을 입력합니다.

   b. (선택 사항) 설명(Comment)에 키 그룹에 대한 설명을 입력합니다.

   c. 퍼블릭 키(Public keys)에서 키 그룹에 추가할 퍼블릭 키를 선택한 다음 추가(Add)를 선택합니다. 키 그룹에 추가할 각 퍼블릭 키에 대해 이 단계를 반복합니다.

5. 키 그룹 만들기(Create Key Pair)를 선택합니다.
6. 키 그룹 이름을 적어 둡니다. 나중에 키 그룹을 CloudFront 배포의 캐시 동작과 연결하는 데 이를 사용합니다. CloudFront API에서 키 그룹 ID를 사용하여 키 그룹을 캐시 동작과 연결합니다.

CloudFront 키 페어 생성(권장되지 않음, AWS 계정 루트 사용자 필요)

> **Important**
>
> 다음 단계를 수행하는 대신 신뢰할 수 있는 키 그룹에 대한 퍼블릭 키를 만드는 것이 좋습니다. 서명된 URL 및 서명된 쿠키에 대한 퍼블릭 키를 만드는 권장 방법은 신뢰할 수 있는 키 그룹에 대한 키 페어 생성(권장) 단원을 참조하세요.

CloudFront 키 페어를 만드는 방법은 다음과 같습니다.

- AWS Management Console에서 키 페어를 만들고 프라이빗 키를 다운로드합니다. 다음 절차를 참조하세요.
- OpenSSL과 같은 애플리케이션을 사용하여 RSA 키 페어를 만든 후 퍼블릭 키를 에 업로드합니다 AWS Management Console RSA 키 페어 생성에 대한 자세한 내용은 신뢰할 수 있는 키 그룹에 대한 키 페어 생성(권장) 단원을 참조하세요.

에서 CloudFront 키 페어 생성AWS Management Console

1. AWS 계정 루트 사용자의 자격 증명을 사용하여 AWS Management Console에 로그인합니다.

   > **Important**
   >
   > IAM 사용자는 CloudFront 키 페어를 만들 수 없습니다. 키 페어를 만들려면 루트 사용자 자격 증명을 사용하여 로그인해야 합니다.

2. 계정 이름을 선택한 다음 내 보안 자격 증명을 선택합니다.
3. CloudFront 키 페어(CloudFront key pairs)를 선택합니다.
4. 활성 키 페어가 하나만 있는지 확인합니다. 활성 키 페어가 이미 두 개라면 키 페어를 만들 수 없습니다.
5. 새 키 페어 생성을 선택합니다.

   > **Note**
   >
   > 사용자 고유의 키 페어를 만들고 퍼블릭 키를 업로드하도록 선택할 수도 있습니다. CloudFront 키 페어는 1024, 2048 또는 4096비트 키를 지원합니다.

6. 키 페어 생성 대화 상자에서 프라이빗 키 파일 다운로드를 선택한 다음 컴퓨터에 파일을 저장합니다.

> **Important**
>
> CloudFront 키 페어의 프라이빗 키를 안전한 위치에 저장하고 파일에 대한 권한을 설정하여 원하는 관리자만 읽을 수 있도록 합니다. 다른 사람이 프라이빗 키를 가지게 되면 이들이 유효한 서명된 URL과 서명된 쿠키를 만들어 콘텐츠를 다운로드할 수 있게 됩니다. 프라이빗 키는 다시 가져올 수 없기 때문에 이를 잃어버리거나 삭제한 경우 새 CloudFront 키 페어를 만들어야 합니다.

7. 키 페어의 키 페어 ID를 기록합니다. (AWS Management Console에서는 액세스 키 ID라고 합니다.) 서명된 URL 또는 서명된 쿠키를 만들 때 이것을 사용합니다.

## 프라이빗 키 재포맷(.NET 및 Java만 해당)

.NET 또는 Java를 사용하여 서명된 URL 또는 서명된 쿠키를 만드는 경우, 기본 PEM 형식의 키 페어에서 프라이빗 키를 사용하여 서명을 만들 수 없습니다. 대신 다음을 수행합니다.

- .NET 프레임워크 – 프라이빗 키를 .NET 프레임워크에서 사용하는 XML 형식으로 변환합니다. 몇 가지 도구를 사용할 수 있습니다.
- Java – 프라이빗 키를 DER 형식으로 변환합니다. 이 작업을 수행하는 한 가지 방법은 다음 OpenSSL 명령을 사용하는 것입니다. 다음 명령에서 `private_key.pem`은 PEM 형식의 프라이빗 키가 들어 있는 파일의 이름이며, `private_key.der`은 명령을 실행한 후 DER 형식의 프라이빗 키가 들어 있는 파일의 이름입니다.

```
openssl pkcs8 -topk8 -nocrypt -in private_key.pem -inform PEM -out private_key.der -
outform DER
```

  인코더가 정확하게 작동하려면 Bouncy Castle Java 암호화 API에 대한 JAR을 프로젝트에 추가한 다음 Bouncy Castle 공급자를 추가합니다.

## 배포에 서명자 추가

서명자는 배포에 대해 서명된 URL 및 서명된 쿠키를 만들 수 있는 신뢰할 수 있는 키 그룹(권장) 또는 CloudFront 키 페어입니다. CloudFront 배포와 함께 서명된 URL 또는 서명된 쿠키를 사용하려면 서명자를 지정해야 합니다.

서명자는 캐시 동작과 연결됩니다. 이렇게 하면 같은 배포의 일부 파일에 대해서만 서명된 URL 또는 서명된 쿠키를 요구할 수 있습니다. 배포에는 해당 캐시 동작과 연결된 파일에 대해서만 서명된 URL 또는 쿠키가 필요합니다.

마찬가지로 서명자는 해당 캐시 동작과 연결된 파일에 대해서만 URL 또는 쿠키에 서명할 수 있습니다. 예를 들어, 캐시 동작 하나당 서명자가 하나씩 있고 다른 캐시 동작에는 다른 서명자가 있다면 양쪽의 서명자 모두 상대편의 캐시 동작과 연결된 파일에 대해 서명된 URL 또는 쿠키를 만들 수 없습니다.

> **Important**
>
> 배포에 서명자를 추가하기 전에 다음을 수행합니다.
>
> - 캐시 동작의 경로 패턴과 캐시 동작 순서를 신중하게 정의하여 콘텐츠 액세스 권한을 실수로 사용자에게 부여하거나 일부 공개 콘텐츠에 사용자가 액세스하는 일이 없도록 해야 합니다.
>
>  예를 들어, 하나의 요청이 두 캐시 동작의 경로 패턴과 일치한다고 가정합니다. 첫 번째 캐시 동작에는 서명된 URL 또는 서명된 쿠키가 필요하지 않고, 두 번째 캐시 동작에는 필요합니다. CloudFront에서는 첫 번째와 연결된 캐시 동작을 처리하므로 사용자는 서명된 URL 또는 서명된 쿠키 없이도 파일에 액세스할 수 있습니다.
>
>  경로 패턴에 대한 자세한 내용은 경로 패턴을 참조하세요.
>
> - 콘텐츠를 배포하는 데 이미 사용하고 있는 배포의 경우 서명자를 추가하기 전에 서명된 URL 및 서명된 쿠키를 생성할 준비가 되어 있어야 합니다. 서명자를 추가하면 CloudFront는 유효한 서명된 URL 또는 서명된 쿠키를 포함하지 않는 요청을 거부합니다.

CloudFront 콘솔 또는 CloudFront API를 사용하여 배포에 서명자를 추가할 수 있습니다.

Console

다음 단계에서는 신뢰할 수 있는 키 그룹을 서명자로 추가하는 방법을 보여 줍니다. AWS 계정을 신뢰할 수 있는 서명자로 추가할 수도 있지만 권장하지 않습니다.

콘솔을 사용하여 배포에 서명자를 추가하려면

1. 신뢰할 수 있는 서명자로 사용할 키 그룹의 키 그룹 ID를 적어 둡니다. 자세한 내용은 신뢰할 수 있는 키 그룹에 대한 키 페어 생성(권장) 섹션을 참조하세요.
2. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다
3. 서명된 URL 또는 서명된 쿠키로 보호하려는 파일의 배포를 선택합니다.

> **Note**
>
> 새 배포에 서명자를 추가하려면 배포를 만들 때 6단계에서 설명하는 것과 동일한 설정을 지정합니다.

4. 동작 탭을 선택합니다.
5. 경로 패턴이 서명된 URL 또는 서명된 쿠키로 보호하려는 파일과 일치하는 캐시 동작을 선택한 다음 편집을 선택합니다.
6. 동작 편집(Edit Behavior) 페이지에서 다음을 수행합니다.
   a. 최종 사용자 액세스 제한(서명된 URL 또는 서명된 쿠키 사용)(Restrict Viewer Access (Use Signed URLs or Signed Cookies)에서 예를 클릭합니다.
   b. 신뢰할 수 있는 키 그룹 또는 신뢰할 수 있는 서명자(Trusted Key Groups or Trusted Signer)에서 신뢰할 수 있는 키 그룹(Trusted Key Groups)을 선택합니다.
   c. 신뢰할 수 있는 키 그룹(Trusted Key Groups)에서 추가할 키 그룹을 선택한 다음 추가를 선택합니다. 두 개 이상의 키 그룹을 추가하려면 이 과정을 반복합니다.
7. 예, 편집합니다를 선택하여 캐시 동작을 업데이트합니다.

## API

CloudFront API를 사용하여 신뢰할 수 있는 키 그룹을 서명자로 추가할 수 있습니다. 기존 배포 또는 새 배포에 서명자를 추가할 수 있습니다. 두 경우 모두 `TrustedKeyGroups` 요소에 값을 지정합니다.

AWS 계정을 신뢰할 수 있는 서명자로 추가할 수도 있지만 권장하지 않습니다.

Amazon CloudFront API 참조에서 다음 주제를 참조하세요.

- 기존 배포 업데이트 – UpdateDistribution
- 새로운 배포 생성 – CreateDistribution

### 키 페어 교체

서명된 URL 및 서명된 쿠키에 대한 키 페어를 주기적으로 교체(변경)하는 것이 좋습니다. 아직 만료되지 않은 서명된 URL 또는 서명된 쿠키를 무효화하지 않고 URL 또는 쿠키를 만들기 위해 사용 중인 키 페어를 교체하려면 다음 작업을 수행하세요.

1. 새 키 페어를 만들고 키 그룹에 퍼블릭 키를 추가합니다. 자세한 내용은 신뢰할 수 있는 키 그룹에 대한 키 페어 생성(권장) 단원을 참조하세요.
2. 이전 단계에서 새 키 그룹을 만든 경우 키 그룹을 배포에 서명자로 추가합니다.

> **Important**
>
> 키 그룹에서 기존 퍼블릭 키를 제거하거나 배포에서 키 그룹을 제거하지 마세요. 새것만 추가하세요.

3. 새로운 키 페어의 프라이빗 키를 사용하여 서명을 만들도록 애플리케이션을 업데이트합니다. 새 프라이빗 키로 서명된 URL 또는 서명된 쿠키가 작동하는지 확인합니다.
4. 이전 프라이빗 키로 서명한 URL 또는 쿠키의 만료 날짜가 지날 때까지 기다립니다. 그런 다음 키 그룹에서 이전 프라이빗 키를 제거합니다. 2단계에서 새 키 그룹을 만든 경우 배포에서 이전 키 그룹을 제거합니다.

## 서명된 URL 또는 서명된 쿠키 사용 결정

CloudFront 서명된 URL 및 서명된 쿠키는 기본 기능이 같습니다. 바로 콘텐츠에 액세스할 수 있는 대상을 제어하는 기능입니다. CloudFront를 통해 프라이빗 콘텐츠를 제공 중이며 서명된 URL 또는 서명된 쿠키 중 하나를 결정해야 하는 경우, 다음 사항을 고려하세요.

다음과 같은 경우 서명된 URL을 사용합니다.

- 애플리케이션 설치 파일을 다운로드하는 등 개별 파일에 대한 액세스를 제한하고 싶은 경우
- 사용자가 쿠키를 지원하지 않는 클라이언트(예: 사용자 지정 HTTP 클라이언트)를 사용하는 경우

다음과 같은 경우 서명된 쿠키를 사용합니다.

- HLS 형식의 비디오 파일 전체 또는 웹 사이트의 구독자 영역에 있는 파일 전체 등 제한된 파일 여러 개에 대한 액세스 권한을 제공하려는 경우
- 현재의 URL을 변경하고 싶지 않은 경우

현재 서명된 URL을 사용하지 않는데 (서명되지 않은) URL에 다음과 같은 쿼리 문자열 파라미터가 들어 있다면 서명된 URL 또는 서명된 쿠키를 사용할 수 없습니다.

- `Expires`
- `Policy`
- `Signature`
- `Key-Pair-Id`

CloudFront는 이러한 쿼리 문자열 파라미터가 포함된 URL을 서명된 URL로 간주하기 때문에 서명된 쿠키를 살펴보지 않을 것입니다.

### 서명된 URL과 서명된 쿠키 모두 사용

서명된 URL은 서명된 쿠키보다 우선합니다. 같은 파일에 대한 액세스를 제어할 때 서명된 URL과 서명된 쿠키를 모두 사용하고 최종 사용자는 서명된 URL로 파일을 요청하는 경우, CloudFront는 서명된 URL만을 기준으로 최종 사용자에게 파일을 반환할지 여부를 판단합니다.

## 서명된 URL 사용

서명된 URL에는 만료 날짜 및 시간 같은 추가 정보가 포함되므로 콘텐츠에 대한 액세스를 보다 세부적으로 제어할 수 있습니다. 이러한 추가 정보는 미리 준비된(canned) 정책 또는 사용자 지정 정책에 따라 정책 설명에 나타납니다. 미리 준비된(canned) 정책과 사용자 지정 정책 간의 차이점은 이어지는 두 단원에 설명되어 있습니다.

> **Note**
>
> 같은 배포에 대해 미리 준비된(canned) 정책과 사용자 지정 정책으로 각각 서명된 URL을 만들 수 있습니다.

주제


- 서명된 URL에 대해 미리 준비된 정책 또는 사용자 지정 정책 사용 결정
- 서명된 URL의 작동 방식
- 서명된 URL의 유효 기간 결정
- CloudFront가 서명된 URL에서 만료 날짜 및 시간을 확인하는 시기
- 예제 코드 및 타사 도구
- 미리 준비된 정책을 사용하여 서명된 URL 생성

- 사용자 지정 정책을 사용하여 서명된 URL 생성

## 서명된 URL에 대해 미리 준비된 정책 또는 사용자 지정 정책 사용 결정

서명된 URL을 만들 때 JSON 형식의 정책 설명을 작성하여 서명된 URL의 제약 조건(예: URL의 유효 기간)을 지정합니다. 미리 준비된(canned) 정책 또는 사용자 지정 정책을 사용할 수 있습니다. 다음은 미리 준비된(canned) 정책과 사용자 지정 정책을 비교한 것입니다.

| 설명 | 미리 준비된 정책 | 사용자 정의 정책 |
| --- | --- | --- |
| 정책 설명은 여러 파일에 재사용할 수 있습니다. 정책 설명을 재사용하려면 `Resource` 객체에 와일드카드 문자를 사용해야 합니다. 자세한 내용은 사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명에서 지정한 값 섹션을 참조하세요.) | 아니요 | 예 |
| 날짜 및 시간을 지정하여 사용자가 콘텐츠에 액세스를 시작할 수 있습니다. | 아니요 | 예(선택 사항) |
| 사용자의 콘텐츠 액세스를 중단할 날짜 및 시간을 지정할 수 있습니다. | 예 | 예 |
| 콘텐츠에 액세스할 수 있는 사용자의 IP 주소 또는 IP 주소 범위를 지정할 수 있습니다. | 아니요 | 예(선택 사항) |
| 서명된 URL에는 base64 인코딩 버전의 정책이 포함되므로 URL 길이가 더 깁니다. | 아니요 | 예 |

미리 준비된(canned) 정책으로 서명된 URL을 만드는 방법에 대한 자세한 내용은 미리 준비된 정책을 사용하여 서명된 URL 생성 단원을 참조하세요.

사용자 지정 정책으로 서명된 URL을 만드는 방법에 대한 자세한 내용은 사용자 지정 정책을 사용하여 서명된 URL 생성 단원을 참조하세요.

## 서명된 URL의 작동 방식

여기서는 서명된 URL에 알맞게 CloudFront 및 Amazon S3를 구성하는 방법과 사용자가 서명된 URL로 파일을 요청할 때 CloudFront가 대응하는 방식을 살펴봅니다.

1. CloudFront 배포에서 CloudFront가 URL 서명을 확인하는 데 사용할 수 있는 퍼블릭 키가 포함된 신뢰할 수 있는 키 그룹을 하나 이상 지정합니다. 해당 프라이빗 키를 사용하여 URL에 서명합니다.

   CloudFront는 RSA 2048 및 ECDSA 256 키 서명을 사용하여 서명된 URL을 지원합니다.

   자세한 내용은 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정 섹션을 참조하세요.
2. 애플리케이션을 개발하여 사용자에게 콘텐츠 액세스를 허용할지 여부를 판단하고, 액세스를 제한할 애플리케이션 부분 또는 파일에 대해 서명된 URL을 만듭니다. 자세한 정보는 다음 주제를 참조하세요.

   - 미리 준비된 정책을 사용하여 서명된 URL 생성
   - 사용자 지정 정책을 사용하여 서명된 URL 생성
3. 사용자가 서명된 URL이 필요한 파일을 요청합니다.
4. 애플리케이션은 해당 사용자에게 파일 액세스 권한이 있는지 확인합니다. 사용자가 로그인했는지, 콘텐츠 액세스 비용을 지불했는지, 기타 액세스 요구 사항을 충족했는지 등을 확인합니다.
5. 애플리케이션은 서명된 URL을 만들어 사용자에게 반환합니다.
6. 사용자는 서명된 URL을 통해 콘텐츠를 다운로드하거나 스트리밍할 수 있습니다.

   이 단계는 자동으로 진행되기 때문에 사용자는 콘텐츠에 액세스하기 위해 아무것도 하지 않아도 됩니다. 예를 들어, 사용자가 웹 브라우저에서 콘텐츠에 액세스한다면 애플리케이션은 서명된 URL을 브라우저에 반환합니다. 브라우저는 즉시 서명된 URL을 사용하여 사용자 개입 없이 CloudFront 엣지 캐시의 파일에 액세스합니다.
7. CloudFront는 퍼블릭 키를 사용하여 서명을 확인하고 URL이 아직 변조되지 않았음을 확인합니다. 유효하지 않은 서명인 경우 요청을 거부합니다.

   서명이 유효한 경우, CloudFront는 URL의 정책 설명을 보고 아직 유효한 정책임을 확인합니다. 미리 준비된(canned) 정책을 사용하는 경우에는 정책 설명을 구성합니다. 예를 들어, URL의 시작 및 종료 날짜와 시간을 지정했다면 CloudFront는 액세스를 허용하려는 기간 동안 사용자가 콘텐츠에 액세스를 시도하고 있음을 확인합니다.

   요청이 정책 설명의 요구 사항을 충족하는 경우, CloudFront는 스탠다드 작업을 수행합니다. 다시 말해 파일이 이미 엣지 캐시에 있는지 여부를 판단하고, 필요에 따라 요청을 오리진으로 전달하고, 사용자에게 파일을 반환합니다.

> **Note**
>
> 서명되지 않은 URL에 쿼리 문자열 파라미터가 포함된 경우 서명하는 URL 부분에 이를 포함시켜야 합니다. 서명된 URL에 서명한 후 서명된 URL에 쿼리 문자열을 추가하면 해당 URL이 HTTP 403 상태를 반환합니다.

## 서명된 URL의 유효 기간 결정

단시간(최소 몇 분) 동안만 유효한 서명된 URL을 사용하여 프라이빗 콘텐츠를 배포할 수 있습니다. 이렇게 단시간 동안 유효한 서명된 URL은 영화 대여 또는 음악 다운로드 등을 주문 방식으로 고객에게 제공하는 등 특정 목적으로 즉석에서 콘텐츠를 배포할 때 유용합니다. 서명된 URL의 효력을 단시간 동안만 유지하려는 경우, 이를 자동으로 만드는 애플리케이션을 개발할 수 있습니다. 사용자가 파일을 다운로드하거나 미디어 파일을 재생하기 시작하면 CloudFront는 URL의 만료 시간을 현재 시간과 비교하여 URL이 아직 유효한지 파악합니다.

또한 장시간(몇 년 정도) 동안 유효한 서명된 URL을 사용하여 프라이빗 콘텐츠를 배포할 수도 있습니다. 장시간 동안 유효한 서명된 URL은 알려진 사용자에게 프라이빗 콘텐츠를 배포할 때 유용합니다. 투자자에게 사업 계획을 배포하거나 직원에게 교육 자료를 배포하는 경우가 여기에 해당합니다. 이러한 장기 서명 URL을 생성하는 애플리케이션을 개발할 수 있습니다.

## CloudFront가 서명된 URL에서 만료 날짜 및 시간을 확인하는 시기

CloudFront는 HTTP 요청이 있을 때 서명된 URL의 만료 날짜 및 시간을 확인합니다. 클라이언트가 만료 시간 직전에 대용량 파일을 다운로드하기 시작한 경우, 다운로드 도중 만료 시간이 지나도 다운로드는 완료됩니다. TCP 연결이 끊어진 경우, 클라이언트가 만료 시간 이후에 다운로드를 다시 시작하는 것은 불가능합니다.

클라이언트가 범위 GET을 사용하여 파일을 작은 조각으로 가져오는 경우, 만료 시간 이후의 GET 요청은 실패합니다. 범위 GET에 대한 자세한 내용은 CloudFront에서 객체에 대한 부분적인 요청을 처리하는 방법(범위 GET)을 참조하세요.

## 예제 코드 및 타사 도구

서명된 URL의 해시 및 서명된 부분을 만드는 예제 코드에 대해서는 다음 주제를 참조하세요.

- Perl을 사용한 URL 서명 생성
- PHP를 사용한 URL 서명 생성
- C# 및 .NET Framework를 사용한 URL 서명 생성

- Java를 사용한 URL 서명 생성

## 미리 준비된 정책을 사용하여 서명된 URL 생성

미리 준비된 정책을 사용하여 서명된 URL을 만들려면 다음 단계를 수행합니다.

미리 준비된 정책을 사용하여 서명된 URL을 만들려면

1. .NET 또는 Java를 사용하여 서명된 URL을 만드는 중인데 키 페어의 프라이빗 키를 기본 .pem 형식에서 .NET 또는 Java와 상환되는 형식으로 다시 포맷하지 않았다면 지금 포맷하세요. 자세한 내용은 프라이빗 키 재포맷(.NET 및 Java만 해당) 섹션을 참조하세요.
2. 다음 값을 연결합니다. 이 예제의 서명된 URL의 형식을 사용할 수 있습니다.

```
https://d111111abcdef8.cloudfront.net/
image.jpg?color=red&size=medium&Expires=1767290400&Signature=nitfHRCrtziwO2HwPfWw~yYDhUF5Ew
j19DzZrvDh6hQ73lDx~-ar3UocvvRQVw6EkC~GdpGQyyOSKQim-
TxAnW7d8F5Kkai9HVx0FIu-5jcQb0UEmatEXAMPLE3ReXySpLSMj0yCd3ZAB4UcBCAqEijkytL6f3fVYNGQI6&Key-
Pair-Id=K2JCJMDEHXQW5F
```

모든 빈 공백(탭과 줄바꿈 문자 포함)을 제거합니다. 애플리케이션 코드의 문자열에 이스케이프 문자를 포함해야 할 수도 있습니다. 모든 값에는 일종의 `String`이 있습니다.

1. ***### ## URL***

   기본 URL이란 서명된 URL을 사용하지 않고 파일에 액세스할 때 사용할 CloudFront URL을 말합니다. 자체 쿼리 문자열 파라미터가 있는 경우 여기에 포함됩니다. 이전 예제에서 기본 URL은 `https://d111111abcdef8.cloudfront.net/image.jpg`입니다. 웹 배포의 URL 형식에 대한 자세한 내용은 CloudFront에서 파일에 대한 URL 형식 사용자 지정을 참조하세요.

   - 다음 CloudFront URL은 배포의 이미지 파일을 위한 것입니다(CloudFront 도메인 이름 사용). `image.jpg`는 `images` 디렉터리에 있습니다. URL에 있는 파일의 경로는 HTTP 서버 또는 Amazon S3 버킷의 파일 경로와 일치해야 합니다.

     `https://d111111abcdef8.cloudfront.net/images/image.jpg`

   - 다음 CloudFront URL은 쿼리 문자열을 포함합니다.

     `https://d111111abcdef8.cloudfront.net/images/image.jpg?size=large`

   - 다음 CloudFront URL은 배포의 이미지 파일을 위한 것입니다. 모두 대체 도메인 이름을 사용합니다. 두 번째는 쿼리 문자열을 포함합니다.

```
https://www.example.com/images/image.jpg

https://www.example.com/images/image.jpg?color=red
```

- 다음 CloudFront URL은 대체 도메인 이름과 HTTPS 프로토콜을 사용하는 배포의 이미지 파일을 위한 것입니다.

```
https://www.example.com/images/image.jpg
```

2. **?**

?는 기본 URL 뒤에 쿼리 파라미터가 있음을 나타냅니다. 쿼리 파라미터를 지정하지 않은 경우에도 ?를 포함합니다.

> **Note**
>
> 순서에 관계없이 다음 쿼리 파라미터를 지정할 수 있습니다.

3. ***## ## ### ####(## ##&***

(선택 사항) 자체 쿼리 문자열 파라미터를 입력할 수 있습니다. 이렇게 하려면 `color=red&size=medium`처럼 각 문자열 파라미터 간에 앰퍼샌드(&)를 추가합니다. 쿼리 문자열 파라미터는 URL 내에서 원하는 순서로 지정할 수 있습니다.

> **Important**
>
> 쿼리 문자열 파라미터의 이름은 `Expires`, `Signature`, `Key-Pair-Id`로 지정할 수 없습니다.

4. **Expires=*Unix ## ##(#) # ## ###(UTC) ### ## # ##***

URL에서 파일에 대한 액세스 허용을 중지하게 할 날짜 및 시간입니다.

Unix 시간 형식(초) 및 협정 세계시(UTC) 기준의 URL 만료 날짜 및 시간을 지정합니다. 예를 들면 2026년 1월 1일 오전 10시(UTC)를 이 주제의 시작 부분에 있는 예와 같이 Unix 시간 형식의 `1767290400`으로 변환합니다.

에포크 시간을 사용하려면 9223372036854775807(금요일, 2262년 4월 11일 23:47:16.854 UTC) 이전 날짜에 대한 64비트 정수를 지정합니다.

UTC에 대한 자세한 내용은 RFC 3339, 인터넷의 날짜 및 시간: 타임스탬프를 참조하세요.

5. **&Signature=*## ### ## # ### ##***

   JSON 정책 설명을 해시, 서명 및 base64로 인코딩한 버전입니다. 자세한 내용은 미리 준비된 정책을 사용하는 서명된 URL에 대한 서명 생성 섹션을 참조하세요.

6. **&Key-Pair-Id=*### #### # ## ## ## #### ## ## CloudFront ### ## ### # ID***

   CloudFront 퍼블릭 키의 ID입니다(예: K2JCJMDEHXQW5F). CloudFront는 서명된 URL을 확인할 때 사용할 퍼블릭 키를 퍼블릭 키 ID로 판단합니다. CloudFront는 서명의 정보를 정책 설명의 정보와 비교하고 URL이 변조되지 않았음을 확인합니다.

   이 퍼블릭 키는 배포에서 신뢰할 수 있는 서명자인 키 그룹에 속해야 합니다. 자세한 내용은 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정 섹션을 참조하세요.

## 미리 준비된 정책을 사용하는 서명된 URL에 대한 서명 생성

미리 준비된 정책을 사용하는 서명된 URL의 서명을 만들려면 다음 절차를 완료하세요.

주제

- 미리 준비된 정책을 사용하는 서명된 URL의 정책 설명 생성
- 미리 준비된 정책을 사용하는 서명된 URL에 대한 서명 생성

### 미리 준비된 정책을 사용하는 서명된 URL의 정책 설명 생성

미리 준비된 정책을 사용하여 서명된 URL을 만드는 경우, `Signature` 파라미터는 정책 설명의 해시 및 서명된 버전입니다. 미리 준비된 정책을 사용하는 서명된 URL의 경우, 사용자 지정 정책을 사용하는 서명된 URL과 마찬가지로 URL에 정책 설명을 포함하지 않습니다. 정책 설명을 만들려면 다음 절차를 수행하세요.

미리 준비된 정책을 사용하는 서명된 URL에 대한 정책 설명을 만들려면

1. 다음 JSON 형식과 UTF-8 문자 인코딩을 사용하여 정책 설명을 구성합니다. 지정된 모든 문장 부호 및 기타 리터럴 값을 정확히 포함해야 합니다. `Resource` 및 `DateLessThan` 파라미터에 대한 내용은 미리 준비된 정책을 사용하는 서명된 URL에 대한 정책 설명에서 지정한 값를 참조하세요.

```
{
    "Statement": [
        {
            "Resource": "base URL or stream name",
            "Condition": {
                "DateLessThan": {
                    "AWS:EpochTime": ending date and time in Unix time format and
 UTC
                }
            }
        }
    ]
}
```

2. 정책 설명에서 모든 공백(탭과 줄바꿈 문자 포함)을 제거합니다. 애플리케이션 코드의 문자열에 이스케이프 문자를 포함해야 할 수도 있습니다.

미리 준비된 정책을 사용하는 서명된 URL에 대한 정책 설명에서 지정한 값

미리 준비된 정책에 대한 정책 설명을 만들 때 다음 값을 지정합니다.

리소스

> **Note**
>
> `Resource`에 대해 값을 하나만 지정할 수 있습니다.

쿼리 문자열(있는 경우)을 포함하지만 CloudFront `Expires`, `Signature` 및 `Key-Pair-Id` 파라미터를 제외한 기본 URL입니다. 예를 들면 다음과 같습니다.

```
https://d111111abcdef8.cloudfront.net/images/horizon.jpg?
size=large&license=yes
```

다음을 참조하세요.

- 프로토콜(Protocol) – 값은 `http://` 또는 `https://`로 시작해야 합니다.
- 쿼리 문자열 파라미터(Query string parameters) – 쿼리 문자열 파라미터가 없는 경우, 물음표를 생략합니다.
- 대체 도메인 이름(Alternate domain names) – URL에 대체 도메인 이름(CNAME)을 지정하는 경우, 웹 페이지 또는 애플리케이션의 파일을 참조할 때 대체 도메인 이름을 지정해야 합니다. 객체에 대한 Amazon S3 URL을 지정하지 마세요.

DateLessThan

Unix 시간 형식(초) 및 협정 세계시(UTC) 기준의 URL 만료 날짜 및 시간. 예를 들면 2026년 1월 1일 오전 10시(UTC)를 Unix 시간 형식의 1767290400으로 변환합니다.

이 값은 서명된 URL의 `Expires` 쿼리 문자열 파라미터 값과 일치해야 합니다. 값을 인용 부호로 묶지 마세요.

자세한 내용은 CloudFront가 서명된 URL에서 만료 날짜 및 시간을 확인하는 시기 단원을 참조하세요.

미리 준비된 정책을 사용하는 서명된 URL에 대한 정책 설명 예제

서명된 URL에 다음 예제의 정책 설명을 사용할 경우, 사용자는 UTC 기준 2026년 1월 1일 오전 10시까지 `https://d111111abcdef8.cloudfront.net/horizon.jpg` 파일에 액세스할 수 있습니다.

```
{
    "Statement": [
        {
            "Resource": "https://d111111abcdef8.cloudfront.net/horizon.jpg?
size=large&license=yes",
            "Condition": {
                "DateLessThan": {
                    "AWS:EpochTime": 1767290400
                }
            }
        }
    ]
}
```

미리 준비된 정책을 사용하는 서명된 URL에 대한 서명 생성

서명된 URL의 `Signature` 파라미터 값을 만들려면 미리 준비된 정책을 사용하는 서명된 URL의 정책 설명 생성에서 만든 정책 설명을 해시하고 서명합니다.

다음을 참조하여 정책 설명을 해시, 서명 및 인코딩하는 방법에 대한 추가적인 내용과 예제를 확인하세요.

- base64 인코딩 및 암호화를 위한 Linux 명령 및 OpenSSL
- 서명 URL에 대한 서명을 만드는 코드 예제

옵션 1: 미리 준비된 정책을 사용하여 서명을 만들려면

1. SHA-1 해시 함수와 생성된 RSA 또는 ECDSA 프라이빗 키를 사용하여 미리 준비된 정책을 사용하는 서명된 URL에 대한 정책 설명을 만들려면 절차에서 만든 정책 설명을 해시하고 서명합니다. 공백이 삭제된 버전의 정책 설명을 사용합니다.

   해시 함수에 필요한 프라이빗 키의 경우 배포에 대해 신뢰할 수 있는 활성 키 그룹에 퍼블릭 키가 있는 프라이빗 키를 사용합니다.

   > **Note**
   >
   > 정책 설명을 해시 및 서명하는 방법은 프로그래밍 언어와 플랫폼에 따라 달라집니다. 샘플 코드에 대한 내용은 서명 URL에 대한 서명을 만드는 코드 예제를 참조하세요.

2. 해시 및 서명된 문자열에서 공백(탭과 줄바꿈 문자 포함)을 제거합니다.
3. MIME base64 인코딩 기준으로 문자열을 base64로 인코딩합니다. 자세한 내용은 RFC 2045, MIME(Multipurpose Internet Mail Extensions) Part One: Format of Internet Message Bodies의 Section 6.8, Base64 Content-Transfer-Encoding을 참조하세요.
4. URL 쿼리 문자열에서 사용할 수 없는 문자를 유효한 문자로 교체합니다. 아래 표에 사용할 수 없는 문자와 유효한 문자가 나열되어 있습니다.

| 무효 문자를 | 교체할 유효한 문자 |
| --- | --- |
| + | - (하이픈) |
| = | _ (밑줄) |
| / | ~ (물결표) |

5. 서명된 URL의 `&Signature=` 뒤에 결과 값을 추가하고 미리 준비된 정책을 사용하여 서명된 URL을 만들려면로 돌아가서 서명된 URL 부분을 마저 연결합니다.

## 사용자 지정 정책을 사용하여 서명된 URL 생성

사용자 지정 정책을 사용하여 서명된 URL을 만들려면 다음 절차를 완료합니다.

사용자 지정 정책을 사용하여 서명된 URL을 만들려면

1. .NET 또는 Java를 사용하여 서명된 URL을 만드는 중인데 키 페어의 프라이빗 키를 기본 .pem 형식에서 .NET 또는 Java와 상환되는 형식으로 다시 포맷하지 않았다면 지금 포맷하세요. 자세한 내용은 프라이빗 키 재포맷(.NET 및 Java만 해당) 섹션을 참조하세요.
2. 다음 값을 연결합니다. 이 예제의 서명된 URL의 형식을 사용할 수 있습니다.

```
https://d111111abcdef8.cloudfront.net/
image.jpg?color=red&size=medium&Policy=eyANCiAgICEXAMPLEW1lbnQiOiBbeyANCiAgICAgICJSZXNvdXJj
j19DzZrvDh6hQ73lDx~-ar3UocvvRQVw6EkC~GdpGQyyOSKQim-
TxAnW7d8F5Kkai9HVx0FIu-5jcQb0UEmatEXAMPLE3ReXySpLSMj0yCd3ZAB4UcBCAqEijkytL6f3fVYNGQI6&Key-
Pair-Id=K2JCJMDEHXQW5F
```

모든 빈 공백(탭과 줄바꿈 문자 포함)을 제거합니다. 애플리케이션 코드의 문자열에 이스케이프 문자를 포함해야 할 수도 있습니다. 모든 값에는 일종의 `String`이 있습니다.

1. ***### ## URL***

   기본 URL이란 서명된 URL을 사용하지 않고 파일에 액세스할 때 사용할 CloudFront URL을 말합니다. 자체 쿼리 문자열 파라미터가 있는 경우 여기에 포함됩니다. 이전 예제에서 기본 URL은 `https://d111111abcdef8.cloudfront.net/image.jpg`입니다. 웹 배포의 URL 형식에 대한 자세한 내용은 CloudFront에서 파일에 대한 URL 형식 사용자 지정을 참조하세요.

   다음 예는 배포에 대해 지정하는 값을 보여줍니다.

   - 다음 CloudFront URL은 배포의 이미지 파일을 위한 것입니다(CloudFront 도메인 이름 사용). `image.jpg`는 `images` 디렉터리에 있습니다. URL에 있는 파일의 경로는 HTTP 서버 또는 Amazon S3 버킷의 파일 경로와 일치해야 합니다.

     `https://d111111abcdef8.cloudfront.net/images/image.jpg`

   - 다음 CloudFront URL은 쿼리 문자열을 포함합니다.

     `https://d111111abcdef8.cloudfront.net/images/image.jpg?size=large`

   - 다음 CloudFront URL은 배포의 이미지 파일을 위한 것입니다. 둘 다 대체 도메인 이름을 사용하며 두 번째는 쿼리 문자열을 포함합니다.

```
https://www.example.com/images/image.jpg

https://www.example.com/images/image.jpg?color=red
```

- 다음 CloudFront URL은 대체 도메인 이름과 HTTPS 프로토콜을 사용하는 배포의 이미지 파일을 위한 것입니다.

```
https://www.example.com/images/image.jpg
```

2. **?**

   ?는 기본 URL 뒤에 쿼리 문자열 파라미터가 있음을 나타냅니다. 쿼리 파라미터를 지정하지 않은 경우에도 ?를 포함합니다.

   > **Note**
   >
   > 순서에 관계없이 다음 쿼리 파라미터를 지정할 수 있습니다.

3. ***## ## ### ####(## ##*****&**

   (선택 사항) 자체 쿼리 문자열 파라미터를 입력할 수 있습니다. 이렇게 하려면 `color=red&size=medium`처럼 각 문자열 파라미터 간에 앰퍼샌드(&)를 추가합니다. 쿼리 문자열 파라미터는 URL 내에서 원하는 순서로 지정할 수 있습니다.

   > **Important**
   >
   > 쿼리 문자열 파라미터의 이름은 `Policy`, `Signature`, `Key-Pair-Id`로 지정할 수 없습니다.

   자체 파라미터를 추가하는 경우, 마지막 파라미터를 포함하여 각 파라미터 뒤에 &를 추가합니다.

4. **Policy=*## ### base64 ### ##***

   공백이 제거된 JSON 형식의 정책 설명은 base64로 인코딩합니다. 자세한 내용은 [사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명 생성](#) 섹션을 참조하세요.

정책 설명은 서명된 URL이 사용자에게 부여하는 액세스를 제어합니다. 여기에는 파일의 URL, 만료 날짜 및 시간, URL의 효력이 발생하는 날짜 및 시간(선택 사항), 파일에 액세스할 수 있는 IP 주소 또는 IP 주소 범위(선택 사항)가 포함됩니다.

5. **&Signature=*## ### ## # ### ##***

   JSON 정책 설명을 해시, 서명 및 base64로 인코딩한 버전입니다. 자세한 내용은 사용자 지정 정책을 사용하는 서명된 URL에 대한 서명 생성 섹션을 참조하세요.

6. **&Key-Pair-Id=*### #### # ## ## ## #### ## ## CloudFront ### ## ### # ID***

   CloudFront 퍼블릭 키의 ID입니다(예: `K2JCJMDEHXQW5F`). CloudFront는 서명된 URL을 확인할 때 사용할 퍼블릭 키를 퍼블릭 키 ID로 판단합니다. CloudFront는 서명의 정보를 정책 설명의 정보와 비교하고 URL이 변조되지 않았음을 확인합니다.

   이 퍼블릭 키는 배포에서 신뢰할 수 있는 서명자인 키 그룹에 속해야 합니다. 자세한 내용은 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정 섹션을 참조하세요.

사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명 생성

다음 단계를 완료하여 사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 명령문을 만듭니다.

다양한 방법으로 파일에 대한 액세스를 제어하는 예시 정책 명령문에 대한 내용은 the section called “사용자 지정 정책을 사용하는 서명된 URL에 대한 예제 정책 설명” 섹션을 참조하세요.

사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명을 만들려면

1. 다음 JSON 형식을 사용하여 정책 설명을 구성하세요. 다음 보다 작음(<) 및 다음보다 큼(>) 기호 및 해당 기호 내의 설명을 사용자 지정 값으로 바꿉니다. 자세한 내용은 the section called “사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명에서 지정한 값” 섹션을 참조하세요.

```
{
    "Statement": [
        {
            "Resource": "<Optional but recommended: URL of the file>",
            "Condition": {
                "DateLessThan": {
                 "AWS:EpochTime": <Required: ending date and time in Unix time
 format and UTC>
                },
```

```
                "DateGreaterThan": {
                 "AWS:EpochTime": <Optional: beginning date and time in Unix time
 format and UTC>
                },
                "IpAddress": {
                 "AWS:SourceIp": "<Optional: IP address>"
                }
            }
        }
    ]
}
```

유의할 사항:

- 정책에는 명령문을 하나만 포함할 수 있습니다.
- UTF-8 문자 인코딩을 사용하세요.
- 지정된 대로 정확하게 모든 문장 부호 및 파라미터 이름을 포함하세요. 파라미터 이름의 약어는 허용되지 않습니다.
- `Condition` 섹션의 파라미터 순서는 중요하지 않습니다.
- `Resource`, `DateLessThan`, `DateGreaterThan`, `IpAddress`에 대한 값 관련 내용은 the section called "사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명에서 지정한 값"을 참조하세요.

2. 정책 설명에서 모든 공백(탭과 줄바꿈 문자 포함)을 제거합니다. 애플리케이션 코드의 문자열에 이스케이프 문자를 포함해야 할 수도 있습니다.
3. MIME base64 인코딩 기준으로 정책 설명을 Base64로 인코딩합니다. 자세한 내용은 RFC 2045, MIME(Multipurpose Internet Mail Extensions) Part One: Format of Internet Message Bodies의 Section 6.8, Base64 Content-Transfer-Encoding을 참조하세요.
4. URL 쿼리 문자열에서 사용할 수 없는 문자를 유효한 문자로 교체합니다. 아래 표에 사용할 수 없는 문자와 유효한 문자가 나열되어 있습니다.

| 무효 문자를 | 교체할 유효한 문자 |
| --- | --- |
| + | - (하이픈) |
| = | _ (밑줄) |
| / | ~ (물결표) |

5. 서명된 URL의 `Policy=` 뒤에 결과 값을 추가하세요.
6. 정책 설명을 해시, 서명 및 base64로 인코딩하여 서명된 URL에 대한 서명을 만드세요. 자세한 내용은 the section called "사용자 지정 정책을 사용하는 서명된 URL에 대한 서명 생성" 섹션을 참조하세요.

사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명에서 지정한 값

사용자 지정 정책에 대한 정책 설명을 생성할 때 다음 값을 지정합니다.

리소스

있는 경우 쿼리 문자열은 포함하지만 CloudFront `Policy`, `Signature` 및 `Key-Pair-Id` 파라미터는 제외한 기본 URL입니다. 예제:

```
https://d111111abcdef8.cloudfront.net/images/horizon.jpg\?
size=large&license=yes
```

`Resource`에 대해 값을 하나만 지정할 수 있습니다.

> **Important**
>
> 정책에서 `Resource` 파라미터를 생략할 수 있지만 생략하는 경우 서명된 URL을 가진 사람은 누구나 서명된 URL을 만드는 데 사용한 키 페어와 연결된 어떠한 배포에서든 모든 파일에 액세스할 수 있다는 뜻이 됩니다.

유의할 사항:

- 프로토콜(Protocol) – 값은 `http://`, `https://` 또는 `*://`로 시작해야 합니다.
- 쿼리 문자열 파라미터 - URL에 쿼리 문자열 파라미터가 있는 경우 백슬래시 문자(\)를 사용하지 않고 쿼리 문자열을 시작하는 물음표 문자(?)를 이스케이프 처리합니다. 예제:

  ```
  https://d111111abcdef8.cloudfront.net/images/horizon.jpg?
  size=large&license=yes
  ```
- 와일드카드 문자 - 정책의 URL에 와일드카드 문자를 사용할 수 있습니다. 다음 와일드카드 문자가 지원됩니다.
  - 별표(*). 0개 이상의 문자와 매칭됩니다.
  - 물음표(?). 정확히 한 문자와 매칭됩니다.

CloudFront가 정책의 URL을 HTTP 요청의 URL과 매칭하면 정책의 URL은 네 개의 섹션, 즉 프로토콜, 도메인, 경로, 쿼리 문자열로 구분됩니다.

```
[protocol]://[domain]/[path]\?[query string]
```

정책의 URL에 와일드카드 문자를 사용하는 경우 와일드카드 매칭은 와일드카드가 포함된 섹션의 경계 내에서만 적용됩니다. 예를 들어, 다음 정책에서 이 URL을 고려해 보세요.

```
https://www.example.com/hello*world
```

이 예시에서 별표 와일드카드(`*`)는 경로 섹션 내에만 적용되므로 URL `https://www.example.com/helloworld` 및 `https://www.example.com/hello-world`와는 매칭되지만 URL `https://www.example.net/hello?world`와는 매칭되지 않습니다.

와일드카드 매칭을 위한 섹션 경계에는 다음과 같은 예외가 적용됩니다.

- 경로 섹션의 후행 별표는 쿼리 문자열 섹션의 별표를 의미합니다. 예를 들어, `http://example.com/hello*`은 `http://example.com/hello*\?*`과 같습니다.
- 도메인 섹션의 후행 별표는 경로와 쿼리 문자열 섹션의 별표를 의미합니다. 예를 들어, `http://example.com*`은 `http://example.com*/*\?*`과 같습니다.
- 정책의 URL은 프로토콜 섹션을 생략하고 도메인 섹션에서 별표로 시작할 수 있습니다. 이 경우 프로토콜 섹션은 암시적으로 별표로 설정됩니다. 예를 들어, 정책의 `*example.com` URL은 `*://*example.com/`에 해당합니다.
- 별표(`"Resource": "*"`)는 그 자체로 모든 URL과 매칭됩니다.

예를 들어, 정책의 `https://d111111abcdef8.cloudfront.net/*game_download.zip*` 값은 다음 URL 모두와 매칭됩니다.

- `https://d111111abcdef8.cloudfront.net/game_download.zip`
- `https://d111111abcdef8.cloudfront.net/example_game_download.zip?license=yes`
- `https://d111111abcdef8.cloudfront.net/test_game_download.zip?license=temp`

- 대체 도메인 이름 – 정책의 URL에 대체 도메인 이름(CNAME)을 지정하는 경우, 웹 페이지 또는 애플리케이션에서 HTTP 요청이 대체 도메인 이름을 사용해야 합니다. 정책의 파일에 Amazon S3 URL을 지정하지 마시기 바랍니다.

DateLessThan

Unix 시간 형식(초) 및 협정 세계시(UTC) 기준의 URL 만료 날짜 및 시간. 정책에서 값을 인용 부호로 묶지 마세요. UTC에 대한 자세한 내용은 인터넷의 날짜 및 시간: 타임스탬프를 참조하세요.

예를 들면 2023년 1월 31일 오전 10시(UTC)를 Unix 시간 형식의 1675159200으로 변환합니다.

`Condition` 섹션에서 유일한 필수 파라미터입니다. CloudFront는 이 값을 통해 프라이빗 콘텐츠에 대한 영구적인 액세스를 방지합니다.

자세한 내용은 the section called "CloudFront가 서명된 URL에서 만료 날짜 및 시간을 확인하는 시기" 섹션을 참조하세요.

DateGreaterThan(선택 사항)

Unix 시간 형식(초)과 협정 세계시(UTC)의 URL에 대한 선택적 시작 날짜 및 시간. 사용자는 지정된 날짜 및 시간 이전에 파일에 액세스할 수 없습니다. 값을 인용 부호로 묶지 마세요.

IpAddress(선택 사항)

HTTP 요청을 수행하는 클라이언트의 IP 주소입니다. 다음 사항에 유의하세요.

- 파일에 액세스하는 IP 주소를 허용하려면 `IpAddress` 파라미터를 생략합니다.
- 하나의 IP 주소 또는 하나의 IP 주소 범위를 지정할 수 있습니다. 클라이언트의 IP 주소가 두 개의 다른 범위 중 하나에 있는 경우, 정책을 사용하여 액세스를 허용할 수 없습니다.
- 단일 IP 주소에서 액세스를 허용하기 위해 다음을 지정합니다.

  ```
  "IPv4 IP ##/32"
  ```

- IP 주소 범위는 스탠다드 IPv4 CIDR 형식(예: `192.0.2.0/24`)을 지정해야 합니다. 자세한 내용은 Classless Inter-domain Routing(CIDR): 인터넷 주소 할당 및 집계 계획을 참조하세요.

> **Important**
>
> IPv6 형식의 IP 주소(예: 2001:0db8:85a3::8a2e:0370:7334)는 지원되지 않습니다.

`IpAddress`를 포함하는 사용자 지정 정책을 사용할 경우 배포에 대해 IPv6를 사용하도록 설정하지 마세요. IP 주소로 일부 콘텐츠에 대한 액세스를 제한하고, 다른 콘텐츠에 대해서는 IPv6 요청을 지원하려면 두 가지 배포를 만들면 됩니다. 자세한 내용은 the section called "IPv6 활성화(뷰어 요청)" 주제에서 the section called "모든 배포 설정" 단원을 참조하세요.

## 사용자 지정 정책을 사용하는 서명된 URL에 대한 예제 정책 설명

다음 예제 정책 설명은 특정 파일, 디렉터리의 모든 파일 또는 키 페어 ID에 연결된 모든 파일의 액세스 제어 방법을 보여줍니다. 예제는 또한 개별 IP 주소 또는 IP 주소 범위에서 액세스를 제어하는 방법과 지정된 날짜 및 시간 이후 사용자의 서명된 URL 사용 방지 방법을 보여줍니다.

이러한 예시를 복사하여 붙여 넣는 경우, 공백(탭과 줄바꿈 문자 포함)을 제거하고 값을 자체 값과 교체한 후 닫는 괄호(})뒤에 줄바꿈 문자를 포함합니다.

자세한 내용은 the section called "사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명에서 지정한 값" 섹션을 참조하세요.

주제

- 예제 정책 설명: IP 주소 범위에서 하나의 파일에 액세스
- 예제 정책 설명: IP 주소 범위에서 디렉터리의 모든 파일에 액세스
- 예제 정책 설명: 하나의 IP 주소에서 키 페어 ID에 연결된 모든 파일에 액세스

### 예제 정책 설명: IP 주소 범위에서 하나의 파일에 액세스

다음 서명된 URL의 예시 사용자 지정 정책은 UTC 기준 2023년 1월 31일 오전 10시까지 `192.0.2.0/24` 범위의 IP 주소에서 `https://d111111abcdef8.cloudfront.net/game_download.zip` 파일에 액세스할 수 있는 사용자를 지정합니다.

```
{
    "Statement": [
        {
            "Resource": "https://d111111abcdef8.cloudfront.net/game_download.zip",
            "Condition": {
                "IpAddress": {
                    "AWS:SourceIp": "192.0.2.0/24"
                },
                "DateLessThan": {
                    "AWS:EpochTime": 1675159200
                }
            }
        }
    ]
}
```

예제 정책 설명: IP 주소 범위에서 디렉터리의 모든 파일에 액세스

다음 예시 사용자 지정 정책은 `Resource` 파라미터의 와일드카드 문자(`*`)로 표시된 바와 같이 `training` 디렉터리의 파일에 대한 서명된 URL을 만들 수 있도록 허용합니다. 사용자는 UTC 기준 2023년 1월 31일 오전 10시까지 `192.0.2.0/24` 범위의 IP 주소에서 파일에 액세스할 수 있습니다.

```
{
    "Statement": [
        {
            "Resource": "https://d111111abcdef8.cloudfront.net/training/*",
            "Condition": {
                "IpAddress": {
                    "AWS:SourceIp": "192.0.2.0/24"
                },
                "DateLessThan": {
                    "AWS:EpochTime": 1675159200
                }
            }
        }
    ]
}
```

이 정책과 함께 사용하는 서명된 URL에는 다음과 같이 특정 파일을 식별하는 URL이 있습니다.

`https://d111111abcdef8.cloudfront.net/training/orientation.pdf`

예제 정책 설명: 하나의 IP 주소에서 키 페어 ID에 연결된 모든 파일에 액세스

다음 예시 사용자 지정 정책은 `Resource` 파라미터의 와일드카드 문자(`*`)로 표시된 대로 배포와 연결된 파일에 대한 서명된 URL을 만들 수 있도록 허용합니다. 서명된 URL은 `http://` 프로토콜이 아닌 `https://` 프로토콜을 사용해야 합니다. 사용자는 IP 주소 `192.0.2.10/32`를 사용해야 합니다. CIDR 표기법의 `192.0.2.10/32` 값은 단일 IP 주소 `192.0.2.10`을 가리킵니다. 파일은 UTC 기준 2023년 1월 31일 오전 10시에서 2023년 2월 2일 오전 10시까지만 사용할 수 있습니다.

```
{
    "Statement": [
       {
            "Resource": "https://*",
            "Condition": {
                "IpAddress": {
                    "AWS:SourceIp": "192.0.2.10/32"
```

```
                },
                "DateGreaterThan": {
                    "AWS:EpochTime": 1675159200
                },
                "DateLessThan": {
                    "AWS:EpochTime": 1675332000
                }
            }
        }
    ]
}
```

이 정책과 사용하는 서명된 URL에는 다음과 같이 특정 CloudFront 배포의 특정 파일을 식별하는 URL이 있습니다.

`https://d111111abcdef8.cloudfront.net/training/orientation.pdf`

또한 서명된 URL은 키 페어 ID를 포함하고 이는 URL에서 지정한 배포 (d111111abcdef8.cloudfront.net)의 신뢰할 수 있는 키 그룹과 연결되어야 합니다.

사용자 지정 정책을 사용하는 서명된 URL에 대한 서명 생성

서명된 URL에 대한 서명은 정책 설명의 해시, 서명 및 base64로 인코딩된 버전의 사용자 지정 정책을 사용합니다. 사용자 지정 정책에 대한 서명을 만들려면 다음 단계를 수행합니다.

다음을 참조하여 정책 설명을 해시, 서명 및 인코딩하는 방법에 대한 추가적인 내용과 예제를 확인하세요.

- base64 인코딩 및 암호화를 위한 Linux 명령 및 OpenSSL
- 서명 URL에 대한 서명을 만드는 코드 예제

옵션 1: 사용자 지정 정책을 사용하여 서명을 만들려면

1. SHA-1 해시 함수와 생성된 RSA 또는 ECDSA 프라이빗 키를 사용하여 사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명을 만들려면 절차에서 만든 JSON 정책 설명을 해시하고 서명합니다. 더 이상 공백을 포함하지 않지만 base64로 인코딩되지 않은 정책 설명 버전을 사용합니다.

   해시 함수에 필요한 프라이빗 키의 경우 배포에 대해 신뢰할 수 있는 활성 키 그룹에 퍼블릭 키가 있는 프라이빗 키를 사용합니다.

> **Note**
>
> 정책 설명을 해시 및 서명하는 방법은 프로그래밍 언어와 플랫폼에 따라 달라집니다. 샘플 코드에 대한 내용은 서명 URL에 대한 서명을 만드는 코드 예제를 참조하세요.

2. 해시 및 서명된 문자열에서 공백(탭과 줄바꿈 문자 포함)을 제거합니다.
3. MIME base64 인코딩 기준으로 문자열을 base64로 인코딩합니다. 자세한 내용은 RFC 2045, MIME(Multipurpose Internet Mail Extensions) Part One: Format of Internet Message Bodies의 Section 6.8, Base64 Content-Transfer-Encoding을 참조하세요.
4. URL 쿼리 문자열에서 사용할 수 없는 문자를 유효한 문자로 교체합니다. 아래 표에 사용할 수 없는 문자와 유효한 문자가 나열되어 있습니다.

| 무효 문자를 | 교체할 유효한 문자 |
| --- | --- |
| + | - (하이픈) |
| = | _ (밑줄) |
| / | ~ (물결표) |

5. 서명된 URL의 `&Signature=` 뒤에 결과 값을 추가하고 사용자 지정 정책을 사용하여 서명된 URL을 만들려면로 돌아가서 서명된 URL 부분을 마저 연결합니다.

## 서명된 쿠키 사용

현재의 URL을 변경하지 않으려는 경우나 여러 제한된 파일(예: 웹 사이트의 구독자 영역에 있는 전체 파일)에 대한 액세스 권한을 제공하려는 경우, CloudFront 서명된 쿠키를 사용하여 콘텐츠 액세스를 제어할 수 있습니다. 이 주제에서는 서명된 쿠키를 사용할 때 고려해야 할 사항과 미리 준비된 정책 및 사용자 지정 정책을 사용하여 서명된 쿠키를 설정하는 방법을 다룹니다.

주제


- 서명된 쿠키에 대해 미리 준비된 정책 또는 사용자 지정 정책 사용 결정
- 서명된 쿠키 작동 방식
- 서명된 쿠키 악용 방지
- CloudFront가 서명된 쿠키의 만료 날짜 및 시간을 확인하는 시기
- 샘플 코드 및 타사 도구

- 미리 준비된 정책을 사용하여 서명된 쿠키 설정
- 사용자 지정 정책을 사용하여 서명된 쿠키 설정
- PHP를 사용하여 서명된 쿠키 생성

## 서명된 쿠키에 대해 미리 준비된 정책 또는 사용자 지정 정책 사용 결정

서명된 쿠키를 만들 때 JSON 형식의 정책 설명을 작성하여 서명된 쿠키의 제약 조건(예: 쿠키의 유효 기간)을 지정합니다. 미리 준비된 정책 또는 사용자 지정 정책을 사용할 수 있습니다. 다음 테이블은 미리 준비된 정책과 사용자 지정 정책을 비교합니다.

| 설명 | 미리 준비된 정책 | 사용자 정의 정책 |
| --- | --- | --- |
| 정책 설명은 여러 파일에 재사용할 수 있습니다. 정책 설명을 재사용하려면 `Resource` 객체에 와일드카드 문자를 사용해야 합니다. 자세한 내용은 서명된 쿠키에서 사용자 지정 정책을 위한 정책 설명에서 지정한 값 섹션을 참조하세요.) | 아니요 | 예 |
| 날짜 및 시간을 지정하여 사용자가 콘텐츠에 액세스를 시작할 수 있습니다. | 아니요 | 예(선택 사항) |
| 사용자의 콘텐츠 액세스를 중단할 날짜 및 시간을 지정할 수 있습니다. | 예 | 예 |
| 콘텐츠에 액세스할 수 있는 사용자의 IP 주소 또는 IP 주소 범위를 지정할 수 있습니다. | 아니요 | 예(선택 사항) |

미리 준비된(canned) 정책으로 서명된 쿠키를 만드는 방법에 대한 자세한 내용은 미리 준비된 정책을 사용하여 서명된 쿠키 설정을 참조하세요.

사용자 지정 정책으로 서명된 쿠키를 만드는 방법에 대한 자세한 내용은 사용자 지정 정책을 사용하여 서명된 쿠키 설정을 참조하세요.

## 서명된 쿠키 작동 방식

서명된 쿠키에 대한 CloudFront 구성 방법과 사용자가 서명된 쿠키를 포함한 요청을 제출할 때 CloudFront 반응에 대한 개요가 있습니다.

1. CloudFront 배포에서 CloudFront가 URL 서명을 확인하는 데 사용할 수 있는 퍼블릭 키가 포함된 신뢰할 수 있는 키 그룹을 하나 이상 지정합니다. 해당 프라이빗 키를 사용하여 URL에 서명합니다.

   자세한 내용은 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정 단원을 참조하세요.

2. 사용자가 콘텐츠에 액세스해야 하는지 여부를 파악하고, 필요하면 최종 사용자에게 `Set-Cookie` 헤더 세 개를 보내도록 애플리케이션을 개발합니다. (각 `Set-Cookie` 헤더는 이름-값 페어를 하나만 포함할 수 있고 CloudFront 서명된 쿠키에는 이름-값 페어 3개가 필요합니다.) 최종 사용자가 프라이빗 콘텐츠를 요청하기 전에 `Set-Cookie` 헤더를 최종 사용자에게 보내야 합니다. 쿠키의 만료 시간을 짧게 설정한 경우, 사용자가 계속 액세스할 수 있도록 후속 요청에 대해 `Set-Cookie` 헤더 세 개를 더 보낼 수 있습니다.

   일반적으로 CloudFront 배포에는 캐시 동작이 2개 이상 있는데 하나는 인증이 필요 없고 다른 하나는 필요합니다. 사이트 보안에 대한 오류 페이지에는 로그인 페이지로 가는 리디렉터 또는 링크가 있습니다.

   쿠키를 기반으로 파일을 캐시하도록 배포를 구성하는 경우, CloudFront는 서명된 쿠키의 속성을 기반으로 별도의 파일을 캐시하지 않습니다.

3. 사용자가 웹 사이트에 로그인하여 콘텐츠에 대한 비용을 지불하거나 액세스에 필요한 자격 요건을 충족합니다.

4. 애플리케이션은 응답을 통해 `Set-Cookie` 헤더를 반환하고 최종 사용자는 이름-값 페어를 저장합니다.

5. 사용자가 파일을 요청합니다.

   사용자의 브라우저 또는 기타 최종 사용자는 4단계의 이름-값 페어를 불러와서 이를 `Cookie` 헤더의 요청에 추가합니다. 이것이 서명된 쿠키입니다.

6. CloudFront는 퍼블릭 키를 사용하여 서명된 쿠키의 서명을 확인하고 쿠키가 아직 변조되지 않았음을 확인합니다. 유효하지 않은 서명인 경우 요청을 거부합니다.

   쿠키의 서명이 유효한 경우, CloudFront는 쿠키의 정책 설명을 보거나 미리 준비된(canned) 정책을 사용하는 경우, 정책 설명을 구성하여 요청이 아직 유효함을 확인합니다. 예를 들어, 쿠키의 시작 및 종료 날짜와 시간을 지정하는 경우, CloudFront는 액세스가 허용된 시간 동안 콘텐츠 액세스를 시도한 사용자를 확인합니다.

   요청이 정책 설명의 요구 사항을 충족하는 경우, CloudFront는 제한되지 않은 콘텐츠와 같은 방식으로 콘텐츠를 제공합니다. 다시 말해 파일이 이미 엣지 캐시에 있는지 여부를 판단하고, 필요에 따라 요청을 오리진으로 전달하고, 사용자에게 파일을 반환합니다.

## 서명된 쿠키 악용 방지

`Domain` 헤더에 `Set-Cookie` 파라미터를 지정할 때는 루트 도메인 이름을 가진 사용자의 액세스 가능성을 줄이도록 최대한 정확한 값을 지정하세요. 예를 들자면 example.com보다 app.example.com이 더 좋고, example.com을 제어하지 않을 때는 특히 그렇습니다. 이렇게 하면 www.example.com을 통한 콘텐츠 액세스를 방지할 수 있습니다.

이런 유형의 공격을 방지하려면 다음을 수행하세요.

- `Expires` 및 `Max-Age` 쿠키 속성을 제외하여 `Set-Cookie` 헤더에서 세션 쿠키를 만들도록 합니다. 세션 쿠키는 사용자가 브라우저를 닫을 때 자동으로 삭제되며, 따라서 권한 없는 제3자가 콘텐츠에 액세스할 가능성을 줄여 줍니다.
- `Secure` 속성을 포함하여 최종 사용자가 이를 요청할 때 쿠키가 암호화되도록 하세요.
- 가능하면 사용자 지정 정책을 사용하고 최종 사용자의 IP 주소를 포함합니다.
- `CloudFront-Expires` 속성에서 사용자의 콘텐츠 액세스를 허용할 기간을 기준으로 하여 가장 짧고 합리적인 만료 시간을 지정하세요.

## CloudFront가 서명된 쿠키의 만료 날짜 및 시간을 확인하는 시기

CloudFront는 서명된 쿠키가 아직 유효한지 파악하기 위해 HTTP 요청 시 쿠키의 만료 날짜 및 시간을 확인합니다. 클라이언트가 만료 시간 직전에 대용량 파일을 다운로드하기 시작한 경우, 다운로드 도중 만료 시간이 지나도 다운로드는 완료됩니다. TCP 연결이 끊어진 경우, 클라이언트가 만료 시간 이후에 다운로드를 다시 시작하는 것은 불가능합니다.

클라이언트가 범위 GET을 사용하여 파일을 작은 조각으로 가져오는 경우, 만료 시간 이후의 GET 요청은 실패합니다. 범위 GET에 대한 자세한 내용은 CloudFront에서 객체에 대한 부분적인 요청을 처리하는 방법(범위 GET)을 참조하세요.

## 샘플 코드 및 타사 도구

프라이빗 콘텐츠에 대한 샘플 코드는 서명된 URL에 대한 서명을 만드는 방법만 보여줍니다. 그러나 서명된 쿠키에 대한 서명 생성 프로세스도 이와 유사하기 때문에 대부분의 샘플 코드는 여전히 관계가 있습니다. 자세한 내용은 다음 항목을 참조하세요.

- Perl을 사용한 URL 서명 생성
- PHP를 사용한 URL 서명 생성
- C# 및 .NET Framework를 사용한 URL 서명 생성
- Java를 사용한 URL 서명 생성

## 미리 준비된 정책을 사용하여 서명된 쿠키 설정

미리 준비된 정책을 사용하여 서명된 쿠키를 설정하려면 다음 단계를 완료하세요. 서명을 만들려면 미리 준비된 정책을 사용하는 서명된 쿠키에 대한 서명 생성 단원을 참조하세요.

미리 준비된 정책을 사용하여 서명된 쿠키를 설정하려면

1. .NET 또는 Java를 사용하여 서명된 쿠키를 생성하는 중인데 키 페어의 프라이빗 키를 기본 .pem 형식에서 .NET 또는 Java와 상환되는 형식으로 다시 포맷하지 않았다면 지금 포맷하세요. 자세한 내용은 프라이빗 키 재포맷(.NET 및 Java만 해당) 단원을 참조하세요.
2. `Set-Cookie` 헤더 세 개를 승인된 최종 사용자에게 보내도록 애플리케이션을 프로그래밍합니다. 각 `Set-Cookie` 헤더는 이름-값 페어를 하나만 포함할 수 있고 CloudFront 서명된 쿠키에는 이름-값 페어 3개가 필요하기 때문에 `Set-Cookie` 헤더가 3개 있어야 합니다. 이름-값 페어는 `CloudFront-Expires`, `CloudFront-Signature`, `CloudFront-Key-Pair-Id`입니다. 액세스가 제한되는 파일에 대해 사용자가 첫 번째 요청을 하려면 최종 사용자에게 값이 있어야 합니다.

   > **Note**
   >
   > 일반적으로 `Expires` 및 `Max-Age` 속성을 제외하는 것이 좋습니다. 이러한 속성을 제외하면 사용자가 브라우저를 닫을 때 쿠키가 삭제되므로 권한 없는 사람이 콘텐츠에 액세스할 가능성이 줄어듭니다. 자세한 내용은 서명된 쿠키 악용 방지 단원을 참조하세요.

   쿠키 속성의 이름은 대소문자를 구분합니다.

   줄바꿈은 속성의 가독성을 높이기 위해서만 사용됩니다.

```
Set-Cookie:
CloudFront-Expires=date and time in Unix time format (in seconds) and Coordinated
 Universal Time (UTC);
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly

Set-Cookie:
CloudFront-Signature=hashed and signed version of the policy statement;
Domain=optional domain name;
Path=/optional directory path;
Secure;
```

```
HttpOnly

Set-Cookie:
CloudFront-Key-Pair-Id=public key ID for the CloudFront public key whose
 corresponding private key you're using to generate the signature;
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly
```

(선택 사항) **Domain**

요청된 파일의 도메인 이름입니다. `Domain` 속성을 지정하지 않는 경우, 기본값은 URL의 도메인 이름이고 이는 하위 도메인이 아닌 지정된 도메인 이름에만 적용됩니다. `Domain` 속성을 지정하는 경우, 하위 도메인에도 적용됩니다. 도메인 이름 앞의 점(예: `Domain=.example.com`)은 선택 사항입니다. 또한 `Domain` 속성을 지정하는 경우, URL의 도메인 이름과 `Domain` 속성의 값이 일치해야 합니다.

CloudFront가 배포에 할당하는 도메인 이름을 d111111abcdef8.cloudfront.net과 같이 지정할 수 있으나, *.cloudfront.net을 도메인 이름으로 지정할 수는 없습니다.

URL에 example.com과 같은 대체 도메인 이름을 사용하려면 `Domain` 속성 지정 여부와 상관없이 대체 도메인 이름을 배포에 추가해야 합니다. 자세한 내용은 대체 도메인 이름(CNAME) 주제에서 모든 배포 설정 참조 단원을 참조하세요.

(선택 사항) **Path**

요청된 파일의 경로입니다. `Path` 속성을 지정하지 않는 경우, 기본값은 URL의 경로입니다.

**Secure**

요청을 보내기 전 최종 사용자에게 쿠키의 암호화를 요청하세요. 쿠키 속성이 MITM(중간자 공격)을 당하지 않도록 HTTPS 연결을 통해 `Set-Cookie` 헤더를 보내는 것이 좋습니다.

**HttpOnly**

브라우저(지원되는 경우)가 쿠키 값과 상호 작용하는 방식을 정의합니다. `HttpOnly`를 사용하면 JavaScript에서 쿠키 값에 액세스할 수 없습니다. 이 예방 조치는 크로스 사이트 스크립팅(XSS) 공격을 완화하는 데 도움이 될 수 있습니다. 자세한 내용은 HTTP 쿠키 사용을 참조하세요.

**`CloudFront-Expires`**

Unix 시간 형식(초) 및 협정 세계시(UTC) 기준의 URL 만료 날짜 및 시간을 지정합니다. 예를 들면 2026년 1월 1일 오전 10시(UTC)를 Unix 시간 형식의 1767290400으로 변환합니다.

에포크 시간을 사용하려면 9223372036854775807(금요일, 2262년 4월 11일 23:47:16.854 UTC) 이전 날짜에 대한 64비트 정수를 지정합니다.

UTC에 대한 자세한 내용은 RFC 3339, 인터넷의 날짜 및 시간: 타임스탬프, https://tools.ietf.org/html/rfc3339를 참조하세요.

**`CloudFront-Signature`**

JSON 정책 설명의 해시, 서명 및 base64 인코딩 버전입니다. 자세한 내용은 미리 준비된 정책을 사용하는 서명된 쿠키에 대한 서명 생성 단원을 참조하세요.

**`CloudFront-Key-Pair-Id`**

CloudFront 퍼블릭 키의 ID입니다(예: `K2JCJMDEHXQW5F`). CloudFront는 서명된 URL을 확인할 때 사용할 퍼블릭 키를 퍼블릭 키 ID로 판단합니다. CloudFront는 서명의 정보를 정책 설명의 정보와 비교하고 URL이 변조되지 않았음을 확인합니다.

이 퍼블릭 키는 배포에서 신뢰할 수 있는 서명자인 키 그룹에 속해야 합니다. 자세한 내용은 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정 단원을 참조하세요.

다음 예제는 배포와 연결된 도메인 이름을 파일의 URL에 사용할 때 서명된 쿠키 한 개의 `Set-Cookie` 헤더를 보여줍니다.

```
Set-Cookie: CloudFront-Expires=1426500000; Domain=d111111abcdef8.cloudfront.net; Path=/
images/*; Secure; HttpOnly
Set-Cookie: CloudFront-Signature=yXrSIgyQoeE4FBI4eMKF6ho~CA8_;
 Domain=d111111abcdef8.cloudfront.net; Path=/images/*; Secure; HttpOnly
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F;
 Domain=d111111abcdef8.cloudfront.net; Path=/images/*; Secure; HttpOnly
```

다음 예제는 파일에 대한 URL의 example.org 대체 도메인 이름을 사용할 때 하나의 서명된 쿠키에 대한 예제 `Set-Cookie` 헤더를 보여줍니다.

```
Set-Cookie: CloudFront-Expires=1426500000; Domain=example.org; Path=/images/*; Secure;
 HttpOnly
```

```
Set-Cookie: CloudFront-Signature=yXrSIgyQoeE4FBI4eMKF6ho~CA8_; Domain=example.org;
 Path=/images/*; Secure; HttpOnly
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F; Domain=example.org; Path=/images/*;
 Secure; HttpOnly
```

URL에 example.com과 같은 대체 도메인 이름을 사용하려면 `Domain` 속성 지정 여부와 상관없이 대체 도메인 이름을 배포에 추가해야 합니다. 자세한 내용은 대체 도메인 이름(CNAME) 주제에서 모든 배포 설정 참조 단원을 참조하세요.

미리 준비된 정책을 사용하는 서명된 쿠키에 대한 서명 생성

미리 준비된 정책을 사용하는 서명된 쿠키의 서명을 만들려면 다음 절차를 완료하세요.

주제

- 미리 준비된 정책을 사용하는 서명된 쿠키에 대한 정책 설명 생성
- 정책 설명에 서명하여 미리 준비된 정책을 사용하는 서명된 쿠키에 대한 서명 생성

미리 준비된 정책을 사용하는 서명된 쿠키에 대한 정책 설명 생성

미리 준비된 정책을 사용하는 서명된 쿠키를 설정할 때 `CloudFront-Signature` 속성은 정책 설명의 해시 및 서명된 버전입니다. 미리 준비된 정책을 사용하는 서명된 쿠키의 경우, 사용자 지정 정책을 사용하는 서명된 쿠키와 마찬가지로 `Set-Cookie` 헤더에 정책 설명을 포함하지 않습니다. 정책 설명을 만들려면 다음 단계를 수행합니다.

미리 준비된 정책을 사용하는 서명된 쿠키에 대한 정책 설명을 만들려면

1. 다음 JSON 형식과 UTF-8 문자 인코딩을 사용하여 정책 설명을 구성합니다. 지정된 모든 문장 부호 및 기타 리터럴 값을 정확히 포함해야 합니다. `Resource` 및 `DateLessThan` 파라미터에 대한 내용은 서명된 쿠키에서 미리 준비된 정책을 위한 정책 설명에서 지정한 값를 참조하세요.

```
{
    "Statement": [
        {
            "Resource": "base URL or stream name",
            "Condition": {
                "DateLessThan": {
                    "AWS:EpochTime": ending date and time in Unix time format and
 UTC
                }
            }
```

```
        }
    ]
}
```

2. 정책 설명에서 모든 공백(탭과 줄바꿈 문자 포함)을 제거합니다. 애플리케이션 코드의 문자열에 이스케이프 문자를 포함해야 할 수도 있습니다.

서명된 쿠키에서 미리 준비된 정책을 위한 정책 설명에서 지정한 값

미리 준비된 정책에 대한 정책 설명을 만들 때 다음 값을 지정합니다.

리소스

쿼리 문자열을 포함하는 기본 URL(있는 경우)은 다음과 같습니다.

`https://d111111abcdef8.cloudfront.net/images/horizon.jpg?size=large&license=yes`

`Resource`에 대해 값을 하나만 지정할 수 있습니다.

다음을 참조하세요.

- 프로토콜(Protocol) – 값은 `http://` 또는 `https://`로 시작해야 합니다.
- 쿼리 문자열 파라미터(Query string parameters) – 쿼리 문자열 파라미터가 없는 경우, 물음표를 생략합니다.
- 대체 도메인 이름(Alternate domain names) – URL에 대체 도메인 이름(CNAME)을 지정하는 경우, 웹 페이지 또는 애플리케이션의 파일을 참조할 때 대체 도메인 이름을 지정해야 합니다. 파일에 대한 Amazon S3 URL을 지정하지 마시기 바랍니다.

DateLessThan

Unix 시간 형식(초) 및 협정 세계시(UTC) 기준의 URL 만료 날짜 및 시간. 값을 인용 부호로 묶지 마세요.

예를 들면 2015년 3월 16일 오전 10시(UTC)를 Unix 시간 형식인 1426500000으로 변환합니다.

이 값은 `CloudFront-Expires` 헤더의 `Set-Cookie` 속성 값과 일치해야 합니다. 값을 인용 부호로 묶지 마세요.

자세한 내용은 CloudFront가 서명된 쿠키의 만료 날짜 및 시간을 확인하는 시기 단원을 참조하세요.

미리 준비된 정책에 대한 정책 설명 예제

서명된 쿠키에 다음 예제의 정책 설명을 사용할 경우, 사용자는 UTC 기준 2015년 3월 16일 오전 10시까지 `https://d111111abcdef8.cloudfront.net/horizon.jpg` 파일에 액세스할 수 있습니다.

```
{
    "Statement": [
        {
            "Resource": "https://d111111abcdef8.cloudfront.net/horizon.jpg?
size=large&license=yes",
            "Condition": {
                "DateLessThan": {
                    "AWS:EpochTime": 1426500000
                }
            }
        }
    ]
}
```

정책 설명에 서명하여 미리 준비된 정책을 사용하는 서명된 쿠키에 대한 서명 생성

`CloudFront-Signature` 헤더의 `Set-Cookie` 속성의 값을 만들려면 미리 준비된 정책을 사용하는 서명된 쿠키에 대한 정책 설명을 만들려면에서 만든 정책 설명을 해시하고 서명합니다.

다음 주제를 참조하여 정책 설명을 해시, 서명 및 인코딩하는 방법에 대한 추가적인 내용과 예제를 확인하세요.

- base64 인코딩 및 암호화를 위한 Linux 명령 및 OpenSSL
- 서명 URL에 대한 서명을 만드는 코드 예제

미리 준비된 정책을 사용하는 서명된 쿠키에 대한 서명을 만들려면

1. SHA-1 해시 함수 및 RSA를 사용하여 미리 준비된 정책을 사용하는 서명된 쿠키에 대한 정책 설명을 만들려면 절차에서 만든 RSA 정책 설명을 해시하고 서명합니다. 공백이 삭제된 버전의 정책 설명을 사용합니다.

   해시 함수에 필요한 프라이빗 키의 경우 배포에 대해 신뢰할 수 있는 활성 키 그룹에 퍼블릭 키가 있는 프라이빗 키를 사용합니다.

> **Note**
> 정책 설명을 해시 및 서명하는 방법은 프로그래밍 언어와 플랫폼에 따라 달라집니다. 샘플 코드에 대한 내용은 서명 URL에 대한 서명을 만드는 코드 예제를 참조하세요.

2. 해시 및 서명된 문자열에서 공백(탭과 줄바꿈 문자 포함)을 제거합니다.
3. MIME base64 인코딩 기준으로 문자열을 base64로 인코딩합니다. 자세한 내용은 RFC 2045, MIME(Multipurpose Internet Mail Extensions) Part One: Format of Internet Message Bodies의 Section 6.8, Base64 Content-Transfer-Encoding을 참조하세요.
4. URL 쿼리 문자열에서 사용할 수 없는 문자를 유효한 문자로 교체합니다. 아래 표에 사용할 수 없는 문자와 유효한 문자가 나열되어 있습니다.

| 무효 문자를 | 교체할 유효한 문자 |
| --- | --- |
| + | - (하이픈) |
| = | _ (밑줄) |
| / | ~ (물결표) |

5. 결과로 얻은 값을 `Set-Cookie` 이름-값 페어의 `CloudFront-Signature` 헤더에 포함하세요. 그런 다음, 미리 준비된 정책을 사용하여 서명된 쿠키를 설정하려면 절차로 돌아가서 `Set-Cookie`에 대한 `CloudFront-Key-Pair-Id` 헤더를 추가합니다.

## 사용자 지정 정책을 사용하여 서명된 쿠키 설정

사용자 지정 정책을 사용하는 서명된 쿠키를 설정하려면 다음 단계를 수행합니다.

사용자 지정 정책을 사용하여 서명된 쿠키를 설정하려면

1. .NET 또는 Java를 사용하여 서명된 URL을 만드는 중인데 키 페어의 프라이빗 키를 기본 .pem 형식에서 .NET 또는 Java와 상환되는 형식으로 다시 포맷하지 않았다면 지금 포맷하세요. 자세한 내용은 프라이빗 키 재포맷(.NET 및 Java만 해당) 단원을 참조하세요.
2. `Set-Cookie` 헤더 세 개를 승인된 최종 사용자에게 보내도록 애플리케이션을 프로그래밍합니다. 각 `Set-Cookie` 헤더는 이름-값 페어를 하나만 포함할 수 있고 CloudFront 서명된 쿠키에는 이름-값 페어 3개가 필요하기 때문에 `Set-Cookie` 헤더가 3개 있어야 합니다. 이름-값 페어는

`CloudFront-Policy`, `CloudFront-Signature`, `CloudFront-Key-Pair-Id`입니다. 액세스가 제한되는 파일에 대해 사용자가 첫 번째 요청을 하려면 최종 사용자에게 값이 있어야 합니다.

> **Note**
>
> 일반적으로 `Expires` 및 `Max-Age` 속성을 제외하는 것이 좋습니다. 이렇게 하면 사용자가 브라우저를 닫을 때 쿠키가 삭제되므로 권한 없는 사람이 콘텐츠에 액세스할 가능성이 줄어듭니다. 자세한 내용은 서명된 쿠키 악용 방지 단원을 참조하세요.

쿠키 속성의 이름은 대소문자를 구분합니다.

줄바꿈은 속성의 가독성을 높이기 위해서만 사용됩니다.

```
Set-Cookie:
CloudFront-Policy=base64 encoded version of the policy statement;
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly


Set-Cookie:
CloudFront-Signature=hashed and signed version of the policy statement;
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly

Set-Cookie:
CloudFront-Key-Pair-Id=public key ID for the CloudFront public key whose
 corresponding private key you're using to generate the signature;
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly
```

(선택 사항) **`Domain`**

요청된 파일의 도메인 이름입니다. `Domain` 속성을 지정하지 않는 경우, 기본값은 URL의 도메인 이름이고 이는 하위 도메인이 아닌 지정된 도메인 이름에만 적용됩니다. `Domain` 속성을 지

정하는 경우, 하위 도메인에도 적용됩니다. 도메인 이름 앞의 점(예: `Domain=.example.com`)은 선택 사항입니다. 또한 `Domain` 속성을 지정하는 경우, URL의 도메인 이름과 `Domain` 속성의 값이 일치해야 합니다.

CloudFront가 배포에 할당하는 도메인 이름을 d111111abcdef8.cloudfront.net과 같이 지정할 수 있으나, *.cloudfront.net을 도메인 이름으로 지정할 수는 없습니다.

URL에 example.com과 같은 대체 도메인 이름을 사용하려면 `Domain` 속성 지정 여부와 상관없이 대체 도메인 이름을 배포에 추가해야 합니다. 자세한 내용은 대체 도메인 이름(CNAME) 주제에서 모든 배포 설정 참조 단원을 참조하세요.

(선택 사항) **`Path`**

요청된 파일의 경로입니다. `Path` 속성을 지정하지 않는 경우, 기본값은 URL의 경로입니다.

**`Secure`**

요청을 보내기 전 최종 사용자에게 쿠키의 암호화를 요청하세요. 쿠키 속성이 MITM(중간자 공격)을 당하지 않도록 HTTPS 연결을 통해 `Set-Cookie` 헤더를 보내는 것이 좋습니다.

**`HttpOnly`**

HTTP 또는 HTTPS 요청에 한해 최종 사용자에게 쿠키를 요구하세요.

**`CloudFront-Policy`**

공백이 제거된 JSON 형식의 정책 설명은 base64로 인코딩합니다. 자세한 내용은 사용자 지정 정책을 사용하는 서명된 쿠키에 대한 서명 생성 섹션을 참조하세요.

정책 설명은 서명된 쿠키가 사용자에게 부여하는 액세스를 제어합니다. 여기에는 사용자가 액세스할 수 있는 파일, 만료 날짜 및 시간, URL의 효력이 발생하는 날짜 및 시간(선택 사항), 파일에 액세스할 수 있는 IP 주소 또는 IP 주소 범위(선택 사항)가 포함됩니다.

**`CloudFront-Signature`**

JSON 정책 설명을 해시, 서명 및 base64로 인코딩한 버전입니다. 자세한 내용은 사용자 지정 정책을 사용하는 서명된 쿠키에 대한 서명 생성 단원을 참조하세요.

**`CloudFront-Key-Pair-Id`**

CloudFront 퍼블릭 키의 ID입니다(예: `K2JCJMDEHXQW5F`). CloudFront는 서명된 URL을 확인할 때 사용할 퍼블릭 키를 퍼블릭 키 ID로 판단합니다. CloudFront는 서명의 정보를 정책 설명의 정보와 비교하고 URL이 변조되지 않았음을 확인합니다.

이 퍼블릭 키는 배포에서 신뢰할 수 있는 서명자인 키 그룹에 속해야 합니다. 자세한 내용은 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정 섹션을 참조하세요.

### 사용자 지정 정책의 예제 헤더 **Set-Cookie**

Set-Cookie 헤더 쌍의 다음 예를 참조하십시오.

URL에 example.org와 같은 대체 도메인 이름을 사용하려면 Domain 속성 지정 여부와 상관없이 대체 도메인 이름을 배포에 추가해야 합니다. 자세한 내용은 대체 도메인 이름(CNAME) 주제에서 모든 배포 설정 참조 섹션을 참조하세요.

#### Example예 1

파일 URL에 배포와 연결된 도메인 이름을 사용하는 경우 서명된 쿠키 하나에 Set-Cookie 헤더를 사용할 수 있습니다.

```
Set-Cookie: CloudFront-
Policy=eyJTdGF0ZW1lbnQiOlt7IlJlc291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dh
 Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly
Set-Cookie: CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_;
 Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F;
 Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly
```

#### Example예제 2

파일의 URL에 대체 도메인 이름(example.org)을 사용하는 경우 서명된 쿠키 하나에 Set-Cookie 헤더를 사용할 수 있습니다.

```
Set-Cookie: CloudFront-
Policy=eyJTdGF0ZW1lbnQiOlt7IlJlc291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dh
 Domain=example.org; Path=/; Secure; HttpOnly
Set-Cookie: CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_; Domain=example.org;
 Path=/; Secure; HttpOnly
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F; Domain=example.org; Path=/; Secure;
 HttpOnly
```

#### Example예 3

파일의 URL에 배포와 연결된 도메인 이름을 사용하는 경우 서명된 요청에 Set-Cookie 헤더 쌍을 사용할 수 있습니다.

```
Set-Cookie: CloudFront-
Policy=eyJTdGF0ZW1lbnQiOlt7IlJlc291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dh
 Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly
Set-Cookie: CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_;
 Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F;
 Domain=dd111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly
```

Example예 4

파일 URL에 배포와 연결된 대체 도메인 이름(example.org)을 사용하는 경우 서명된 요청 하나에 `Set-Cookie` 헤더 쌍을 사용할 수 있습니다.

```
Set-Cookie: CloudFront-
Policy=eyJTdGF0ZW1lbnQiOlt7IlJlc291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dh
 Domain=example.org; Path=/; Secure; HttpOnly
Set-Cookie: CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_; Domain=example.org;
 Path=/; Secure; HttpOnly
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F; Domain=example.org; Path=/; Secure;
 HttpOnly
```

### 사용자 지정 정책을 사용하는 서명된 쿠키에 대한 정책 설명 생성

사용자 지정 정책에 대한 정책 설명을 만들려면 다음 단계를 수행합니다. 다양한 방법으로 파일에 대한 액세스를 제어하는 몇 가지 예제 정책 설명에 대한 내용은 사용자 지정 정책을 사용하는 서명된 쿠키에 대한 예제 정책 설명 단원을 참조하세요.

사용자 지정 정책을 사용하는 서명된 쿠키에 대한 정책 설명을 만들려면

1. 다음 JSON 형식을 사용하여 정책 설명을 구성하세요.

```
{
    "Statement": [
        {
            "Resource": "URL of the file",
            "Condition": {
                "DateLessThan": {
                    "AWS:EpochTime":required ending date and time in Unix time
 format and UTC
                },
                "DateGreaterThan": {
```

```
                    "AWS:EpochTime":optional beginning date and time in Unix time
 format and UTC
                },
                "IpAddress": {
                    "AWS:SourceIp": "optional IP address"
                }
            }
        }
    ]
}
```

다음을 참조하세요.

- 단 한 개의 명령문만 포함시킬 수 있습니다.
- UTF-8 문자 인코딩을 사용하세요.
- 지정된 대로 정확하게 모든 문장 부호 및 파라미터 이름을 포함하세요. 파라미터 이름의 약어는 허용되지 않습니다.
- `Condition` 섹션의 파라미터 순서는 중요하지 않습니다.
- `Resource`, `DateLessThan`, `DateGreaterThan`, `IpAddress`에 대한 값 관련 내용은 서명된 쿠키에서 사용자 지정 정책을 위한 정책 설명에서 지정한 값을 참조하세요.

2. 정책 설명에서 모든 공백(탭과 줄바꿈 문자 포함)을 제거합니다. 애플리케이션 코드의 문자열에 이스케이프 문자를 포함해야 할 수도 있습니다.
3. MIME base64 인코딩 기준으로 정책 설명을 Base64로 인코딩합니다. 자세한 내용은 RFC 2045, MIME(Multipurpose Internet Mail Extensions) Part One: Format of Internet Message Bodies의 Section 6.8, Base64 Content-Transfer-Encoding을 참조하세요.
4. URL 쿼리 문자열에서 사용할 수 없는 문자를 유효한 문자로 교체합니다. 아래 표에 사용할 수 없는 문자와 유효한 문자가 나열되어 있습니다.

| 무효 문자를 | 교체할 유효한 문자 |
| --- | --- |
| + | - (하이픈) |
| = | _ (밑줄) |
| / | ~ (물결표) |

5. `Set-Cookie` 뒤 `CloudFront-Policy=` 헤더의 결과 값을 포함하세요.
6. 정책 설명을 해시, 서명 및 base64로 인코딩하여 `Set-Cookie`의 `CloudFront-Signature` 헤더에 대한 서명을 만드세요. 자세한 내용은 사용자 지정 정책을 사용하는 서명된 쿠키에 대한 서명 생성 단원을 참조하세요.

서명된 쿠키에서 사용자 지정 정책을 위한 정책 설명에서 지정한 값

사용자 지정 정책에 대한 정책 설명을 생성할 때 다음 값을 지정합니다.

리소스

쿼리 문자열을 포함하는 기본 URL(있는 경우):

```
https://d111111abcdef8.cloudfront.net/images/horizon.jpg?
size=large&license=yes
```

> **Important**
>
> `Resource` 파라미터를 생략하는 경우, 사용자는 서명된 URL을 만들 때 사용하는 키 페어에 연결된 배포 관련 모든 파일에 액세스할 수 있습니다.

`Resource`에 대해 값을 하나만 지정할 수 있습니다.

다음을 참조하세요.

- 프로토콜(Protocol) – 값은 `http://` 또는 `https://`로 시작해야 합니다.
- 쿼리 문자열 파라미터(Query string parameters) – 쿼리 문자열 파라미터가 없는 경우, 물음표를 생략합니다.
- 와일드카드(Wildcards) – 0개 이상의 문자(*)에 해당하거나 문자열 내 정확히 1문자(?)에 해당하는 와일드카드를 문자열 내 모든 곳에 사용할 수 있습니다. 예를 들어 값은

  ```
  https://d111111abcdef8.cloudfront.net/*game_download.zip*
  ```

  다음 파일을 포함합니다.

  - `https://d111111abcdef8.cloudfront.net/game_download.zip`
  - `https://d111111abcdef8.cloudfront.net/example_game_download.zip?license=yes`

- `https://d111111abcdef8.cloudfront.net/test_game_download.zip?license=temp`
- 대체 도메인 이름 – URL에 대체 도메인 이름(CNAME)을 지정하는 경우, 웹 페이지 또는 애플리케이션의 파일을 참조할 때 대체 도메인 이름을 지정해야 합니다. 파일에 대한 Amazon S3 URL을 지정하지 마시기 바랍니다.

DateLessThan

Unix 시간 형식(초) 및 협정 세계시(UTC) 기준의 URL 만료 날짜 및 시간. 값을 인용 부호로 묶지 마세요.

예를 들면 2015년 3월 16일 오전 10시(UTC)를 Unix 시간 형식인 1426500000으로 변환합니다.

자세한 내용은 CloudFront가 서명된 쿠키의 만료 날짜 및 시간을 확인하는 시기 단원을 참조하세요.

DateGreaterThan(선택 사항)

Unix 시간 형식(초)과 협정 세계시(UTC)의 URL에 대한 선택적 시작 날짜 및 시간. 사용자는 지정된 날짜 및 시간 이전에 파일에 액세스할 수 없습니다. 값을 인용 부호로 묶지 마세요.

IpAddress(선택 사항)

클라이언트의 IP 주소는 GET 요청을 합니다. 다음을 참조하세요.

- 파일에 액세스하는 IP 주소를 허용하려면 `IpAddress` 파라미터를 생략합니다.
- 하나의 IP 주소 또는 하나의 IP 주소 범위를 지정할 수 있습니다. 예를 들면, 클라이언트의 IP 주소가 두 개의 다른 범위 중 하나에 있는 경우, 정책을 설정하여 액세스를 허용할 수 없습니다.
- 단일 IP 주소에서 액세스를 허용하기 위해 다음을 지정합니다.

  `"IPv4 IP ##/32"`
- IP 주소 범위는 스탠다드 IPv4 CIDR 형식(예: `192.0.2.0/24`)을 지정해야 합니다. 자세한 내용은 RFC 4632, Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan, https://tools.ietf.org/html/rfc4632를 참조하세요.

> **Important**
>
> IPv6 형식의 IP 주소(예: 2001:0db8:85a3::8a2e:0370:7334)는 지원되지 않습니다.

`IpAddress`를 포함하는 사용자 지정 정책을 사용할 경우 배포에 대해 IPv6를 사용하도록 설정하지 마세요. IP 주소로 일부 콘텐츠에 대한 액세스를 제한하고, 다른 콘텐츠에 대해서는 IPv6 요

청을 지원하려면 두 가지 배포를 만들면 됩니다. 자세한 내용은 IPv6 활성화(뷰어 요청) 주제에서 모든 배포 설정 참조 단원을 참조하세요.

## 사용자 지정 정책을 사용하는 서명된 쿠키에 대한 예제 정책 설명

다음 예제 정책 설명은 특정 파일, 디렉터리의 모든 파일 또는 키 페어 ID에 연결된 모든 파일의 액세스 제어 방법을 보여줍니다. 예제는 또한 개별 IP 주소 또는 IP 주소 범위에서 액세스를 제어하는 방법과 지정된 날짜 및 시간 이후 사용자의 서명된 쿠키 사용 방지 방법을 보여줍니다.

이러한 예제를 복사하여 붙여 넣는 경우, 공백(탭과 줄바꿈 문자 포함)을 제거하고 값을 자체 값과 교체한 후 닫는 괄호(})뒤에 줄바꿈 문자를 포함합니다.

자세한 내용은 서명된 쿠키에서 사용자 지정 정책을 위한 정책 설명에서 지정한 값 섹션을 참조하세요.

주제

- 예제 정책 설명: IP 주소 범위에서 하나의 파일에 액세스
- 예제 정책 설명: IP 주소 범위에서 디렉터리의 모든 파일에 액세스
- 예제 정책 설명: 하나의 IP 주소에서 키 페어 ID에 연결된 모든 파일에 액세스

### 예제 정책 설명: IP 주소 범위에서 하나의 파일에 액세스

다음 서명된 쿠키의 예제 사용자 지정 정책은 UTC 기준 2023년 1월 1일 오전 10시까지 `https://d111111abcdef8.cloudfront.net/game_download.zip` 범위의 IP 주소에서 `192.0.2.0/24` 파일에 액세스할 수 있는 사용자를 지정합니다.

```
{
    "Statement": [
        {
            "Resource": "https://d111111abcdef8.cloudfront.net/game_download.zip",
            "Condition": {
                "IpAddress": {
                    "AWS:SourceIp": "192.0.2.0/24"
                },
                "DateLessThan": {
                    "AWS:EpochTime": 1767290400
                }
            }
        }
    ]
```

```
}
```

예제 정책 설명: IP 주소 범위에서 디렉터리의 모든 파일에 액세스

다음 예제 사용자 지정 정책은 `training` 파라미터의 * 와일드카드 문자에 표시된 바와 같이 `Resource` 디렉터리의 파일에 대한 서명된 쿠키를 만들 수 있도록 허용합니다. 사용자는 UTC 기준 2013년 1월 1일 오전 10시까지 `192.0.2.0/24` 범위의 IP 주소에서 파일에 액세스할 수 있습니다.

```
{
    "Statement": [
        {
            "Resource": "https://d111111abcdef8.cloudfront.net/training/*",
            "Condition": {
                "IpAddress": {
                    "AWS:SourceIp": "192.0.2.0/24"
                },
                "DateLessThan": {
                    "AWS:EpochTime": 1767290400
                }
            }
        }
    ]
}
```

이 정책을 사용하는 서명된 쿠키는 각자 특정 파일을 식별하는 다음과 같은 기본 URL을 포함합니다.

`https://d111111abcdef8.cloudfront.net/training/orientation.pdf`

예제 정책 설명: 하나의 IP 주소에서 키 페어 ID에 연결된 모든 파일에 액세스

다음 샘플 사용자 지정 정책은 `Resource` 파라미터의 * 와일드카드 문자 표시된 바와 같이 배포와 연결된 파일에 대한 서명된 쿠키를 설정할 수 있도록 허용합니다. 사용자는 IP 주소 `192.0.2.10/32`를 사용해야 합니다. CIDR 표기법의 `192.0.2.10/32` 값은 단일 IP 주소 `192.0.2.10`을(를) 가리킵니다. 파일은 UTC 기준 2013년 1월 1일 오전 10시에서 2013년 1월 2일 오전 10시까지만 사용할 수 있습니다.

```
{
    "Statement": [
        {
            "Resource": "https://*",
            "Condition": {
```

```
                "IpAddress": {
                    "AWS:SourceIp": "192.0.2.10/32"
                },
                "DateGreaterThan": {
                    "AWS:EpochTime": 1767290400
                },
                "DateLessThan": {
                    "AWS:EpochTime": 1767376800
                }
            }
        }
    ]
}
```

이 정책을 사용하는 서명된 쿠키는 각각 특정 CloudFront 배포의 특정 파일을 식별하는 다음과 같은 기본 URL을 포함합니다.

`https://d111111abcdef8.cloudfront.net/training/orientation.pdf`

또한 서명된 쿠키는 키 페어 ID를 포함하고 이는 기본 URL에서 지정한 배포 (d111111abcdef8.cloudfront.net)의 신뢰할 수 있는 키 그룹과 연결되어야 합니다.

사용자 지정 정책을 사용하는 서명된 쿠키에 대한 서명 생성

서명된 쿠키에 대한 서명은 정책 설명의 해시, 서명 및 base64로 인코딩된 버전의 사용자 지정 정책을 사용합니다.

다음을 참조하여 정책 설명을 해시, 서명 및 인코딩하는 방법에 대한 추가적인 내용과 예제를 확인하세요.

- base64 인코딩 및 암호화를 위한 Linux 명령 및 OpenSSL
- 서명 URL에 대한 서명을 만드는 코드 예제

사용자 지정 정책으로 서명된 쿠키에 대한 서명을 만들려면

1. SHA-1 해시 함수 및 RSA를 사용하여 사용자 지정 정책을 사용하는 서명된 URL에 대한 정책 설명을 만들려면 절차에서 만든 JSON 정책 설명을 해시하고 서명합니다. 더 이상 공백을 포함하지 않지만 base64로 인코딩되지 않은 정책 설명 버전을 사용합니다.

   해시 함수에 필요한 프라이빗 키의 경우 배포에 대해 신뢰할 수 있는 활성 키 그룹에 퍼블릭 키가 있는 프라이빗 키를 사용합니다.

> **Note**
>
> 정책 설명을 해시 및 서명하는 방법은 프로그래밍 언어와 플랫폼에 따라 달라집니다. 샘플 코드에 대한 내용은 서명 URL에 대한 서명을 만드는 코드 예제를 참조하세요.

2. 해시 및 서명된 문자열에서 공백(탭과 줄바꿈 문자 포함)을 제거합니다.
3. MIME base64 인코딩 기준으로 문자열을 base64로 인코딩합니다. 자세한 내용은 RFC 2045, MIME(Multipurpose Internet Mail Extensions) Part One: Format of Internet Message Bodies의 Section 6.8, Base64 Content-Transfer-Encoding을 참조하세요.
4. URL 쿼리 문자열에서 사용할 수 없는 문자를 유효한 문자로 교체합니다. 아래 표에 사용할 수 없는 문자와 유효한 문자가 나열되어 있습니다.

| 무효 문자를 | 교체할 유효한 문자 |
| --- | --- |
| + | - (하이픈) |
| = | _ (밑줄) |
| / | ~ (물결표) |

5. `Set-Cookie` 이름-값 페어에 대한 `CloudFront-Signature=` 헤더의 결과 값을 포함하고 사용자 지정 정책을 사용하여 서명된 쿠키를 설정하려면으로 돌아가서 `Set-Cookie`에 대한 `CloudFront-Key-Pair-Id` 헤더를 추가하세요.

## PHP를 사용하여 서명된 쿠키 생성

다음 코드 예제는 동영상에 대한 링크를 만드는 PHP를 사용한 URL 서명 생성의 예제와 유사합니다. 그러나 이 예제에서는 코드의 URL에 서명하는 대신 `create_signed_cookies()` 함수를 사용하여 쿠키에 서명합니다. 클라이언트 측 플레이어는 쿠키를 사용하여 CloudFront 배포에 대한 각 요청을 인증합니다.

이 접근 방식은 클라이언트가 매니페스트, 세그먼트 및 관련 재생 자산을 검색하기 위해 여러 요청을 해야 하는 HLS(HTTP Live Streaming) 또는 DASH(Dynamic Adaptive Streaming over HTTP)와 같은 콘텐츠를 스트리밍하는 데 유용합니다. 클라이언트는 서명된 쿠키를 사용하여 각 세그먼트에 대해 서명된 새 URL을 만들 필요 없이 각 요청을 인증할 수 있습니다.

> **Note**
>
> - URL 서명을 만드는 것은 서명된 쿠키를 사용해 프라이빗 콘텐츠를 제공하는 프로세스의 한 부분에 불과합니다. 자세한 내용은 서명된 쿠키 사용 섹션을 참조하세요.

주제

- RSA SHA-1 서명 생성
- 서명된 쿠키 생성
- 전체 코드

다음 섹션에서는 코드 예제를 각 부분으로 나눕니다. 아래에서 전체 코드 샘플을 확인할 수 있습니다.

RSA SHA-1 서명 생성

이 코드 예제에서는 다음을 수행합니다.

1. `rsa_sha1_sign` 함수는 정책 문을 해시하고 서명합니다. 필요한 인수는 배포의 신뢰할 수 있는 키 그룹에 있는 퍼블릭 키에 해당하는 프라이빗 키와 정책 설명입니다.
2. 다음으로 `url_safe_base64_encode` 함수는 서명의 안전한 URL 버전을 만듭니다.

```
function rsa_sha1_sign($policy, $private_key_filename) {
    $signature = "";
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);
    openssl_sign($policy, $signature, $pkeyid);
    openssl_free_key($pkeyid);
    return $signature;
}

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
```

```
}
```

### 서명된 쿠키 생성

다음 코드는 다음 쿠키 속성(예: `CloudFront-Expires`, `CloudFront-Signature`, `CloudFront-Key-Pair-Id`)을 사용하여 서명된 쿠키를 만듭니다. 코드는 사용자 지정 정책을 사용합니다.

```
function create_signed_cookies($resource, $private_key_filename, $key_pair_id,
 $expires, $client_ip = null) {
    $policy = array(
        'Statement' => array(
            array(
                'Resource' => $resource,
                'Condition' => array(
                    'DateLessThan' => array('AWS:EpochTime' => $expires)
                )
            )
        )
    );

    if ($client_ip) {
        $policy['Statement'][0]['Condition']['IpAddress'] = array('AWS:SourceIp' =>
 $client_ip . '/32');
    }

    $policy = json_encode($policy);
    $encoded_policy = url_safe_base64_encode($policy);
    $signature = rsa_sha1_sign($policy, $private_key_filename);
    $encoded_signature = url_safe_base64_encode($signature);

    return array(
        'CloudFront-Policy' => $encoded_policy,
        'CloudFront-Signature' => $encoded_signature,
        'CloudFront-Key-Pair-Id' => $key_pair_id
    );
}
```

자세한 내용은 [사용자 지정 정책을 사용하여 서명된 쿠키 설정](#) 섹션을 참조하세요.

## 전체 코드

다음 예제 코드는 PHP로 CloudFront 서명 쿠키를 만드는 전체 데모를 제공합니다. demo-php.zip 파일에서 전체 예제를 다운로드할 수 있습니다.

다음 예제에서는 IPv4 및 IPv6 주소 범위를 모두 허용하도록 `$policy Condition` 요소를 수정할 수 있습니다. 예제는 Amazon Simple Storage Service 사용 설명서의 IAM 정책에서 IPv6 주소 사용을 참조합니다.

```
<?php

function rsa_sha1_sign($policy, $private_key_filename) {
    $signature = "";
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);
    openssl_sign($policy, $signature, $pkeyid);
    openssl_free_key($pkeyid);
    return $signature;
}

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
}

function create_signed_cookies($resource, $private_key_filename, $key_pair_id,
 $expires, $client_ip = null) {
    $policy = array(
        'Statement' => array(
            array(
                'Resource' => $resource,
                'Condition' => array(
                    'DateLessThan' => array('AWS:EpochTime' => $expires)
                )
            )
        )
    );
```

```
    if ($client_ip) {
        $policy['Statement'][0]['Condition']['IpAddress'] = array('AWS:SourceIp' =>
 $client_ip . '/32');
    }

    $policy = json_encode($policy);
    $encoded_policy = url_safe_base64_encode($policy);
    $signature = rsa_sha1_sign($policy, $private_key_filename);
    $encoded_signature = url_safe_base64_encode($signature);

    return array(
        'CloudFront-Policy' => $encoded_policy,
        'CloudFront-Signature' => $encoded_signature,
        'CloudFront-Key-Pair-Id' => $key_pair_id
    );
}



$private_key_filename = '/home/test/secure/example-priv-key.pem';
$key_pair_id = 'K2JCJMDEHXQW5F';
$base_url = 'https://d1234.cloudfront.net';

$expires = time() + 3600; // 1 hour from now

// Get the viewer real IP from the x-forward-for header as $_SERVER['REMOTE_ADDR']
 will return viewer facing IP. An alternative option is to use CloudFront-Viewer-
Address header. Note that this header is a trusted CloudFront immutable header. Example
 format: IP:PORT ("CloudFront-Viewer-Address": "1.2.3.4:12345")
$client_ip = $_SERVER['HTTP_X_FORWARDED_FOR'];


// For HLS manifest and segments (using wildcard)
$hls_resource = $base_url . '/sign/*';
$signed_cookies = create_signed_cookies($hls_resource, $private_key_filename,
 $key_pair_id, $expires, $client_ip);

// Set the cookies
$cookie_domain = parse_url($base_url, PHP_URL_HOST);
foreach ($signed_cookies as $name => $value) {
    setcookie($name, $value, $expires, '/', $cookie_domain, true, true);
}

?>
```

```
<!DOCTYPE html>
<html>
<head>
    <title>CloudFront Signed HLS Stream with Cookies</title>
</head>
<body>
    <h1>Amazon CloudFront Signed HLS Stream with Cookies</h1>
    <h2>Expires at <?php echo gmdate('Y-m-d H:i:s T', $expires); ?> only viewable by IP
 <?php echo $client_ip; ?></h2>

    <div id='hls-video'>
        <video id="video" width="640" height="360" controls></video>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/hls.js@latest"></script>
    <script>
        var video = document.getElementById('video');
        var manifestUrl = '<?php echo $base_url; ?>/sign/manifest.m3u8';

        if (Hls.isSupported()) {
            var hls = new Hls();
            hls.loadSource(manifestUrl);
            hls.attachMedia(video);
        }
        else if (video.canPlayType('application/vnd.apple.mpegurl')) {
            video.src = manifestUrl;
        }
    </script>
</body>
</html>
```

서명된 쿠키를 사용하는 대신 서명된 URL을 사용할 수 있습니다. 자세한 내용은 PHP를 사용한 URL 서명 생성 섹션을 참조하세요.

## base64 인코딩 및 암호화를 위한 Linux 명령 및 OpenSSL

다음 Linux 명령줄 명령과 OpenSSL을 사용하여 정책 설명을 해시 및 서명한 후, 서명을 base64로 인코딩하여 URL 쿼리 문자열 파라미터에서 사용할 수 없는 문자를 유효한 문자와 교체하세요.

OpenSSL에 대한 자세한 내용은 를 참조하세요https://www.openssl.org

```
cat policy | tr -d "\n" | tr -d " \t\n\r" | openssl sha1 -sign private_key.pem |
 openssl base64 -A | tr -- '+=/' '-_~'
```

앞의 명령에서:

- `cat` 은 `policy` 파일을 읽습니다.
- `tr -d "\n" | tr -d " \t\n\r"`은 `cat`에 의해 추가된 공백과 줄 바꿈 문자를 제거합니다.
- OpenSSL은 SHA-1을 사용하여 파일을 해시하고 프라이빗 키 파일 `private_key.pem`을 사용하여 서명합니다. 프라이빗 키 서명은 RSA 2048 또는 ECDSA 256일 수 있습니다.
- OpenSSL은 해시, 서명된 정책 문을 base64로 인코딩합니다.
- `tr` 은 URL 쿼리 문자열 파라미터에 사용할 수 없는 문자를 유효한 문자로 교체합니다.

서명을 만드는 방법을 보여주는 코드 예시는 서명 URL에 대한 서명을 만드는 코드 예제을 참조하세요.

## 서명 URL에 대한 서명을 만드는 코드 예제

이 단원에는 서명된 URL에 대한 서명을 만드는 방법을 담은 애플리케이션 예제가 수록되어 있으며 이를 다운로드할 수 있습니다. 예제는 Perl, PHP, C#, Java로 제공됩니다. 이러한 예제를 사용하여 서명된 URL을 만들 수 있습니다. Perl 스크립트는 Linux 및 masOS 플랫폼에서 실행됩니다. PHP 예제는 PHP를 실행하는 모든 서버에서 작동합니다. C# 예제는 NET Framework를 사용합니다.

JavaScript(Node.js)의 예제 코드는 AWS 개발자 블로그의 Node.js에서 Amazon CloudFront 서명 URL 생성을 참조하세요.

Python의 예제 코드는 AWS SDK for Python(Boto3) API 참조의 Amazon CloudFront에 대한 서명된 URL 생성과 Boto3 GitHub 리포지토리의 이 예제 코드를 참조하세요.

주제


- Perl을 사용한 URL 서명 생성
- PHP를 사용한 URL 서명 생성
- C# 및 .NET Framework를 사용한 URL 서명 생성
- Java를 사용한 URL 서명 생성


### Perl을 사용한 URL 서명 생성

이 단원에는 프라이빗 콘텐츠에 대한 서명을 생성하는 데 사용할 수 있는 Linux/Mac 플랫폼용 Perl 스크립트가 포함되어 있습니다. 서명을 생성하려면, CloudFront URL, 서명자의 프라이빗 키 경로, 키 ID

와 URL에 대한 만료 날짜를 지정하는 명령줄 인수를 포함한 스크립트를 실행합니다. 또한 이 도구는 서명된 URL을 디코딩할 수 있습니다.

> **Note**
>
> URL 서명 생성은 서명된 URL을 통해 프라이빗 콘텐츠를 제공하는 프로세스의 한 부분에 불과합니다. 종단 간 프로세스에 대한 자세한 내용은 서명된 URL 사용 단원을 참조합니다.

주제

- 서명된 URL을 생성하기 위한 Perl 스크립트의 소스

## 서명된 URL을 생성하기 위한 Perl 스크립트의 소스

서명된 CloudFront URL을 만들기 위해 다음과 같은 Perl 소스 코드를 사용할 수 있습니다. 이 코드의 명령에는 명령줄 전환 및 이 도구의 기능에 대한 자세한 내용이 포함되어 있습니다.

```
#!/usr/bin/perl -w

# Copyright 2008 Amazon Technologies, Inc.  Licensed under the Apache License, Version
 2.0 (the "License");
# you may not use this file except in compliance with the License. You may obtain a
 copy of the License at:
#
# https://aws.amazon.com/apache2.0
#
# This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
 KIND, either express or implied.
# See the License for the specific language governing permissions and limitations under
 the License.

=head1 cfsign.pl

cfsign.pl - A tool to generate and verify Amazon CloudFront signed URLs

=head1 SYNOPSIS

This script uses an existing RSA key pair to sign and verify Amazon CloudFront signed
 URLs

View the script source for details as to which CPAN packages are required beforehand.
```

```
For help, try:

cfsign.pl --help

URL signing examples:

cfsign.pl --action encode --url https://images.my-website.com/gallery1.zip --policy
 sample_policy.json --private-key privkey.pem --key-pair-id mykey

cfsign.pl --action encode --url https://images.my-website.com/gallery1.zip --expires
 1257439868 --private-key privkey.pem --key-pair-id mykey

URL decode example:

cfsign.pl --action decode --url "http//mydist.cloudfront.net/?Signature=AGO-
PgxkYo99MkJFHvjfGXjG1QDEXeaDb4Qtzmy85wqyJjK7eKojQWa4BCRcow__&Policy=eyJTdGF0ZW1lbnQiOlt7IlJlc29
Pair-Id=mykey"


To generate an RSA key pair, you can use openssl and the following commands:

# Generate a 2048 bit key pair
openssl genrsa -out private-key.pem 2048
openssl rsa -in private-key.pem -pubout -out public-key.pem


=head1 OPTIONS

=over 8

=item B<--help>

Print a help message and exits.

=item B<--action> [action]

The action to execute.  action can be one of:

  encode - Generate a signed URL (using a canned policy or a user policy)
  decode - Decode a signed URL

=item B<--url>
```

```
The URL to en/decode

=item B<--stream>

The stream to en/decode

=item B<--private-key>

The path to your private key.

=item B<--key-pair-id>

The key pair identifier.

=item B<--policy>

The CloudFront policy document.

=item B<--expires>

The Unix epoch time when the URL is to expire. If both this option and
the --policy option are specified, --policy will be used. Otherwise, this
option alone will use a canned policy.

=back

=cut

use strict;
use warnings;

# you might need to use CPAN to get these modules.
# run perl -MCPAN -e "install <module>" to get them.
# The openssl command line will also need to be in your $PATH.
use File::Temp qw/tempfile/;
use File::Slurp;
use Getopt::Long;
use IPC::Open2;
use MIME::Base64 qw(encode_base64 decode_base64);
use Pod::Usage;
use URI;

my $CANNED_POLICY
```

```
    = '{"Statement":[{"Resource":"<RESOURCE>","Condition":{"DateLessThan":
{"AWS:EpochTime":<EXPIRES>}}}]}';

my $POLICY_PARAM      = "Policy";
my $EXPIRES_PARAM     = "Expires";
my $SIGNATURE_PARAM   = "Signature";
my $KEY_PAIR_ID_PARAM = "Key-Pair-Id";

my $verbose = 0;
my $policy_filename = "";
my $expires_epoch = 0;
my $action = "";
my $help = 0;
my $key_pair_id = "";
my $url = "";
my $stream = "";
my $private_key_filename = "";

my $result = GetOptions("action=s"      => \$action,
                        "policy=s"      => \$policy_filename,
                        "expires=i"     => \$expires_epoch,
                        "private-key=s" => \$private_key_filename,
                        "key-pair-id=s" => \$key_pair_id,
                        "verbose"       => \$verbose,
                        "help"          => \$help,
                        "url=s"         => \$url,
                        "stream=s"      => \$stream,
                      );

if ($help or !$result) {
    pod2usage(1);
    exit;
}

if ($url eq "" and $stream eq "") {
    print STDERR "Must include a stream or a URL to encode or decode with the --stream
 or --url option\n";
    exit;
}

if ($url ne "" and $stream ne "") {
    print STDERR "Only one of --url and --stream may be specified\n";
    exit;
}
```

```
if ($url ne "" and !is_url_valid($url)) {
    exit;
}

if ($stream ne "") {
    exit unless is_stream_valid($stream);

    # The signing mechanism is identical, so from here on just pretend we're
    # dealing with a URL
    $url = $stream;
}

if ($action eq "encode") {
    # The encode action will generate a private content URL given a base URL,
    # a policy file (or an expires timestamp) and a key pair id parameter
    my $private_key;
    my $public_key;
    my $public_key_file;

    my $policy;
    if ($policy_filename eq "") {
        if ($expires_epoch == 0) {
            print STDERR "Must include policy filename with --policy argument or an
 expires" .
                         "time using --expires\n";
        }

        $policy = $CANNED_POLICY;
        $policy =~ s/<EXPIRES>/$expires_epoch/g;
        $policy =~ s/<RESOURCE>/$url/g;
    } else {
        if (! -e $policy_filename) {
            print STDERR "Policy file $policy_filename does not exist\n";
            exit;
        }
        $expires_epoch = 0; # ignore if set
        $policy = read_file($policy_filename);
    }

    if ($private_key_filename eq "") {
        print STDERR "You must specific the path to your private key file with --
private-key\n";
        exit;
```

```
    }

    if (! -e $private_key_filename) {
        print STDERR "Private key file $private_key_filename does not exist\n";
        exit;
    }

    if ($key_pair_id eq "") {
        print STDERR "You must specify a key pair id with --key-pair-id\n";
        exit;
    }

    my $encoded_policy = url_safe_base64_encode($policy);
    my $signature = rsa_sha1_sign($policy, $private_key_filename);
    my $encoded_signature = url_safe_base64_encode($signature);

    my $generated_url = create_url($url, $encoded_policy, $encoded_signature,
 $key_pair_id, $expires_epoch);


    if ($stream ne "") {
        print "Encoded stream (for use within a swf):\n" . $generated_url . "\n";
        print "Encoded and escaped stream (for use on a webpage):\n" .
 escape_url_for_webpage($generated_url) . "\n";
    } else {
        print "Encoded URL:\n" . $generated_url . "\n";
    }
} elsif ($action eq "decode") {
    my $decoded = decode_url($url);
    if (!$decoded) {
        print STDERR "Improperly formed URL\n";
        exit;
    }

    print_decoded_url($decoded);
} else {
    # No action specified, print help.  But only if this is run as a program (caller
 will be empty)
    pod2usage(1) unless caller();
}

# Decode a private content URL into its component parts
sub decode_url {
    my $url = shift;
```

```
    if ($url =~ /(.*)\?(.*)/) {
        my $base_url = $1;
        my $params = $2;

        my @unparsed_params = split(/&/, $params);
        my %params = ();
        foreach my $param (@unparsed_params) {
            my ($key, $val) = split(/=/, $param);
            $params{$key} = $val;
        }

        my $encoded_signature = "";
        if (exists $params{$SIGNATURE_PARAM}) {
            $encoded_signature = $params{"Signature"};
        } else {
            print STDERR "Missing Signature URL parameter\n";
            return 0;
        }

        my $encoded_policy = "";
        if (exists $params{$POLICY_PARAM}) {
            $encoded_policy = $params{$POLICY_PARAM};
        } else {
            if (!exists $params{$EXPIRES_PARAM}) {
                print STDERR "Either the Policy or Expires URL parameter needs to be
specified\n";
                return 0;
            }

            my $expires = $params{$EXPIRES_PARAM};

            my $policy = $CANNED_POLICY;
            $policy =~ s/<EXPIRES>/$expires/g;

            my $url_without_cf_params = $url;
            $url_without_cf_params =~ s/$SIGNATURE_PARAM=[^&]*&?//g;
            $url_without_cf_params =~ s/$POLICY_PARAM=[^&]*&?//g;
            $url_without_cf_params =~ s/$EXPIRES_PARAM=[^&]*&?//g;
            $url_without_cf_params =~ s/$KEY_PAIR_ID_PARAM=[^&]*&?//g;

            if ($url_without_cf_params =~ /(.*)\?$/) {
                $url_without_cf_params = $1;
            }
```

```
            $policy =~ s/<RESOURCE>/$url_without_cf_params/g;

            $encoded_policy = url_safe_base64_encode($policy);
        }

        my $key = "";
        if (exists $params{$KEY_PAIR_ID_PARAM}) {
            $key = $params{$KEY_PAIR_ID_PARAM};
        } else {
            print STDERR "Missing $KEY_PAIR_ID_PARAM parameter\n";
            return 0;
        }

        my $policy = url_safe_base64_decode($encoded_policy);

        my %ret = ();
        $ret{"base_url"} = $base_url;
        $ret{"policy"} = $policy;
        $ret{"key"} = $key;

        return \%ret;
    } else {
        return 0;
    }
}

# Print a decoded URL out
sub print_decoded_url {
    my $decoded = shift;

    print "Base URL: \n" . $decoded->{"base_url"} . "\n";
    print "Policy: \n" . $decoded->{"policy"} . "\n";
    print "Key: \n" . $decoded->{"key"} . "\n";
}

# Encode a string with base 64 encoding and replace some invalid URL characters
sub url_safe_base64_encode {
    my ($value) = @_;

    my $result = encode_base64($value);
    $result =~ tr|+=/|-_~|;

    return $result;
```

```
}

# Decode a string with base 64 encoding.  URL-decode the string first
# followed by reversing any special character ("+=/") translation.
sub url_safe_base64_decode {
    my ($value) = @_;

    $value =~ s/%([0-9A-Fa-f]{2})/chr(hex($1))/eg;
    $value =~ tr|-_~|+=/|;

    my $result = decode_base64($value);

    return $result;
}

# Create a private content URL
sub create_url {
    my ($path, $policy, $signature, $key_pair_id, $expires) = @_;

    my $result;
    my $separator = $path =~ /\?/ ? '&' : '?';
    if ($expires) {
        $result = "$path$separator$EXPIRES_PARAM=$expires&$SIGNATURE_PARAM=$signature&
$KEY_PAIR_ID_PARAM=$key_pair_id";
    } else {
        $result = "$path$separator$POLICY_PARAM=$policy&$SIGNATURE_PARAM=$signature&
$KEY_PAIR_ID_PARAM=$key_pair_id";
    }
    $result =~ s/\n//g;

    return $result;
}

# Sign a document with given private key file.
# The first argument is the document to sign
# The second argument is the name of the private key file
sub rsa_sha1_sign {
    my ($to_sign, $pvkFile) = @_;
    print "openssl sha1 -sign $pvkFile $to_sign\n";

    return write_to_program($pvkFile, $to_sign);
}

# Helper function to write data to a program
```

```
sub write_to_program {
my ($keyfile, $data) = @_;
unlink "temp_policy.dat" if (-e "temp_policy.dat");
unlink "temp_sign.dat" if (-e "temp_sign.dat");

write_file("temp_policy.dat", $data);

system("openssl dgst -sha1 -sign \"$keyfile\" -out temp_sign.dat temp_policy.dat");

my $output = read_file("temp_sign.dat");

    return $output;
}

# Read a file into a string and return the string
sub read_file {
    my ($file) = @_;

    open(INFILE, "<$file") or die("Failed to open $file: $!");
    my $str = join('', <INFILE>);
    close INFILE;

    return $str;
}

sub is_url_valid {
    my ($url) = @_;

    # HTTP distributions start with http[s]:// and are the correct thing to sign
    if ($url =~ /^https?:\/\//) {
        return 1;
    } else {
        print STDERR "CloudFront requires absolute URLs for HTTP distributions\n";
        return 0;
    }
}

sub is_stream_valid {
    my ($stream) = @_;

    if ($stream =~ /^rtmp:\/\// or $stream =~ /^\/?cfx\/st/) {
        print STDERR "Streaming distributions require that only the stream name is
 signed.\n";
```

```
        print STDERR "The stream name is everything after, but not including, cfx/st/
\n";
        return 0;
    } else {
        return 1;
    }
}

# flash requires that the query parameters in the stream name are url
# encoded when passed in through javascript, etc.  This sub handles the minimal
# required url encoding.
sub escape_url_for_webpage {
    my ($url) = @_;

    $url =~ s/\?/%3F/g;
    $url =~ s/=/%3D/g;
    $url =~ s/&/%26/g;

    return $url;
}

1;
```

## PHP를 사용한 URL 서명 생성

PHP를 실행하는 모든 웹 서버는 이 PHP 예제 코드를 사용하여 프라이빗 CloudFront 배포에 대한 정책 설명 및 서명을 만듭니다. 예제 전체에서는 CloudFront 스트리밍을 사용하여 비디오 스트림을 재생하는 서명된 URL 링크가 있는 정상 웹 페이지를 만듭니다. demo-php.zip 파일에서 전체 예제를 다운로드할 수 있습니다.

> **참고**
>
> - URL 서명 생성은 서명된 URL을 통해 프라이빗 콘텐츠를 제공하는 프로세스의 한 부분에 불과합니다. 전체 프로세스에 대한 자세한 내용은 서명된 URL 사용을 참조합니다.
> - AWS SDK for PHP에서 `UrlSigner` 클래스를 사용하여 서명된 URL을 생성할 수도 있습니다. 자세한 내용은 AWS SDK for PHP API 참조의 Class UrlSigner를 참조하세요.

### 주제

- RSA SHA-1 서명 생성

- 준비된 정책 생성
- 사용자 지정 정책 생성
- 전체 코드 예제

다음 섹션에서는 코드 예제를 각 부분으로 나눕니다. 아래 전체 코드 예제에서 확인할 수 있습니다.

RSA SHA-1 서명 생성

이 코드 예제에서는 다음을 수행합니다.

- `rsa_sha1_sign` 함수는 정책 문을 해시하고 서명합니다. 필요한 인수는 배포의 신뢰할 수 있는 키 그룹에 있는 퍼블릭 키에 해당하는 프라이빗 키와 정책 설명입니다.
- 다음으로 `url_safe_base64_encode` 함수는 서명의 안전한 URL 버전을 만듭니다.

```
function rsa_sha1_sign($policy, $private_key_filename) {
    $signature = "";

    // load the private key
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);

    // compute signature
    openssl_sign($policy, $signature, $pkeyid);

    // free the key from memory
    openssl_free_key($pkeyid);

    return $signature;
}

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    // replace unsafe characters +, = and / with
    // the safe characters -, _ and ~
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
```

```
}
```

다음 코드 조각은 `get_canned_policy_stream_name()` 함수 및 `get_custom_policy_stream_name()`을 사용하여 준비된 사용자 지정 정책을 만듭니다. CloudFront는 정책을 사용하여 만료 시간 지정을 비롯해 동영상 스트리밍용 URL을 만듭니다.

그런 다음 준비된 정책 또는 사용자 지정 정책을 사용하여 콘텐츠에 대한 액세스를 관리하는 방법을 결정할 수 있습니다. 선택할 항목에 대한 자세한 내용은 [서명된 URL에 대해 미리 준비된 정책 또는 사용자 지정 정책 사용 결정] 섹션을 참조하시기 바랍니다.

준비된 정책 생성

다음 예제 코드는 서명에 대한 미리 준비된 정책 설명을 구성합니다.

> **Note**
>
> `$expires` 변수는 문자열이 아닌 정수여야 하는 날짜/타임 스탬프입니다.

```
function get_canned_policy_stream_name($video_path, $private_key_filename,
 $key_pair_id, $expires) {
    // this policy is well known by CloudFront, but you still need to sign it, since it
 contains your parameters
    $canned_policy = '{"Statement":[{"Resource":"' . $video_path . '","Condition":
{"DateLessThan":{"AWS:EpochTime":'. $expires . '}}}]}';
    // the policy contains characters that cannot be part of a URL, so we base64 encode
 it
    $encoded_policy = url_safe_base64_encode($canned_policy);
    // sign the original policy, not the encoded version
    $signature = rsa_sha1_sign($canned_policy, $private_key_filename);
    // make the signature safe to be included in a URL
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, null, $encoded_signature,
 $key_pair_id, $expires);
    // URL-encode the query string characters
    return $stream_name;
}
```

미리 준비된 정책에 대한 자세한 내용은 [미리 준비된 정책을 사용하여 서명된 URL 생성]을 참조하세요.

### 사용자 지정 정책 생성

다음 예제 코드는 서명에 대한 사용자 지정 정책 설명을 구성합니다.

```
function get_custom_policy_stream_name($video_path, $private_key_filename,
 $key_pair_id, $policy) {
    // the policy contains characters that cannot be part of a URL, so we base64 encode
 it
    $encoded_policy = url_safe_base64_encode($policy);
    // sign the original policy, not the encoded version
    $signature = rsa_sha1_sign($policy, $private_key_filename);
    // make the signature safe to be included in a URL
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, $encoded_policy, $encoded_signature,
 $key_pair_id, null);
    // URL-encode the query string characters
    return $stream_name;
}
```

사용자 지정 정책에 대한 내용은 사용자 지정 정책을 사용하여 서명된 URL 생성을 참조하세요.

### 전체 코드 예제

다음 예제 코드는 PHP로 CloudFront 서명 URL을 만드는 전체 예제를 제공합니다. demo-php.zip 파일에서 전체 예제를 다운로드할 수 있습니다.

다음 예제에서는 IPv4 및 IPv6 주소 범위를 모두 허용하도록 `$policy` `Condition` 요소를 수정할 수 있습니다. 예제는 Amazon Simple Storage Service 사용 설명서의 IAM 정책에서 IPv6 주소 사용을 참조합니다.

```
<?php

function rsa_sha1_sign($policy, $private_key_filename) {
    $signature = "";

    // load the private key
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);
```

```
    // compute signature
    openssl_sign($policy, $signature, $pkeyid);

    // free the key from memory
    openssl_free_key($pkeyid);

    return $signature;
}

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    // replace unsafe characters +, = and / with the safe characters -, _ and ~
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
}

function create_stream_name($stream, $policy, $signature, $key_pair_id, $expires) {
    $result = $stream;
    // if the stream already contains query parameters, attach the new query parameters
 to the end
    // otherwise, add the query parameters
    $separator = strpos($stream, '?') == FALSE ? '?' : '&';
    // the presence of an expires time means we're using a canned policy
    if($expires) {
        $result .= $separator . "Expires=" . $expires . "&Signature=" . $signature .
 "&Key-Pair-Id=" . $key_pair_id;
    }
    // not using a canned policy, include the policy itself in the stream name
    else {
        $result .= $separator . "Policy=" . $policy . "&Signature=" . $signature .
 "&Key-Pair-Id=" . $key_pair_id;
    }

    // new lines would break us, so remove them
    return str_replace('\n', '', $result);
}


function get_canned_policy_stream_name($video_path, $private_key_filename,
 $key_pair_id, $expires) {
    // this policy is well known by CloudFront, but you still need to sign it, since it
 contains your parameters
```

```
    $canned_policy = '{"Statement":[{"Resource":"' . $video_path . '","Condition":
{"DateLessThan":{"AWS:EpochTime":'. $expires . '}}}]}';
    // the policy contains characters that cannot be part of a URL, so we base64 encode
 it
    $encoded_policy = url_safe_base64_encode($canned_policy);
    // sign the original policy, not the encoded version
    $signature = rsa_sha1_sign($canned_policy, $private_key_filename);
    // make the signature safe to be included in a URL
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, null, $encoded_signature,
 $key_pair_id, $expires);
    // URL-encode the query string characters
    return $stream_name;
}

function get_custom_policy_stream_name($video_path, $private_key_filename,
 $key_pair_id, $policy) {
    // the policy contains characters that cannot be part of a URL, so we base64 encode
 it
    $encoded_policy = url_safe_base64_encode($policy);
    // sign the original policy, not the encoded version
    $signature = rsa_sha1_sign($policy, $private_key_filename);
    // make the signature safe to be included in a URL
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, $encoded_policy, $encoded_signature,
 $key_pair_id, null);
    // URL-encode the query string characters
    return $stream_name;
}


// Path to your private key.  Be very careful that this file is not accessible
// from the web!

$private_key_filename = '/home/test/secure/example-priv-key.pem';
$key_pair_id = 'K2JCJMDEHXQW5F';

// Make sure you have "Restrict viewer access" enabled on this path behaviour and using
 the above Trusted key groups (recommended).
$video_path = 'https://example.com/secure/example.mp4';
```

```
$expires = time() + 300; // 5 min from now
$canned_policy_stream_name = get_canned_policy_stream_name($video_path,
 $private_key_filename, $key_pair_id, $expires);

// Get the viewer real IP from the x-forward-for header as $_SERVER['REMOTE_ADDR']
 will return viewer facing IP. An alternative option is to use CloudFront-Viewer-
Address header. Note that this header is a trusted CloudFront immutable header. Example
 format: IP:PORT ("CloudFront-Viewer-Address": "1.2.3.4:12345")
$client_ip = $_SERVER['HTTP_X_FORWARDED_FOR'];
$policy =
'{'.
    '"Statement":['.
        '{'.
            '"Resource":"'. $video_path . '",'.
            '"Condition":{'.
                '"IpAddress":{"AWS:SourceIp":"' . $client_ip . '/32"},'.
                '"DateLessThan":{"AWS:EpochTime":' . $expires . '}'.
            '}'.
        '}'.
    ']' .
    '}';
$custom_policy_stream_name = get_custom_policy_stream_name($video_path,
 $private_key_filename, $key_pair_id, $policy);

?>

<html>

<head>
    <title>CloudFront</title>
</head>

<body>
    <h1>Amazon CloudFront</h1>
    <h2>Canned Policy</h2>
    <h3>Expires at <?php echo gmdate('Y-m-d H:i:s T', $expires); ?></h3>
    <br />

    <div id='canned'>The canned policy video will be here: <br>

        <video width="640" height="360" autoplay muted controls>
        <source src="<?php echo $canned_policy_stream_name; ?>" type="video/mp4">
        Your browser does not support the video tag.
```

```
        </video>
    </div>

    <h2>Custom Policy</h2>
    <h3>Expires at <?php echo gmdate('Y-m-d H:i:s T', $expires); ?> only viewable by IP
 <?php echo $client_ip; ?></h3>
    <div id='custom'>The custom policy video will be here: <br>

         <video width="640" height="360" autoplay muted controls>
         <source src="<?php echo $custom_policy_stream_name; ?>" type="video/mp4">
         Your browser does not support the video tag.
        </video>
    </div>

</body>

</html>
```

추가 URL 서명 예제를 알아보려면 다음 주제를 참조하시기 바랍니다.

- Perl을 사용한 URL 서명 생성
- C# 및 .NET Framework를 사용한 URL 서명 생성
- Java를 사용한 URL 서명 생성

서명된 URL을 사용하여 서명을 만드는 대신 서명된 쿠키를 사용할 수 있습니다. 자세한 내용은 PHP를 사용하여 서명된 쿠키 생성 섹션을 참조하세요.

## C# 및 .NET Framework를 사용한 URL 서명 생성

이 섹션의 C# 예제는 사용자 지정 및 미리 준비된 정책 설명을 사용하여 CloudFront 프라이빗 배포에 대한 서명을 만드는 방법을 보여주는 예제 애플리케이션을 구현합니다. 이 예제는 .NET 애플리케이션에서 유용할 수 있는 AWS SDK for .NET를 기반으로 한 유틸리티 함수를 포함합니다.

SDK for .NET를 사용하여 서명된 URL과 서명된 쿠키를 생성할 수도 있습니다. SDK for .NET API 참조에서 다음 주제를 참조하세요.

- 서명된 URL – AmazonCloudFrontUrlSigner
- 서명된 쿠키 – AmazonCloudFrontCookieSigner

코드를 다운로드하려면 C#의 서명 코드를 참조하세요.

> **참고**
>
> - `AmazonCloudFrontUrlSigner` 및 `AmazonCloudFrontCookieSigner` 클래스가 별도의 패키지로 이전되었습니다. 해당 클래스 사용에 대한 자세한 내용은 AWS SDK for .NET(V4) 개발자 안내서의 CookieSigner and UrlSigner를 참조하시기 바랍니다.
> - URL 서명 생성은 서명된 URL을 통해 프라이빗 콘텐츠를 제공하는 프로세스의 한 부분에 불과합니다. 자세한 내용은 서명된 URL 사용 섹션을 참조하세요. 서명된 쿠키 사용에 대한 자세한 내용은 서명된 쿠키 사용 섹션을 참조하세요.

.NET Framework에서 RSA 키 사용

.NET Framework에서 RSA 키를 사용하려면 AWS가 제공한 .pem 파일을 .NET Framework에서 사용하는 XML 형식으로 변환해야 합니다.

변환 후, RSA 프라이빗 키 파일의 형식은 다음과 같습니다.

Example: XML .NET Framework 형식의 RSA 프라이빗 키

```
<RSAKeyValue>
  <Modulus>
    wO5IvYCP5UcoCKDo1dcspoMehWBZcyfs9QEzGi6Oe5y+ewGr1oW+vB2GPB
    ANBiVPcUHTFWhwaIBd3oglmF0lGQljP/jOfmXHUK2kUUnLnJp+oOBL2NiuFtqcW6h/L5lIpD8Yq+NRHg
    Ty4zDsyr2880MvXv88yEFURCkqEXAMPLE=
  </Modulus>
  <Exponent>AQAB</Exponent>
  <P>
    5bmKDaTz
    npENGVqz4Cea8XPH+sxt+2VaAwYnsarVUoSBeVt8WLloVuZGG9IZYmH5KteXEu7fZveYd9UEXAMPLE==
  </P>
  <Q>
    1v9l/WN1a1N3rOK4VGoCokx7kR2SyTMSbZgF9IWJNOugR/WZw7HTnjipO3c9dy1Ms9pUKwUF4
    6d7049EXAMPLE==
  </Q>
  <DP>
    RgrSKuLWXMyBH+/l1Dx/I4tXuAJIrlPyo+VmiOc7b5NzHptkSHEPfR9s1
    OK0VqjknclqCJ3Ig86OMEtEXAMPLE==
  </DP>
  <DQ>
    pjPjvSFw+RoaTu0pgCA/jwW/FGyfN6iim1RFbkT4
    z49DZb2IM885f3vf35eLTaEYRYUHQgZtChNEV0TEXAMPLE==
```

```
  </DQ>
  <InverseQ>
    nkvOJTg5QtGNgWb9i
    cVtzrL/1pFEOHbJXwEJdU99N+7sMK+1066DL/HSBUCD63qD4USpnf0myc24in0EXAMPLE==</InverseQ>
  <D>
      Bc7mp7XYHynuPZxChjWNJZIq+A73gm0ASDv6At7F8Vi9r0xUlQe/v0AQS3ycN8QlyR4XMbzMLYk
      3yjxFDXo4ZKQtOGzLGteCU2srANiLv26/imXA8FVidZftTAtLviWQZBVPTeYIA69ATUYPEq0a5u5wjGy
      UOij9OWyuEXAMPLE=
   </D>
</RSAKeyValue>
```

### C#의 미리 준비된 정책 서명 메서드

다음 C# 코드는 다음을 수행하여 미리 준비된 정책을 사용하는 서명된 URL을 만듭니다.

- 정책 설명을 만듭니다.
- SHA1을 사용하여 정책 설명을 해시하고, RSA와 해당 퍼블릭 키가 신뢰할 수 있는 키 그룹에 있는 프라이빗 키를 사용하여 결과에 서명합니다.
- Base64 방식으로 해시 및 서명된 정책 설명을 인코딩하고 특수 문자를 교체하여 URL 요청 파라미터로 사용하기에 안전한 문자열을 만드세요.
- 값을 연결합니다.

완전한 구현에 대한 내용은 C#의 서명 코드에서 예제를 참조하세요.

> **Note**
>
> CloudFront에 퍼블릭 키를 업로드하면 `keyId`가 반환됩니다. 자세한 내용은
>
> 6
>
> &Key-Pair-Id를 참조하세요.

### Example: C#의 미리 준비된 정책 서명 메서드

```
public static string ToUrlSafeBase64String(byte[] bytes)
{
    return System.Convert.ToBase64String(bytes)
        .Replace('+', '-')
        .Replace('=', '_')
        .Replace('/', '~');
}
```

```
public static string CreateCannedPrivateURL(string urlString,
    string durationUnits, string durationNumber, string pathToPolicyStmnt,
    string pathToPrivateKey, string keyId)
{
    // args[] 0-thisMethod, 1-resourceUrl, 2-seconds-minutes-hours-days
    // to expiration, 3-numberOfPreviousUnits, 4-pathToPolicyStmnt,
    // 5-pathToPrivateKey, 6-keyId

    TimeSpan timeSpanInterval = GetDuration(durationUnits, durationNumber);

    // Create the policy statement.
    string strPolicy = CreatePolicyStatement(pathToPolicyStmnt,
        urlString,
        DateTime.Now,
        DateTime.Now.Add(timeSpanInterval),
        "0.0.0.0/0");
    if ("Error!" == strPolicy) return "Invalid time frame." +
        "Start time cannot be greater than end time.";

    // Copy the expiration time defined by policy statement.
    string strExpiration = CopyExpirationTimeFromPolicy(strPolicy);

    // Read the policy into a byte buffer.
    byte[] bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);

    // Initialize the SHA1CryptoServiceProvider object and hash the policy data.
    using (SHA1CryptoServiceProvider
        cryptoSHA1 = new SHA1CryptoServiceProvider())
    {
        bufferPolicy = cryptoSHA1.ComputeHash(bufferPolicy);

        // Initialize the RSACryptoServiceProvider object.
        RSACryptoServiceProvider providerRSA = new RSACryptoServiceProvider();
        XmlDocument xmlPrivateKey = new XmlDocument();

        // Load your private key, which you created by converting your
        // .pem file to the XML format that the .NET framework uses.
        // Several tools are available.
        xmlPrivateKey.Load(pathToPrivateKey);

        // Format the RSACryptoServiceProvider providerRSA and
        // create the signature.
        providerRSA.FromXmlString(xmlPrivateKey.InnerXml);
```

```
        RSAPKCS1SignatureFormatter rsaFormatter =
            new RSAPKCS1SignatureFormatter(providerRSA);
        rsaFormatter.SetHashAlgorithm("SHA1");
        byte[] signedPolicyHash = rsaFormatter.CreateSignature(bufferPolicy);

        // Convert the signed policy to URL-safe base64 encoding and
        // replace unsafe characters + = / with the safe characters - _ ~
        string strSignedPolicy = ToUrlSafeBase64String(signedPolicyHash);

        // Concatenate the URL, the timestamp, the signature,
        // and the key pair ID to form the signed URL.
        return urlString +
            "?Expires=" +
            strExpiration +
            "&Signature=" +
            strSignedPolicy +
            "&Key-Pair-Id=" +
            keyId;
    }
}
```

C#의 사용자 지정 정책 서명 메서드

다음 C# 코드는 다음을 수행하여 사용자 지정 정책을 사용하는 서명된 URL을 만듭니다.

1. 정책 설명을 만듭니다.
2. Base64 방식으로 정책 설명을 인코딩하고 특수 문자를 교체하여 URL 요청 파라미터로 사용하기에 안전한 문자열을 만드세요.
3. SHA1을 사용하여 정책 설명을 해시하고, RSA와 해당 퍼블릭 키가 신뢰할 수 있는 키 그룹에 있는 프라이빗 키를 사용하여 결과를 암호화합니다.
4. Base64 방식으로 해시된 정책 설명을 인코딩하고 특수 문자를 교체하여 URL 요청 파라미터로 사용하기에 안전한 문자열을 만드세요.
5. 값을 연결합니다.

완전한 구현에 대한 내용은 C#의 서명 코드에서 예제를 참조하세요.

> **Note**
>
> CloudFront에 퍼블릭 키를 업로드하면 `keyId`가 반환됩니다. 자세한 내용은
>
> 6
>
> &Key-Pair-Id를 참조하세요.

Example: C#의 사용자 지정 정책 서명 메서드

```
public static string ToUrlSafeBase64String(byte[] bytes)
{
    return System.Convert.ToBase64String(bytes)
        .Replace('+', '-')
        .Replace('=', '_')
        .Replace('/', '~');
}

public static string CreateCustomPrivateURL(string urlString,
    string durationUnits, string durationNumber, string startIntervalFromNow,
    string ipaddress, string pathToPolicyStmnt, string pathToPrivateKey,
    string keyId)
{
    // args[] 0-thisMethod, 1-resourceUrl, 2-seconds-minutes-hours-days
    // to expiration, 3-numberOfPreviousUnits, 4-starttimeFromNow,
    // 5-ip_address, 6-pathToPolicyStmt, 7-pathToPrivateKey, 8-keyId

    TimeSpan timeSpanInterval = GetDuration(durationUnits, durationNumber);
    TimeSpan timeSpanToStart = GetDurationByUnits(durationUnits,
        startIntervalFromNow);
    if (null == timeSpanToStart)
        return "Invalid duration units." +
            "Valid options: seconds, minutes, hours, or days";

    string strPolicy = CreatePolicyStatement(
        pathToPolicyStmnt, urlString, DateTime.Now.Add(timeSpanToStart),
        DateTime.Now.Add(timeSpanInterval), ipaddress);

    // Read the policy into a byte buffer.
    byte[] bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);

    // Convert the policy statement to URL-safe base64 encoding and
    // replace unsafe characters + = / with the safe characters - _ ~
```

```
    string urlSafePolicy = ToUrlSafeBase64String(bufferPolicy);

    // Initialize the SHA1CryptoServiceProvider object and hash the policy data.
    byte[] bufferPolicyHash;
    using (SHA1CryptoServiceProvider cryptoSHA1 =
        new SHA1CryptoServiceProvider())
    {
        bufferPolicyHash = cryptoSHA1.ComputeHash(bufferPolicy);

        // Initialize the RSACryptoServiceProvider object.
        RSACryptoServiceProvider providerRSA = new RSACryptoServiceProvider();
        XmlDocument xmlPrivateKey = new XmlDocument();

        // Load your private key, which you created by converting your
        // .pem file to the XML format that the .NET framework uses.
        // Several tools are available.
        xmlPrivateKey.Load(pathToPrivateKey);

        // Format the RSACryptoServiceProvider providerRSA
        // and create the signature.
        providerRSA.FromXmlString(xmlPrivateKey.InnerXml);
        RSAPKCS1SignatureFormatter RSAFormatter =
            new RSAPKCS1SignatureFormatter(providerRSA);
        RSAFormatter.SetHashAlgorithm("SHA1");
        byte[] signedHash = RSAFormatter.CreateSignature(bufferPolicyHash);

        // Convert the signed policy to URL-safe base64 encoding and
        // replace unsafe characters + = / with the safe characters - _ ~
        string strSignedPolicy = ToUrlSafeBase64String(signedHash);

        return urlString +
            "?Policy=" +
            urlSafePolicy +
            "&Signature=" +
            strSignedPolicy +
            "&Key-Pair-Id=" +
            keyId;
    }
}
```

## 서명 생성을 위한 유틸리티 메서드

다음 메서드는 파일에서 정책 설명을 가져오고 서명 생성을 위해 시간 간격을 구문 분석합니다.

## Example: 서명 생성을 위한 유틸리티 메서드

```
public static string CreatePolicyStatement(string policyStmnt,
   string resourceUrl,
   DateTime startTime,
   DateTime endTime,
   string ipAddress)

{
   // Create the policy statement.
   FileStream streamPolicy = new FileStream(policyStmnt, FileMode.Open,
 FileAccess.Read);
   using (StreamReader reader = new StreamReader(streamPolicy))
   {
      string strPolicy = reader.ReadToEnd();

      TimeSpan startTimeSpanFromNow = (startTime - DateTime.Now);
      TimeSpan endTimeSpanFromNow = (endTime - DateTime.Now);
      TimeSpan intervalStart =
         (DateTime.UtcNow.Add(startTimeSpanFromNow)) -
         new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);
      TimeSpan intervalEnd =
         (DateTime.UtcNow.Add(endTimeSpanFromNow)) -
         new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);

      int startTimestamp = (int)intervalStart.TotalSeconds; // START_TIME
      int endTimestamp = (int)intervalEnd.TotalSeconds;  // END_TIME

      if (startTimestamp > endTimestamp)
         return "Error!";

      // Replace variables in the policy statement.
      strPolicy = strPolicy.Replace("RESOURCE", resourceUrl);
      strPolicy = strPolicy.Replace("START_TIME", startTimestamp.ToString());
      strPolicy = strPolicy.Replace("END_TIME", endTimestamp.ToString());
      strPolicy = strPolicy.Replace("IP_ADDRESS", ipAddress);
      strPolicy = strPolicy.Replace("EXPIRES", endTimestamp.ToString());
      return strPolicy;
   }
}

public static TimeSpan GetDuration(string units, string numUnits)
{
   TimeSpan timeSpanInterval = new TimeSpan();
```

```
   switch (units)
   {
      case "seconds":
         timeSpanInterval = new TimeSpan(0, 0, 0, int.Parse(numUnits));
         break;
      case "minutes":
         timeSpanInterval = new TimeSpan(0, 0, int.Parse(numUnits), 0);
         break;
      case "hours":
         timeSpanInterval = new TimeSpan(0, int.Parse(numUnits), 0 ,0);
         break;
      case "days":
         timeSpanInterval = new TimeSpan(int.Parse(numUnits),0 ,0 ,0);
         break;
      default:
         Console.WriteLine("Invalid time units;" +
            "use seconds, minutes, hours, or days");
         break;
   }
   return timeSpanInterval;
}

private static TimeSpan GetDurationByUnits(string durationUnits,
   string startIntervalFromNow)
{
   switch (durationUnits)
   {
      case "seconds":
         return new TimeSpan(0, 0, int.Parse(startIntervalFromNow));
      case "minutes":
         return new TimeSpan(0, int.Parse(startIntervalFromNow), 0);
      case "hours":
         return new TimeSpan(int.Parse(startIntervalFromNow), 0, 0);
      case "days":
         return new TimeSpan(int.Parse(startIntervalFromNow), 0, 0, 0);
      default:
         return new TimeSpan(0, 0, 0, 0);
   }
}

public static string CopyExpirationTimeFromPolicy(string policyStatement)
{
   int startExpiration = policyStatement.IndexOf("EpochTime");
   string strExpirationRough = policyStatement.Substring(startExpiration +
```

```
      "EpochTime".Length);
   char[] digits = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' };

   List<char> listDigits = new List<char>(digits);
   StringBuilder buildExpiration = new StringBuilder(20);

   foreach (char c in strExpirationRough)
   {
      if (listDigits.Contains(c))
         buildExpiration.Append(c);
   }
   return buildExpiration.ToString();
}
```

다음 사항도 참조하세요.

- Perl을 사용한 URL 서명 생성
- PHP를 사용한 URL 서명 생성
- Java를 사용한 URL 서명 생성

## Java를 사용한 URL 서명 생성

다음 코드 예제 외에도 AWS SDK for Java(버전 1)의 `CloudFrontUrlSigner` 유틸리티 클래스를 사용하여 CloudFront 서명 URL을 만들 수 있습니다.

자세한 예제는 AWS SDK Code Examples Code Library에서 AWS SDK를 사용하여 서명된 URL 및 쿠키 만들기를 참조합니다.

> **Note**
>
> 서명된 URL 생성은 CloudFront를 통해 프라이빗 콘텐츠를 제공하는 프로세스의 한 부분에 불과합니다. 전체 프로세스에 대한 자세한 내용은 서명된 URL 사용을 참조하세요.

다음 예제는 CloudFront 서명된 URL을 만드는 방법을 보여줍니다.

Example Java 정책 및 서명 암호화 메서드

```
package org.example;

import java.time.Instant;
```

```
import java.time.temporal.ChronoUnit;
import software.amazon.awssdk.services.cloudfront.CloudFrontUtilities;
import software.amazon.awssdk.services.cloudfront.model.CannedSignerRequest;
import software.amazon.awssdk.services.cloudfront.url.SignedUrl;

public class Main {

    public static void main(String[] args) throws Exception {
        CloudFrontUtilities cloudFrontUtilities = CloudFrontUtilities.create();
        Instant expirationDate = Instant.now().plus(7, ChronoUnit.DAYS);
        String resourceUrl = "https://a1b2c3d4e5f6g7.cloudfront.net";
        String keyPairId = "K1UA3WV15I7JSD";
        CannedSignerRequest cannedRequest = CannedSignerRequest.builder()
                .resourceUrl(resourceUrl)
                .privateKey(new java.io.File("/path/to/private_key.pem").toPath())
                .keyPairId(keyPairId)
                .expirationDate(expirationDate)
                .build();
        SignedUrl signedUrl =
 cloudFrontUtilities.getSignedUrlWithCannedPolicy(cannedRequest);
        String url = signedUrl.url();
        System.out.println(url);

    }
}
```

추가 참고:

- Perl을 사용한 URL 서명 생성
- PHP를 사용한 URL 서명 생성
- C# 및 .NET Framework를 사용한 URL 서명 생성

# AWS 오리진에 대한 액세스 제한

다음과 같은 이점을 제공하는 방식으로 CloudFront 및 일부 AWS 오리진을 구성할 수 있습니다.

- 공개적으로 액세스할 수 없도록 AWS 오리진에 대한 액세스를 제한합니다.
- 뷰어(사용자)가 지정된 CloudFront 배포를 통해서만 AWS 오리진의 콘텐츠에 액세스할 수 있는지 확인합니다. 이렇게 하면 뷰어가 오리진에서 직접 또는 의도하지 않은 CloudFront 배포를 통해 콘텐츠에 액세스할 수 없습니다.

이렇게 하려면 인증된 요청을 AWS 오리진으로 보내도록 CloudFront를 구성하고 CloudFront의 인증된 요청에 대한 액세스만 허용하도록 AWS 오리진을 구성합니다. 자세한 정보는 호환되는 AWS 오리진 유형에 대한 다음 주제를 참조합니다.

주제

- AWS Elemental MediaPackage v2 오리진에 대한 액세스 제한
- AWS Elemental MediaStore 오리진에 대한 액세스 제한
- AWS Lambda 함수 URL 오리진에 대한 액세스 제한
- Amazon S3 오리진에 대한 액세스 제한
- VPC 오리진으로 액세스 제한

## AWS Elemental MediaPackage v2 오리진에 대한 액세스 제한

CloudFront는 MediaPackage v2 오리진에 대한 액세스를 제한하기 위한 오리진 액세스 제어(OAC)를 제공합니다.

> **Note**
>
> CloudFront OAC는 MediaPackage v2만 지원합니다. MediaPackage v1은 지원되지 않습니다.

주제

- 새 OAC 생성
- 오리진 액세스 제어를 위한 고급 설정

### 새 OAC 생성

다음 주제에 설명된 단계를 완료하여 CloudFront에서 새 OAC를 설정합니다.

주제

- 사전 조건
- MediaPackage v2 오리진에 액세스할 수 있는 CloudFront 권한 부여
- OAC 생성

사전 조건

OAC를 생성하고 설정하기 전에 MediaPackage v2 오리진과 함께 CloudFront 배포가 있어야 합니다. 자세한 내용은 MediaStore 컨테이너 또는 MediaPackage 채널 사용 섹션을 참조하세요.

MediaPackage v2 오리진에 액세스할 수 있는 CloudFront 권한 부여

OAC를 만들거나 CloudFront 배포에서 설정하기 전에 CloudFront에 MediaPackage v2 오리진에 액세스할 수 있는 권한이 있는지 확인합니다. 이 작업은 CloudFront 배포를 생성한 후, 배포 구성에서 MediaPackage v2 오리진에 OAC를 추가하기 전에 수행합니다.

IAM 정책을 사용하여 CloudFront 서비스 위탁자(`cloudfront.amazonaws.com`)가 오리진에 액세스하도록 허용합니다. 이 정책의 `Condition` 요소는 요청이 MediaPackage v2 오리진을 포함하는 CloudFront 배포를 대신하는 경우에만 CloudFront가 MediaPackage v2 오리진에 액세스할 수 있도록 허용합니다. OAC를 추가하려는 MediaPackage v2 오리진이 있는 배포입니다.

Example: CloudFront 배포에서 OAC에 대한 읽기 전용 액세스를 허용하는 IAM 정책

다음 정책은 MediaPackage v2 오리진에 대한 CloudFront 배포(*E1PDK09ESKHJWT*) 액세스를 허용합니다. 오리진은 `Resource` 요소에 지정된 ARN입니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCloudFrontServicePrincipal",
            "Effect": "Allow",
            "Principal": {"Service": "cloudfront.amazonaws.com"},
            "Action": "mediapackagev2:GetObject",
            "Resource": "arn:aws:mediapackagev2:us-
east-1:123456789012:channelGroup/channel-group-name/channel/channel-name/
originEndpoint/origin_endpoint_name",
            "Condition": {
                "StringEquals": {"AWS:SourceArn":
 "arn:aws:cloudfront::123456789012:distribution/E1PDK09ESKHJWT"}
            }
        }
    ]
}
```

> **참고**
>
> - MQAR 기능 및 오리진 액세스 제어(OAC)를 사용 설정한 경우 IAM 정책에 `mediapackagev2:GetHeadObject` 작업을 추가합니다. MediaPackage v2 오리진으로 HEAD 요청을 보내려면 MQAR에 이 권한이 필요합니다. MQAR에 대한 자세한 내용은 미디어 품질 인식 복원력 섹션을 참조하시기 바랍니다.
> - MediaPackage v2 오리진에 대한 권한이 없는 배포를 생성하는 경우 CloudFront 콘솔에서 정책 복사를 선택한 다음 엔드포인트 권한 업데이트를 선택할 수 있습니다. 그런 다음 복사한 권한을 엔드포인트에 연결할 수 있습니다. 자세한 내용은 AWS Elemental MediaPackage 사용자 설명서의 엔드포인트 정책 필드를 참조합니다.

## OAC 생성

OAC를 생성하려면 AWS Management Console, CloudFormation, AWS CLI 또는 CloudFront API를 사용할 수 있습니다.

### Console

OAC를 만들려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Origin access(원본 액세스)를 선택합니다.
3. Create control setting(제어 설정 생성)을 선택합니다.
4. 새로운 OAC 생성 양식에서 다음을 수행합니다.
   a. OAC의 이름을 입력하고 선택적으로 설명을 입력합니다.
   b. 서명 동작의 경우 기본 설정인 요청 서명(권장)을 유지하는 것이 좋습니다. 자세한 내용은 the section called “오리진 액세스 제어를 위한 고급 설정” 섹션을 참조하세요.
5. 오리진 유형으로는 MediaPackage V2를 선택합니다.
6. 생성을 선택합니다.

> **Tip**
>
> OAC를 생성한 후 이름을 기록해 둡니다. 다음 절차에서 필요합니다.

배포의 MediaPackage v2 오리진에 OAC를 추가하려는 경우

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. OAC를 추가하려는 MediaPackage V2 오리진이 있는 배포를 선택한 다음 오리진 탭을 선택합니다.
3. OAC를 추가할 MediaPackage V2 오리진을 선택한 다음 편집을 선택합니다.
4. 오리진의 프로토콜에 HTTPS만 해당을 선택합니다.
5. 오리진 액세스 제어 드롭다운 메뉴에서 사용할 OAC를 선택합니다.
6. 변경 사항 저장를 선택합니다.

모든 CloudFront 엣지 로케이션에 배포가 시작됩니다. 엣지 로케이션은 새 구성을 수신하면 MediaPackage V2 오리진으로 보내는 모든 요청에 서명합니다.

CloudFormation

CloudFormation에 OAC를 생성하려면 `AWS::CloudFront::OriginAccessControl` 리소스 유형을 사용합니다. 다음 예는 OAC를 생성하기 위한 YAML 형식의 CloudFormation 템플릿 구문을 보여줍니다.

```
Type: AWS::CloudFront::OriginAccessControl
Properties:
  OriginAccessControlConfig:
      Description: An optional description for the origin access control
      Name: ExampleOAC
      OriginAccessControlOriginType: mediapackagev2
      SigningBehavior: always
      SigningProtocol: sigv4
```

자세한 내용은 AWS CloudFormation 사용 설명서의 AWS::CloudFront::OriginAccessControl을 참조하세요.

CLI

AWS Command Line Interface(AWS CLI)를 사용하여 오리진 액세스 제어를 생성하려면 aws cloudfront create-origin-access-control 명령을 사용합니다. 각 개별 파라미터를 명령줄 입력으로 지정하는 대신 입력 파일을 사용하여 명령의 입력 파라미터를 제공할 수 있습니다.

오리진 액세스 제어를 생성하려면(입력 파일과 CLI)

1. 다음 명령을 사용하여 이름이 `origin-access-control.yaml`인 파일을 생성합니다. 이 파일에는 create-origin-access-control 명령에 대한 모든 입력 파라미터가 들어 있습니다.

```
aws cloudfront create-origin-access-control --generate-cli-skeleton yaml-input >
 origin-access-control.yaml
```

2. 방금 생성한 `origin-access-control.yaml` 파일을 엽니다. 파일을 편집하여 OAC의 이름, 설명(선택 사항)을 추가하고 `SigningBehavior`를 `always`로 변경합니다. 그런 다음 파일을 저장합니다.

   다른 OAC 설정에 대한 자세한 내용은 the section called "오리진 액세스 제어를 위한 고급 설정" 섹션을 참조하세요.

3. 다음 명령으로 `origin-access-control.yaml` 파일의 입력 파라미터를 사용하여 오리진 액세스 제어를 만듭니다.

```
aws cloudfront create-origin-access-control --cli-input-yaml file://origin-
access-control.yaml
```

   명령 출력에 있는 `Id` 값을 기록해 둡니다. CloudFront 배포의 MediaPackage v2 오리진에 OAC를 추가하려면 필요합니다.

OAC를 기존 배포의 MediaPackage v2 오리진에 연결하는 방법(입력 파일과 CLI)

1. 다음 명령을 사용하여 OAC를 추가할 CloudFront 배포에 대한 배포 구성을 저장합니다. 배포에 MediaPackage v2 오리진이 있어야 합니다.

```
aws cloudfront get-distribution-config --id <CloudFront distribution ID> --
output yaml > dist-config.yaml
```

2. 방금 생성한 `dist-config.yaml`이라는 파일을 엽니다. 파일을 편집하여 다음과 같이 변경합니다.

   - `Origins` 객체에서 `OriginAccessControlId`라는 필드에 OAC의 ID를 추가합니다.
   - `OriginAccessIdentity`라는 필드에서 값을 제거합니다(있는 경우).

- `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

완료되면 파일을 저장합니다.

3. 오리진 액세스 제어를 사용하도록 배포를 업데이트하려면 다음 명령을 사용합니다.

```
aws cloudfront update-distribution --id <CloudFront distribution ID> --cli-input-yaml file://dist-config.yaml
```

모든 CloudFront 엣지 로케이션에 배포가 시작됩니다. 엣지 로케이션은 새 구성을 수신하면 MediaPackage v2 오리진으로 보내는 모든 요청에 서명합니다.

API

CloudFront API를 사용하여 OAC를 생성하려면 CreateOriginAccessControl을 사용합니다. 이 API 호출에서 지정하는 필드에 대한 자세한 내용은 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

OAC를 생성한 후 다음 API 호출 중 하나를 사용하여 배포의 MediaPackage v2 오리진에 연결할 수 있습니다.

- 기존 배포에 연결하려면 UpdateDistribution을 사용합니다.
- 새 배포에 연결하려면 CreateDistribution을 사용합니다.

이 두 API 호출에 대해 오리진 내부의 `OriginAccessControlId` 필드에 OAC ID를 제공합니다. 이러한 API 호출에서 지정하는 다른 필드에 대한 자세한 내용은 모든 배포 설정 참조와 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

## 오리진 액세스 제어를 위한 고급 설정

CloudFront OAC 기능에는 특정 사용 사례만을 위한 고급 설정이 포함되어 있습니다. 고급 설정이 특별히 필요한 경우가 아니면 권장 설정을 사용하세요.

OAC에는 서명 동작(콘솔) 또는 `SigningBehavior`(API, CLI 및 CloudFormation)라는 설정이 포함되어 있습니다. 이 설정은 다음 옵션을 제공합니다.

항상 오리진 요청에 서명(권장 설정)

이 설정의 이름은 콘솔에서 서명 요청(권장)이고 API, CLI, CloudFormation에서 `always`이며, 이 설정을 사용하는 것이 좋습니다. 이 설정을 사용하면 CloudFront가 항상 MediaPackage v2 오리진으로 보내는 모든 요청에 서명합니다.

오리진 요청 서명 안 함

이 설정의 이름은 콘솔에서 Do not sign requests(요청 서명 안 함)이고 API, CLI, CloudFormation에서 `never`입니다. 이 OAC를 사용하는 모든 배포의 모든 오리진에 대한 OAC를 끄려면 이 설정을 사용합니다. 오리진 액세스 제어를 사용하는 모든 오리진 및 배포에서 OAC를 하나씩 제거하는 것과 비교하여 시간과 노력을 절약할 수 있습니다. 이 설정을 사용하면 CloudFront가 MediaPackage v2 오리진으로 보내는 모든 요청에 서명하지 않습니다.

> **Warning**
>
> 이 설정을 사용하려면 MediaPackage v2 오리진에 공개적으로 액세스할 수 있어야 합니다. 공개적으로 액세스할 수 없는 MediaPackage v2 오리진에 이 설정을 사용하는 경우 CloudFront가 오리진에 액세스할 수 없습니다. MediaPackage v2 오리진은 CloudFront에 오류를 반환하고 CloudFront 해당 오류를 뷰어에 전달합니다. 자세한 내용은 AWS Elemental MediaPackage 사용 설명서의 MediaPackage의 정책 및 권한에 대한 MediaPackage v2 정책 예제를 참조하세요.

뷰어(클라이언트) **`Authorization`** 헤더 재정의 안 함

이 설정의 이름은 콘솔에서 Do not override authorization header(승인 헤더 재정의 안 함)이고 API, CLI, CloudFormation에서 `no-override`입니다. 해당 뷰어 요청에 `Authorization` 헤더가 포함되지 않은 경우에만 CloudFront에서 오리진 요청에 서명하도록 하려면 이 설정을 사용합니다. 이 설정을 사용하면 CloudFront에서는 뷰어 요청에 `Authorization` 헤더가 있는 경우 이를 전달하지만 뷰어 요청에 `Authorization` 헤더가 포함되어 있지 않으면 오리진 요청에 서명합니다(자체 `Authorization` 헤더 추가).

> **Warning**
>
> 뷰어 요청에서 `Authorization` 헤더를 전달하려면 이 오리진 액세스 제어와 연결된 MediaPackage v2 오리진을 사용하는 모든 캐시 동작에 대한 `Authorization` 헤더를 캐시 정책에 반드시 추가해야 합니다.

# AWS Elemental MediaStore 오리진에 대한 액세스 제한

CloudFront는 AWS Elemental MediaStore 오리진에 대한 액세스를 제한할 수 있도록 오리진 액세스 제어(OAC)를 제공합니다.

주제


- 새 오리진 액세스 제어 생성
- 오리진 액세스 제어를 위한 고급 설정


## 새 오리진 액세스 제어 생성

다음 주제에 설명된 단계를 완료하여 CloudFront에서 새 오리진 액세스 제어를 설정합니다.

주제


- 사전 조건
- CloudFront에 MediaStore 오리진에 액세스할 수 있는 권한 부여
- 오리진 액세스 제어 생성


### 사전 조건

오리진 액세스 제어(OAC)를 생성하고 설정하기 전에 MediaStore 오리진과 함께 CloudFront 배포가 있어야 합니다.

### CloudFront에 MediaStore 오리진에 액세스할 수 있는 권한 부여

오리진 액세스 제어를 만들거나 CloudFront 배포에서 이를 설정하기 전에 CloudFront에 MediaStore 오리진에 액세스할 수 있는 권한이 있는지 확인합니다. 이 작업은 CloudFront 배포를 생성한 후 배포 구성에서 MediaStore 오리진에 OAC를 추가하기 전에 수행해야 합니다.

MediaStore 컨테이너 정책을 사용하여 CloudFront 서비스 위탁자(`cloudfront.amazonaws.com`)가 오리진에 액세스하도록 허용합니다. 정책의 `Condition` 요소를 사용하여 MediaStore 오리진이 포함된 CloudFront 배포를 대신하는 요청인 경우에만 CloudFront에서 MediaStore 컨테이너에 액세스하도록 허용합니다. OAC를 추가하려는 MediaStore 오리진이 있는 배포입니다.

다음은 CloudFront 배포에서 MediaStore 오리진에 액세스할 수 있도록 허용하는 MediaStore 컨테이너 정책의 예제입니다.

Example: CloudFront 배포에서 OAC에 대한 읽기 전용 액세스를 허용하도록 사용 설정된 MediaStore 컨테이너 정책

JSON

```
{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Sid": "AllowCloudFrontServicePrincipalReadOnly",
                "Effect": "Allow",
                "Principal": {
                  "Service": "cloudfront.amazonaws.com"
                },
                "Action": [
                  "mediastore:GetObject"
                ],
                "Resource": "arn:aws:mediastore:us-
east-1:111122223333:container/<container name>/*",
                "Condition": {
                    "StringEquals": {
                      "AWS:SourceArn":
 "arn:aws:cloudfront::111122223333:distribution/CloudFront-distribution-ID"
                    },
                    "Bool": {
                      "aws:SecureTransport": "true"
                    }
                }
            }
        ]
}
```

Example: CloudFront 배포에서 OAC에 대한 읽기 및 쓰기 액세스를 허용하도록 사용 설정된 MediaStore 컨테이너 정책

JSON

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

```
            {
                "Sid": "AllowCloudFrontServicePrincipalReadWrite",
                "Effect": "Allow",
                "Principal": {
                  "Service": "cloudfront.amazonaws.com"
                },
                "Action": [
                  "mediastore:GetObject",
                  "mediastore:PutObject"
                ],
                "Resource": "arn:aws:mediastore:us-
east-1:111122223333:container/container-name/*",
                "Condition": {
                    "StringEquals": {
                      "AWS:SourceArn":
 "arn:aws:cloudfront::111122223333:distribution/CloudFront-distribution-ID"
                    },
                    "Bool": {
                      "aws:SecureTransport": "true"
                    }
                }
            }
        ]
}
```

> **Note**
>
> 쓰기 액세스를 허용하려면 CloudFront 배포의 동작 설정에 PUT을 포함하기 위해 Allowed HTTP methods(허용된 HTTP 메서드)를 구성해야 합니다.

## 오리진 액세스 제어 생성

OAC를 생성하려면 AWS Management Console, CloudFormation, AWS CLI 또는 CloudFront API를 사용할 수 있습니다.

Console

오리진 액세스 제어를 생성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Origin access(원본 액세스)를 선택합니다.
3. Create control setting(제어 설정 생성)을 선택합니다.
4. Create control setting(제어 설정 생성) 양식을 사용하여 다음을 수행합니다.
   a. Details(세부 정보) 창에서 오리진 액세스 제어의 Name(이름) 및 선택적 Description(설명)을 입력합니다.
   b. Settings(설정) 창에서 기본 설정인 Sign requests(recommended)(요청 서명(권장))를 유지하는 것이 좋습니다. 자세한 내용은 the section called "오리진 액세스 제어를 위한 고급 설정" 섹션을 참조하세요.
5. Origin type(오리진 유형) 드롭다운에서 MediaStore를 선택합니다.
6. 생성(Create)을 선택합니다.

   OAC를 생성한 후 Name(이름)을 기록해 둡니다. 다음 절차에서 필요합니다.

배포의 MediaStore 오리진에 오리진 액세스 제어를 추가하는 방법

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. OAC를 추가하려는 MediaStore 오리진이 있는 배포를 선택한 다음 Origins(오리진) 탭을 선택합니다.
3. OAC를 추가할 MediaStore 오리진을 선택한 다음 Edit(편집)을 선택합니다.
4. 오리진의 Protocol(프로토콜)에 HTTPS only(HTTPS만 해당)를 선택합니다.
5. Origin access control(원본 액세스 제어) 드롭다운 메뉴에서 사용할 OAC를 선택합니다.
6. 변경 사항 저장을 선택합니다.

모든 CloudFront 엣지 로케이션에 배포가 시작됩니다. 엣지 로케이션은 새 구성을 수신하면 MediaStore 버킷 오리진으로 보내는 모든 요청에 서명합니다.

CloudFormation

CloudFormation에 오리진 액세스 제어(OAC)를 생성하려면 `AWS::CloudFront::OriginAccessControl` 리소스 유형을 사용합니다. 다음 예는 오리진 액세스 제어를 생성하기 위한 YAML 형식의 CloudFormation 템플릿 구문을 보여줍니다.

```
Type: AWS::CloudFront::OriginAccessControl
Properties:
  OriginAccessControlConfig:
      Description: An optional description for the origin access control
      Name: ExampleOAC
      OriginAccessControlOriginType: mediastore
      SigningBehavior: always
      SigningProtocol: sigv4
```

자세한 내용은 AWS CloudFormation 사용 설명서의 AWS::CloudFront::OriginAccessControl을 참조하세요.

CLI

AWS Command Line Interface(AWS CLI)를 사용하여 오리진 액세스 제어를 생성하려면 aws cloudfront create-origin-access-control 명령을 사용합니다. 각 개별 파라미터를 명령줄 입력으로 지정하는 대신 입력 파일을 사용하여 명령의 입력 파라미터를 제공할 수 있습니다.

오리진 액세스 제어를 생성하려면(입력 파일과 CLI)

1. 다음 명령을 사용하여 이름이 `origin-access-control.yaml`인 파일을 생성합니다. 이 파일에는 create-origin-access-control 명령에 대한 모든 입력 파라미터가 들어 있습니다.

   ```
   aws cloudfront create-origin-access-control --generate-cli-skeleton yaml-input >
    origin-access-control.yaml
   ```

2. 방금 생성한 `origin-access-control.yaml` 파일을 엽니다. 파일을 편집하여 OAC의 이름, 설명(선택 사항)을 추가하고 `SigningBehavior`를 `always`로 변경합니다. 그런 다음 파일을 저장합니다.

   다른 OAC 설정에 대한 자세한 내용은 the section called “오리진 액세스 제어를 위한 고급 설정” 섹션을 참조하세요.

3. 다음 명령으로 `origin-access-control.yaml` 파일의 입력 파라미터를 사용하여 오리진 액세스 제어를 만듭니다.

```
aws cloudfront create-origin-access-control --cli-input-yaml file://origin-
access-control.yaml
```

명령 출력에 있는 `Id` 값을 기록해 둡니다. CloudFront 배포의 MediaStore 오리진에 OAC를 추가하려면 필요합니다.

OAC를 기존 배포의 MediaStore 오리진에 연결하는 방법(입력 파일과 CLI)

1. 다음 명령을 사용하여 OAC를 추가할 CloudFront 배포에 대한 배포 구성을 저장합니다. 배포에 MediaStore 오리진이 있어야 합니다.

```
aws cloudfront get-distribution-config --id <CloudFront distribution ID> --
output yaml > dist-config.yaml
```

2. 방금 생성한 `dist-config.yaml`이라는 파일을 엽니다. 파일을 편집하여 다음과 같이 변경합니다.

   - `Origins` 객체에서 `OriginAccessControlId`라는 필드에 OAC의 ID를 추가합니다.
   - `OriginAccessIdentity`라는 필드에서 값을 제거합니다(있는 경우).
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

   완료되면 파일을 저장합니다.

3. 오리진 액세스 제어를 사용하도록 배포를 업데이트하려면 다음 명령을 사용합니다.

```
aws cloudfront update-distribution --id <CloudFront distribution ID> --cli-
input-yaml file://dist-config.yaml
```

모든 CloudFront 엣지 로케이션에 배포가 시작됩니다. 엣지 로케이션은 새 구성을 수신하면 MediaStore 오리진으로 보내는 모든 요청에 서명합니다.

API

CloudFront API를 사용하여 오리진 액세스 제어를 생성하려면 CreateOriginAccessControl을 사용합니다. 이 API 호출에서 지정하는 필드에 대한 자세한 내용은 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

오리진 액세스 제어를 생성한 후 다음 API 호출 중 하나를 사용하여 배포의 MediaStore 오리진에 연결할 수 있습니다.

- 기존 배포에 연결하려면 UpdateDistribution을 사용합니다.
- 새 배포에 연결하려면 CreateDistribution을 사용합니다.

이 두 API 호출에 대해 오리진 내부의 `OriginAccessControlId` 필드에 오리진 액세스 제어 ID를 제공합니다. 이러한 API 호출에서 지정하는 다른 필드에 대한 자세한 내용은 모든 배포 설정 참조와 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

## 오리진 액세스 제어를 위한 고급 설정

CloudFront 오리진 액세스 제어 기능에는 특정 사용 사례만을 위한 고급 설정이 포함되어 있습니다. 고급 설정이 특별히 필요한 경우가 아니면 권장 설정을 사용하세요.

오리진 액세스 제어에는 Signing behavior(서명 동작)(콘솔) 또는 `SigningBehavior`(API, CLI 및 CloudFormation)라는 설정이 포함되어 있습니다. 이 설정은 다음 옵션을 제공합니다.

항상 오리진 요청에 서명(권장 설정)

이 설정의 이름은 콘솔에서 서명 요청(권장)이고 API, CLI, CloudFormation에서 `always`이며, 이 설정을 사용하는 것이 좋습니다. 이 설정을 사용하면 CloudFront가 항상 MediaStore 오리진으로 보내는 모든 요청에 서명합니다.

오리진 요청 서명 안 함

이 설정의 이름은 콘솔에서 Do not sign requests(요청 서명 안 함)이고 API, CLI, CloudFormation에서 `never`입니다. 이 오리진 액세스 제어를 사용하는 모든 배포의 모든 오리진에 대한 오리진 액세스 제어를 끄려면 이 설정을 사용합니다. 오리진 액세스 제어를 사용하는 모든 오리진 및 배포에서 오리진 액세스 제어를 하나씩 제거하는 것과 비교하여 시간과 노력을 절약할 수 있습니다. 이 설정을 사용하면 CloudFront가 MediaStore 오리진으로 보내는 모든 요청에 서명하지 않습니다.

> **Warning**
>
> 이 설정을 사용하려면 MediaStore 오리진에 공개적으로 액세스할 수 있어야 합니다. 공개적으로 액세스할 수 없는 MediaStore 오리진에 이 설정을 사용하는 경우 CloudFront가 오리진에 액세스할 수 없습니다. MediaStore 오리진은 CloudFront에 오류를 반환하고 CloudFront 해당 오류를 뷰어에 전달합니다. 자세한 내용은 Public read access over HTTPS(HTTPS를 통한 공개 읽기 액세스)에 대한 MediaStore 컨테이너 정책 예시를 참조하세요.

뷰어(클라이언트) **`Authorization`** 헤더 재정의 안 함

이 설정의 이름은 콘솔에서 Do not override authorization header(승인 헤더 재정의 안 함)이고 API, CLI, CloudFormation에서 `no-override`입니다. 해당 뷰어 요청에 `Authorization` 헤더가 포함되지 않은 경우에만 CloudFront에서 오리진 요청에 서명하도록 하려면 이 설정을 사용합니다. 이 설정을 사용하면 CloudFront에서는 뷰어 요청에 `Authorization` 헤더가 있는 경우 이를 전달하지만 뷰어 요청에 `Authorization` 헤더가 포함되어 있지 않으면 오리진 요청에 서명합니다(자체 `Authorization` 헤더 추가).

> **Warning**
>
> 뷰어 요청에서 `Authorization` 헤더를 전달하려면 이 오리진 액세스 제어와 연결된 MediaStore 오리진을 사용하는 모든 캐시 동작에 대한 `Authorization` 헤더를 캐시 정책에 반드시 추가해야 합니다.

## AWS Lambda 함수 URL 오리진에 대한 액세스 제한

CloudFront는 Lambda 함수 URL 오리진에 대한 액세스를 제한하기 위한 오리진 액세스 제어(OAC)를 제공합니다.

주제

- 새 OAC 생성
- 오리진 액세스 제어를 위한 고급 설정
- 예제 템플릿 코드

## 새 OAC 생성

다음 주제에 설명된 단계를 완료하여 CloudFront에서 새 OAC를 설정합니다.

> **Important**
>
> Lambda 함수 URL과 함께 PUT 또는 POST 메서드를 사용하는 경우 사용자가 요청을 CloudFront에 전송할 때 본문의 SHA256을 컴퓨팅하고 `x-amz-content-sha256` 헤더에 있는 요청 본문의 페이로드 해시 값을 포함해야 합니다. Lambda는 서명되지 않은 페이로드를 지원하지 않습니다.

주제

- 사전 조건
- Lambda 함수 URL에 액세스할 수 있는 CloudFront 권한 부여
- OAC 생성

### 사전 조건

OAC를 생성하고 설정하기 전에 Lambda 함수 URL을 오리진으로 사용하는 CloudFront 배포가 있어야 합니다. OAC를 사용하려면 `AWS_IAM`을 `AuthType` 파라미터의 값으로 지정해야 합니다. 자세한 내용은 Lambda 함수 URL 사용 섹션을 참조하세요.

### Lambda 함수 URL에 액세스할 수 있는 CloudFront 권한 부여

OAC를 만들거나 CloudFront 배포에서 설정하기 전에 CloudFront에 Lambda 함수 URL에 액세스할 수 있는 권한이 있는지 확인합니다. 이 작업은 CloudFront 배포를 생성한 후 배포 구성의 Lambda 함수 URL에 OAC를 추가하기 전에 수행해야 합니다.

> **Note**
>
> Lambda 함수 URL에 대한 IAM 정책을 업데이트하려면 AWS Command Line Interface(AWS CLI)을 사용해야 합니다. Lambda 콘솔에서의 IAM 정책 편집은 현재 지원되지 않습니다.

다음 AWS CLI 명령은 Lambda 함수 URL에 대한 CloudFront 서비스 보안 주체 (`cloudfront.amazonaws.com`) 액세스 권한을 부여합니다. 이 정책의 `Condition` 요소는 요청이

Lambda 함수 URL을 포함하는 CloudFront 배포를 대신하는 경우에만 CloudFront가 Lambda에 액세스할 수 있도록 허용합니다. OAC를 추가하려는 Lambda 함수 URL 오리진이 있는 배포입니다.

Example: CloudFront 배포에서 OAC에 대한 읽기 전용 액세스를 허용하도록 정책을 업데이트하는 AWS CLI 명령입니다.

다음 AWS CLI 명령을 실행하면 CloudFront 배포(*E1PDK09ESKHJWT*)가 Lambda *FUNCTION_URL_NAME*에 액세스할 수 있습니다.

```
aws lambda add-permission \
--statement-id "AllowCloudFrontServicePrincipal" \
--action "lambda:InvokeFunctionUrl" \
--principal "cloudfront.amazonaws.com" \
--source-arn "arn:aws:cloudfront::123456789012:distribution/E1PDK09ESKHJWT" \
--function-name FUNCTION_URL_NAME
```

```
aws lambda add-permission \
--statement-id "AllowCloudFrontServicePrincipalInvokeFunction" \
--action "lambda:InvokeFunction" \
--principal "cloudfront.amazonaws.com" \
--source-arn "arn:aws:cloudfront::123456789012:distribution/E1PDK09ESKHJWT" \
--function-name FUNCTION_URL_NAME
```

> **Note**
>
> 배포를 생성했는데 Lambda 함수 URL에 대한 권한이 없는 경우 CloudFront 콘솔에서 Copy CLI 명령을 선택한 다음 명령줄 터미널에서 이 명령을 입력할 수 있습니다. 자세한 정보는 AWS Lambda 개발자 안내서의 AWS 서비스에 대한 액세스 부여를 참조합니다.

## OAC 생성

OAC를 생성하려면 AWS Management Console, CloudFormation, AWS CLI 또는 CloudFront API를 사용할 수 있습니다.

### Console

OAC를 만들려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.

2. 탐색 창에서 Origin access(원본 액세스)를 선택합니다.
3. Create control setting(제어 설정 생성)을 선택합니다.
4. 새로운 OAC 생성 양식에서 다음을 수행합니다.

   a. OAC의 이름을 입력하고 선택적으로 설명을 입력합니다.

   b. 서명 동작의 경우 기본 설정인 요청 서명(권장)을 유지하는 것이 좋습니다. 자세한 내용은 the section called "오리진 액세스 제어를 위한 고급 설정" 섹션을 참조하세요.
5. 오리진 유형에 대해 Lambda를 선택합니다.
6. 생성(Create)을 선택합니다.

> **Tip**
>
> OAC를 생성한 후 이름을 기록해 둡니다. 다음 절차에서 필요합니다.

배포의 Lambda 함수 URL에 오리진 액세스 제어를 추가하려는 경우

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다
2. OAC를 추가하려는 Lambda 함수 URL이 있는 배포를 선택한 다음 오리진 탭을 선택합니다.
3. OAC를 추가할 Lambda 함수 URL을 선택한 다음 편집을 선택합니다.
4. 오리진의 프로토콜에 HTTPS만 해당을 선택합니다.
5. 오리진 액세스 제어 드롭다운 메뉴에서 사용할 OAC를 선택합니다.
6. 변경 사항 저장를 선택합니다.

모든 CloudFront 엣지 로케이션에 배포가 시작됩니다. 엣지 로케이션은 새 구성을 수신하면 Lambda 함수 URL로 보내는 모든 요청에 서명합니다.

CloudFormation

CloudFormation에 OAC를 생성하려면 `AWS::CloudFront::OriginAccessControl` 리소스 유형을 사용합니다. 다음 예는 OAC를 생성하기 위한 YAML 형식의 CloudFormation 템플릿 구문을 보여줍니다.

```
Type: AWS::CloudFront::OriginAccessControl
Properties:
  OriginAccessControlConfig:
      Description: An optional description for the origin access control
```

```
    Name: ExampleOAC
    OriginAccessControlOriginType: lambda
    SigningBehavior: always
    SigningProtocol: sigv4
```

자세한 내용은 AWS CloudFormation 사용 설명서의 AWS::CloudFront::OriginAccessControl을 참조하세요.

CLI

AWS Command Line Interface(AWS CLI)를 사용하여 오리진 액세스 제어를 생성하려면 aws cloudfront create-origin-access-control 명령을 사용합니다. 각 개별 파라미터를 명령줄 입력으로 지정하는 대신 입력 파일을 사용하여 명령의 입력 파라미터를 제공할 수 있습니다.

오리진 액세스 제어를 생성하려면(입력 파일과 CLI)

1. 다음 명령을 사용하여 이름이 `origin-access-control.yaml`인 파일을 생성합니다. 이 파일에는 create-origin-access-control 명령에 대한 모든 입력 파라미터가 들어 있습니다.

```
aws cloudfront create-origin-access-control --generate-cli-skeleton yaml-input >
 origin-access-control.yaml
```

2. 방금 생성한 `origin-access-control.yaml` 파일을 엽니다. 파일을 편집하여 OAC의 이름, 설명(선택 사항)을 추가하고 `SigningBehavior`를 `always`로 변경합니다. 그런 다음 파일을 저장합니다.

   다른 OAC 설정에 대한 자세한 내용은 the section called “오리진 액세스 제어를 위한 고급 설정” 섹션을 참조하세요.

3. 다음 명령으로 `origin-access-control.yaml` 파일의 입력 파라미터를 사용하여 오리진 액세스 제어를 만듭니다.

```
aws cloudfront create-origin-access-control --cli-input-yaml file://origin-
access-control.yaml
```

   명령 출력에 있는 `Id` 값을 기록해 둡니다. CloudFront 배포의 Lambda 함수 URL에 OAC를 추가하려면 필요합니다.

OAC를 기존 배포의 Lambda 함수 URL에 연결하려는 경우(입력 파일과 CLI)

1. 다음 명령을 사용하여 OAC를 추가할 CloudFront 배포에 대한 배포 구성을 저장합니다. 배포에 오리진과 함께 Lambda 함수 URL이 있어야 합니다.

```
aws cloudfront get-distribution-config --id <CloudFront distribution ID> --
output yaml > dist-config.yaml
```

2. 방금 생성한 `dist-config.yaml`이라는 파일을 엽니다. 파일을 편집하여 다음과 같이 변경합니다.

   - `Origins` 객체에서 `OriginAccessControlId`라는 필드에 OAC의 ID를 추가합니다.
   - `OriginAccessIdentity`라는 필드에서 값을 제거합니다(있는 경우).
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

   완료되면 파일을 저장합니다.

3. 오리진 액세스 제어를 사용하도록 배포를 업데이트하려면 다음 명령을 사용합니다.

```
aws cloudfront update-distribution --id <CloudFront distribution ID> --cli-
input-yaml file://dist-config.yaml
```

모든 CloudFront 엣지 로케이션에 배포가 시작됩니다. 엣지 로케이션은 새 구성을 수신하면 Lambda 함수 URL로 보내는 모든 요청에 서명합니다.

API

CloudFront API를 사용하여 OAC를 생성하려면 CreateOriginAccessControl을 사용합니다. 이 API 호출에서 지정하는 필드에 대한 자세한 내용은 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

OAC를 생성한 후 다음 API 호출 중 하나를 사용하여 배포의 Lambda 함수 URL에 연결할 수 있습니다.

- 기존 배포에 연결하려면 UpdateDistribution을 사용합니다.
- 새 배포에 연결하려면 CreateDistribution을 사용합니다.

이 두 API 호출에 대해 오리진 내부의 `OriginAccessControlId` 필드에 OAC ID를 제공합니다. 이러한 API 호출에서 지정하는 다른 필드에 대한 자세한 내용은 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조합니다.

## 오리진 액세스 제어를 위한 고급 설정

CloudFront OAC 기능에는 특정 사용 사례만을 위한 고급 설정이 포함되어 있습니다. 고급 설정이 특별히 필요한 경우가 아니면 권장 설정을 사용하세요.

OAC에는 서명 동작(콘솔) 또는 `SigningBehavior`(API, CLI 및 CloudFormation)라는 설정이 포함되어 있습니다. 이 설정은 다음 옵션을 제공합니다.

항상 오리진 요청에 서명(권장 설정)

이 설정의 이름은 콘솔에서 서명 요청(권장)이고 API, CLI, CloudFormation에서 `always`이며, 이 설정을 사용하는 것이 좋습니다. 이 설정을 사용하면 CloudFront가 항상 Lambda 함수 URL으로 보내는 모든 요청에 서명합니다.

오리진 요청 서명 안 함

이 설정의 이름은 콘솔에서 Do not sign requests(요청 서명 안 함)이고 API, CLI, CloudFormation에서 `never`입니다. 이 OAC를 사용하는 모든 배포의 모든 오리진에 대한 OAC를 끄려면 이 설정을 사용합니다. OAC를 사용하는 모든 오리진 및 배포에서 OAC를 하나씩 제거하는 것과 비교하여 시간과 노력을 절약할 수 있습니다. 이 설정을 사용하면 CloudFront가 Lambda 함수 URL로 보내는 모든 요청에 서명하지 않습니다.

> **Warning**
>
> 이 설정을 사용하려면 Lambda 함수 URL이 공개적으로 액세스할 수 있어야 합니다. 공개적으로 액세스할 수 없는 Lambda 함수 URL에 이 설정을 사용하는 경우 CloudFront가 오리진에 액세스할 수 없습니다. Lambda 함수 URL은 CloudFront에 오류를 반환하고 CloudFront 해당 오류를 뷰어에 전달합니다. 자세한 내용은 AWS Lambda 사용 설명서의 Lambda 함수 URL에 대한 보안 및 인증 모델을 참조하세요.

뷰어(클라이언트) **`Authorization`** 헤더 재정의 안 함

이 설정의 이름은 콘솔에서 Do not override authorization header(승인 헤더 재정의 안 함)이고 API, CLI, CloudFormation에서 `no-override`입니다. 해당 뷰어 요청에 `Authorization` 헤더가 포함

되지 않은 경우에만 CloudFront에서 오리진 요청에 서명하도록 하려면 이 설정을 사용합니다. 이 설정을 사용하면 CloudFront에서는 뷰어 요청에 `Authorization` 헤더가 있는 경우 이를 전달하지만 뷰어 요청에 `Authorization` 헤더가 포함되어 있지 않으면 오리진 요청에 서명합니다(자체 `Authorization` 헤더 추가).

> **Warning**
>
> - 이 설정을 사용하는 경우, Lambda 함수 URL에 대해 CloudFront 배포의 이름 또는 CNAME 대신 Signature Version 4 서명을 지정해야 합니다. CloudFront가 뷰어 요청의 `Authorization` 헤더를 Lambda 함수 URL로 전달하면 Lambda는 Lambda URL 도메인의 호스트에 대해 서명을 검증합니다. 서명이 Lambda URL 도메인을 기반으로 하지 않는 경우 서명의 호스트는 Lambda URL 오리진에서 사용하는 호스트와 일치하지 않습니다. 따라서 요청이 실패하여 서명 검증 오류가 발생합니다.
> - 뷰어 요청에서 `Authorization` 헤더를 전달하려면 이 오리진 액세스 제어와 연결된 Lambda 함수 URL을 사용하는 모든 캐시 동작에 대한 `Authorization` 헤더를 캐시 정책에 반드시 추가해야 합니다.

## 예제 템플릿 코드

CloudFront 오리진이 OAC와 연결된 Lambda 함수 URL인 경우 다음 Python 스크립트를 사용하여 `POST` 메서드로 Lambda 함수에 파일을 업로드할 수 있습니다.

이 코드는 기본 서명 동작이 오리진 요청에 항상 서명으로 설정된 OAC를 구성했고 승인 헤더 재정의 안 함 설정을 선택하지 않았다고 가정합니다.

이 구성을 통해 OAC는 Lambda 호스트 이름을 사용하여 Lambda로 SigV4 인증을 올바르게 관리할 수 있습니다. 페이로드는 `IAM_AUTH` 유형으로 지정된 Lambda 함수 URL에 대해 승인된 IAM 자격 증명의 SigV4를 사용하여 서명됩니다.

템플릿은 클라이언트 측의 `POST` 요청에 대해 x-amz-content-sha256 헤더에서 서명된 페이로드 해시 값을 처리하는 방법을 보여줍니다. 특히 이 템플릿은 양식 데이터 페이로드를 관리하도록 설계되었습니다. 템플릿은 CloudFront를 통해 Lambda 함수 URL에 대한 보안 파일 업로드를 활성화하고 AWS 인증 메커니즘을 사용하여 승인된 요청만 Lambda 함수에 액세스할 수 있도록 합니다.

> 코드에 다음 기능이 포함됩니다.
>
> - x-amz-content-sha256 헤더에 페이로드 해시를 포함하기 위한 요구 사항을 충족합니다.

- 보안 AWS 서비스 액세스를 위해 SigV4 인증 사용
- 멀티파트 양식 데이터를 사용하여 파일 업로드 지원
- 요청 예외에 대한 오류 처리 포함

```
import boto3
from botocore.auth import SigV4Auth
from botocore.awsrequest import AWSRequest
import requests
import hashlib
import os


def calculate_body_hash(body):
    return hashlib.sha256(body).hexdigest()


def sign_request(request, credentials, region, service):
    sigv4 = SigV4Auth(credentials, service, region)
    sigv4.add_auth(request)


def upload_file_to_lambda(cloudfront_url, file_path, region):
    # AWS credentials
    session = boto3.Session()
    credentials = session.get_credentials()

    # Prepare the multipart form-data
    boundary = "------------------------boundary"

    # Read file content
    with open(file_path, 'rb') as file:
        file_content = file.read()

    # Get the filename from the path
    filename = os.path.basename(file_path)

    # Prepare the multipart body
    body = (
        f'--{boundary}\r\n'
        f'Content-Disposition: form-data; name="file"; filename="{filename}"\r\n'
        f'Content-Type: application/octet-stream\r\n\r\n'
```

```
    ).encode('utf-8')
    body += file_content
    body += f'\r\n--{boundary}--\r\n'.encode('utf-8')

    # Calculate SHA256 hash of the entire body
    body_hash = calculate_body_hash(body)

    # Prepare headers
    headers = {
        'Content-Type': f'multipart/form-data; boundary={boundary}',
        'x-amz-content-sha256': body_hash
    }

    # Create the request
    request = AWSRequest(
        method='POST',
        url=cloudfront_url,
        data=body,
        headers=headers
    )

    # Sign the request
    sign_request(request, credentials, region, 'lambda')

    # Get the signed headers
    signed_headers = dict(request.headers)

    # Print request headers before sending
    print("Request Headers:")
    for header, value in signed_headers.items():
        print(f"{header}: {value}")

    try:
        # Send POST request with signed headers
        response = requests.post(
            cloudfront_url,
            data=body,
            headers=signed_headers
        )

        # Print response status and content
        print(f"\nStatus code: {response.status_code}")
        print("Response:", response.text)
```

```
        # Print response headers
        print("\nResponse Headers:")
        for header, value in response.headers.items():
            print(f"{header}: {value}")

    except requests.exceptions.RequestException as e:
        print(f"An error occurred: {e}")


# Usage
cloudfront_url = "https://d111111abcdef8.cloudfront.net"
file_path = r"filepath"
region = "us-east-1"  # example: "us-west-2"

upload_file_to_lambda(cloudfront_url, file_path, region)
```

## Amazon S3 오리진에 대한 액세스 제한

CloudFront는 Amazon S3 오리진에 인증된 요청을 전송하는 두 가지 방법으로 오리진 액세스 제어(OAC)와 오리진 액세스 ID(OAI)를 제공합니다. OAC는 Amazon S3와 같은 오리진을 보호하는 데 도움이 됩니다.

OAC는 다음 기능을 지원하므로 OAC를 대신 사용할 것을 권장합니다.

- 2022년 12월 이후에 출시된 옵트인 리전을 포함하여 모든 AWS 리전 리전의 모든 Amazon S3 버킷
- Amazon S3 AWS KMS를 사용한 서버 측 암호화(SSE-KMS)
- Amazon S3에 대한 동적 요청(PUT 및 DELETE)

OAI는 이러한 기능을 지원하지 않거나 이러한 시나리오에서 추가 해결 방법을 필요로 합니다. 이미 OAI를 사용하고 있고 마이그레이션하려는 경우 the section called “오리진 액세스 ID(OAI)에서 오리진 액세스 제어(OAC)로 마이그레이션” 섹션을 참조하세요.

> **참고**
>
> - Amazon S3 버킷 오리진과 함께 CloudFront OAC를 사용하는 경우 Amazon S3 객체 소유권을 새 Amazon S3 버킷의 기본값인 버킷 소유자 적용으로 설정해야 합니다. ACL이 필요한 경우 버킷 소유자 선호 설정을 사용하여 CloudFront를 통해 업로드된 객체에 대한 제어를 유지합니다.

- 오리진이 웹 사이트 엔드포인트로 구성된 Amazon S3 버킷인 경우 사용자 지정 오리진으로 CloudFront와 함께 버킷을 설정해야 합니다. 즉, OAC(또는 OAI)를 사용할 수 없습니다. OAC는 Lambda@Edge를 사용한 오리진 리디렉션을 지원하지 않습니다.

다음 주제에서는 Amazon S3 오리진에서 OAC를 사용하는 방법을 설명합니다.

주제


- the section called “새 오리진 액세스 제어 생성”
- the section called “S3 버킷에 OAC가 연결된 배포 삭제”
- the section called “오리진 액세스 ID(OAI)에서 오리진 액세스 제어(OAC)로 마이그레이션”
- the section called “오리진 액세스 제어를 위한 고급 설정”


## 새 오리진 액세스 제어 생성

다음 주제에 설명된 단계를 완료하여 CloudFront에서 새 오리진 액세스 제어를 설정합니다.

주제


- 사전 조건
- CloudFront에 S3 버킷에 액세스할 수 있는 권한 부여
- 오리진 액세스 제어 생성


### 사전 조건

오리진 액세스 제어(OAC)를 생성하고 설정하기 전에 Amazon S3 버킷 오리진과 함께 CloudFront 배포가 있어야 합니다. 이 오리진은 웹 사이트 엔드포인트로 구성된 버킷이 아니라 일반 S3 버킷이어야 합니다. S3 버킷 오리진과 함께 CloudFront 배포를 설정하는 방법에 대한 자세한 내용은 the section called “표준 배포 시작” 섹션을 참조하세요.

> **Important**
>
> OAC를 사용하여 Amazon S3 오리진을 보호하는 경우 CloudFront와 Amazon S3 간의 통신은 항상 HTTPS를 통해 이루어지지만 항상 요청에 서명하도록 선택한 경우에만 그렇습니다. 콘솔에서 서명 요청 (권장)을 선택하거나 CloudFront API, AWS CLI 또는 CloudFormation에서 `always`를 지정해야 합니다.

대신 요청 서명 안 함 또는 승인 헤더 재정의 안 함 옵션을 선택하면 CloudFront는 다음 정책에서 지정한 연결 프로토콜을 사용합니다.

- 뷰어 프로토콜 정책
- 오리진 프로토콜 정책(사용자 지정 오리진 전용)

예를 들어 승인 헤더 재정의 안 함을 선택하고 CloudFront와 Amazon S3 오리진 간에 HTTPS를 사용하려면 뷰어 사용자 프로토콜 정책에서 HTTP를 HTTPS로 리디렉션 또는 HTTPS 전용을 사용합니다.

CloudFront에 S3 버킷에 액세스할 수 있는 권한 부여

오리진 액세스 제어(OAC)를 만들거나 CloudFront 배포에서 이를 설정하기 전에 CloudFront에 S3 버킷 오리진에 액세스할 수 있는 권한이 있는지 확인합니다. 이 작업은 CloudFront 배포를 생성한 후 배포 구성에서 S3 오리진에 OAC를 추가하기 전에 수행해야 합니다.

S3 버킷 정책을 사용하여 CloudFront 서비스 위탁자(`cloudfront.amazonaws.com`)가 버킷에 액세스하도록 허용합니다. 정책의 `Condition` 요소를 사용하여 S3 오리진이 포함된 CloudFront 배포를 위한 요청인 경우에만 CloudFront에서 버킷에 액세스하도록 허용합니다. OAC를 추가하려는 S3 오리진이 있는 배포입니다.

버킷 정책 추가 또는 수정에 대한 자세한 내용은 Amazon S3 사용 설명서의 Amazon S3 콘솔을 사용하여 버킷 정책 추가를 참조합니다.

다음은 CloudFront 배포에서 OAC가 S3 오리진에 액세스할 수 있도록 허용하는 S3 버킷 정책의 예제입니다.

Example CloudFront 배포에서 OAC에 대한 읽기 전용 액세스를 허용하도록 사용 설정된 S3 버킷 정책

JSON

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

```
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceArn":
 "arn:aws:cloudfront::111122223333:distribution/<CloudFront distribution ID>"
        }
      }
    }
  ]
}
```

Example CloudFront 배포에서 OAC에 대한 읽기 및 쓰기 액세스를 허용하도록 사용 설정된 S3 버킷 정책

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudFrontServicePrincipalReadWrite",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceArn":
 "arn:aws:cloudfront::111122223333:distribution/CloudFront-distribution-ID>"
        }
      }
    }
  ]
}
```

### SSE-KMS

S3 버킷 오리진의 객체가 AWS Key Management Service(SSE-KMS)로 서버 측 암호화를 사용하여 암호화된 경우 CloudFront 배포에 AWS KMS 키를 사용할 권한이 있는지 확인해야 합니다. CloudFront 배포에 KMS 키를 사용할 수 있는 권한을 부여하려면 KMS 키 정책에 스테이트먼트를 추가합니다. 키 정책을 수정하는 방법에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 키 정책 변경을 참조하세요.

#### Example KMS 키 정책 설명

다음 예제는 OAC가 있는 CloudFront 배포에서 SSE-KMS용 KMS 키에 액세스할 수 있도록 허용하는 AWS KMS 정책 스테이트먼트를 보여줍니다.

```
{
    "Sid": "AllowCloudFrontServicePrincipalSSE-KMS",
    "Effect": "Allow",
    "Principal": {
        "Service": [
            "cloudfront.amazonaws.com"
        ]
     },
    "Action": [
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:GenerateDataKey*"
    ],
    "Resource": "*",
    "Condition": {
            "StringEquals": {
                "AWS:SourceArn":
 "arn:aws:cloudfront::111122223333:distribution/<CloudFront distribution ID>"
            }
        }
}
```

### 오리진 액세스 제어 생성

오리진 액세스 제어(OAC)를 생성하려면 AWS Management Console, CloudFormation, AWS CLI 또는 CloudFront API를 사용할 수 있습니다.

Console

오리진 액세스 제어를 생성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Origin access(원본 액세스)를 선택합니다.
3. Create control setting(제어 설정 생성)을 선택합니다.
4. Create control setting(제어 설정 생성) 양식을 사용하여 다음을 수행합니다.
   a. Details(세부 정보) 창에서 오리진 액세스 제어의 Name(이름) 및 선택적 Description(설명)을 입력합니다.
   b. Settings(설정) 창에서 기본 설정인 Sign requests(recommended)(요청 서명(권장))를 유지하는 것이 좋습니다. 자세한 내용은 the section called "오리진 액세스 제어를 위한 고급 설정" 섹션을 참조하세요.
5. Origin type(오리진 유형) 드롭다운에서 S3를 선택합니다.
6. 생성(Create)을 선택합니다.

   OAC를 생성한 후 Name(이름)을 기록해 둡니다. 다음 절차에서 필요합니다.

배포의 S3 오리진에 오리진 액세스 제어를 추가하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. OAC를 추가하려는 S3 오리진이 있는 배포를 선택한 다음 Origins(원본) 탭을 선택합니다.
3. OAC를 추가할 S3 오리진을 선택한 다음 Edit(편집)을 선택합니다.
4. 원본 액세스의 경우 원본 액세스 제어 설정(권장)를 선택합니다.
5. Origin access control(원본 액세스 제어) 드롭다운 메뉴에서 사용할 OAC를 선택합니다.
6. 변경 사항 저장을 선택합니다.

모든 CloudFront 엣지 로케이션에 배포가 시작됩니다. 엣지 로케이션은 새 구성을 수신하면 S3 버킷 오리진으로 보내는 모든 요청에 서명합니다.

CloudFormation

CloudFormation에 오리진 액세스 제어(OAC)를 생성하려면
`AWS::CloudFront::OriginAccessControl` 리소스 유형을 사용합니다. 다음 예는 오리진 액세스 제어를 생성하기 위한 YAML 형식의 CloudFormation 템플릿 구문을 보여줍니다.

```
Type: AWS::CloudFront::OriginAccessControl
Properties:
  OriginAccessControlConfig:
      Description: An optional description for the origin access control
      Name: ExampleOAC
      OriginAccessControlOriginType: s3
      SigningBehavior: always
      SigningProtocol: sigv4
```

자세한 내용은 AWS CloudFormation 사용 설명서의 AWS::CloudFront::OriginAccessControl을 참조하세요.

CLI

AWS Command Line Interface(AWS CLI)를 사용하여 오리진 액세스 제어를 생성하려면 aws cloudfront create-origin-access-control 명령을 사용합니다. 각 개별 파라미터를 명령줄 입력으로 지정하는 대신 입력 파일을 사용하여 명령의 입력 파라미터를 제공할 수 있습니다.

오리진 액세스 제어를 생성하려면(입력 파일과 CLI)

1. 다음 명령을 사용하여 이름이 `origin-access-control.yaml`인 파일을 생성합니다. 이 파일에는 create-origin-access-control 명령에 대한 모든 입력 파라미터가 들어 있습니다.

   ```
   aws cloudfront create-origin-access-control --generate-cli-skeleton yaml-input >
    origin-access-control.yaml
   ```

2. 방금 생성한 `origin-access-control.yaml` 파일을 엽니다. 파일을 편집하여 OAC의 이름, 설명(선택 사항)을 추가하고 `SigningBehavior`를 `always`로 변경합니다. 그런 다음 파일을 저장합니다.

   다른 OAC 설정에 대한 자세한 내용은 the section called “오리진 액세스 제어를 위한 고급 설정” 섹션을 참조하세요.

3. 다음 명령으로 `origin-access-control.yaml` 파일의 입력 파라미터를 사용하여 오리진 액세스 제어를 만듭니다.

   ```
   aws cloudfront create-origin-access-control --cli-input-yaml file://origin-
   access-control.yaml
   ```

명령 출력에 있는 `Id` 값을 기록해 둡니다. CloudFront 배포의 S3 버킷 오리진에 OAC를 추가하려면 필요합니다.

OAC를 기존 배포의 S3 버킷 오리진에 연결하려면(입력 파일과 CLI)

1. 다음 명령을 사용하여 OAC를 추가할 CloudFront 배포에 대한 배포 구성을 저장합니다. 배포에는 S3 버킷 오리진이 있어야 합니다.

```
aws cloudfront get-distribution-config --id <CloudFront distribution ID> --
output yaml > dist-config.yaml
```

2. 방금 생성한 `dist-config.yaml`이라는 파일을 엽니다. 파일을 편집하여 다음과 같이 변경합니다.

   - `Origins` 객체에서 `OriginAccessControlId`라는 필드에 OAC의 ID를 추가합니다.
   - `OriginAccessIdentity`라는 필드에서 값을 제거합니다(있는 경우).
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

   완료되면 파일을 저장합니다.

3. 오리진 액세스 제어를 사용하도록 배포를 업데이트하려면 다음 명령을 사용합니다.

```
aws cloudfront update-distribution --id <CloudFront distribution ID> --cli-
input-yaml file://dist-config.yaml
```

모든 CloudFront 엣지 로케이션에 배포가 시작됩니다. 엣지 로케이션은 새 구성을 수신하면 S3 버킷 오리진으로 보내는 모든 요청에 서명합니다.

API

CloudFront API를 사용하여 오리진 액세스 제어를 생성하려면 CreateOriginAccessControl을 사용합니다. 이 API 호출에서 지정하는 필드에 대한 자세한 내용은 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

오리진 액세스 제어를 생성한 후 다음 API 호출 중 하나를 사용하여 배포의 S3 버킷 오리진에 연결할 수 있습니다.

- 기존 배포에 연결하려면 UpdateDistribution을 사용합니다.
- 새 배포에 연결하려면 CreateDistribution을 사용합니다.

이 두 API 호출에 대해 오리진 내부의 `OriginAccessControlId` 필드에 오리진 액세스 제어 ID를 제공합니다. 이러한 API 호출에서 지정하는 다른 필드에 대한 자세한 내용은 모든 배포 설정 참조와 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

## S3 버킷에 OAC가 연결된 배포 삭제

S3 버킷에 OAC가 연결된 배포를 삭제해야 하는 경우 S3 버킷 오리진을 삭제하기 전에 배포를 삭제해야 합니다. 또는 오리진 도메인 이름에 지역을 포함시킬 수도 있습니다. 이렇게 할 수 없는 경우 삭제하기 전에 공개로 전환하여 배포에서 OAC를 제거할 수 있습니다. 자세한 내용은 배포 삭제 섹션을 참조하세요.

## 오리진 액세스 ID(OAI)에서 오리진 액세스 제어(OAC)로 마이그레이션

레거시 오리진 액세스 ID(OAI)에서 오리진 액세스 제어(OAC)로 마이그레이션하려면 먼저 S3 버킷 오리진을 업데이트하여 OAC가 있는 배포 및 OAI 모두에서 버킷의 콘텐츠에 액세스할 수 있도록 합니다. 이렇게 하면 전환 중에 CloudFront가 버킷에 대한 액세스 권한을 잃지 않습니다. OAC가 있는 배포 및 OAI 모두에서 S3 버킷에 액세스할 수 있도록 하려면 각 위탁자에 대해 하나씩 두 개의 스테이트먼트를 포함하도록 버킷 정책을 업데이트합니다.

다음 예제 S3 버킷 정책은 OAC가 있는 배포 및 OAI 모두에서 S3 오리진에 액세스할 수 있도록 허용합니다.

Example OAC가 있는 CloudFront 배포 및 OAI에서 읽기 전용 액세스를 허용하도록 사용 설정된 S3 버킷 정책

JSON

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

```
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::<S3 bucket name>/*",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceArn":
 "arn:aws:cloudfront::111122223333:distribution/<CloudFront distribution ID>"
                }
            }
        },
        {
            "Sid": "AllowLegacyOAIReadOnly",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
 Identity <origin access identity ID>"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::<S3 bucket name>/*"
        }
    ]
}
```

OAI와 OAC에 대한 액세스를 모두 허용하도록 S3 오리진의 버킷 정책을 업데이트한 후 OAI 대신 OAC를 사용하도록 배포 구성을 업데이트할 수 있습니다. 자세한 내용은 the section called “새 오리진 액세스 제어 생성” 섹션을 참조하세요.

배포가 완전히 배포되면 버킷 정책에서 OAI에 대한 액세스를 허용하는 명령문을 제거할 수 있습니다. 자세한 내용은 the section called “CloudFront에 S3 버킷에 액세스할 수 있는 권한 부여” 섹션을 참조하세요.

## 오리진 액세스 제어를 위한 고급 설정

CloudFront 오리진 액세스 제어 기능에는 특정 사용 사례만을 위한 고급 설정이 포함되어 있습니다. 고급 설정이 특별히 필요한 경우가 아니면 권장 설정을 사용하세요.

오리진 액세스 제어에는 Signing behavior(서명 동작)(콘솔) 또는 `SigningBehavior`(API, CLI 및 CloudFormation)라는 설정이 포함되어 있습니다. 이 설정은 다음 옵션을 제공합니다.

항상 오리진 요청에 서명(권장 설정)

이 설정의 이름은 콘솔에서 Sign requests(recommended)(서명 요청(권장))이고 API, CLI, CloudFormation에서 `always`이며, 이 설정을 사용하는 것이 좋습니다. 이 설정을 사용하면 CloudFront가 항상 S3 버킷 오리진으로 보내는 모든 요청에 서명합니다.

오리진 요청 서명 안 함

이 설정의 이름은 콘솔에서 Do not sign requests(요청 서명 안 함)이고 API, CLI, CloudFormation에서 `never`입니다. 이 오리진 액세스 제어를 사용하는 모든 배포의 모든 오리진에 대한 오리진 액세스 제어를 끄려면 이 설정을 사용합니다. 오리진 액세스 제어를 사용하는 모든 오리진 및 배포에서 오리진 액세스 제어를 하나씩 제거하는 것과 비교하여 시간과 노력을 절약할 수 있습니다. 이 설정을 사용하면 CloudFront가 S3 버킷 오리진으로 보내는 모든 요청에 서명하지 않습니다.

> **Warning**
>
> 이 설정을 사용하려면 S3 버킷 오리진에 공개적으로 액세스할 수 있어야 합니다. 공개적으로 액세스할 수 없는 S3 버킷 오리진에 이 설정을 사용하는 경우 CloudFront가 오리진에 액세스할 수 없습니다. S3 버킷 오리진은 CloudFront에 오류를 반환하고 CloudFront 해당 오류를 뷰어에 전달합니다.

뷰어(클라이언트) **`Authorization`** 헤더 재정의 안 함

이 설정의 이름은 콘솔에서 Do not override authorization header(승인 헤더 재정의 안 함)이고 API, CLI, CloudFormation에서 `no-override`입니다. 해당 뷰어 요청에 `Authorization` 헤더가 포함되지 않은 경우에만 CloudFront에서 오리진 요청에 서명하도록 하려면 이 설정을 사용합니다. 이 설정을 사용하면 CloudFront에서는 뷰어 요청에 `Authorization` 헤더가 있는 경우 이를 전달하지만 뷰어 요청에 `Authorization` 헤더가 포함되어 있지 않으면 오리진 요청에 서명합니다(자체 `Authorization` 헤더 추가).

> **Warning**
>
> 뷰어 요청에서 `Authorization` 헤더를 전달하려면 이 오리진 액세스 제어와 연결된 S3 버킷 오리진을 사용하는 모든 캐시 동작에 대한 `Authorization` 헤더를 캐시 정책에 반드시 추가해야 합니다.

# 오리진 액세스 ID 사용(레거시, 권장하지 않음)

## 오리진 액세스 ID 개요

CloudFront 오리진 액세스 ID(OAI)는 오리진 액세스 제어(OAC)와 유사한 기능을 제공하지만 모든 시나리오에서 기능이 작동하는 것은 아닙니다. 특히 OAI는 다음을 지원하지 않습니다.

- 옵트인 리전을 포함한 모든 AWS 리전의 Amazon S3 버킷
- Amazon S3 AWS KMS를 사용한 서버 측 암호화(SSE-KMS)
- Amazon S3에 대한 동적 요청(PUT, POST 또는 DELETE)
- 2023년 1월 이후 새롭게 AWS 리전 출시

> **Tip**
> 대신 OAC를 사용할 것을 권장합니다. OAC를 설정하려면 새 오리진 액세스 제어 생성 섹션을 참조하세요. OAI에서 OAC로 마이그레이션하는 방법에 대한 자세한 내용은 the section called "오리진 액세스 ID(OAI)에서 오리진 액세스 제어(OAC)로 마이그레이션" 섹션을 참조하세요.

## 오리진 액세스 ID에 Amazon S3 버킷의 파일을 읽을 수 있는 권한 부여

OAI를 생성하거나 CloudFront 콘솔을 사용하여 배포에 추가할 때 Amazon S3 버킷 정책을 자동으로 업데이트하여 OAI에 버킷 액세스 권한을 부여할 수 있습니다. 또는 버킷 정책을 수동으로 생성하거나 업데이트하도록 선택할 수 있습니다. 어떤 방법을 사용하든 권한을 검토하여 다음 사항을 확인하세요.

- CloudFront를 통해 파일을 요청하는 최종 사용자를 대신하여 CloudFront OAI가 버킷의 파일에 액세스할 수 있습니다.
- 최종 사용자는 Amazon S3 URL을 사용하여 CloudFront 외부의 파일에 액세스할 수 없습니다.

> **Important**
> CloudFront가 지원하는 모든 HTTP 메서드를 수락하고 전달하도록 CloudFront를 구성하는 경우 CloudFront OAI에 원하는 권한을 부여해야 합니다. 예를 들어 DELETE 메서드를 사용하는 요청을 수락하고 전달하도록 CloudFront를 구성하는 경우, 뷰어가 원하는 파일만 삭제할 수 있도록 DELETE 요청을 적절하게 처리하도록 버킷 정책을 구성합니다.

### Amazon S3 버킷 정책 사용

다음과 같은 방법으로 버킷 정책을 만들거나 업데이트하여 Amazon S3 버킷의 파일에 대한 액세스 권한을 CloudFront OAI에 부여할 수 있습니다.

- Amazon S3 콘솔에서 Amazon S3 버킷의 Permissions(권한) 탭 사용
- Amazon S3 API에서 PutBucketPolicy 사용
- CloudFront 콘솔 사용 OAI를 CloudFront 콘솔의 원본 설정에 추가할 때 예, 버킷 정책을 업데이트합니다(Yes, update the bucket policy)를 선택하여 사용자를 대신해 버킷 정책을 업데이트하도록 CloudFront에 지시할 수 있습니다.

버킷 정책을 수동으로 업데이트하는 경우 다음 사항을 확인하세요.

- 정책에서 올바른 OAI를 `Principal`로 지정합니다.
- 최종 사용자를 대신해 객체에 액세스하는 데 필요한 권한을 OAI에 부여합니다.

자세한 내용은 다음 단원을 참조하세요.

### 버킷 정책에서 OAI를 **`Principal`**로 지정

Amazon S3 버킷 정책에서 OAI를 `Principal`로 지정하려면 OAI ID가 포함된 OAI의 Amazon 리소스 이름(ARN)을 사용합니다. 예제:

```
"Principal": {
    "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access Identity <origin
 access identity ID>"
}
```

CloudFront 콘솔의 보안, 오리진 액세스, ID(레거시)에서 OAI ID를 찾습니다. 또는 CloudFront API의 ListCloudFrontOriginAccessIdentities를 사용할 수도 있습니다.

### OAI에 권한 부여

OAI에 Amazon S3 버킷의 객체에 액세스할 수 있는 권한을 부여하려면 정책에서 특정 Amazon S3 API 작업과 관련된 작업을 사용합니다. 예를 들어 `s3:GetObject` 작업을 통해 OAI가 버킷의 객체를 읽을 수 있습니다. 자세한 내용은 다음 섹션의 예제를 참조하거나 Amazon Simple Storage Service 사용 설명서의 Amazon S3 작업을 참조하세요.

### Amazon S3 버킷 정책 예제

다음 예제에서는 CloudFront OAI가 S3 버킷에 액세스하도록 허용하는 Amazon S3 버킷 정책을 보여줍니다.

CloudFront 콘솔의 보안, 오리진 액세스, ID (레거시)에서 OAI ID를 찾을 수 있습니다. 또는 CloudFront API의 ListCloudFrontOriginAccessIdentities를 사용할 수도 있습니다.

#### Example OAI에 읽기 액세스 권한을 부여하는 Amazon S3 버킷 정책

다음 예제에서는 OAI가 지정된 버킷(`s3:GetObject`)의 객체를 읽을 수 있도록 허용합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Id": "PolicyForCloudFrontPrivateContent",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
 Identity <origin access identity ID>"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::<S3 bucket name>/*"
        }
    ]
}
```

#### Example OAI에 읽기 및 쓰기 액세스 권한을 부여하는 Amazon S3 버킷 정책

다음 예제에서는 OAI가 지정된 버킷(`s3:GetObject` 및 `s3:PutObject`)의 객체를 읽고 쓸 수 있도록 허용합니다. 이렇게 하면 최종 사용자가 CloudFront를 통해 Amazon S3 버킷에 파일을 업로드할 수 있습니다.

JSON

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

```
    "Id": "PolicyForCloudFrontPrivateContent",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
 Identity <origin access identity ID>"
            },
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::<S3 bucket name>/*"
        }
    ]
}
```

Amazon S3 객체 ACL 사용(권장되지는 않음)

> **Important**
>
> Amazon S3 버킷 정책을 사용하여 OAI에 S3 버킷에 대한 액세스 권한을 부여하는 것이 좋습니다. 이 섹션에 설명된 대로 액세스 제어 목록(ACL)을 사용할 수 있지만, 권장하지는 않습니다. Amazon S3는 S3 객체 소유권을 버킷 소유자 적용으로 설정할 것을 권장합니다. 즉, 버킷 및 그 안의 객체에 대해 ACL이 비활성화됩니다. 객체 소유권에 이 설정을 적용하는 경우 버킷 정책을 사용하여 OAI에 대한 액세스 권한을 부여해야 합니다(이전 섹션 참조).
> 다음 섹션은 ACL이 필요한 레거시 사용 사례에만 해당됩니다.

다음과 같은 방법으로 파일의 ACL을 만들거나 업데이트하여 Amazon S3 버킷의 파일에 대한 액세스 권한을 CloudFront OAI에 부여할 수 있습니다.

- Amazon S3 콘솔에서 Amazon S3 객체의 Permissions(권한) 탭 사용
- Amazon S3 API에서 PutObjectAcl 사용

ACL을 사용하여 OAI에 액세스 권한을 부여하는 경우 Amazon S3 정식 사용자 ID를 사용하여 OAI를 지정해야 합니다. CloudFront 콘솔의 보안, 오리진 액세스, ID (레거시)에서 이 ID를 찾을 수 있습니다. CloudFront API를 사용하는 경우, OAI를 만들 때 반환된 `S3CanonicalUserId` 요소의 값을 사용하거나 CloudFront API에서 ListCloudFrontOriginAccessIdentities를 호출합니다.

서명 버전 4 인증만 지원하는 Amazon S3 리전에서 오리진 액세스 ID 사용

새 Amazon S3 리전에서는 인증된 요청에 대해 서명 버전 4를 사용해야 합니다. (각 Amazon S3 리전에서 지원되는 서명 버전은 AWS 일반 참조의 Amazon Simple Storage Service 엔드포인트 및 할당량을 참조하세요.) 오리진 액세스 ID를 사용 중이고 버킷이 서명 버전 4가 필요한 리전 중 하나에 있는 경우 다음에 유의해야 합니다.

- `DELETE`, `GET`, `HEAD`, `OPTIONS` 및 `PATCH` 요청은 자격과 관계없이 지원됩니다.
- `POST` 요청은 지원되지 않습니다.

## VPC 오리진으로 액세스 제한

CloudFront를 사용하여 가상 프라이빗 클라우드(VPC) 프라이빗 서브넷에 호스팅되는 애플리케이션에서 콘텐츠를 제공할 수 있습니다. 프라이빗 서브넷의 Application Load Balancer(ALB), Network Load Balancer(NLB), EC2 인스턴스를 VPC 오리진으로 사용할 수 있습니다.

다음은 VPC 오리진을 사용하려는 몇 가지 이유입니다.

- 보안 - VPC 오리진은 프라이빗 서브넷에 로드 밸런서와 EC2 인스턴스를 배치하여 애플리케이션의 보안 태세를 강화하도록 설계되어 CloudFront를 단일 진입점으로 만듭니다. 사용자 요청은 프라이빗 보안 연결을 통해 CloudFront에서 VPC 오리진으로 이동하므로 애플리케이션의 보안을 강화합니다.
- 관리 - VPC 오리진은 CloudFront와 오리진 간의 보안 연결에 필요한 운영 오버헤드를 줄입니다. 퍼블릭 액세스 권한이 없는 프라이빗 서브넷으로 오리진을 이동할 수 있으며, 오리진에 대한 액세스를 제한하기 위해 액세스 제어 목록(ACL) 또는 기타 메커니즘을 구현할 필요가 없습니다. 이렇게 하면 CloudFront로 웹 애플리케이션을 보호하기 위해 차별화되지 않은 개발 작업에 투자할 필요가 없습니다.
- 확장성 및 성능 - VPC 오리진을 사용하면 웹 애플리케이션을 보호할 수 있으므로 CloudFront를 통해 보안을 개선하고 고성능 및 글로벌 확장성을 유지하면서 중요한 비즈니스 애플리케이션을 성장시키는 데 집중할 시간을 확보할 수 있습니다. VPC 오리진은 보안 관리를 간소화하고 운영 복잡성을 줄여주므로 CloudFront를 애플리케이션의 단일 진입점으로 사용할 수 있습니다.

> **Tip**
>
> CloudFront는 조직에 있든 없든 간에 AWS 계정에서 VPC 오리진 공유를 지원합니다. CloudFront 콘솔에서 VPC 오리진을 공유하거나 AWS Resource Access Manager(AWS RAM)를 사용할 수 있습니다. 자세한 내용은 CloudFront에서 공유 리소스 작업 섹션을 참조하세요.

## 사전 조건

CloudFront 배포에 대한 VPC 오리진을 만들기 전에 다음을 완료해야 합니다.

- Amazon VPC에 가상 프라이빗 클라우드(VPC)를 만듭니다.
  - VPC는 VPC 오리진에 대해 지원되는 AWS 리전 중 하나에 있어야 합니다. 자세한 내용은 VPC 오리진에서 지원되는 AWS 리전 섹션을 참조하세요.
  - VPC 서브넷과 연결된 네트워크 ACL은 VPC 오리진에서 클라이언트 IP 주소 보존이 활성화된 경우, 송신(아웃바운드) 트래픽에 적용됩니다. 하지만 트래픽이 VPC 오리진을 통해 종료되도록 허용하려면 ACL을 인바운드 규칙과 아웃바운드 규칙 모두로 구성해야 합니다.

    예를 들어, 임시 소스 포트를 사용하는 TCP 및 UDP 클라이언트가 VPC 오리진을 통해 엔드포인트에 연결되도록 허용하려면 엔드포인트의 서브넷을 임시 TCP 또는 UDP 포트(포트 범위 1024~65535, 대상 0.0.0.0/0)로 향하는 아웃바운드 트래픽을 허용하는 네트워크 ACL과 연결합니다. 또한 일치하는 인바운드 규칙(포트 범위 1024-65535, 소스 0.0.0.0/0)을 생성합니다.

  VPC 생성에 대한 자세한 내용은 Amazon VPC 사용 설명서의 VPC 및 기타 VPC 리소스 생성을 참조하시기 바랍니다.
- 요청에서 다음 내용을 포함합니다.
  - 인터넷 게이트웨이 - VPC 오리진 리소스가 있는 VPC에 인터넷 게이트웨이를 추가해야 합니다. 인터넷 게이트웨이는 VPC가 인터넷에서 트래픽을 수신할 수 있음을 나타내는 데 필요합니다. 인터넷 게이트웨이는 서브넷 내의 오리진으로 트래픽을 라우팅하는 데 사용되지 않으므로 라우팅 정책을 업데이트할 필요가 없습니다.
  - 사용 가능한 IPv4 주소가 최소 하나 이상인 프라이빗 서브넷 - CloudFront는 사용자가 CloudFront에서 VPC 오리진 리소스를 정의한 후 CloudFront가 생성하는 서비스 관리형 탄력적 네트워크 인터페이스(ENI)를 사용하여 서브넷으로 라우팅합니다. ENI 생성 프로세스에 성공하려면 프라이빗 서브넷에 사용 가능한 IPv4 주소가 하나 이상 있어야 합니다. IPv4 주소는 프라이빗 주소일 수 있으며 추가 비용은 없습니다.

    > **Note**
    >
    > IPv6 전용 서브넷은 지원되지 않습니다.
- 프라이빗 서브넷에서 Application Load Balancer, Network Load Balancer, 오리진으로 사용할 EC2 인스턴스를 시작합니다.
  - 시작하는 리소스는 완전히 배포되고 활성 상태여야 VPC 오리진에 사용할 수 있습니다.

- Gateway Load Balancer, 듀얼 스택 Network Load Balancer 및 TLS 리스너가 있는 Network Load Balancer는 오리진으로 추가할 수 없습니다.
- VPC 오리진으로 사용하려면 Network Load Balancer에 보안 그룹이 연결되어 있어야 합니다.
- CloudFront 관리형 접두사 목록을 명시적으로 허용하도록 VPC 프라이빗 오리진의 보안 그룹을 업데이트합니다. 자세한 내용은 CloudFront 관리형 접두사 목록 사용 섹션을 참조하세요.

> **Note**
>
> CloudFront-VPCOrigins-Service-SG는 VPC 오리진에 사용되는 보안 그룹의 AWS 예약 이름입니다. 보안 그룹에 다른 이름을 지정해야 합니다. 자세한 내용은 보안 그룹 생성을 참조하세요.

- VPC 오리진이 생성된 후 VPC 오리진에서의 트래픽만 허용하도록 보안 그룹을 추가로 제한할 수 있습니다. 이렇게 하려면 관리형 접두사 목록에서 허용된 트래픽 소스를 CloudFront 보안 그룹으로 업데이트합니다.

> **Note**
>
> VPC 오리진의 경우 CloudFront에서 Lambda@Edge를 사용한 WebSockets, gRPC 트래픽, 오리진 요청 및 오리진 응답 트리거가 지원되지 않습니다. 자세한 내용은 Lambda@Edge 설명서의 요청 및 응답 작업 수행 페이지를 참고하시기 바랍니다.

## VPC 오리진 생성(새 배포)

다음 절차에서는 CloudFront 콘솔에서 새 CloudFront 배포에 대한 VPC 오리진을 만드는 방법을 보여줍니다. 또는 AWS CLI나 AWS SDK에서 CreateVpcOrigin 및 CreateDistribution API 작업을 사용할 수 있습니다.

새 CloudFront 배포에 대한 VPC 오리진을 만들려면 다음과 같이 합니다.

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. VPC 오리진, VPC 오리진 생성을 선택합니다.
3. 필수 필드를 입력합니다. 오리진 ARN에서 Application Load Balancer, Network Load Balancer 또는 EC2 인스턴스의 ARN을 선택합니다. ARN이 표시되지 않으면 특정 리소스 ARN을 복사하여 여기에 붙여넣을 수 있습니다.

4. VPC 오리진 생성을 선택합니다.
5. VPC 오리진 상태가 배포됨으로 변경될 때까지 기다립니다. 이 프로세스는 최대 15분이 걸릴 수 있습니다.
6. 배포, 배포 생성을 선택합니다.
7. 원본 도메인의 드롭다운 목록에서 VPC 오리진 리소스를 선택합니다.

   VPC 오리진이 EC2 인스턴스인 경우 인스턴스의 프라이빗 IP DNS 이름을 복사하여 원본 도메인 필드에 붙여넣습니다.
8. 배포 생성을 완료합니다. 자세한 내용은 콘솔에서 CloudFront 배포 생성 섹션을 참조하세요.

## VPC 오리진 생성(기존 배포)

다음 절차에서는 CloudFront 콘솔에서 기존 CloudFront 배포에 대한 VPC 오리진을 만드는 방법을 보여줍니다. 이를 통해 애플리케이션의 지속적인 가용성을 보장할 수 있습니다. 또는 AWS CLI나 AWS SDK와 함께 CreateVpcOrigin 및 UpdateDistributionWithStagingConfig API 작업을 사용할 수 있습니다.

필요에 따라 스테이징 배포를 만들지 않고 기존 배포에 VPC 오리진을 추가하도록 선택할 수 있습니다.

기존 CloudFront 배포에 대한 VPC 오리진을 만들려면 다음과 같이 합니다.

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. VPC 오리진, VPC 오리진 생성을 선택합니다.
3. 필수 필드를 입력합니다. 오리진 ARN에서 Application Load Balancer, Network Load Balancer 또는 EC2 인스턴스의 ARN을 선택합니다. ARN이 표시되지 않으면 특정 리소스 ARN을 복사하여 여기에 붙여넣을 수 있습니다.
4. VPC 오리진 생성을 선택합니다.
5. VPC 오리진 상태가 배포됨으로 변경될 때까지 기다립니다. 이 프로세스는 최대 15분이 걸릴 수 있습니다.
6. 탐색 창에서 Distributions(배포)를 선택합니다.
7. 배포 ID를 선택합니다.
8. 일반 탭의 지속적 배포에서 스테이징 배포 생성을 선택합니다. 자세한 내용은 CloudFront 지속적 배포를 사용하여 CDN 구성 변경을 안전하게 테스트 섹션을 참조하세요.
9. 스테이징 배포 생성 마법사의 단계에 따라 스테이징 배포를 만듭니다. 다음 단계를 포함합니다.

   - 오리진에서 원본 생성을 선택합니다.

- 원본 도메인의 드롭다운 메뉴에서 VPC 오리진 리소스를 선택합니다.

  VPC 오리진이 EC2 인스턴스인 경우 인스턴스의 프라이빗 IP DNS 이름을 복사하여 원본 도메인 필드에 붙여넣습니다.
- 오리진 생성(Create Origin)을 선택합니다.

10. 스테이징 배포에서 VPC 오리진을 테스트합니다.
11. 스테이징 배포 구성을 기본 배포로 승격합니다. 자세한 내용은 스테이징 배포 구성 승격 섹션을 참조하세요.
12. 서브넷을 프라이빗으로 설정하여 VPC 오리진에 대한 퍼블릭 액세스를 제거합니다. 이렇게 하면 인터넷을 통해 VPC 오리진을 검색할 수 없지만 CloudFront는 여전히 프라이빗 액세스 권한을 갖습니다. 자세한 내용은 Amazon VPC 사용 설명서의 서브넷을 라우팅 테이블과 연결 또는 연결 해제를 참조하시기 바랍니다.

## VPC 오리진 업데이트

다음 절차에서는 CloudFront 콘솔에서 CloudFront 배포의 VPC 오리진을 업데이트하는 방법을 보여줍니다. 또는 AWS CLI나 AWS SDK와 함께 UpdateDistribution 및 UpdateVpcOrigin API 작업을 사용할 수 있습니다.

CloudFront 배포의 기존 VPC 오리진을 업데이트하려면 다음과 같이 합니다.

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Distributions(배포)를 선택합니다.
3. 배포 ID를 선택합니다.
4. 동작 탭을 선택합니다.
5. VPC 오리진이 캐시 동작의 기본 오리진인지 여부를 확인합니다.
6. 오리진 탭을 선택합니다.
7. 업데이트할 VPC 오리진을 선택하고 삭제를 선택합니다. 이렇게 하면 배포에서 VPC 오리진의 연결이 해제됩니다. 2~7단계를 반복하여 VPC 오리진을 다른 배포에서 연결 해제합니다.
8. VPC 오리진을 선택합니다.
9. VPC 오리진을 선택하고 편집을 선택합니다.
10. 업데이트를 수행하고 VPC 오리진 업데이트를 선택합니다.
11. VPC 오리진 상태가 배포됨으로 변경될 때까지 기다립니다. 이 프로세스는 최대 15분이 걸릴 수 있습니다.

12. 탐색 창에서 Distributions(배포)를 선택합니다.
13. 배포 ID를 선택합니다.
14. 오리진 탭을 선택합니다.
15. 오리진 생성(Create Origin)을 선택합니다.
16. 원본 도메인의 드롭다운 메뉴에서 VPC 오리진 리소스를 선택합니다.

    VPC 오리진이 EC2 인스턴스인 경우 인스턴스의 프라이빗 IP DNS 이름을 복사하여 원본 도메인 필드에 붙여넣습니다.
17. 오리진 생성(Create Origin)을 선택합니다. 이렇게 하면 VPC 오리진이 배포와 다시 연결됩니다. 12~17단계를 반복하여 업데이트된 VPC 오리진을 다른 배포와 연결합니다.

## VPC 오리진에서 지원되는 AWS 리전

VPC 오리진은 현재 다음 상업용 AWS 리전에서 지원됩니다. 가용 영역(AZ) 예외가 기록됩니다.

| 리전 이름 | 리전 |
| --- | --- |
| 미국 동부(오하이오) | `us-east-2` |
| 미국 동부(버지니아 북부) | `us-east-1 (except AZ use1-az3)` |
| 미국 서부(캘리포니아 북부) | `us-west-1 (except AZ usw1-az2)` |
| 미국 서부(오리건) | `us-west-2` |
| 아프리카(케이프타운) | `af-south-1` |
| 아시아 태평양(홍콩) | `ap-east-1` |
| 아시아 태평양(뭄바이) | `ap-south-1` |
| 아시아 태평양(하이데라바드) | `ap-south-2` |
| 아시아 태평양(자카르타) | `ap-southeast-3` |
| 아시아 태평양(멜버른) | `ap-southeast-4` |
| 아시아 태평양(오사카) | `ap-northeast-3` |

| 리전 이름 | 리전 |
| --- | --- |
| 아시아 태평양(싱가포르) | ap-southeast-1 |
| 아시아 태평양(시드니) | ap-southeast-2 |
| 아시아 태평양(도쿄) | ap-northeast-1 (except AZ apne1-az3) |
| 아시아 태평양(서울) | ap-northeast-2 (except AZ apne2-az1) |
| 아시아 태평양(태국) | ap-southeast-7 |
| 아시아 태평양(말레이시아) | ap-southeast-5 |
| 아시아 태평양(타이베이) | ap-east-2 |
| 캐나다(중부) | ca-central-1 (except AZ cac1-az3) |
| 캐나다 서부(캘거리) | ca-west-1 |
| 유럽(프랑크푸르트) | eu-central-1 |
| 유럽(아일랜드) | eu-west-1 |
| 유럽(런던) | eu-west-2 |
| 유럽(밀라노) | eu-south-1 |
| 유럽(파리) | eu-west-3 |
| 유럽(스페인) | eu-south-2 |
| 유럽(스톡홀름) | eu-north-1 |
| 유럽(취리히) | eu-central-2 |
| 이스라엘(텔아비브) | il-central-1 |
| 중동(바레인) | me-south-1 |

| 리전 이름 | 리전 |
| --- | --- |
| 중동(UAE) | me-central-1 |
| 남아메리카(상파울루) | sa-east-1 |
| 멕시코(중부) | mx-central-1 |

# Application Load Balancer에 대한 액세스 제한

Amazon CloudFront에서는 내부 및 인터넷 연결 Application Load Balancer를 모두 사용할 수 있습니다. VPC 오리진을 사용하여 CloudFront의 프라이빗 서브넷 내에서 내부 Application Load Balancer를 사용할 수 있습니다. CloudFront VPC 오리진을 사용하면 퍼블릭 인터넷에 노출시키지 않고 프라이빗 VPC 서브넷에 호스팅된 애플리케이션에서 콘텐츠를 제공할 수 있습니다. 자세한 내용은 VPC 오리진으로 액세스 제한 섹션을 참조하세요.

CloudFront에서 인터넷 연결 Application Load Balancer를 사용하는 경우 다음 보안 완화 조치를 사용하여 사용자가 Application Load Balancer에 직접 액세스하지 못하도록 하고 CloudFront를 통한 액세스만 허용할 수 있습니다.

1. Application Load Balancer로 보내는 요청에 사용자 지정 HTTP 헤더를 추가하도록 CloudFront를 구성합니다.
2. 사용자 지정 HTTP 헤더가 포함된 요청만 전달하도록 Application Load Balancer를 구성합니다.
3. HTTPS를 사용하도록 하여 이 솔루션의 보안을 강화합니다.

CloudFront는 지연 시간을 줄이고 일부 분산 서비스 거부(DDoS) 공격을 완화하는 데 도움이 됩니다.

사용 사례에 따라 인터넷을 통해 직접 CloudFront와 Application Load Balancer 모두에서 웹 애플리케이션에 이중 액세스해야 하는 경우 다음과 같이 웹 애플리케이션 API를 분할하는 것이 좋습니다.

- CloudFront를 통과해야 하는 API. 이 경우 별도의 프라이빗 Application Load Balancer를 오리진으로 사용하는 것이 좋습니다.
- Application Load Balancer를 통해 액세스해야 하는 API. 이 경우 CloudFront를 우회합니다.

또한 CloudFront는 Elastic Load Balancing에서 인터넷 경계 Application Load Balancer를 통해 제공되는 웹 애플리케이션 또는 기타 콘텐츠의 객체를 캐싱하고 사용자(뷰어)에게 직접 제공하여 Application

Load Balancer의 로드를 줄입니다. 인터넷 경계 로드 밸런서는 공개적으로 확인 가능한 DNS 이름이 있으며 인터넷을 통해 클라이언트의 요청을 대상으로 라우팅합니다.

자세한 내용은 다음 항목을 참조하세요. 이 단계를 완료하면 사용자가 CloudFront를 통해서만 Application Load Balancer에 액세스할 수 있습니다.

주제

- 사용자 지정 HTTP 헤더를 요청에 추가하도록 CloudFront 구성
- 특정 헤더가 포함된 요청만 전달하도록 Application Load Balancer 구성
- (선택 사항) 이 솔루션의 보안 강화
- (선택 사항) CloudFront의 AWS-managed 접두사 목록을 사용하여 오리진에 대한 액세스를 제한합니다.

## 사용자 지정 HTTP 헤더를 요청에 추가하도록 CloudFront 구성

오리진(이 경우 Application Load Balancer)으로 보내는 요청에 사용자 지정 HTTP 헤더를 추가하도록 CloudFront를 구성할 수 있습니다.

> **Important**
>
> 이 사용 사례에서는 사용자 지정 헤더의 이름과 값을 기밀로 유지해야 합니다. 헤더 이름과 값을 기밀로 유지하지 않으면 다른 HTTP 클라이언트에서 Application Load Balancer로 직접 보내는 요청에 헤더 이름과 값을 포함할 수 있습니다. 그러면 Application Load Balancer가 CloudFront의 요청이 아닌 것을 CloudFront의 요청처럼 처리할 수 있습니다. 이를 방지하려면 사용자 정의 헤더의 이름과 값을 기밀로 유지해야 합니다.

CloudFront 콘솔, CloudFormation 또는 CloudFront API를 사용하여 오리진 요청에 사용자 지정 HTTP 헤더를 추가하도록 CloudFront를 구성할 수 있습니다.

사용자 지정 HTTP 헤더를 추가하려면(CloudFront 콘솔)

CloudFront 콘솔에서 오리진 설정의 오리진 사용자 지정 헤더 설정을 사용합니다. 헤더 이름 및 해당 값을 입력합니다.

> **Note**
> 프로덕션 환경에서는 무작위로 생성된 헤더 이름과 값을 사용합니다. 헤더 이름과 값은 사용자 이름 및 암호와 같은 보안 자격 증명으로 취급합니다.

기존 CloudFront 배포에 대한 오리진을 생성 또는 편집할 때와 새 배포를 생성할 때 오리진 사용자 지정 헤더 설정을 편집할 수 있습니다. 자세한 내용은 배포 업데이트 및 배포 생성(을)를 참조하세요.

사용자 지정 HTTP 헤더 추가(CloudFormation)

CloudFormation 템플릿에서 다음 예제와 같이 `OriginCustomHeaders` 속성을 사용합니다.

> **Note**
> 이 예제에서 헤더 이름과 값은 데모용입니다. 프로덕션 환경에서는 무작위로 생성된 값을 사용합니다. 헤더 이름과 값은 사용자 이름 및 암호와 같은 보안 자격 증명으로 취급해야 합니다.

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  TestDistribution:
    Type: 'AWS::CloudFront::Distribution'
    Properties:
      DistributionConfig:
        Origins:
          - DomainName: app-load-balancer.example.com
            Id: Example-ALB
            CustomOriginConfig:
              OriginProtocolPolicy: https-only
              OriginSSLProtocols:
                - TLSv1.2
            OriginCustomHeaders:
              - HeaderName: X-Custom-Header
                HeaderValue: random-value-1234567890
        Enabled: 'true'
        DefaultCacheBehavior:
          TargetOriginId: Example-ALB
          ViewerProtocolPolicy: allow-all
```

```
          CachePolicyId: 658327ea-f89d-4fab-a63d-7e88639e58f6
        PriceClass: PriceClass_All
        ViewerCertificate:
          CloudFrontDefaultCertificate: 'true'
```

자세한 내용은 AWS CloudFormation사용 설명서의 오리진 및 OriginCustomHeader 속성을 참조합니다.

사용자 지정 HTTP 헤더를 추가하려면(CloudFront API)

CloudFront API에서 `CustomHeaders` 안에 `Origin` 객체를 사용합니다. 자세한 내용은 Amazon CloudFront API 참조의 CreateDistribution 및 UpdateDistribution과 SDK 또는 기타 API 클라이언트 설명서를 참조합니다.

오리진 사용자 지정 헤더로 지정할 수 없는 헤더 이름이 몇 개 있습니다. 자세한 내용은 CloudFront에서 오리진 요청에 추가할 수 없는 사용자 지정 헤더 섹션을 참조하세요.

## 특정 헤더가 포함된 요청만 전달하도록 Application Load Balancer 구성

Application Load Balancer로 보내는 요청에 사용자 지정 HTTP 헤더를 추가하도록 CloudFront를 구성(이전 섹션 참조)한 후 이 사용자 지정 헤더가 포함된 요청만 전달하도록 로드 밸런서를 구성할 수 있습니다. 새 규칙을 추가하고 로드 밸런서의 리스너에서 기본 규칙을 수정하면 됩니다.

사전 조건

다음 절차를 사용하려면 리스너가 하나 이상인 Application Load Balancer가 필요합니다. 아직 생성하지 않았다면 Application Load Balancer 사용 설명서에서 Application Load Balancer 만들기를 참조합니다.

다음 절차에서는 HTTPS 수신기를 수정합니다. 동일한 프로세스를 사용하여 HTTP 수신기를 수정할 수 있습니다.

Application Load Balancer 수신기의 규칙을 업데이트하려면

1. 새 규칙을 추가합니다. 규칙 추가의 지침을 따르고 다음과 같이 수정하세요.
   - CloudFront 배포의 오리진인 로드 밸런서에 규칙을 추가합니다.
   - 조건 추가에서 Http 헤더를 선택합니다. CloudFront에서 오리진 사용자 지정 헤더로 추가한 HTTP 헤더 이름과 값을 지정합니다.
   - 작업 추가에서 다음으로 전달:을 선택합니다. 요청을 전달할 대상 그룹을 선택합니다.

2. 로드 밸런서의 리스너에서 기본 규칙을 편집합니다. 규칙 편집의 지침을 따르고 다음과 같이 수정하세요.

   - CloudFront 배포의 오리진인 로드 밸런서의 기본 규칙을 편집합니다.
   - 기본 작업을 삭제한 다음 작업 추가에서 고정 응답 반환을 선택합니다.
   - 응답 코드에 **403**을 입력합니다.
   - 응답 본문에 **Access denied**를 입력합니다.

이 단계를 완료하면 2개의 규칙이 로드 밸런서 리스너에 추가됩니다. 한 규칙은 HTTP 헤더가 포함된 요청(CloudFront에서 오는 요청)을 전달합니다. 또 다른 규칙은 다른 모든 요청(CloudFront에서 오지 않은 요청)에 고정 응답을 보냅니다.

CloudFront 배포와 Application Load Balancer로 요청을 보내 솔루션이 작동하는지 확인할 수 있습니다. CloudFront에 대한 요청은 웹 애플리케이션 또는 콘텐츠를 반환하고 Application Load Balancer로 직접 전송된 요청은 일반 텍스트 메시지 `403`와 함께 `Access denied` 응답을 반환합니다.

## (선택 사항) 이 솔루션의 보안 강화

이 솔루션의 보안을 강화하려면 Application Load Balancer로 요청을 보낼 때 항상 HTTPS를 사용하도록 CloudFront 배포를 구성할 수 있습니다. 이 솔루션이 작동하려면 사용자 지정 헤더 이름과 값을 기밀로 유지해야 합니다. HTTPS를 사용하면 도청자가 헤더 이름과 값을 검색할 수 없습니다. 또한 헤더 이름과 값을 주기적으로 교체하는 것이 좋습니다.

오리진 요청에 HTTPS 사용

오리진 요청에 HTTPS를 사용하도록 CloudFront를 구성하려면 오리진 프로토콜 정책 설정을 HTTPS 전용으로 설정합니다. 이 설정은 CloudFront 콘솔, CloudFormation 및 CloudFront API에서 사용할 수 있습니다. 자세한 내용은 프로토콜(사용자 지정 오리진만 해당) 섹션을 참조하세요.

오리진 요청에 HTTPS를 사용하도록 CloudFront를 구성하는 경우에도 다음이 적용됩니다.

- 오리진 요청 정책에 따라 `Host` 헤더를 오리진에 전달하도록 CloudFront를 구성해야 합니다. AllViewer 관리형 오리진 요청 정책을 사용할 수 있습니다.
- Application Load Balancer에 HTTPS 리스너가 있는지 확인합니다 (이전 섹션 참조). 자세한 내용은 Application Load Balancer 사용 설명서에서 HTTPS 리스너 생성을 참조합니다. HTTPS 리스너를 사용하려면 Application Load Balancer로 라우팅된 도메인 이름과 일치하는 SSL/TLS 인증서가 있어야 합니다.

- CloudFront용 SSL/TLS 인증서는 AWS Certificate Manager(ACM)의 `us-east-1` AWS 리전 에서만 요청(또는 가져오기)할 수 있습니다. CloudFront는 글로벌 서비스이므로 해당 지역의 인증서를 CloudFront 배포와 관련된 모든 `us-east-1` 리전에 자동으로 배포합니다.
  - 예를 들어 `ap-southeast-2` 리전에 Application Load Balancer(ALB)가 있는 경우 `ap-southeast-2` 리전(CloudFront와 ALB 오리진 간 HTTPS 사용)과 `us-east-1` 리전(최종 사용자와 CloudFront 간 HTTPS 사용) 모두에서 SSL/TLS 인증서를 구성해야 합니다. 두 인증서는 Application Load Balancer로 라우팅되는 도메인 이름과 일치해야 합니다. 자세한 내용은 AWS 리전 for AWS Certificate Manager 섹션을 참조하세요.
- 웹 애플리케이션의 최종 사용자(최종 사용자 또는 클라이언트라고도 함)가 HTTPS를 사용할 수 있는 경우 최종 사용자의 HTTPS 연결을 기본적으로 사용하거나 요구하도록 CloudFront를 구성할 수도 있습니다. 이렇게 하려면 뷰어 프로토콜 정책 설정을 사용 합니다. 최종 사용자를 HTTP에서 HTTPS로 리디렉션하거나 HTTP를 사용하는 요청을 거부하도록 설정할 수 있습니다. 이 설정은 CloudFront 콘솔, CloudFormation 및 CloudFront API에서 사용할 수 있습니다. 자세한 내용은 뷰어 프로토콜 정책 섹션을 참조하세요.

헤더 이름과 값 교체

HTTPS를 사용하는 것에 더해 헤더 이름과 값을 주기적으로 교체하는 것이 좋습니다. 이 작업을 수행하는 간략한 단계는 다음과 같습니다.

1. Application Load Balancer로 보내는 요청에 추가 사용자 지정 HTTP 헤더를 추가하도록 CloudFront를 구성합니다.
2. 추가 사용자 지정 HTTP 헤더가 포함된 요청만 전달하도록 Application Load Balancer 리스너 규칙을 업데이트합니다.
3. Application Load Balancer로 보내는 요청에 원래 사용자 지정 HTTP 헤더를 추가하지 않도록 CloudFront를 구성합니다.
4. 원래 사용자 지정 HTTP 헤더가 포함된 요청을 전달하지 않도록 Application Load Balancer 리스너 규칙을 업데이트합니다.

이러한 단계를 수행하는 방법에 대한 자세한 내용은 이전 섹션을 참조하세요.

## (선택 사항) CloudFront의 AWS-managed 접두사 목록을 사용하여 오리진에 대한 액세스를 제한합니다.

Application Load Balancer에 대한 액세스를 더욱 제한하려면 서비스가 AWS-managed 접두사 목록을 사용할 때만 CloudFront의 트래픽을 수락하도록 Application Load Balancer와 연결된 보안 그룹을 구

성할 수 있습니다. 이렇게 하면 CloudFront에서 시작되지 않은 트래픽이 네트워크 계층(계층 3) 또는 전송 계층(계층 4)의 Application Load Balancer에 도달하는 것을 방지할 수 있습니다.

자세한 내용은 Amazon CloudFront의 AWS-managed 접두사 목록을 사용하여 오리진에 대한 액세스 제한 블로그 게시물을 참조합니다.

# 콘텐츠의 지리적 배포 제한

지리적 차단이라고도 하는 지리적 제한을 사용하여 특정 지리적 위치에 있는 사용자가 Amazon CloudFront 배포를 통해 배포한 콘텐츠에 액세스하는 것을 차단할 수 있습니다. 지리적 제한을 사용하는 데에는 다음 두 가지 옵션이 있습니다.

- CloudFront 지리적 제한 기능을 사용합니다. 이 옵션을 사용하면 배포와 연결된 파일 전체에 대한 액세스를 제한하고 국가 수준에서 액세스를 제한합니다.
- 타사 지리적 위치 서비스를 사용합니다. 이 옵션을 사용하면 배포와 연결된 파일의 하위 집합에 대한 액세스를 제한하거나 국가 수준이 아닌 더욱 세분화된 범위에서 액세스를 제한합니다.

주제


- CloudFront 지리적 제한 사용
- 서드 파티 지리적 위치 서비스 사용


## CloudFront 지리적 제한 사용

사용자가 콘텐츠를 요청할 경우 일반적으로 CloudFront는 사용자가 위치한 장소와 상관없이 요청한 콘텐츠를 제공합니다. 특정 지리적 위치에 있는 사용자가 콘텐츠에 액세스하는 것을 차단해야 하는 경우, CloudFront 지리적 제한 기능을 사용하여 다음 중 하나를 실행할 수 있습니다.

- 허용 목록에 있는 승인된 국가에 있는 사용자만 콘텐츠에 액세스할 수 있도록 권한을 부여합니다.
- 사용자가 거부 목록의 금지된 국가 중 하나에 있는 경우 액세스를 금지합니다.

예를 들어, 콘텐츠를 배포하도록 허용되지 않는 특정 국가로부터 요청이 온 경우 CloudFront 지리적 제한을 사용하여 이 요청을 차단할 수 있습니다.

> **Note**
>
> CloudFront에서는 서드 파티 데이터베이스를 사용하여 사용자의 위치를 확인합니다. IP 주소 및 국가 간 매핑 정확도는 리전에 따라 다릅니다. 최근 테스트에 따르면 전반적인 정확도는

99.8%입니다. CloudFront에서 사용자 위치를 확인할 수 없는 경우 CloudFront는 사용자가 요청한 콘텐츠를 제공합니다.

지리적 제한이 작동하는 방식은 다음과 같습니다.

1. 리히텐슈타인에서만 콘텐츠를 배포할 권리를 보유하고 있다고 가정합니다. CloudFront 배포를 업데이트하고 리히텐슈타인만 포함된 허용 목록을 추가합니다. (또는 리히텐슈타인을 제외한 모든 국가가 포함된 거부 목록을 추가할 수도 있습니다.)
2. 모나코에 있는 사용자가 콘텐츠를 요청하고 DNS는 요청을 이탈리아 밀라노에 있는 CloudFront 엣지 로케이션으로 라우팅합니다.
3. 밀라노의 엣지 로케이션에서는 이 배포를 찾아 모나코에 있는 사용자의 콘텐츠 다운로드가 허용되지 않았음을 확인합니다.
4. CloudFront에서는 HTTP 상태 코드 `403 (Forbidden)`을 사용자에게 반환합니다.

선택적으로 CloudFront에서 사용자 지정 오류 메시지를 사용자에게 반환하도록 구성할 수 있으며, CloudFront에서 요청된 파일에 대한 오류 응답을 캐싱하려는 시간을 지정할 수 있습니다. 기본값은 10초입니다. 자세한 내용은 특정 HTTP 상태 코드에 대한 사용자 지정 오류 페이지 생성 섹션을 참조하세요.

지리적 제한은 전체 배포에 적용됩니다. 하나의 제한을 콘텐츠의 특정 부분에 적용하고 또 다른 제한을 다른 부분에 적용해야 할 경우(또는 이 부분에 대한 제한이 없는 경우), 별도의 CloudFront 배포를 만들거나 서드 파티 지리적 위치 서비스를 사용해야 합니다.

CloudFront 액세스 표준 로그(액세스 로그)를 사용하는 경우, `sc-status`(HTTP 상태 코드)의 값이 `403`인 로그 항목을 검색하여 CloudFront에서 거부된 요청을 식별할 수 있습니다. 그러나 표준 로그만 사용해서는 사용자의 위치에 따라 CloudFront에서 거부된 요청과 사용자가 다른 이유로 파일에 대한 액세스 권한이 없어 CloudFront에서 거부된 요청을 구별할 수 없습니다. Digital Element 또는 MaxMind와 같은 타사 지리적 위치 서비스가 있는 경우, 액세스 로그에 있는 `c-ip`(클라이언트 IP) 열의 IP 주소에 따라 요청의 위치를 식별할 수 있습니다. CloudFront 표준 로그에 대한 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하세요.

다음 절차는 CloudFront 콘솔을 사용하여 기존 배포에 지리적 제한을 추가하는 방법을 설명합니다. 콘솔을 사용하여 배포를 생성하는 방법에 대한 자세한 내용은 배포 생성 섹션을 참조하세요.

CloudFront 웹 배포에 지리적 제한을 추가하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 배포를 선택한 후 업데이트하려는 배포를 선택합니다.
3. 보안 탭을 선택한 다음 지리적 제한을 선택합니다.
4. 편집을 선택합니다.
5. 허용 목록(Allow list)을 선택하여 허용된 국가의 목록을 생성하거나 차단 목록(Block list)을 선택하여 차단된 국가의 목록을 생성합니다.
6. 원하는 국가를 목록에 추가한 다음 변경 사항 저장(Save changes)을 선택합니다.

## 서드 파티 지리적 위치 서비스 사용

CloudFront 지리적 제한 기능을 사용하여 지정된 웹 배포를 통해 배포하는 전체 파일에 대해 국가 수준에서 콘텐츠의 배포를 제어할 수 있습니다. 국가의 경계를 따르지 않는 지리적 제한이 있는 경우 또는 해당 배포를 통해 제공하는 파일 중 일부에 대해서만 액세스를 제한하려는 경우, CloudFront와 서드 파티 지리적 위치 서비스를 조합할 수 있습니다. 이를 통해 국가에 따라서나 도시, 우편번호에 따라서는 물론이고 경도/위도에 따라서도 콘텐츠를 제어할 수 있습니다.

타사 지리적 위치 서비스를 사용하는 경우 CloudFront 서명된 URL을 사용하는 것이 좋습니다. 이렇게 하면 URL이 더 이상 유효하지 않게 된 이후의 만료 날짜 및 시간을 지정할 수 있습니다. 또한 오리진으로는 Amazon S3 버킷을 사용하는 것이 좋습니다. 이렇게 하면 CloudFront 오리진 액세스 제어를 사용하여 사용자가 오리진에서 콘텐츠에 직접 액세스하는 것을 차단할 수 있기 때문입니다. 서명된 URL 및 오리진 액세스 제어에 대한 자세한 내용은 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공 섹션을 참조하세요.

다음 단계에서는 서드 파티 지리적 위치 서비스를 사용하여 파일에 대한 액세스를 제어하는 방법을 설명합니다.

서드 파티 지리적 위치 서비스를 사용하여 CloudFront 배포의 파일에 대한 액세스를 제한하려면

1. 지리적 위치 서비스를 통해 계정을 얻습니다.
2. Amazon S3 버킷에 콘텐츠를 업로드합니다.
3. 프라이빗 콘텐츠를 제공하도록 Amazon CloudFront 및 Amazon S3를 구성합니다. 자세한 내용은 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공 단원을 참조하세요.
4. 다음을 수행하도록 웹 애플리케이션을 작성합니다.

- 각 사용자 요청에 대한 IP 주소를 지리적 위치 서비스에 전송합니다.
- 지리적 위치 서비스로부터의 반환 값을 평가하여 CloudFront에서 콘텐츠를 배포하려는 위치에 사용자가 있는지 여부를 확인합니다.
- 사용자의 위치에 콘텐츠를 배포하려면 CloudFront 콘텐츠에 대해 서명된 URL을 생성합니다. 해당 위치에 콘텐츠를 배포하지 않으려면 HTTP 상태 코드 `403 (Forbidden)`를 사용자에게 반환합니다. 또는 사용자 지정 오류 메시지를 반환하도록 CloudFront를 구성할 수 있습니다. 자세한 내용은 the section called "특정 HTTP 상태 코드에 대한 사용자 지정 오류 페이지 생성" 섹션을 참조하세요.

자세한 내용은 사용 중인 지리적 위치 서비스의 설명서를 참조하세요.

웹 서버 변수를 사용하여 웹 사이트를 방문한 사용자의 IP 주소를 가져올 수 있습니다. 다음 경고를 기록해 두세요.

- 웹 서버가 로드 밸런서를 통해 인터넷에 연결되지 않은 경우, 웹 서버 변수를 사용하여 원격 IP 주소를 가져올 수 있습니다. 그러나 이 IP 주소는 사용자의 IP 주소와 항상 일치하지는 않으며, 사용자가 인터넷에 연결된 방법에 따라 프록시 서버의 IP 주소가 될 수도 있습니다.
- 웹 서버가 로드 밸런서를 통해 인터넷에 연결된 경우, 웹 서버 변수에는 사용자의 IP 주소가 아닌 로드 밸런서의 IP 주소가 포함될 수 있습니다. 이러한 구성에서는 `X-Forwarded-For` HTTP 헤더의 마지막 IP 주소를 사용하는 것이 좋습니다. 이 헤더는 대개 두 개 이상의 IP 주소를 포함하고 있으며, 이들 중 대부분은 프록시나 로드 밸런서용 IP 주소입니다. 목록의 마지막 IP 주소는 사용자의 지리적 위치와 가장 관련성이 높은 주소입니다.

웹 서버가 로드 밸런서에 연결되지 않은 경우, IP 주소 스푸핑을 방지하기 위해 `X-Forwarded-For` 헤더 대신 웹 서버 변수를 사용하는 것이 좋습니다.

# 필드 수준 암호화를 사용하여 민감한 데이터 보호

Amazon CloudFront를 사용하면 HTTPS를 통해 오리진 서버에 대한 종단 간 보안 연결을 적용할 수 있습니다. 필드 레벨 암호화는 추가 보안 레이어를 추가하여 시스템 처리 전체에서 특정 데이터를 보호하고 특정 애플리케이션만 이를 볼 수 있도록 합니다.

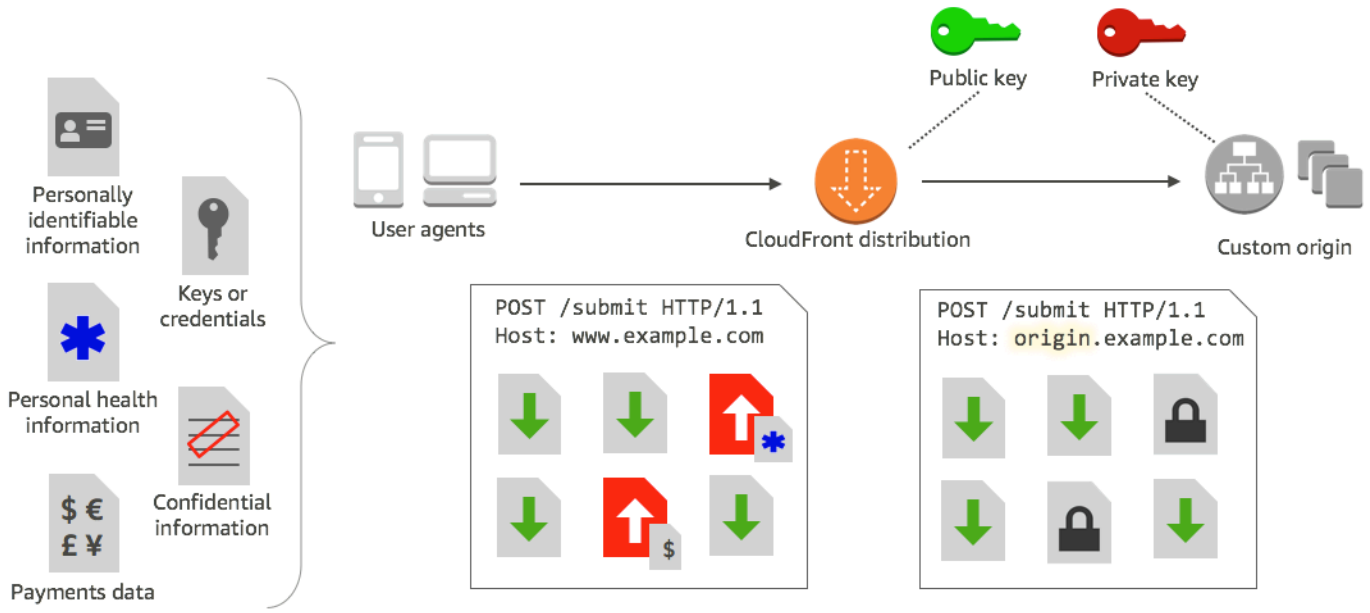
필드 레벨 암호화를 통해 사용자가 민감한 정보를 웹 서버에 안전하게 업로드할 수 있습니다. 사용자가 제공한 민감한 정보는 사용자에게 가까운 엣지에서 암호화되고 전체 애플리케이션 스택에서 암호화를

유지하므로 데이터가 필요하고 이를 해독할 자격 증명을 보유한 애플리케이션만 이 작업을 수행할 수 있습니다.

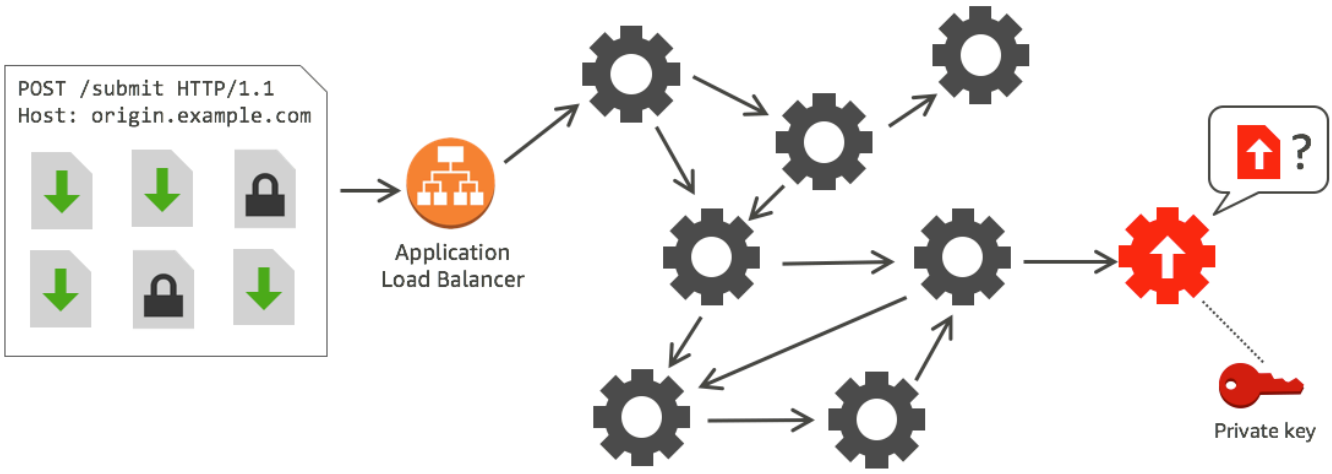
필드 레벨 암호화를 사용하려면 CloudFront 배포를 구성하여 암호화되기를 원하는 POST 요청의 필드 세트와 암호화에 사용할 퍼블릭 키를 지정할 수 있습니다. 한 요청에서 최대 10개의 데이터 필드를 암호화할 수 있습니다. (필드 레벨 암호화된 요청의 모든 데이터를 암호화할 수는 없습니다. 암호화할 개별 필드를 지정해야 합니다.)

필드 레벨 암호화가 포함된 HTTPS 요청이 오리진에 전달될 때 요청은 오리진 애플리케이션 또는 하위 시스템 전체에 라우팅되고, 민감한 데이터는 계속 암호화되어 민감한 데이터의 침해 또는 우발적 데이터 손실의 위험을 줄입니다. 크레딧 번호에 대한 액세스를 필요로 하는 결제 처리 시스템과 같이 업무상 사유에 따라 민감한 데이터에 대한 액세스를 필요로 하는 구성 요소는 적절한 프라이빗 키를 사용하여 해독하고 데이터에 액세스할 수 있습니다.

> **Note**
>
> 필드 레벨 암호화를 사용하려면 오리진이 chunked 인코딩을 지원해야 합니다.




CloudFront 필드 레벨 암호화는 퍼블릭 키 암호화라고도 하는 비대칭 암호화를 사용합니다. CloudFront에 퍼블릭 키를 입력하면 지정한 모든 민감한 데이터가 자동적으로 암호화됩니다. CloudFront에 입력한 키는 암호화된 값의 해독에 사용할 수 없습니다. 이는 프라이빗 키만이 가능합니다.




주제

- 필드 레벨 암호화 개요
- 필드 레벨 암호화 설정
- 오리진의 데이터 필드 해독

## 필드 레벨 암호화 개요

다음 단계는 필드 레벨 암호화 설정의 개요를 제공합니다. 자세한 단계는 필드 레벨 암호화 설정 단원을 참조하세요.

1. 퍼블릭 키-프라이빗 키 페어를 가져옵니다. CloudFront에서 필드 레벨 암호화를 설정하기 전에 퍼블릭 키를 얻고 이를 추가해야 합니다.
2. 필드 레벨 암호화 프로필을 생성합니다. CloudFront에서 생성한 필드 레벨 암호화 프로필은 암호화하고자 하는 필드를 정의합니다.
3. 필드 레벨 암호화 구성을 생성합니다. 구성은 요청의 콘텐츠 유형 또는 쿼리 인수를 기반으로 특정 데이터 필드 암호화에 사용할 프로필을 지정합니다. 여러 시나리오에 대해 원하는 요청 전달 동작 옵션을 선택할 수도 있습니다. 예를 들어 요청 URL의 쿼리 인수로 지정된 프로파일 이름이 CloudFront에 없는 경우에 대한 동작을 설정할 수 있습니다.
4. 캐시 동작에 연결합니다. 배포에 대한 캐시 동작의 구성을 연결하여 CloudFront가 데이터를 암호화할 시기를 지정합니다.

# 필드 레벨 암호화 설정

다음 단계를 따라 필드 레벨 암호화 사용을 시작합니다. 필드 레벨 암호화에 대한 할당량(이전에는 제한이라고 함)에 대한 자세한 내용은 할당량 단원을 참조하세요.

- 1단계: RSA 키 페어 생성
- 2단계: CloudFront에 퍼블릭 키 추가
- 3단계: 필드 레벨 암호화 프로필 생성
- 4단계: 구성 생성
- 5단계: 캐시 동작에 구성 추가

## 1단계: RSA 키 페어 생성

먼저, 퍼블릭 키와 프라이빗 키를 포함하는 RSA 키 페어를 생성해야 합니다. 퍼블릭 키를 사용하면 CloudFront가 데이터를 암호화할 수 있으며, 프라이빗 키를 사용하면 오리진의 구성 요소가 암호화된 필드를 해독할 수 있습니다. OpenSSL 또는 기타 도구를 사용해서 키 페어를 만들 수 있습니다. 키 크기는 2048비트여야 합니다.

예를 들어 OpenSSL을 사용하는 경우, 다음 명령으로 길이가 2048비트인 키 페어를 만들고 `private_key.pem` 파일에 저장할 수 있습니다.

```
openssl genrsa -out private_key.pem 2048
```

이렇게 만들어진 파일은 퍼블릭 키와 프라이빗 키를 모두 포함합니다. 그 파일에서 퍼블릭 키를 추출하려면 다음 명령을 실행합니다.

```
openssl rsa -pubout -in private_key.pem -out public_key.pem
```

퍼블릭 키 파일(`public_key.pem`)에는 다음 단계에서 붙여 넣는 암호화된 키 값이 포함됩니다.

## 2단계: CloudFront에 퍼블릭 키 추가

RSA 키 페어를 얻은 후 퍼블릭 키를 CloudFront에 추가합니다.

CloudFront에 퍼블릭 키를 추가하려면(콘솔)

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.

2. 탐색 창에서 퍼블릭 키를 선택합니다.
3. Add public key(퍼블릭 키 추가)를 선택합니다.
4. 키 이름에 키의 고유 이름을 입력합니다. 이름은 공백을 포함할 수 없으며, 영숫자 문자, 밑줄(_) 및 하이픈(-)만 포함할 수 있습니다. 문자는 최대 128자입니다.
5. Key value(키 값)에 `-----BEGIN PUBLIC KEY-----` 및 `-----END PUBLIC KEY-----` 행을 포함한 퍼블릭 키의 암호화된 키 값을 붙여 넣습니다.
6. 설명에 코멘트를 추가합니다(선택 사항). 예를 들어, 퍼블릭 키의 만료 날짜를 포함할 수 있습니다.
7. 추가를 선택합니다.

이 절차의 단계를 반복하여 CloudFront에서 사용할 키를 더 추가할 수 있습니다.

## 3단계: 필드 레벨 암호화 프로필 생성

CloudFront에 하나 이상의 퍼블릭 키를 추가한 이후 CloudFront에 어떤 필드를 암호화할지 알리는 프로필을 생성합니다.

필드 레벨 암호화 프로필을 생성하려면(콘솔)

1. 왼쪽 탐색 창에서 Field-level encryption(필드 레벨 암호화)을 선택합니다.
2. Create profile(프로파일 생성)을 선택합니다.
3. 다음 필드를 입력합니다.

   프로필 이름

   프로필의 고유한 이름을 입력합니다. 이름은 공백을 포함할 수 없으며, 영숫자 문자, 밑줄(_) 및 하이픈(-)만 포함할 수 있습니다. 문자는 최대 128자입니다.

   퍼블릭 키 이름

   드롭다운 목록에서, 2단계에서 CloudFront에 추가한 퍼블릭 키의 이름을 선택합니다. CloudFront는 키를 사용하여 이 프로필에서 지정한 필드를 암호화합니다.

   공급자 이름

   키 페어를 얻은 공급자와 같이 키를 식별하는 데 도움이 될 구문을 입력합니다. 프라이빗 키와 함께 이 정보는 애플리케이션이 데이터 필드를 해독할 때 필요합니다. 공급자 이름은 공백을 포함할 수 없으며, 영숫자 문자, 콜론(:), 밑줄(_) 및 하이픈(-)만 포함할 수 있습니다. 문자는 최대 128자입니다.

일치시킬 필드 이름 패턴

CloudFront에서 암호화하고자 하는 데이터 필드의 이름 또는 요청에서 데이터 필드 이름을 식별하는 패턴을 입력합니다. + 옵션을 사용하여 이 키를 사용하여 암호화하고자 하는 모든 필드를 추가합니다.

필드 이름 패턴의 경우 데이터 필드의 전체 이름(예: DateOfBirth)을 입력하거나 와일드카드 문자(*)를 포함하여 이름의 첫 부분(예: CreditCard*)만을 입력할 수 있습니다. 필드 이름 패턴은 영숫자 문자, 대괄호([ 및 ]), 마침표(.), 밑줄(_) 및 하이픈(-)만을 포함할 수 있으며, 추가로 와일드카드 문자(*)를 선택적으로 포함할 수 있습니다.

다른 필드 이름 패턴과 중첩되는 문자를 사용하지 마세요. 예를 들어, ABC*라는 필드 이름 패턴을 보유한 경우 또 다른 AB* 필드 이름 패턴을 추가할 수는 없습니다. 또한 필드 이름은 대소문자를 구분하며, 사용할 수 있는 최대 문자는 128자입니다.

Comment

(선택 사항) 이 프로필에 대한 설명을 입력합니다. 사용할 수 있는 최대 문자 수는 128자입니다.

4. 필드에 내용을 입력한 후 Create profile(프로파일 생성)을 선택합니다.
5. 프로필을 더 추가하고자 하는 경우 프로필 추가를 선택합니다.

## 4단계: 구성 생성

하나 이상의 필드 레벨 암호화 프로필을 생성한 이후, 암호화할 데이터를 포함하는 요청의 콘텐츠 유형, 암호화에 사용할 프로필, CloudFront의 암호화 처리 방법을 지정하는 기타 옵션을 지정하는 구성을 생성합니다.

예를 들어, CloudFront가 데이터를 암호화할 수 없을 때 다음 시나리오에서 CloudFront가 요청을 차단하거나 오리진으로 전달할지 여부를 지정할 수 있습니다.

- 요청의 콘텐츠 유형이 구성에 없을 때 – 구성에 콘텐츠 유형을 추가하지 않은 경우 CloudFront가 해당 콘텐츠 유형을 포함한 요청을 데이터 필드 암호화 없이 오리진에 전달할지 아니면 요청을 차단하고 오류를 반환할지 지정할 수 있습니다.

> **Note**
>
> 구성에 콘텐츠 유형을 추가했지만 해당 유형을 사용할 프로필을 지정하지 않은 경우 CloudFront에서 해당 콘텐츠 유형을 포함한 요청을 항상 오리진으로 전달합니다.

- 쿼리 인수에 입력된 프로필 이름이 알 수 없는 이름임 – 배포에 존재하지 않는 프로필 이름을 포함하는 `fle-profile` 쿼리 인수를 지정할 때 CloudFront가 요청을 데이터 필드 암호화 없이 오리진으로 보낼지 아니면 요청을 차단하고 오류를 반환할지 지정할 수 있습니다.

구성에서 URL에 쿼리 인수로 프로필을 입력함으로써 해당 쿼리에 대한 콘텐츠 유형으로 매핑한 프로필을 무시할지 여부를 지정할 수도 있습니다. 기본적으로 CloudFront는 콘텐츠 유형에 매핑한 프로필을 사용합니다(이를 지정한 경우). 이를 통해 기본적으로 사용할 프로필을 가질 수 있지만 다른 프로필을 적용하고자 하는 특정 요청을 정할 수도 있습니다.

예를 들어, 사용할 쿼리 인수로 (구성에 있는) **SampleProfile**을 지정할 수 있습니다. 그런 다음, `https://d1234.cloudfront.net?fle-profile=SampleProfile` 대신 URL `https://d1234.cloudfront.net`을 사용하여 CloudFront가 요청의 콘텐츠 유형에 대해 설정한 프로필 대신 이 요청에서 **SampleProfile**을 사용하도록 할 수 있습니다.

단일 계정에 대해 최대 10개의 구성을 생성할 수 있으며 계정에 대한 모든 배포의 캐시 동작에 구성 중 하나를 연결할 수 있습니다.

필드 레벨 암호화 구성을 생성하려면(콘솔)

1. Field-level encryption(필드 레벨 암호화) 페이지에서 구성 만들기를 선택합니다.

   참고: 최소 하나의 프로필을 생성하지 않은 경우 구성 생성 옵션이 표시되지 않습니다.
2. 다음 필드를 입력하여 사용할 프로필을 지정합니다. (일부 필드는 변경할 수 없습니다.)

   콘텐츠 유형(변경할 수 없음)

   콘텐츠 유형은 `application/x-www-form-urlencoded`로 설정되며 변경할 수 없습니다.

   기본 프로필 ID(선택 사항)

   드롭다운 목록 가운데 콘텐츠 유형 필드에서 콘텐츠 유형을 매핑하고자 하는 프로필을 선택합니다.

   콘텐츠 형식(변경할 수 없음)

   콘텐츠 형식은 `URLencoded`로 설정되며 변경할 수 없습니다.
3. 다음 옵션에 대한 CloudFront 기본 동작을 변경하고자 하는 경우 해당 확인란을 선택합니다.

요청의 콘텐츠 유형이 구성되지 않을 때 오리진으로 요청 전달

요청의 콘텐츠 유형에 대해 사용할 프로필을 지정하지 않았다면 요청이 오리진으로 이동하도록 허용할 경우 확인란을 선택합니다.

입력된 쿼리 인수를 포함한 콘텐츠 유형에 대한 프로필 무시

쿼리 인수에 입력된 프로필이 콘텐츠 유형에 대해 지정한 프로필을 무시하도록 허용할 경우 확인란을 선택합니다.

4. 확인란을 선택하여 쿼리 인수가 기본 프로필을 무시하도록 허용할 경우 구성에 대한 다음 추가 필드를 입력해야 합니다. 쿼리에 사용할 쿼리 인수 매핑 가운데 최대 5개를 생성할 수 있습니다.

   쿼리 인수

   `fle-profile` 쿼리 인수에 대해 URL에서 포함하고자 하는 값을 입력합니다. 이 값은 CloudFront에게 이 쿼리의 필드 레벨 암호화에 대한 쿼리 인수와 연결된 프로필 ID(다음 필드에서 지정)를 사용하도록 합니다.

   사용할 수 있는 최대 문자 수는 128자입니다. 값에는 공백이 포함될 수 없으며, 영숫자 문자와 다음 문자만을 사용할 수 있습니다. 대시(-), 마침표(.), 밑줄(_), 별표(*) 더하기 기호(+), 퍼센트(%).

   프로필 ID

   드롭다운 목록 가운데 쿼리 인수에 입력한 값과 연결하고자 하는 프로필을 선택합니다.

   쿼리 인수에 지정된 프로필이 존재하지 않을 때 요청을 오리진에 전달

   쿼리 인수에서 지정된 프로필이 CloudFront에서 정의되지 않은 경우 요청이 오리진으로 이동하도록 허용할 경우 확인란을 선택합니다.

## 5단계: 캐시 동작에 구성 추가

필드 레벨 암호화를 사용하려면 배포에 대한 값으로 구성 ID를 추가함으로써 배포에 대한 캐시 동작과 구성을 연결합니다.

> **Important**
>
> 필드 레벨 암호화 구성을 캐시 동작에 연결하려면 항상 HTTPS를 사용하고 최종 사용자의 HTTP POST 및 PUT 요청을 수락하도록 배포를 구성해야 합니다. 즉, 다음 조건이 충족되어야 합니다.

- 캐시 동작의 Viewer Protocol Policy(최종 사용자 프로토콜 정책)을 Redirect HTTP to HTTPS(HTTP에서 HTTPS로 리디렉션) 또는 HTTPS Only(HTTPS만)로 설정해야 합니다. (CloudFormation 또는 CloudFront API에서는 `ViewerProtocolPolicy`를 `redirect-to-https` 또는 `https-only`로 설정해야 합니다.)
- 캐시 동작의 Allowed HTTP Methods(허용되는 HTTP 메서드)를 GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE로 설정해야 합니다. (CloudFormation 또는 CloudFront API에서 `AllowedMethods`를 `GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE`로 설정해야 합니다. 이러한 값은 임의의 순서로 지정할 수 있습니다.)
- 오리진 설정의 Origin Protocol Policy(오리진 프로토콜 정책)를 Match Viewer(최종 사용자와 일치) 또는 HTTPS Only(HTTPS만)로 설정해야 합니다. (CloudFormation 또는 CloudFront API에서는 `OriginProtocolPolicy`를 `match-viewer` 또는 `https-only`로 설정해야 합니다.)

자세한 내용은 모든 배포 설정 참조 섹션을 참조하세요.

## 오리진의 데이터 필드 해독

CloudFront는 AWS Encryption SDK를 사용하여 데이터 필드를 암호화합니다. 데이터는 애플리케이션 스택에 걸쳐 암호화를 유지하며, 이를 해독할 자격 증명을 보유한 애플리케이션만이 액세스할 수 있습니다.

암호화 이후 암호 텍스트는 base64 암호화됩니다. 애플리케이션이 오리진의 텍스트를 해독할 때 우선 암호 텍스트를 디코딩한 다음 AWS 암호화 SDK를 사용하여 데이터를 복호화해야 합니다.

다음 코드 예제는 애플리케이션이 오리진의 데이터를 해독하는 방법을 보여줍니다. 다음을 참조하세요.

- 예제를 간소화하기 위해 이 샘플은 작업 디렉터리에 있는 파일에서 퍼블릭 및 프라이빗 키(DER 형식)를 로드합니다. 실제로는 프라이빗 키를 오프라인 하드웨어 보안 모듈과 같은 안전한 오프라인 위치에 저장하고, 퍼블릭 키를 개발 팀에 배포합니다.
- CloudFront는 데이터를 암호화하는 동안 특정 정보를 사용하며 동일한 파라미터 세트를 오리진에서 사용하여 이를 해독해야 합니다. MasterKey를 초기화하는 동안 CloudFront가 사용하는 파라미터는 다음을 포함합니다.
  - PROVIDER_NAME: 필드 레벨 암호화 프로필을 생성할 때 이 값을 지정했습니다. 여기에서 동일한 값을 사용합니다.

- KEY_NAME: 퍼블릭 키를 CloudFront로 업로드할 때 퍼블릭 키의 이름을 생성했으며, 프로필에서 키 이름을 지정했습니다. 여기에서 동일한 값을 사용합니다.
- ALGORITHM: CloudFront는 `RSA/ECB/OAEPWithSHA-256AndMGF1Padding`을 암호화 알고리즘으로 사용합니다. 따라서 동일한 알고리즘을 사용하여 데이터를 해독해야 합니다.

- 입력으로 암호 텍스트를 포함한 다음 샘플 프로그램을 실행하는 경우 암호화된 데이터가 콘솔에 출력됩니다. 자세한 내용은 AWS 암호화 SDK의 Java 예제 코드를 참조하세요.

## 샘플 코드

```
import java.nio.file.Files;
import java.nio.file.Paths;
import java.security.KeyFactory;
import java.security.PrivateKey;
import java.security.PublicKey;
import java.security.spec.PKCS8EncodedKeySpec;
import java.security.spec.X509EncodedKeySpec;

import org.apache.commons.codec.binary.Base64;

import com.amazonaws.encryptionsdk.AwsCrypto;
import com.amazonaws.encryptionsdk.CryptoResult;
import com.amazonaws.encryptionsdk.jce.JceMasterKey;

/**
 * Sample example of decrypting data that has been encrypted by CloudFront field-level
 encryption.
 */
public class DecryptExample {

    private static final String PRIVATE_KEY_FILENAME = "private_key.der";
    private static final String PUBLIC_KEY_FILENAME = "public_key.der";
    private static PublicKey publicKey;
    private static PrivateKey privateKey;

    // CloudFront uses the following values to encrypt data, and your origin must use
 same values to decrypt it.
    // In your own code, for PROVIDER_NAME, use the provider name that you specified
 when you created your field-level
    // encryption profile. This sample uses 'DEMO' for the value.
    private static final String PROVIDER_NAME = "DEMO";
```

```
    // In your own code, use the key name that you specified when you added your public
key to CloudFront. This sample
    // uses 'DEMOKEY' for the key name.
    private static final String KEY_NAME = "DEMOKEY";
    // CloudFront uses this algorithm when encrypting data.
    private static final String ALGORITHM = "RSA/ECB/OAEPWithSHA-256AndMGF1Padding";

    public static void main(final String[] args) throws Exception {

        final String dataToDecrypt = args[0];

        // This sample uses files to get public and private keys.
        // In practice, you should distribute the public key and save the private key
in secure storage.
        populateKeyPair();

        System.out.println(decrypt(debase64(dataToDecrypt)));
    }

    private static String decrypt(final byte[] bytesToDecrypt) throws Exception {
        // You can decrypt the stream only by using the private key.

        // 1. Instantiate the SDK
        final AwsCrypto crypto = new AwsCrypto();

        // 2. Instantiate a JCE master key
        final JceMasterKey masterKey = JceMasterKey.getInstance(
                publicKey,
                privateKey,
                PROVIDER_NAME,
                KEY_NAME,
                ALGORITHM);

        // 3. Decrypt the data
        final CryptoResult <byte[], ? > result = crypto.decryptData(masterKey,
bytesToDecrypt);
        return new String(result.getResult());
    }

    // Function to decode base64 cipher text.
    private static byte[] debase64(final String value) {
        return Base64.decodeBase64(value.getBytes());
    }
```

```
    private static void populateKeyPair() throws Exception {
        final byte[] PublicKeyBytes =
 Files.readAllBytes(Paths.get(PUBLIC_KEY_FILENAME));
        final byte[] privateKeyBytes =
 Files.readAllBytes(Paths.get(PRIVATE_KEY_FILENAME));
        publicKey = KeyFactory.getInstance("RSA").generatePublic(new
 X509EncodedKeySpec(PublicKeyBytes));
        privateKey = KeyFactory.getInstance("RSA").generatePrivate(new
 PKCS8EncodedKeySpec(privateKeyBytes));
    }
}
```

# CloudFront를 사용한 온디맨드 비디오 및 라이브 스트리밍 비디오

CloudFront를 사용하여 HTTP 오리진에서 온디맨드 비디오(VOD) 또는 라이브 스트리밍 비디오를 제공할 수 있습니다. 클라우드에서 비디오 워크플로를 설정하는 한 가지 방법은 CloudFront를 AWS Media Services와 함께 사용하는 것입니다.

주제


- 스트리밍 비디오 정보
- CloudFront를 사용한 온디맨드 비디오 제공
- CloudFront 및 AWS Media Services를 사용하여 동영상 스트리밍 제공
- 미디어 품질 인식 복원력


## 스트리밍 비디오 정보

CloudFront에서 콘텐츠를 배포하려면 먼저 인코더를 사용하여 비디오 콘텐츠를 패키징해야 합니다. 패키징 프로세스에서는 오디오, 비디오 및 캡션 콘텐츠를 포함하는 세그먼트가 생성됩니다. 또한 특정 순서로 재생할 세그먼트와 시점을 설명하는 매니페스트 파일도 생성됩니다. 일반적인 패키지 포맷은 MPEG DASH, Apple HLS, Microsoft Smooth Streaming, CMAF입니다.

VOD 스트리밍

VOD 스트리밍의 경우, 비디오 콘텐츠가 서버에 저장되고 뷰어가 언제든지 비디오 콘텐츠를 볼 수 있습니다. 최종 사용자가 스트리밍할 수 있는 애셋을 만들려면 AWS Elemental MediaConvert와 같은 인코더를 사용하여 미디어 파일을 포맷하고 패키징합니다.

비디오를 올바른 포맷으로 패키징했으면 서버나 Amazon S3 버킷에 저장한 후 최종 사용자들이 요청할 때 CloudFront를 사용하여 제공할 수 있습니다.

라이브 비디오 스트리밍

라이브 비디오 스트리밍의 경우 비디오 콘텐츠가 라이브 이벤트 발생 시 실시간으로 스트리밍되거나 24x7 라이브 채널로 설정됩니다. 브로드캐스트 및 스트리밍 제공을 위한 라이브 출력을 만들려면 AWS Elemental MediaLive와 같은 인코더를 사용하여 비디오를 압축하고 시청 디바이스에 맞게 포맷합니다.

비디오를 인코딩한 후, AWS Elemental MediaStore에 저장하거나 AWS Elemental MediaPackage를 사용하여 다른 전송 형식으로 변환할 수 있습니다. 이러한 오리진을 사용하여 콘텐츠를 제공하도록 CloudFront 배포를 설정합니다. 이러한 서비스와 함께 사용할 배포를 만드는 절차와 지침을 보려면 AWS Elemental MediaStore를 오리진으로 사용하여 비디오를 제공합니다. 및 AWS Elemental MediaPackage를 사용하여 포맷된 라이브 비디오 제공 단원을 참조하십시오.

Wowza 및 Unified Streaming에서도 CloudFront를 통해 비디오를 스트리밍하는 데 사용할 수 있는 도구를 제공합니다. Wowza를 CloudFront와 함께 사용하는 방법은 Wowza 설명서 웹 사이트에서 Bring your Wowza Streaming Engine license to CloudFront live HTTP streaming(Wowza 스트리밍 엔진 라이선스를 CloudFront 라이브 HTTP 스트리밍으로 가져오기)을 참조하십시오. VOD 스트리밍에 CloudFront와 함께 Unified Streaming을 사용하는 방법에 대한 자세한 내용은 Unified Streaming 설명서 웹 사이트의 CloudFront를 참조하세요.

## CloudFront를 사용한 온디맨드 비디오 제공

CloudFront를 통해 온디맨드 비디오(VOD) 스트리밍을 제공하려면 다음 서비스를 사용합니다.

- Amazon S3를 사용하여 콘텐츠를 원본 포맷으로 저장하고 트랜스코딩된 비디오를 저장합니다.
- 인코더(예: AWS Elemental MediaConvert)를 사용하여 비디오를 스트리밍 포맷으로 트랜스코딩.
- CloudFront를 사용하여 트랜스코딩된 비디오를 최종 사용자에게 제공합니다. Microsoft Smooth Streaming을 사용하려면 Microsoft Smooth Streaming용 온디맨드 비디오 구성 단원을 참조하십시오.

CloudFront를 사용하여 VOD 솔루션을 생성하려면

1. Amazon S3 버킷에 콘텐츠를 업로드합니다. Amazon S3 사용에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서를 참조하세요.
2. MediaConvert 작업을 사용하여 콘텐츠를 트랜스코딩합니다. 이 작업은 최종 사용자들이 사용할 플레이어에 필요한 포맷으로 비디오를 변환합니다. 작업을 사용하여 다양한 해상도와 비트레이트를 갖는 애셋을 만들 수도 있습니다. 이러한 애셋은 최종 사용자가 사용 가능한 대역폭에 따라 화질을 조정하는 가변 비트레이트(ABR) 스트리밍에 사용됩니다. MediaConvert 는 트랜스코딩된 비디오를 S3 버킷에 저장합니다.
3. CloudFront 배포를 사용하여 변환된 콘텐츠를 제공합니다. 최종 사용자는 언제든지 모든 디바이스에서 콘텐츠를 볼 수 있습니다.

# Microsoft Smooth Streaming용 온디맨드 비디오 구성

다음과 같은 방법으로 CloudFront를 사용하여 Microsoft Smooth Streaming 포맷으로 트랜스코딩한 온디맨드 비디오(VOD) 콘텐츠를 배포할 수 있습니다.

- Microsoft IIS를 실행하고 Smooth Streaming을 배포 오리진으로 지원하는 웹 서버를 지정합니다.
- CloudFront 배포의 캐시 동작에서 Smooth Streaming을 활성화합니다. 배포에 여러 캐시 동작을 사용할 수 있기 때문에 하나의 배포를 Smooth Streaming 미디어 파일뿐 아니라 다른 콘텐츠에도 사용할 수 있습니다.

> **Important**
>
> Microsoft IIS를 실행하는 웹 서버를 오리진으로 지정하는 경우 CloudFront 배포의 캐시 동작에서 Smooth Streaming을 활성화하지 마십시오. Smooth Streaming을 캐시 동작으로 활성화하는 경우 CloudFront는 Microsoft IIS 서버를 오리진으로 사용할 수 없습니다.

캐시 동작에서 Smooth Streaming을 활성화할 경우, 즉 Microsoft IIS를 실행하는 서버를 사용하지 않을 경우 다음 사항에 유의하십시오.

- 콘텐츠가 해당 캐시 동작의 경로 패턴 값과 일치하는 경우 동일한 캐시 동작을 사용하여 다른 콘텐츠를 배포할 수 있습니다.
- CloudFront는 Smooth Streaming 미디어 파일에 Amazon S3 버킷 또는 사용자 지정 오리진을 사용할 수 있습니다. 캐시 동작에 대해 Smooth Streaming을 활성화하는 경우 CloudFront는 Microsoft IIS 서버를 오리진으로 사용할 수 없습니다.
- Smooth Streaming 포맷의 미디어 파일은 무효화할 수 없습니다. 만료되기 전에 파일을 업데이트하려는 경우 파일의 이름을 변경해야 합니다. 자세한 내용은 CloudFront가 배포하는 콘텐츠 추가, 제거 또는 교체 섹션을 참조하세요.

Smooth Streaming 클라이언트에 대한 자세한 내용은 Microsoft 설명서 웹 사이트의 Smooth Streaming를 참조합니다.

Microsoft IIS 웹 서버가 오리진이 아닌 경우 CloudFront를 사용하여 Smooth Streaming 파일을 배포하려면

1. 미디어 파일을 Smooth Streaming fragmented-MP4 포맷으로 트랜스코딩합니다.
2. 다음 중 하나를 수행하십시오.

- CloudFront 콘솔을 사용하는 경우: 배포를 생성하거나 업데이트할 때 하나 이상의 배포 캐시 동작에서 Smooth Streaming을 활성화합니다.
- CloudFront API를 사용하는 경우: 하나 이상의 배포 캐시 동작에 대해 `SmoothStreaming` 요소를 `DistributionConfig` 복합 형식에 추가합니다.

3. Smooth Streaming 파일을 오리진에 업로드합니다.
4. `clientaccesspolicy.xml` 또는 `crossdomainpolicy.xml` 파일을 만들고 이 파일을 배포의 루트에서 액세스할 수 있는 위치(예: `https://d111111abcdef8.cloudfront.net/clientaccesspolicy.xml`)에 추가합니다. 다음은 정책의 예입니다.

```
<?xml version="1.0" encoding="utf-8"?>
<access-policy>
<cross-domain-access>
<policy>
<allow-from http-request-headers="*">
<domain uri="*"/>
</allow-from>
<grant-to>
<resource path="/" include-subpaths="true"/>
</grant-to>
</policy>
</cross-domain-access>
</access-policy>
```

   자세한 내용은 Microsoft Developer Network 웹 사이트의 Making a Service Available Across Domain Boundaries를 참조하십시오.

5. 애플리케이션(예: 미디어 플레이어)의 링크로, 미디어 파일의 URL을 다음 형식으로 지정합니다.

   `https://d111111abcdef8.cloudfront.net/video/presentation.ism/Manifest`

# CloudFront 및 AWS Media Services를 사용하여 동영상 스트리밍 제공

AWS Media Services와 CloudFront를 사용하여 전 세계 시청자에게 라이브 콘텐츠를 제공하려면 다음 지침을 참조하세요.

AWS Elemental MediaLive를 사용하여 라이브 비디오 스트림을 실시간으로 인코딩합니다. 큰 비디오 스트림을 인코딩하는 경우 MediaLive에서 최종 사용자에게 배포할 수 있는 더 작은 버전(인코딩)으로 압축합니다.

라이브 비디오 스트림을 압축한 후 다음 두 가지 주요 방법 중 하나를 사용하여 콘텐츠를 준비하고 제공할 수 있습니다.

- 필요한 포맷으로 콘텐츠를 변환한 후 제공 - 여러 포맷의 콘텐츠가 필요한 경우 AWS Elemental MediaPackage를 사용하여 다양한 디바이스 유형에 맞게 콘텐츠를 패키징합니다. 콘텐츠를 패키징할 때 추가 기능을 구현하고 콘텐츠의 무단 사용을 방지하도록 DRM(Digital Rights Management)을 추가할 수도 있습니다. CloudFront를 사용하여 MediaPackage가 포맷한 콘텐츠를 제공하는 방법에 대한 단계별 지침은 AWS Elemental MediaPackage를 사용하여 포맷된 라이브 비디오 제공 단원을 참조하십시오.
- 확장 가능 오리진을 사용하여 콘텐츠를 저장하고 제공 - MediaLive에서 뷰어가 사용하는 모든 디바이스에 필요한 포맷으로 콘텐츠를 인코딩한 경우, AWS Elemental MediaStore와 같이 확장성이 우수한 오리진을 사용하여 콘텐츠를 제공할 수 있습니다. CloudFront를 사용하여 MediaStore 컨테이너에 저장된 콘텐츠를 제공하는 방법에 대한 단계별 지침은 AWS Elemental MediaStore를 오리진으로 사용하여 비디오를 제공합니다. 단원을 참조하십시오.

이러한 옵션 중 하나를 사용하여 오리진을 설정했으면 이제 CloudFront를 사용하여 최종 사용자에게 라이브 스트리밍 비디오를 배포할 수 있습니다.

> **Tip**
>
> 가용성이 높은 실시간 시청 환경을 제공하기 위해 자동으로 서비스를 배포하는 AWS 솔루션에 대한 자세한 내용을 알아볼 수 있습니다. 이 솔루션을 자동으로 배포하는 절차를 보려면 라이브 스트리밍 자동 배포를 참조하십시오.

주제

- AWS Elemental MediaStore를 오리진으로 사용하여 비디오를 제공합니다.
- AWS Elemental MediaPackage를 사용하여 포맷된 라이브 비디오 제공
- AWS Elemental MediaPackage로 동영상 온디맨드 콘텐츠 제공

# AWS Elemental MediaStore를 오리진으로 사용하여 비디오를 제공합니다.

AWS Elemental MediaStore 컨테이너에 저장된 비디오가 있으면 CloudFront 배포를 만들어 콘텐츠를 제공할 수 있습니다.

시작하려면 CloudFront에 MediaStore 컨테이너에 대한 액세스 권한을 부여합니다. 그런 다음 CloudFront 배포를 생성하고 MediaStore에서 작동하도록 구성합니다.

AWS Elemental MediaStore 컨테이너에서 콘텐츠를 제공하려면

1. Amazon CloudFront가 AWS Elemental MediaStore 컨테이너에 액세스하도록 허용의 절차에 따라 이 단계로 돌아가서 배포를 생성합니다.
2. 다음 설정을 사용하여 배포를 생성합니다.

   a. 오리진 도메인 - MediaStore 컨테이너에 할당된 데이터 엔드포인트. 드롭다운 목록에서 해당 라이브 비디오의 MediaStore 컨테이너를 선택합니다.

   b. 오리진 경로 - 객체가 저장되는 MediaStore 컨테이너의 폴더 구조. 자세한 내용은 the section called “오리진 경로” 섹션을 참조하세요.

   c. 사용자 지정 헤더 추가 - 요청을 오리진으로 전달할 때 CloudFront에 사용자 지정 헤더를 추가하고자 하는 경우 헤더 이름과 값을 추가합니다.

   d. 뷰어 프로토콜 정책 - HTTP를 HTTPS로 리디렉션을 선택합니다. 자세한 내용은 the section called “뷰어 프로토콜 정책” 섹션을 참조하세요.

   e. 캐시 정책 및 오리진 요청 정책

      - 캐시 정책(Cache policy)에서 정책 생성(Create policy)을 선택한 그런 다음 캐시 요구 및 세그먼트 기간에 적합한 캐시 정책을 생성합니다. 정책을 생성한 후 캐시 정책의 목록을 새로 고치고 방금 생성한 정책을 선택합니다.
      - 오리진 요청 정책(Origin request policy)의 드롭다운 목록에서 CORS-CustomOrigin을 선택합니다.

   기타 설정의 경우 다른 기술 요구 사항 또는 비즈니스 요구 사항을 기반으로 특정 값을 설정할 수 있습니다. 배포에 대한 모든 옵션 목록과 설정에 대한 정보는 the section called “모든 배포 설정” 단원을 참조하십시오.
3. 미디어 플레이어 등 애플리케이션의 링크에 대해 CloudFront를 사용하여 배포한 다른 객체에 사용한 것과 같은 포맷으로 미디어 파일의 이름을 지정합니다.

# AWS Elemental MediaPackage를 사용하여 포맷된 라이브 비디오 제공

AWS Elemental MediaPackage를 사용하여 라이브 스트림을 포맷한 경우 CloudFront 배포를 만들고 라이브 스트림을 제공하도록 캐시 동작을 구성하면 됩니다. 다음 프로세스에서는 이미 MediaPackage를 사용하여 라이브 비디오를 위한 채널을 만들었고 엔드포인트를 추가했다는 전제하에 설명을 진행합니다.

MediaPackage용 CloudFront 배포를 수동으로 생성하려면 다음 단계를 수행하십시오.

1단계: CloudFront 배포 생성 및 구성

MediaPackage를 사용하여 생성한 라이브 비디오 채널을 위한 CloudFront 배포를 설정하려면 다음 절차를 완료합니다.

라이브 비디오 채널을 위한 배포를 만들려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포 생성을 선택합니다.
3. 다음과 같이 배포를 위한 설정을 선택합니다.

   오리진 도메인

   MediaPackage 라이브 비디오 채널과 엔드포인트가 있는 오리진입니다. 텍스트 필드를 선택한 다음 드롭다운 목록에서 해당 라이브 비디오의 MediaPackage 오리진 도메인을 선택합니다. 한 개의 도메인을 여러 오리진 엔드포인트에 매핑할 수 있습니다.

   다른 AWS 계정을 사용하여 오리진 도메인을 만든 경우 필드에 해당 오리진 URL 값을 입력합니다. 오리진은 HTTPS URL이어야 합니다.

   예를 들어 `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.m3u8`과 같은 HLS 엔드포인트의 경우, 오리진 도메인은 `3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com`입니다.

   자세한 내용은 the section called “오리진 도메인” 섹션을 참조하세요.

   오리진 경로

   콘텐츠가 제공되는 MediaPackage 엔드포인트의 경로입니다.

오리진 경로의 작동 방식에 대한 자세한 내용은 the section called "오리진 경로" 단원을 참조하세요.

> **Important**
>
> CloudFront 배포의 어딘가로 라우팅하려면 와일드카드 경로 *가 필요합니다. 명시적 경로와 일치하지 않는 요청이 실제 오리진으로 라우팅되지 않도록 하려면 해당 와일드카드 경로에 대해 '더미' 오리진을 만듭니다.

Example: '더미' 오리진 생성

다음 예에서는 엔드포인트 `abc123` 및 `def456`가 '실제' 오리진으로 라우팅되지만 다른 엔드포인트의 비디오 콘텐츠에 대한 요청은 적절한 하위 도메인 없이 `mediapackage.us-west-2.amazonaws.com`으로 라우팅되어 HTTP `404` 오류가 발생합니다.

MediaPackage 엔드포인트:

```
https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/
index.m3u8
https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/def456/
index.m3u8
```

CloudFront 오리진 A:

```
Domain: 3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com
Path: None
```

CloudFront 오리진 B:

```
Domain: mediapackage.us-west-2.amazonaws.com
Path: None
```

CloudFront 캐시 동작:

```
1. Path: /out/v1/abc123/* forward to Origin A
2. Path: /out/v1/def456/* forward to Origin A
```

```
3. Path: * forward to Origin B
```

그밖의 배포 설정에 대해, 다른 기술 요구 사항 또는 비즈니스 요구 사항을 기반으로 특정 값을 설정합니다. 배포에 대한 모든 옵션 목록과 설정에 대한 정보는 the section called “모든 배포 설정” 단원을 참조하십시오.

다른 배포 설정 선택을 마치면 배포 생성(Create Distribution)을 선택합니다.

4. 방금 생성한 배포를 선택한 다음 동작(Behaviors)을 선택합니다.
5. 기본 캐시 동작을 선택한 다음 편집(Edit)을 선택합니다. 오리진에 대해 선택한 채널에 적합한 캐시 동작 설정을 지정합니다. 나중에 한 개 이상의 오리진을 추가한 후 그에 대한 캐시 동작 설정을 편집해 보겠습니다.
6. CloudFront 배포(CloudFront distributions) 페이지로 이동합니다.
7. 배포의 마지막 수정(Last modified) 열 값이 배포 중(Deploying)에서 날짜 및 시간으로 바뀔 때까지 기다립니다. 이렇게 상태가 바뀌면 CloudFront에서 배포를 생성한 것입니다.

2단계: MediaPackage 엔드포인트의 도메인에 오리진 추가

다음 단계를 반복하여 각 MediaPackage 채널 엔드포인트를 배포에 추가합니다. '더미' 오리진을 만들어야 한다는 점에 유의하세요.

다른 엔드포인트를 오리진으로 추가하려면

1. CloudFront 콘솔에서 채널에 대해 생성한 배포를 선택합니다.
2. 오리진(Origins)을 선택한 다음 오리진 생성(Create origin)을 선택합니다.
3. 오리진 도메인(Origin domain)의 드롭다운 목록에서 해당 채널의 MediaPackage 엔드포인트를 선택합니다.
4. 그밖의 설정에 대해, 다른 기술 요구 사항 또는 비즈니스 요구 사항을 기반으로 값을 설정합니다. 자세한 내용은 the section called “오리진 설정” 섹션을 참조하세요.
5. 오리진 생성(Create Origin)을 선택합니다.

3단계: 모든 엔드포인트에 대한 캐시 동작 구성

각 엔드포인트에 대해 요청을 올바르게 라우팅하는 경로 패턴을 추가하도록 캐시 동작을 구성해야 합니다. 지정하는 경로 패턴은 제공하는 비디오 포맷에 따라 다릅니다. 다음 절차에는 Apple HLS, CMAF, DASH, Microsoft Smooth Streaming 포맷에 대해 사용할 경로 패턴 정보가 나와 있습니다.

일반적으로 각 엔드포인트에 대해 두 개의 캐시 동작을 설정합니다.

- 패턴 매니페스트(해당 파일에 대한 인덱스)
- 세그먼트(비디오 콘텐츠의 파일)

엔드포인트에 대한 캐시 동작을 만들려면

1. CloudFront 콘솔에서 채널에 대해 생성한 배포를 선택합니다.
2. 동작(Behaviors)을 선택한 다음 동작 만들기(Create behavior)를 선택합니다.
3. 경로 패턴에서 특정 MediaPackage `OriginEndpoint` GUID를 경로 접두사로 사용합니다.

   경로 패턴

   `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.m3u8`과 같은 HLS 엔드포인트의 경우, 다음 두 가지 캐시 동작을 만듭니다.

   - 상위 및 하위 매니페스트에 대해 `/out/v1/abc123/*.m3u8`을 사용합니다.
   - 콘텐츠 세그먼트에 대해 `/out/v1/abc123/*.ts`를 사용합니다.

   `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.m3u8`과 같은 CMAF 엔드포인트의 경우, 다음 두 가지 캐시 동작을 만듭니다.

   - 상위 및 하위 매니페스트에 대해 `/out/v1/abc123/*.m3u8`을 사용합니다.
   - 콘텐츠 세그먼트에 대해 `/out/v1/abc123/*.mp4`를 사용합니다.

   `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.mpd`와 같은 DASH 엔드포인트의 경우, 다음 두 가지 캐시 동작을 만듭니다.

   - 상위 매니페스트에 대해 `/out/v1/abc123/*.mpd`를 사용합니다.
   - 콘텐츠 세그먼트에 대해 `/out/v1/abc123/*.mp4`를 사용합니다.

   `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.ism`과 같은 Microsoft Smooth Streaming 엔드포인트의 경우, 한 개의 매니페스트만 제공되므로 캐시 동작을 `out/v1/abc123/index.ism/*` 하나만 만듭니다.
4. 각 캐시 동작에 대해 다음 설정의 값을 지정합니다.

   뷰어 프로토콜 정책

   Redirect HTTP to HTTPS(HTTP를 HTTPS로 재지정)를 선택합니다.

캐시 정책 및 오리진 요청 정책

캐시 정책(Cache policy)에서 정책 생성(Create policy)을 선택합니다. 새 캐시 정책에 대해 다음 설정을 지정합니다.

Minimum TTL

5초 미만으로 설정하면 부실한 콘텐츠 서비스 방지에 도움이 됩니다.

쿼리 문자열

쿼리 문자열(Query strings)(캐시 키 설정(Cache key settings)에 있음)에서 지정된 쿼리 문자열 포함(Include specified query strings)을 선택합니다. 허용(Allow)에서 다음 값을 입력하고 항목 추가(Add item)를 선택하여 값을 추가합니다.

- CloudFront가 캐시 기반으로 사용할 쿼리 문자열 파라미터로 `m`을 추가합니다. MediaPackage 응답에는 항상 `?m=###` 태그가 포함되어 엔드포인트의 수정 시간을 캡처합니다. 이 태그에 대해 다른 값을 사용하여 이미 콘텐츠가 캐싱된 경우 CloudFront는 캐싱된 버전을 제공하는 대신 새로운 매니페스트를 요청합니다.
- MediaPackage에서 시간 이동 보기 기능을 사용하려는 경우, 매니페스트 요청(`start`, `end` 및 `*.m3u8`)에 대한 캐시 동작에 `*.mpd` 및 `index.ism/*`를 추가 쿼리 문자열 파라미터로 지정합니다. 이렇게 하면 매니페스트 요청에서 요청된 시간 기간에 해당하는 콘텐츠가 제공됩니다. 시간 이동 보기 및 콘텐츠 시작/종료 요청 파라미터 포맷에 대한 자세한 내용은 AWS Elemental MediaPackage 사용 설명서의 시간 이동 보기를 참조하세요.
- MediaPackage에서 매니페스트 필터링 기능을 사용하는 경우, 매니페스트 요청(`*.m3u8`, `*.mpd`, `index.ism/*`)에 대한 캐시 동작에 사용할 캐시 정책에 `aws.manifestfilter`를 추가 쿼리 문자열 파라미터로 지정합니다. 이렇게 하면 매니페스트 필터링 기능이 작동하는 데 필요한 MediaPackage 오리진에 `aws.manifestfilter` 쿼리 문자열을 전달하도록 배포가 구성됩니다. 자세한 내용은 AWS Elemental MediaPackage 사용 설명서의 매니페스트 필터링을 참조하세요.
- 지연 시간이 짧은 HLS(LL-HLS)를 사용하는 경우 매니페스트 요청(`*.m3u8`)의 캐시 동작과 함께 사용하는 캐시 정책에 대한 추가 쿼리 문자열 파라미터로 `_HLS_msn` 및 `_HLS_part`를 지정합니다. 이렇게 하면 LL-HLS 차단 재생 목록 기능이 작동하는 데 필요한 MediaPackage 오리진에 `_HLS_msn` 및 `_HLS_part` 쿼리 문자열을 전달하도록 배포가 구성됩니다.

5. 생성(Create)을 선택합니다.
6. 캐시 정책을 만든 후 캐시 동작 생성 워크플로로 돌아갑니다. 캐시 정책의 목록을 새로 고치고 방금 생성한 정책을 선택합니다.

7. 동작 만들기(Create behavior)를 선택합니다.
8. 엔드포인트가 Microsoft Smooth Streaming 엔드포인트가 아닌 경우 이 단계를 반복하여 두 번째 캐시 동작을 만듭니다.

4단계: 헤더 기반 MediaPackage CDN 승인 활성화

MediaPackage 엔드포인트와 CloudFront 배포 간에 헤더 기반 미디어 패키지 CDN 승인을 활성화하는 것이 좋습니다. 자세한 내용은 AWS Elemental MediaPackage 사용 설명서의 MediaPackage에서 CDN 승인 활성화를 참조하세요.

5단계: CloudFront를 사용하여 라이브 스트림 채널 제공

배포를 생성하고 오리진을 추가하고 헤더 기반 CDN 승인을 활성화한 후 캐시 동작을 생성하면 CloudFront를 사용하여 라이브 스트림 채널을 제공할 수 있습니다. CloudFront는 캐시 동작에 대해 구성한 설정에 따라 최종 사용자의 요청을 올바른 MediaPackage 엔드포인트로 라우팅합니다.

애플리케이션(예: 미디어 플레이어)의 링크로, CloudFront URL에 독립형 포맷의 미디어 파일의 URL을 지정합니다. 자세한 내용은 the section called “파일 URL 사용자 지정” 섹션을 참조하세요.

## AWS Elemental MediaPackage로 동영상 온디맨드 콘텐츠 제공

AWS Elemental MediaPackage 오리진에서 동영상 온디맨드(VOD) 콘텐츠를 시작하는 경우 CloudFront 배포를 만들고 최적화된 캐시 동작을 구성하여 VOD 콘텐츠를 뷰어에게 제공할 수 있습니다. 다음 프로세스에서는 패키징 구성으로 패키지 그룹을 이미 만들고 MediaPackage로 자산을 수집했다고 가정합니다.

MediaPackage용 CloudFront 배포를 수동으로 생성하려면 다음 단계를 수행하십시오.

1단계: CloudFront 배포 생성 및 구성

MediaPackage를 사용하여 만든 패키징 그룹용 CloudFront 배포를 설정하려면 다음 절차를 완료합니다.

VOD 콘텐츠용 배포를 만들려면 다음과 같이 합니다.

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포 생성을 선택합니다.
3. 다음과 같이 배포를 위한 설정을 선택합니다.

오리진 도메인

MediaPackage 패키징 그룹의 오리진입니다. 텍스트 필드에 오리진 URL 값을 입력합니다. 오리진은 HTTPS URL이어야 합니다.

예를 들어 `https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.m3u8`과 같은 HLS 엔드포인트의 경우, 오리진 도메인은 `3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com`입니다.

자세한 내용은 the section called "오리진 도메인" 섹션을 참조하세요.

오리진 경로

콘텐츠가 제공되는 경로입니다.

오리진 경로의 작동 방식에 대한 자세한 내용은 the section called "오리진 경로" 단원을 참조하세요.

> **Important**
>
> CloudFront 배포의 어딘가로 라우팅하려면 와일드카드 경로 `*`가 필요합니다. 명시적 경로와 일치하지 않는 요청이 실제 오리진으로 라우팅되지 않도록 하려면 해당 와일드카드 경로에 대해 '더미' 오리진을 만듭니다.

Example: '더미' 오리진 생성

다음 예제에서는 `def456` 및 `321xyz`가 '실제' 오리진으로 라우팅되지만 다른 동영상 콘텐츠에 대한 요청은 적절한 하위 도메인 없이 `mediapackage-vod.us-west-2.amazonaws.com`으로 라우팅되어 HTTP `404` 오류가 발생합니다.

두 가지 패키징 구성이 있는 패키징 그룹의 단일 자산에 대한 MediaPackage 콘텐츠 URL은 다음과 같습니다.

```
https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/
abc123/def456/ghi789/index.m3u8
https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/
abc123/321xyz/654uvw/index.m3u8
```

CloudFront 오리진 A:

```
Domain: 3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com
Path: None
```

CloudFront 오리진 B:

```
Domain: mediapackage-vod.us-west-2.amazonaws.com
Path: None
```

CloudFront 캐시 동작:

```
1. Path: /out/v1/*/def456/* forward to Origin A
2. Path: /out/v1/*/321xyz/* forward to Origin A
3. Path: * forward to Origin B
```

그밖의 배포 설정에 대해, 다른 기술 요구 사항 또는 비즈니스 요구 사항을 기반으로 특정 값을 설정합니다. 배포에 대한 모든 옵션 목록과 설정에 대한 정보는 the section called “모든 배포 설정” 단원을 참조하십시오.

다른 배포 설정 선택을 마치면 배포 생성(Create Distribution)을 선택합니다.

4. 방금 생성한 배포를 선택한 다음 동작(Behaviors)을 선택합니다.
5. 기본 캐시 동작을 선택한 다음 편집(Edit)을 선택합니다. 오리진에서 선택한 패키징 구성에 적합한 캐시 동작 설정을 지정합니다. 나중에 한 개 이상의 오리진을 추가한 후 그에 대한 캐시 동작 설정을 편집해 보겠습니다.
6. CloudFront 배포(CloudFront distributions) 페이지로 이동합니다.
7. 배포의 마지막 수정(Last modified) 열 값이 배포 중(Deploying)에서 날짜 및 시간으로 바뀔 때까지 기다립니다. 이렇게 상태가 바뀌면 CloudFront에서 배포를 생성한 것입니다.

### 2단계: MediaPackage 패키징 그룹의 도메인에 오리진 추가

다음 단계를 반복하여 각 MediaPackage 패키징 그룹을 배포에 추가합니다. '더미' 오리진을 만들어야 한다는 점에 유의하세요.

다른 패키징 그룹을 오리진으로 추가하려면 다음과 같이 합니다.

1. CloudFront 콘솔에서 채널에 대해 생성한 배포를 선택합니다.

2. 오리진(Origins)을 선택한 다음 오리진 생성(Create origin)을 선택합니다.
3. 원본 도메인에 MediaPackage 패키징 그룹의 URL을 입력합니다.
4. 그밖의 설정에 대해, 다른 기술 요구 사항 또는 비즈니스 요구 사항을 기반으로 값을 설정합니다. 자세한 내용은 the section called “오리진 설정” 섹션을 참조하세요.
5. 오리진 생성(Create Origin)을 선택합니다.

### 3단계: 모든 패키징 구성에 대한 캐시 동작 구성

각 패키징 구성에 대해 요청을 올바르게 라우팅하는 경로 패턴을 추가하도록 캐시 동작을 구성해야 합니다. 지정하는 경로 패턴은 제공하는 비디오 포맷에 따라 다릅니다. 다음 절차에는 Apple HLS, CMAF, DASH, Microsoft Smooth Streaming 포맷에 대해 사용할 경로 패턴 정보가 나와 있습니다.

일반적으로 각 패키징 구성에 대해 여러 캐시 동작을 설정합니다.

- 패턴 매니페스트(해당 파일에 대한 인덱스)
- 세그먼트(비디오 콘텐츠의 파일) 형식은 구성에 따라 콘텐츠에 둘 이상의 확장자를 사용할 수 있습니다. 각 확장에 캐시 동작이 필요합니다.

패키징 구성에 대한 캐시 동작을 만들려면 다음과 같이 합니다.

1. CloudFront 콘솔에서 채널에 대해 생성한 배포를 선택합니다.
2. 동작(Behaviors)을 선택한 다음 동작 만들기(Create behavior)를 선택합니다.
3. 경로 패턴에서 특정 MediaPackage VOD 패키징 구성 GUID를 경로 접두사로 사용합니다. 이는 MediaPackage VOD 경로의 두 번째 GUID입니다.

   경로 패턴

   `https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.m3u8`과 같이 HLS 콘텐츠의 경우 다음 캐시 동작을 만듭니다.

   - 상위 및 하위 매니페스트에 대해 `/out/v1/*/def456/*.m3u8`을 사용합니다.
   - 콘텐츠 세그먼트의 경우 `/out/v1/*/def456/*.ts`를 사용하고 모든 필요한 세그먼트 확장자에서 반복합니다.

   `https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.m3u8`과 같이 CMAF 콘텐츠의 경우 다음 캐시 동작을 만듭니다.

- 상위 및 하위 매니페스트에 대해 `/out/v1/*/def456/*.m3u8`을 사용합니다.
- 콘텐츠 세그먼트의 경우 `/out/v1/*/def456/*.mp4`를 사용하고 모든 필요한 세그먼트 확장자에서 반복합니다.

`https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.mpd`와 같이 DASH 콘텐츠의 경우 다음 캐시 동작을 만듭니다.

- 상위 매니페스트에 대해 `/out/v1/*/def456/*.mpd`를 사용합니다.
- 콘텐츠 세그먼트에 대해 `/out/v1/*/def456/*.mp4`를 사용합니다.

`https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.ism/Manifest`과 같은 Microsoft Smooth Streaming 엔드포인트의 경우, 한 개의 매니페스트만 제공되므로 캐시 동작을 `out/v1/*/def456/*/index.ism/*` 하나만 만듭니다.

4. 각 캐시 동작에 대해 다음 설정의 값을 지정합니다.

뷰어 프로토콜 정책

Redirect HTTP to HTTPS(HTTP를 HTTPS로 재지정)를 선택합니다.

캐시 정책 및 오리진 요청 정책

캐시 정책(Cache policy)에서 정책 생성(Create policy)을 선택합니다. 새 캐시 정책에 대해 다음 설정을 지정합니다.

Minimum TTL

5초 미만으로 설정하면 부실한 콘텐츠 서비스 방지에 도움이 됩니다.

쿼리 문자열

쿼리 문자열(Query strings)(캐시 키 설정(Cache key settings)에 있음)에서 지정된 쿼리 문자열 포함(Include specified query strings)을 선택합니다. 허용(Allow)에서 다음 값을 입력하고 항목 추가(Add item)를 선택하여 값을 추가합니다.

- MediaPackage에서 매니페스트 필터링 기능을 사용하는 경우, 매니페스트 요청(`*.m3u8`, `*.mpd`, `index.ism/*`)에 대한 캐시 동작에 사용할 캐시 정책에 `aws.manifestfilter`를 추가 쿼리 문자열 파라미터로 지정합니다. 이렇게 하면 매니페스트 필터링 기능이 작동하는 데 필요한 MediaPackage 오리진에 `aws.manifestfilter` 쿼리 문자열을 전달하도록 배포가 구성됩니다. 자세한 내용은 AWS Elemental MediaPackage 사용 설명서의 매니페스트 필터링을 참조하세요.

5. 생성(Create)을 선택합니다.
6. 캐시 정책을 만든 후 캐시 동작 생성 워크플로로 돌아갑니다. 캐시 정책의 목록을 새로 고치고 방금 생성한 정책을 선택합니다.
7. 동작 만들기(Create behavior)를 선택합니다.
8. 엔드포인트가 Microsoft Smooth Streaming 엔드포인트가 아닌 경우 이 단계를 반복하여 두 번째 캐시 동작을 만듭니다.

4단계: 헤더 기반 MediaPackage CDN 승인 활성화

MediaPackage VOD 콘텐츠와 CloudFront 배포 간에 헤더 기반 MediaPackage CDN 승인을 활성화하는 것이 좋습니다. 자세한 내용은 AWS Elemental MediaPackage 사용 설명서의 MediaPackage에서 CDN 승인 활성화를 참조하세요.

5단계: CloudFront를 사용하여 VOD 콘텐츠 제공

배포를 만들고 오리진을 추가하고 캐시 동작을 만들고 헤더 기반 CDN 승인을 활성화하면 CloudFront를 사용하여 VOD 콘텐츠를 제공할 수 있습니다. CloudFront는 캐시 동작에 대해 구성한 설정에 따라 뷰어의 요청을 올바른 MediaPackage VOD 콘텐츠로 라우팅합니다.

애플리케이션(예: 미디어 플레이어)의 링크로, CloudFront URL에 독립형 포맷의 미디어 파일의 URL을 지정합니다. 자세한 내용은 the section called “파일 URL 사용자 지정” 섹션을 참조하세요.

# 미디어 품질 인식 복원력

미디어 품질 인식 복원력(MQAR)은 Amazon CloudFront와 AWS Media Services 간의 통합 기능입니다. MQAR은 미디어 품질 신뢰도 점수(MQCS)를 기반으로 리전 간 오리진 옵션을 자동으로 제공합니다. MQCS는 뷰어가 인식하는 미디어 품질 경험에 영향을 미치는 파라미터를 기반으로 AWS Elemental MediaLive에서 합성됩니다. CloudFront 오리진 그룹 장애 조치 기준에 지정할 수 있는 여러 옵션을 사용하여 복원력이 뛰어난 실시간 이벤트 스트리밍을 제공하도록 CloudFront 및 AWS Media Services를 구성할 수 있습니다.

배포에 대해 MQAR 기능을 활성화하면 CloudFront가 품질 점수가 가장 높은 것으로 간주되는 오리진을 자동으로 선택할 수 있는 권한을 부여합니다.

품질 점수는 블랙 프레임, 고정 또는 삭제된 프레임, 반복 프레임 등 오리진에서 인지된 미디어 스트리밍 품질 문제를 나타냅니다. 예를 들어 AWS Elemental MediaPackage v2 오리진이 서로 다른 두 AWS 리전에 배포되고 한 리전이 다른 리전에 비해 더 높은 미디어 품질 점수를 보고하는 경우, CloudFront는 자동으로 더 높은 점수를 보고하는 오리진으로 전환합니다.

이렇게 하기 위해 CloudFront에서 다음 작업이 수행됩니다.

1. CloudFront는 GET 요청을 기본 MediaPackage 오리진에 전달하고 동시에 보조 MediaPackage 오리진에 대한 HEAD 요청을 시작합니다. CloudFront는 각 오리진의 응답 헤더에서 미디어 품질 점수를 받습니다.
2. 다음으로 CloudFront는 각 오리진의 점수를 추적하고 이 정보를 사용하여 새 요청이 도착할 때 점수가 높은 오리진을 결정합니다.

오리진의 미디어 품질 점수는 실시간으로 변경될 수 있습니다. CloudFront는 MQCS 변경 사항을 사용하여 이를 결정하고 오리진 간에 전환하여 뷰어가 더 높은 미디어 품질의 콘텐츠를 볼 수 있도록 합니다. 자세한 내용은 AWS Elemental MediaPackage V2 사용 설명서의 Leveraging media quality scores with MediaPackage를 참조하시기 바랍니다.

MQAR을 사용하면 CloudFront가 고객에게 잠재적으로 영향을 미칠 수 있는 문제가 있는지 여부를 최대한 빨리 확인할 수 있습니다. 예를 들어 네트워크 연결, 동영상 처리, 오디오 손실 또는 중단, 인코더 속도 문제와 같은 문제는 뷰어의 미디어 품질 점수에 영향을 미칠 수 있습니다.

MQAR은 오리진 간 원활한 전환을 제공하므로 복원력이 뛰어난 리전 간 엔드 투 엔드 미디어 전송 워크플로를 AWS에서 배포하고 뷰어에게 고품질 콘텐츠를 제공할 수 있습니다.

> **Note**
>
> 현재 이 기능은 MediaPackage v2 오리진만 지원합니다.

이 기능을 활성화해 배포하려면 다음 단계를 완료합니다.

1. 아직 완료되지 않은 경우 MediaPackage v2 오리진을 만들고 엔드포인트 구성에서 이 기능을 활성화합니다. 리전 간 배포의 경우 동일한 설정으로 다른 AWS 리전에 보조 채널을 만듭니다. 자세한 내용은 AWS Elemental MediaPackage V2 사용 설명서에서 다음 주제를 참조하시기 바랍니다.
   - 채널 및 엔드포인트 생성
   - 미디어 품질 점수 사용
2. CloudFront에 MediaPackage v2 오리진을 사용하려면 CloudFront 배포를 만들거나 업데이트합니다. 배포 생성 및 배포 업데이트 섹션을 참조하세요.
3. 오리진 그룹을 만들고 두 오리진을 기본 및 보조로 선택합니다. 오리진 그룹에서 미디어 품질 점수 옵션을 활성화합니다. 자세한 내용은 오리진 그룹 생성 섹션을 참조하세요.

4. 배포의 캐시 동작에서 만든 오리진 그룹을 선택합니다. 캐시 동작이 채널 경로 패턴과 일치하는 것이 좋습니다.

CloudFront에서 두 MediaPackage v2 오리진 모두 점수가 동일하다고 판단하면 오리진 그룹에 나열된 기본 오리진으로 요청을 전달합니다. 처음에 선택한 오리진이 오리진 그룹에서 지정한 장애 조치 기준과 일치하는 오류 코드로 응답하면 CloudFront는 미디어 품질 점수와 관계없이 오리진 그룹의 대체 오리진에 요청을 재시도합니다.

> **참고**
>
> - CloudFront는 미디어 품질 점수에 대해 활성화된 오리진 그룹을 활용하는 각 캐시 동작의 품질 점수를 추적합니다. 미디어 품질 점수를 내보내는 여러 채널에 동일한 오리진 그룹을 사용하는 경우 점수가 섞이지 않도록 각 채널의 경로 패턴에 대해 별도의 캐시 동작을 만듭니다. 오리진 그룹 할당량에 대한 자세한 내용은 배포의 일반 할당량 섹션을 참조하시기 바랍니다.
> - 현재 배포의 캐시 동작과 연결된 오리진 방향 트리거(오리진 요청 및 오리진 응답)에서 Lambda@Edge 함수를 사용하는 경우 MQAR을 사용할 수 없습니다. 자세한 내용은 캐시 동작 설정 섹션을 참조하세요.
> - MQAR 기능 및 오리진 액세스 제어(OAC)를 사용 설정한 경우 IAM 정책에 `mediapackagev2:GetHeadObject` 작업을 추가합니다. MediaPackage v2 오리진으로 `HEAD` 요청을 보내려면 MQAR에 이 권한이 필요합니다. OAC에 대한 자세한 내용은 AWS Elemental MediaPackage v2 오리진에 대한 액세스 제한을 참조합니다.

## MQAR 로그 필드

CloudFront는 품질 점수와 선택한 오리진을 반영하기 위해 실시간 액세스 로그에 다음 필드를 제공합니다. CloudFront 실시간 액세스 로그에서 이러한 필드를 활성화할 수 있습니다.

- `r-host`
- `sr-reason`
- `x-edge-mqcs`

자세한 내용은 Fields 65~67을 참조하시기 바랍니다.

# 함수를 사용하여 엣지에서 사용자 지정

Amazon CloudFront를 사용하면 자체 코드를 작성하여 CloudFront 배포에서 HTTP 요청 및 응답을 처리하는 방법을 사용자 지정할 수 있습니다. 코드는 최종 사용자(사용자) 가까이에서 실행되어 지연 시간을 최소화하고 서버나 기타 인프라를 관리할 필요가 없습니다. 코드를 작성하여 CloudFront를 통해 흐르는 요청 및 응답을 조작하고, 기본 인증 및 권한 부여를 수행하고, 엣지에서 HTTP 응답을 생성하는 등의 작업을 수행할 수 있습니다.

CloudFront 배포에 작성하고 연결하는 코드를 엣지 함수라고 합니다. CloudFront는 엣지 함수를 작성하고 관리하는 두 가지 방법을 제공합니다.

CloudFront 함수

지연 시간에 민감한 대규모 CDN 사용자 지정을 위해 JavaScript로 경량 함수를 작성할 수 있습니다. CloudFront 함수 런타임 환경은 밀리초 미만의 시작 시간을 제공하고 초당 수백만 건의 요청을 처리할 수 있도록 즉시 확장되며 매우 안전합니다. CloudFront 함수는 CloudFront의 기본 기능입니다. 즉, CloudFront 내에서 완전히 코드를 빌드, 테스트 및 배포할 수 있습니다.

Lambda@Edge

Lambda@Edge는 더 가까운 전체 애플리케이션 로직 및 복잡한 함수에 대한 강력하면서도 유연한 서버리스 컴퓨팅을 뷰어에게 제공하는 AWS Lambda의 확장으로, 매우 안전합니다. Lambda@Edge 함수는 Node.js 또는 Python 런타임 환경에서 실행됩니다. 단일 AWS 리전에 함수를 게시하고, 함수를 CloudFront 배포에 연결하면 Lambda@Edge에서 자동으로 전 세계에 코드를 복제합니다.

CloudFront에서 AWS WAF를 실행하는 경우, CloudFront 함수 및 Lambda@Edge 모두에 AWS WAF 삽입 헤더를 사용할 수 있습니다. 이는 최종 사용자 및 오리진 요청과 응답에 사용할 수 있습니다.

주제


- CloudFront Functions와 Lambda@Edge 간 차이점
- CloudFront Functions를 사용하여 엣지에서 사용자 지정
- CloudFront 연결 함수를 사용하여 사용자 지정
- Lambda@Edge를 사용하여 엣지에서 사용자 지정
- 엣지 함수에 대한 제한 사항

# CloudFront Functions와 Lambda@Edge 간 차이점

CloudFront 함수와 Lambda@Edge 모두 CloudFront 이벤트에 대한 응답으로 코드를 실행할 수 있는 방법을 제공합니다.

CloudFront Functions는 다음과 같은 사용 사례의 단기 실행 경량 함수에 적합합니다.

- 캐시 키 정규화 – HTTP 요청 속성(헤더, 쿼리 문자열, 쿠키 및 URL 경로)을 변환하여 캐시 적중률을 개선할 수 있도록 최적의 캐시 키를 생성합니다.
- 헤더 조작 – 요청 또는 응답에서 HTTP 헤더를 삽입, 수정 또는 삭제합니다. 예를 들어 모든 요청에 `True-Client-IP` 헤더를 추가할 수 있습니다.
- URL 리디렉션 또는 다시 쓰기 – 요청 정보에 따라 뷰어를 다른 페이지로 리디렉션하거나 한 경로에서 다른 경로로 모든 요청을 다시 씁니다.
- 요청 권한 부여 – 권한 부여 헤더 또는 다른 요청 메타데이터를 검사하여 JSON 웹 토큰(JWT)과 같은 해시된 권한 부여 토큰을 검증합니다.

CloudFront 함수를 시작하려면 CloudFront Functions를 사용하여 엣지에서 사용자 지정 단원을 참조하세요.

Lambda@Edge는 다음과 같은 사용 사례에 적합합니다.

- 완료하는 데 몇 밀리초 이상이 걸리는 함수
- 조정 가능한 CPU 또는 메모리가 필요한 함수
- 서드 파티 라이브러리(다른 AWS 서비스와의 통합을 위한 AWS SDK 포함)에 의존하는 함수
- 처리에 외부 서비스를 사용하기 위해 네트워크 액세스가 필요한 함수
- 파일 시스템 액세스 또는 HTTP 요청 본문에 대한 액세스가 필요한 함수

Lambda@Edge를 시작하려면 Lambda@Edge를 사용하여 엣지에서 사용자 지정 단원을 참조세요.

사용 사례에 맞는 옵션을 적절히 선택할 수 있도록 다음 표를 통해 CloudFront Functions와 Lambda@Edge 간의 차이점을 파악하세요. 오리진 수정 헬퍼 메서드에 적용되는 차이점에 대한 자세한 내용은 CloudFront Functions과 Lambda@Edge 중에서 선택 섹션을 참조하세요.

| | CloudFront 함수 | Lambda@Edge |
|---|---|---|
| 프로그래밍 언어 | JavaScript(ECMAScript 5.1 호환) | Node.js 및 Python |
| 이벤트 소스 | • 최종 사용자 요청<br>• 최종 사용자 응답 | • 최종 사용자 요청<br>• 최종 사용자 응답<br>• 오리진 요청<br>• 오리진 응답 |
| Amazon CloudFront KeyValueStore 지원 | 예<br><br>CloudFront KeyValueStore는 JavaScript 런타임 2.0만 지원합니다. | 아니요 |
| Scale | 초당 최대 수백만 개의 요청 | 리전별 초당 최대 10,000 건의 요청 |
| 함수 지속 시간 | 서브밀리초 | 최대 30초(최종 사용자 요청 및 최종 사용자 응답)<br><br>최대 30초(오리진 요청 및 오리진 응답) |
| 최대 함수 메모리 크기 | 2MB | 128MB(최종 사용자 요청 및 최종 사용자 응답)<br><br>10,240MB(10GB)(오리진 요청 및 오리진 응답)<br><br>자세한 내용은 Lambda@Edge에 대한 할당량 섹션을 참조하세요. |
| 함수 코드 및 포함된 라이브러리의 최대 크기 | 10KB | 50MB(뷰어 요청 및 뷰어 응답) |

| | CloudFront 함수 | Lambda@Edge |
|---|---|---|
| | | 50MB(오리진 요청 및 오리진 응답) |
| 네트워크 액세스 | 아니요 | 예 |
| 파일 시스템 액세스 | 아니요 | 예 |
| 요청 본문에 대한 액세스 | 아니요 | 예 |
| 위치 정보 및 장치 데이터에 대한 액세스 | 예 | No(최종 사용자 요청 및 최종 사용자 응답)<br><br>Yes(오리진 요청 및 오리진 응답) |
| CloudFront 내에서 완전히 빌드 및 테스트 가능 | 예 | 아니요 |
| 함수 로깅 및 지표 | 예 | 예 |

# CloudFront Functions를 사용하여 엣지에서 사용자 지정

CloudFront 함수를 사용하면 지연 시간에 민감한 대규모 CDN 사용자 지정을 위해 JavaScript로 경량 함수를 작성할 수 있습니다. 함수는 CloudFront를 통해 흐르는 요청 및 응답을 조작하고, 기본 인증 및 권한 부여를 수행하고, 엣지에서 HTTP 응답을 생성하는 등의 작업을 수행할 수 있습니다. CloudFront 함수 런타임 환경은 밀리초 미만의 시작 시간을 제공하고 초당 수백만 건의 요청을 처리할 수 있도록 즉시 확장되며 매우 안전합니다. CloudFront 함수는 CloudFront의 기본 기능입니다. 즉, CloudFront 내에서 완전히 코드를 빌드, 테스트 및 배포할 수 있습니다.

CloudFront 배포를 CloudFront 함수와 연결하면 CloudFront가 CloudFront 엣지 로케이션에서 요청 및 응답을 가로채고 함수에 전달합니다. 다음 이벤트가 발생할 때 CloudFront 함수를 간접 호출할 수 있습니다.

- CloudFront가 최종 사용자의 요청을 수신할 때(최종 사용자 요청)
- CloudFront가 최종 사용자에게 응답을 반환하기 전(최종 사용자 응답)
- TLS 연결 설정(연결 요청) 중 - 현재 상호 TLS(mTLS) 연결에 사용 가능

CloudFront Functions에 대한 자세한 내용은 다음 주제를 참조하세요.

주제


- 튜토리얼: CloudFront Functions를 사용하여 간단한 함수 생성
- 튜토리얼: 키 값을 포함하는 CloudFront 함수 생성
- 함수 코드 작성
- 함수 생성
- 함수 테스트
- 함수 업데이트
- 함수 게시
- 배포에 함수 연결
- Amazon CloudFront KeyValueStore


# 튜토리얼: CloudFront Functions를 사용하여 간단한 함수 생성

이 자습서에서는 CloudFront 함수를 시작하는 방법을 설명합니다. 뷰어를 다른 URL로 리디렉션하고 사용자 지정 응답 헤더를 반환하는 간단한 함수를 만들 수 있습니다.

목차


- 사전 조건
- 함수 생성
- 함수 확인


## 사전 조건

CloudFront 함수를 사용하려면 CloudFront 배포가 필요합니다. 계정이 없는 경우 CloudFront 표준 배포 시작 단원을 참조하십시오.

## 함수 생성

CloudFront 콘솔을 사용하여 뷰어를 다른 URL로 리디렉션하고 사용자 지정 응답 헤더를 반환하는 간단한 함수를 만들 수 있습니다.

CloudFront 함수를 생성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 함수를 선택한 후, 함수 생성을 선택합니다.
3. 함수 생성 페이지에서 이름에 *MyFunctionName*과 같은 함수 이름을 입력합니다.
4. (선택 사항) 설명에 **Simple test function**과 같은 기능에 대한 설명을 입력합니다.
5. 런타임의 경우 선택한 기본 JavaScript 버전을 유지합니다.
6. 함수 생성(Create function)을 선택합니다.
7. 다음 함수 코드를 복사합니다. 이 함수 코드는 최종 사용자를 다른 URL로 리디렉션하고 사용자 지정 응답 헤더도 반환합니다.

```
function handler(event) {
    // NOTE: This example function is for a viewer request event trigger.
    // Choose viewer request for event trigger when you associate this function
 with a distribution.
    var response = {
        statusCode: 302,
        statusDescription: 'Found',
        headers: {
            'cloudfront-functions': { value: 'generated-by-CloudFront-Functions' },
            'location': { value: 'https://aws.amazon.com/cloudfront/' }
        }
    };
    return response;
}
```

8. 함수 코드의 경우 코드를 코드 편집기에 붙여넣어 기본 코드를 대체합니다.
9. 변경 사항 저장을 선택합니다.
10. (선택 사항) 함수를 게시하기 전에 테스트할 수 있습니다. 이 자습서에서는 함수를 테스트하는 방법에 대해 설명하지 않습니다. 자세한 내용은 함수 테스트 단원을 참조하십시오.
11. 게시 탭을 선택한 다음 게시 함수를 선택합니다. 함수를 CloudFront 배포와 연결하려면 먼저 함수를 게시해야 합니다.
12. 다음으로 함수를 배포 또는 캐시 동작과 연결할 수 있습니다. *MyFunctionName*페이지에서 게시 탭을 선택합니다.

> **Warning**
> 다음 단계에서는 테스트에 사용할 배포 또는 캐시 동작을 선택합니다. 이 테스트 함수를 프로덕션 환경에서 사용되는 배포 또는 캐시 동작과 연결하지 마세요.

13. Add association을 선택합니다.
14. 연결 대화 상자에서 배포 및/또는 캐시 동작을 선택합니다. 이벤트 유형의 경우 기본값을 유지합니다.
15. 연결 추가를 선택합니다.

    연결된 배포 테이블에 연결된 배포가 표시됩니다.
16. 연결된 배포가 배포될 때까지 몇 분 정도 기다립니다. 그런 다음 배포 상태를 확인하려면 연결된 배포 테이블에서 해당 배포를 선택하고 배포 보기를 선택합니다.

    배포의 상태가 배포됨(Deployed)이면 함수가 작동하는지 확인할 준비가 된 것입니다.

## 함수 확인

함수를 배포한 후 해당 함수가 배포에서 작동하는지 확인할 수 있습니다.

함수를 확인하려면

1. 웹 브라우저에서 배포의 도메인 이름(예: `https://d111111abcdef8.cloudfront.net`)으로 이동합니다.

   이 함수는 브라우저로 리디렉션을 반환하므로 브라우저가 자동으로 `https://aws.amazon.com/cloudfront/`(으)로 이동합니다.
2. 명령줄 창에서 같은 curl 도구를 사용하여 배포의 도메인 이름으로 요청을 보낼 수 있습니다.

   ```
   curl -v https://d111111abcdef8.cloudfront.net/
   ```

   응답에는 리디렉션 응답(`302 Found`)과 함수가 추가한 사용자 지정 응답 헤더가 표시됩니다. 응답은 다음 예와 같이 보일 수 있습니다.

   Example

   ```
   curl -v https://d111111abcdef8.cloudfront.net/
   ```

```
> GET / HTTP/1.1
> Host: d111111abcdef8.cloudfront.net
> User-Agent: curl/7.64.1
> Accept: */*
>
< HTTP/1.1 302 Found
< Server: CloudFront
< Date: Tue, 16 Mar 2021 18:50:48 GMT
< Content-Length: 0
< Connection: keep-alive
< Location: https://aws.amazon.com/cloudfront/
< Cloudfront-Functions: generated-by-CloudFront-Functions
< X-Cache: FunctionGeneratedResponse from cloudfront
< Via: 1.1 3035b31bddaf14eded329f8d22cf188c.cloudfront.net (CloudFront)
< X-Amz-Cf-Pop: PHX50-C2
< X-Amz-Cf-Id: ULZdIz6j43uGBlXyob_JctF9x7CCbwpNniiMlmNbmwzH1YWP9FsEHg==
```

# 튜토리얼: 키 값을 포함하는 CloudFront 함수 생성

이 자습서에서는 CloudFront 함수를 사용하여 키 값을 포함하는 방법을 보여줍니다. 키 값은 키 값 페어의 일부입니다. 사용자는 키 값 페어의 이름을 함수 코드에 포함할 수 있습니다. 함수 실행 시 CloudFront가 이름을 값으로 대체합니다.

키 값 페어는 키 값 저장소에 저장되는 변수입니다. 함수에 하드 코딩된 값 대신 키를 사용하면 함수가 더 유연해집니다. 코드 변경 내용을 배포하지 않고도 키 값을 변경할 수 있습니다. 키 값 페어를 사용하면 함수 크기를 줄일 수도 있습니다. 자세한 내용은 ??? 단원을 참조하십시오.

목차


- 사전 조건
- 키 값 저장소 생성
- 키 값 저장소에 키 값 페어 추가
- 키 값 저장소를 함수와 연결
- 함수 코드 테스트 및 게시


## 사전 조건

CloudFront Functions 함수 및 키 값 저장소를 처음 사용하는 경우 the section called “튜토리얼: 간단한 CloudFront 함수 생성”의 튜토리얼을 따르는 것이 좋습니다.

해당 튜토리얼을 완료한 후에는 이 튜토리얼을 따라 이전에 생성한 함수를 확장할 수 있습니다. 이 튜토리얼에서는 키 값 저장소를 먼저 생성하려는 것이 좋습니다.

## 키 값 저장소 생성

먼저 함수에 사용할 키 값 저장소를 생성합니다.

키 값 저장소를 생성하려면

1. 함수에 포함하려는 키 값 페어를 계획합니다. 키 이름을 적어두세요. 함수에 사용하려는 키 값 페어는 하나의 키 값 저장소에 있어야 합니다.
2. 작업 순서를 결정합니다. 다음 두 가지 방법으로 진행할 수 있습니다.
   - 키 값 저장소를 생성하고 저장소에 키 값 페어를 추가합니다. 그런 다음 함수를 생성(또는 수정)하고 키 이름을 통합합니다.
   - 또는 함수를 생성(또는 수정)하고 사용하려는 키 이름으로 통합합니다. 그런 다음 키 값 저장소를 생성하고 저장소에 키 값 페어를 추가합니다.
3. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
4. 탐색 창에서 함수를 선택한 다음 KeyValueStores 탭을 선택합니다.
5. KeyValueStore 생성을 선택하고 다음과 같이 필드를 작성합니다.
   - 저장소 이름을 입력하고 필요한 경우 설명을 입력합니다.
   - S3 URI를 비워 둡니다. 이 튜토리얼에서는 키 값 페어를 수동으로 입력합니다.
6. 생성(Create)을 선택합니다. 새로운 키 값 저장소의 세부 정보 페이지가 나타납니다. 이 페이지에는 현재 비어 있는 키 값 페어 섹션이 포함되어 있습니다.

## 키 값 저장소에 키 값 페어 추가

그런 다음 이전에 생성한 키 값 저장소에 키 값 페어의 목록을 수동으로 추가합니다.

키 값 저장소에 키 값 페어를 추가하려면

1. 키 값 페어 섹션에서 키 값 페어 추가를 선택합니다.
2. 태그 추가를 선택한 다음 키와 값을 입력합니다. 확인 표시를 선택하여 변경 사항을 확인하고 이 단계를 반복하여 더 추가합니다.

3. 작업을 마쳤으면 변경 사항 저장을 선택하여 키 값 페어를 키 값 저장소에 저장합니다. 확인 대화 상자에서 완료를 선택합니다.

이제 키 값 페어 그룹이 포함된 키 값 저장소가 생겼습니다.

## 키 값 저장소를 함수와 연결

이제 키 값 저장소가 생성되었습니다. 그리고 키 값 저장소의 키 이름을 포함하는 함수를 만들거나 수정했습니다. 이제 키 값 저장소와 함수를 연결할 수 있습니다. 이 연결은 함수 내에서 생성할 수 있습니다.

키 값 저장소를 함수와 연결하려면

1. 탐색 창에서 함수를 선택합니다. 기본적으로 함수 탭이 맨 위에 표시됩니다.
2. 함수 이름을 선택하고 연결된 KeyValueStore 섹션에서 기존 KeyValueStore 연결을 선택합니다.
3. 키 값 저장소를 선택하고 KeyValueStore 연결을 선택합니다.

> **Note**
> 각 함수에는 하나의 키 값 저장소만 연결할 수 있습니다.

## 함수 코드 테스트 및 게시

키 값 저장소를 함수와 연결한 후 함수 코드를 테스트하고 게시할 수 있습니다. 다음을 수행할 때를 포함하여 함수 코드를 수정할 때마다 항상 함수 코드를 테스트해야 합니다.

- 키 값 저장소를 함수와 연결할 때
- 새로운 키 값 페어를 포함하도록 함수와 해당 키 값 저장소를 수정할 때
- 키 값 페어의 값을 변경할 때

함수 코드를 테스트하고 게시하려면

1. 함수를 테스트하는 방법은 the section called “함수 테스트” 섹션을 참조하세요. DEVELOPMENT 스테이지에서 함수를 테스트하도록 선택해야 합니다.

2. LIVE 환경에서 새 키 값 페어 또는 수정된 키 값 페어와 함께 함수를 사용할 준비가 되면 함수를 게시합니다.

   함수를 게시하면 CloudFront는 DEVELOPMENT 스테이지에서 라이브 스테이지로 함수 버전을 복사합니다. 함수에는 새 코드가 포함되며 키 값 저장소와 연결됩니다. 라이브 스테이지에서는 연결을 다시 수행하지 않아도 됩니다.

   함수를 게시하는 방법은 the section called "함수 게시" 섹션을 참조하세요.

# 함수 코드 작성

CloudFront Functions를 사용하여 지연 시간에 민감한 대규모 CDN 사용자 지정을 위해 JavaScript로 경량 함수를 작성할 수 있습니다. 함수 코드는 CloudFront를 통해 흐르는 요청 및 응답을 조작하고, 기본 인증 및 권한 부여를 수행하고, 엣지에서 HTTP 응답을 생성하는 등의 작업을 수행할 수 있습니다.

CloudFront Functions의 함수 코드를 작성하는 데 도움을 받으려면 다음 주제를 참조하세요. 코드 예제는 CloudFront의 CloudFront Functions 예제 및 GitHub의 amazon-cloudfront-functions 리포지토리를 참조하시기 바랍니다.

주제


- 함수 용도 결정
- CloudFront 함수 이벤트 구조
- CloudFront 함수에 대한 JavaScript 런타임 기능
- 키 값 저장소를 위한 도우미 메서드
- 오리진 수정용 헬퍼 메서드
- CloudFront SaaS Manager 속성에 대한 헬퍼 메서드
- async 및 await 사용
- CloudFront Functions에 대한 CWT 지원
- 일반 헬퍼 메서드


## 함수 용도 결정

함수 코드를 작성하기 전에 함수의 용도를 결정합니다. CloudFront 함수의 대부분의 함수는 다음 용도 중 하나를 가지고 있습니다.

주제

- 뷰어 요청 이벤트 유형의 HTTP 요청 수정
- 뷰어 요청 이벤트 유형에 HTTP 응답 생성
- 뷰어 응답 이벤트 유형의 HTTP 응답 수정
- 연결 요청 이벤트 유형에서 mTLS 연결 검증
- 관련 정보

함수의 용도에 관계없이 `handler`은(는) 모든 함수의 진입점입니다. CloudFront에 의해 함수에 전달되는 `event`(이)라고 하는 단일 인수가 필요합니다. `event`은(는) HTTP 요청(그리고 함수가 HTTP 응답을 수정하는 경우 응답)의 표현을 포함하는 JSON 객체입니다.

뷰어 요청 이벤트 유형의 HTTP 요청 수정

함수는 CloudFront가 최종 사용자(클라이언트)로부터 수신하는 HTTP 요청을 수정하고 수정된 요청을 CloudFront로 반환하여 처리를 계속할 수 있습니다. 예를 들어 함수 코드가 캐시 키를 정규화하거나 요청 헤더를 수정할 수 있습니다.

HTTP 요청을 수정하는 함수를 만들고 게시한 후 뷰어 요청 이벤트 유형 연결을 추가해야 합니다. 자세한 내용은 함수 생성 섹션을 참조하세요. 즉, 요청된 객체가 CloudFront 캐시에 있는지 여부를 확인하기 위해 점검하기 전에 CloudFront가 뷰어의 요청을 수신할 때마다 함수가 실행됩니다.

Example예제

다음 유사 코드는 HTTP 요청을 수정하는 함수의 구조를 보여줍니다.

```
function handler(event) {
    var request = event.request;

    // Modify the request object here.

    return request;
}
```

이 함수는 수정된 `request` 객체를 CloudFront로 반환합니다. CloudFront는 CloudFront 캐시에서 캐시 적중률을 확인하고 필요한 경우 오리진에 요청을 전송하여 반환된 요청을 계속 처리합니다.

뷰어 요청 이벤트 유형에 HTTP 응답 생성

함수는 CloudFront에서 캐시된 응답이나 추가 처리를 확인하지 않고 엣지에서 HTTP 응답을 생성하여 최종 사용자(클라이언트)에게 직접 반환할 수 있습니다. 예를 들어 함수 코드는 요청을 새 URL로 리디렉션하거나 권한 부여를 확인하고 권한 없는 요청에 `401` 또는 `403` 응답을 반환할 수 있습니다.

HTTP 응답을 생성하는 함수를 생성할 때는 최종 사용자 요청 이벤트 유형을 선택해야 합니다. 즉, CloudFront가 추가 요청을 처리하기 전에 CloudFront가 최종 사용자의 요청을 수신할 때마다 함수가 실행됩니다.

Example예제

다음 유사 코드는 HTTP 응답을 생성하는 함수의 구조를 보여줍니다.

```
function handler(event) {
    var request = event.request;

    var response = ...; // Create the response object here,
                        // using the request properties if needed.

    return response;
}
```

이 함수는 CloudFront에 `response` 객체를 반환하며, CloudFront는 CloudFront 캐시를 확인하거나 오리진에 요청을 보내지 않고 즉시 최종 사용자에게 반환합니다.

뷰어 응답 이벤트 유형의 HTTP 응답 수정

CloudFront가 HTTP 응답을 최종 사용자(클라이언트)에게 전송하기 전에 함수는 HTTP 응답을 수정할 수 있습니다. 응답이 CloudFront 캐시에서 왔는지 오리진에서 왔는지는 관계없습니다. 예를 들어 함수 코드에서 응답 헤더, 상태 코드 및 본문 콘텐츠를 추가하거나 수정할 수 있습니다.

HTTP 응답을 수정하는 함수를 생성할 때는 최종 사용자 응답 이벤트 유형을 선택해야 합니다. 즉, 응답이 CloudFront 캐시에서 왔는지 오리진에서 왔는지 관계없이 CloudFront가 최종 사용자에게 응답을 반환하기 전에 함수가 실행됩니다.

Example예제

다음 유사 코드는 HTTP 응답을 수정하는 함수의 구조를 보여줍니다.

```
function handler(event) {
    var request = event.request;
    var response = event.response;

    // Modify the response object here,
    // using the request properties if needed.
```

```
    return response;
}
```

이 함수는 수정된 `response` 객체를 CloudFront로 반환하며 CloudFront는 즉시 최종 사용자에게 반환합니다.

연결 요청 이벤트 유형에서 mTLS 연결 검증

연결 함수는 사용자 지정 검증 및 인증 로직을 제공하기 위해 TLS 연결 중에 실행되는 CloudFront Functions의 한 유형입니다. 연결 함수는 현재 상호 TLS(mTLS) 연결에 사용할 수 있으며, 여기서 클라이언트 인증서를 검증하고 표준 인증서 검증을 넘어 사용자 지정 인증 로직을 구현할 수 있습니다. 연결 함수는 TLS 핸드셰이크 프로세스 중에 실행되며 인증서 속성, 클라이언트 IP 주소 또는 기타 기준에 따라 연결을 허용하거나 거부할 수 있습니다.

연결 함수를 생성하고 게시한 후에는 mTLS 지원 배포에 연결 요청 이벤트 유형에 대한 연결을 추가해야 합니다. 이렇게 하면 클라이언트가 CloudFront와 mTLS 연결을 설정하려고 할 때마다 함수가 실행됩니다.

Example

다음 의사 코드는 연결 함수의 구조를 보여줍니다.

```
function connectionHandler(connection) {
    // Validate certificate and connection properties here.

    if (/* validation passes */) {
        connection.allow();
    } else {
        connection.deny();
    }
}
```

함수는 헬퍼 메서드를 사용하여 연결을 허용할지 거부할지를 결정합니다. 뷰어 요청 및 뷰어 응답 함수와 달리 연결 함수는 HTTP 요청 또는 응답을 수정할 수 없습니다.

관련 정보

CloudFront Functions 작업에 대한 자세한 내용은 다음 주제를 참조하세요.

- 이벤트 구조
- JavaScript 런타임 기능

- CloudFront Functions 예제
- 엣지 함수에 대한 제한 사항

## CloudFront 함수 이벤트 구조

CloudFront 함수는 함수를 실행할 때 `event` 객체를 함수 코드에 입력으로 전달합니다. 함수를 테스트할 때 `event` 객체를 생성하여 함수에 전달합니다. 함수를 테스트하기 위해 `event` 객체를 생성할 때 `distributionDomainName` 객체의 `distributionId`, `requestId` 및 `context` 필드를 생략할 수 있습니다. 헤더의 이름이 소문자인지 확인합니다. CloudFront Functions가 프로덕션 환경에서 함수에 전달하는 `event` 객체의 경우 항상 소문자입니다.

다음은 이 이벤트 객체의 구조에 대한 개요를 보여 줍니다.

```
{
    "version": "1.0",
    "context": {
        <context object>
    },
    "viewer": {
        <viewer object>
    },
    "request": {
        <request object>
    },
    "response": {
        <response object>
    }
}
```

자세한 내용은 다음 항목을 참조하세요.

주제

- 버전 필드
- 컨텍스트 객체
- 연결 이벤트 구조
- 최종 사용자 객체
- 요청 객체
- 응답 객체

- 상태 코드 및 본문
- 쿼리 문자열, 헤더 및 쿠키 구조
- 예시 응답 객체
- 예시 이벤트 객체

### 버전 필드

`version` 필드에는 CloudFront 함수 이벤트 객체의 버전을 지정하는 문자열이 포함되어 있습니다. 현재 버전은 `1.0`입니다.

### 컨텍스트 객체

`context` 객체는 이벤트에 대한 컨텍스트 정보를 포함합니다. 여기에는 다음 필드가 포함됩니다.

**`distributionDomainName`**

이벤트와 연결된 표준 배포의 CloudFront 도메인 이름(예: d111111abcdef8.cloudfront.net)입니다.

`distributionDomainName` 필드는 함수가 표준 배포에서 호출될 때만 나타납니다.

**`endpoint`**

이벤트와 연결된 연결 그룹의 CloudFront 도메인 이름(예: d111111abcdef8.cloudfront.net)입니다.

`endpoint` 필드는 함수가 다중 테넌트 배포에서 호출될 때만 나타납니다.

**`distributionId`**

이벤트와 연결된 배포의 ID(예: EDFDVBD6EXAMPLE)입니다.

**`eventType`**

이벤트 유형(`viewer-request` 또는 `viewer-response`)입니다.

**`requestId`**

CloudFront 요청(및 연결된 응답)을 고유하게 식별하는 문자열입니다.

### 연결 이벤트 구조

연결 함수는 뷰어 함수와 다른 이벤트 구조를 수신합니다. 연결 이벤트 구조 및 응답 형식에 대한 자세한 내용은 CloudFront 연결 함수 연결 섹션을 참조하세요.

최종 사용자 객체

`viewer` 객체는 요청을 보낸 최종 사용자(클라이언트)의 IP 주소 값이 있는 `ip` 필드를 포함합니다. 최종 사용자가 HTTP 프록시 또는 로드 밸런서를 사용하여 요청을 전송한 경우 이 값은 프록시 또는 로드 밸런서의 IP 주소입니다.

요청 객체

`request` 객체는 viewer-to-CloudFront HTTP 요청의 표현을 포함합니다. 함수에 전달된 `event` 객체에서 `request` 객체는 CloudFront가 뷰어에게 받은 실제 요청을 나타냅니다.

함수 코드가 CloudFront에 `request` 객체를 반환하는 경우 동일한 구조를 사용해야 합니다.

`request` 객체는 다음 필드를 포함합니다.

**`method`**

요청의 HTTP 메서드. 함수 코드가 `request`를 반환하면 이 필드를 수정할 수 없습니다. 이 필드는 `request` 객체에서 유일한 읽기 전용 필드입니다.

**`uri`**

요청된 객체의 상대 경로입니다.

> **Note**
>
> 함수가 `uri` 값을 수정하는 경우 다음 사항이 적용됩니다.
>
> - 새 `uri` 값은 슬래시(/)로 시작해야 합니다.
> - 함수가 `uri` 값을 변경하는 경우 이로 인해 최종 사용자가 요청 중인 객체가 변경됩니다.
> - 함수가 `uri` 값을 변경하는 경우 이로 인해 요청 또는 요청이 전송되는 오리진에 대한 캐시 동작이 변경되지 않습니다.

**`querystring`**

요청의 쿼리 문자열을 나타내는 객체입니다. 요청에 쿼리 문자열이 포함되지 않더라도 `request` 객체에는 여전히 빈 `querystring` 객체가 포함됩니다.

`querystring` 객체는 요청의 각 쿼리 문자열 파라미터에 대해 하나의 필드를 포함합니다.

**headers**

요청의 HTTP 헤더를 나타내는 객체입니다. 요청에 `Cookie` 헤더가 포함되어 있으면 해당 헤더는 `headers` 객체의 일부가 아닙니다. 쿠키는 `cookies` 객체에서 별도로 표시됩니다.

`headers` 객체는 요청의 각 헤더에 대해 하나의 필드를 포함합니다. 헤더 이름은 이벤트 객체에서 ASCII 소문자로 변환되므로 함수 코드에서 추가될 경우 헤더 이름은 ASCII 소문자여야 합니다. CloudFront Functions이 이벤트 객체를 HTTP 요청으로 다시 변환할 경우 헤더 이름이 ASCII 문자이면 헤더 이름에 있는 각 단어의 첫 글자가 대문자로 표시됩니다. CloudFront Functions은 헤더 이름의 ASCII가 아닌 기호에 변경 사항을 적용하지 않습니다. 예를 들어 `TÈst-header`는 함수 내 `tÈst-header`에 있게 됩니다. ASCII가 아닌 기호(È)는 변경되지 않습니다.

단어는 하이픈(`-`)으로 구분됩니다. 예를 들어 함수 코드가 `example-header-name`이라는 이름의 헤더를 추가하는 경우, CloudFront는 이것을 HTTP 요청에서 `Example-Header-Name`으로 변환합니다.

**cookies**

요청(`Cookie` 헤더)의 쿠키를 나타내는 객체입니다.

`cookies` 객체는 요청의 각 쿠키에 대해 하나의 필드를 포함합니다.

쿼리 문자열, 헤더 및 쿠키의 구조에 대한 자세한 내용은 쿼리 문자열, 헤더 및 쿠키 구조 단원을 참조하세요.

예제 `event` 객체에 대해서는 예시 이벤트 객체 단원을 참조하세요.

응답 객체

`response` 객체는 CloudFront-to-Viewer HTTP 응답의 표현을 포함합니다. 함수에 전달된 `event` 객체에서 `response` 객체는 뷰어 요청에 대한 CloudFront의 실제 응답을 나타냅니다.

함수 코드가 `response` 객체를 반환하는 경우 이 동일한 구조를 사용해야 합니다.

`response` 객체는 다음 필드를 포함합니다.

**statusCode**

응답의 HTTP 상태 코드입니다. 이 값은 문자열이 아닌 정수입니다.

함수는 `statusCode`를 생성하거나 수정할 수 있습니다.

**statusDescription**

응답의 HTTP 상태 설명입니다. 함수 코드에서 응답을 생성하는 경우 이 필드는 선택 사항입니다.

**headers**

응답에서 HTTP 헤더를 나타내는 객체입니다. 응답에 `Set-Cookie` 헤더가 포함되어 있으면 해당 헤더는 `headers` 객체의 일부가 아닙니다. 쿠키는 `cookies` 객체에서 별도로 표시됩니다.

`headers` 객체는 응답의 각 헤더에 대해 하나의 필드를 포함합니다. 헤더 이름은 이벤트 객체에서 소문자로 변환되므로, 헤더 이름은 함수 코드에 의해 추가될 때 소문자여야 합니다. CloudFront 함수가 이벤트 객체를 HTTP 응답으로 다시 변환하는 경우 헤더 이름에 있는 각 단어의 첫 글자가 대문자로 표시됩니다. 단어는 하이픈(`-`)으로 구분됩니다. 예를 들어 함수 코드가 `example-header-name`이라는 이름의 헤더를 추가하는 경우, CloudFront는 이것을 HTTP 응답에서 `Example-Header-Name`으로 변환합니다.

**cookies**

응답(`Set-Cookie` 헤더)에서 쿠키를 나타내는 객체입니다.

`cookies` 객체는 응답의 각 쿠키에 대해 하나의 필드를 포함합니다.

**body**

body 필드 추가는 선택 사항이며 함수에서 지정하지 않는 한 `response` 객체에 표시되지 않습니다. 함수는 CloudFront 캐시 또는 오리진에서 반환된 원래 본문에 액세스할 수 없습니다. 뷰어 응답 함수에서 body 필드를 지정하지 않으면 CloudFront 캐시 또는 오리진에서 반환되는 원래 본문이 뷰어에게 반환됩니다.

CloudFront에서 뷰어에게 사용자 지정 본문을 반환하도록 하려면 `data` 필드에 본문 콘텐츠를 지정하고 `encoding` 필드에 본문 인코딩을 지정하세요. 인코딩을 일반 텍스트(`"encoding": "text"`) 또는 Base64로 인코딩된 콘텐츠(`"encoding": "base64"`)로 지정할 수 있습니다.

단축키로 body 필드(`"body": "<specify the body content here>"`)에서 직접 본문 콘텐츠를 지정할 수도 있습니다. 이 작업을 수행할 때는 `data` 및 `encoding` 필드를 생략하세요. 이 경우 CloudFront는 본문을 일반 텍스트로 취급합니다.

**encoding**

body 콘텐츠(`data`필드)의 인코딩입니다. 유일하게 유효한 인코딩은 `text`과 `base64`입니다.

`encoding`을 `base64`로 지정하지만 본문이 유효한 base64가 아닌 경우 CloudFront는 오류를 반환합니다.

**data**

body 콘텐츠입니다.

수정된 상태 코드 및 본문 콘텐츠에 대한 자세한 내용은 상태 코드 및 본문를 참조하세요.

헤더 및 쿠키의 구조에 대한 자세한 내용은 쿼리 문자열, 헤더 및 쿠키 구조 단원을 참조하세요.

예제 `response` 객체에 대해서는 예시 응답 객체 단원을 참조하세요.

상태 코드 및 본문

CloudFront Functions를 사용하여 뷰어 응답 상태 코드를 업데이트하고 응답 본문 전체를 새것으로 바꾸거나 제거할 수 있습니다. CloudFront 캐시 또는 오리진에서 응답의 여러 측면을 평가한 후 뷰어 응답을 업데이트하는 일반적인 시나리오는 다음과 같습니다.

- 상태를 변경하여 HTTP 200 상태 코드를 설정하고 정적 본문 콘텐츠를 생성하여 뷰어에게 반환합니다.
- 상태를 변경하여 HTTP 301 또는 302 상태 코드를 설정하고 사용자를 다른 웹 페이지로 리디렉션합니다.
- 뷰어 응답의 본문을 제공할지 아니면 삭제할지 결정합니다.

> **Note**
> 오리진에서 400 이상의 HTTP 오류를 반환하는 경우 CloudFront Function이 실행되지 않습니다. 자세한 내용은 모든 엣지 함수에 대한 제한 사항을 참조하세요.

HTTP 응답 본문을 사용하여 작업 중일 때 CloudFront Functions가 응답 본문에 액세스하지 못합니다. 원하는 값으로 설정하여 본문 콘텐츠를 바꾸거나 값을 비어 있음으로 설정하여 본문을 제거할 수 있습니다. 함수의 본문 필드를 업데이트하지 않은 경우 CloudFront 캐시 또는 오리진에서 반환된 원래 본문이 뷰어에게 반환됩니다.

> **Tip**
> CloudFront Functions를 사용하여 본문을 교체하는 경우`content-encoding`, `content-type` 또는 `content-length` 등의 해당 헤더를 새 본문 콘텐츠에 맞게 정렬해야 합니다.

예를 들어 CloudFront 오리진 또는 캐시가 `content-encoding: gzip`을 반환하지만 뷰어 응답 함수가 일반 텍스트인 본문을 설정하는 경우 함수는 `content-encoding` 및 `content-type` 헤더도 그에 따라 변경해야 합니다.

CloudFront Function이 400 이상의 HTTP 오류를 반환하도록 구성된 경우 동일한 상태 코드에 대해 지정한 사용자 지정 오류 페이지가 뷰어에게 표시되지 않습니다.

쿼리 문자열, 헤더 및 쿠키 구조

쿼리 문자열, 헤더 및 쿠키는 동일한 구조를 공유합니다. 쿼리 문자열은 요청에 표시될 수 있습니다. 헤더는 요청 및 응답에 표시됩니다. 쿠키는 요청 및 응답에 표시됩니다.

각 쿼리 문자열, 헤더 또는 쿠키는 상위 `querystring`, `headers` 또는 `cookies` 객체 내에서 고유한 필드입니다. 필드 이름은 쿼리 문자열, 헤더 또는 쿠키의 이름입니다. 각 필드에는 쿼리 문자열, 헤더 또는 쿠키의 값이 있는 `value` 속성이 포함됩니다.

목차


- 쿼리 문자열 값 또는 쿼리 문자열 객체
- 헤더에 대한 특별 고려 사항
- 중복 쿼리 문자열, 헤더 및 쿠키(multiValue 배열)
- 쿠키 속성


쿼리 문자열 값 또는 쿼리 문자열 객체

함수는 쿼리 문자열 객체 외에도 쿼리 문자열 값을 반환할 수 있습니다. 쿼리 문자열 값을 사용하여 쿼리 문자열 파라미터를 사용자 지정 순서로 정렬할 수 있습니다.

Example예제

함수 코드에서 쿼리 문자열을 수정하려면 다음과 같은 코드를 사용합니다.

```
var request = event.request;
request.querystring =
 'ID=42&Exp=1619740800&TTL=1440&NoValue=&querymv=val1&querymv=val2,val3';
```

헤더에 대한 특별 고려 사항

헤더의 경우, 헤더 이름은 이벤트 객체에서 소문자로 변환되므로, 헤더 이름은 함수 코드에 의해 추가될 때 소문자여야 합니다. CloudFront 함수가 이벤트 객체를 HTTP 요청 또는 응답으로 다시 변환하는

경우 헤더 이름에 있는 각 단어의 첫 글자가 대문자로 표시됩니다. 단어는 하이픈(-)으로 구분됩니다. 예를 들어 함수 코드가 `example-header-name`이라는 이름의 헤더를 추가하는 경우, CloudFront는 이것을 HTTP 요청 또는 응답에서 `Example-Header-Name`으로 변환합니다.

Example예제

HTTP 요청에서 다음 `Host` 헤더를 고려합니다.

```
Host: video.example.com
```

이 헤더는 `request` 객체에서 다음과 같이 표시됩니다.

```
"headers": {
    "host": {
        "value": "video.example.com"
    }
}
```

함수 코드의 `Host` 헤더에 액세스하려면 다음과 같은 코드를 사용합니다.

```
var request = event.request;
var host = request.headers.host.value;
```

함수 코드에서 헤더를 추가하거나 수정하려면 다음과 같은 코드를 사용합니다(이 코드는 `X-Custom-Header` 값이 있는 `example value`(이)라는 이름의 헤더를 추가함).

```
var request = event.request;
request.headers['x-custom-header'] = {value: 'example value'};
```

중복 쿼리 문자열, 헤더 및 쿠키(**`multiValue`** 배열)

HTTP 요청 또는 응답은 동일한 이름의 하나 이상의 쿼리 문자열, 헤더 또는 쿠키를 포함할 수 있습니다. 이 경우 중복 쿼리 문자열, 헤더 또는 쿠키가 `request` 또는 `response` 객체의 한 필드로 축소되지만 이 필드에는 `multiValue`(이)라는 추가 속성이 포함되어 있습니다. `multiValue` 속성에는 중복 쿼리 문자열, 헤더 또는 쿠키의 각 값이 포함된 배열이 포함됩니다.

Example예제

예를 들어 다음 `Accept` 헤더가 있는 HTTP 요청을 고려합니다.

```
Accept: application/json
Accept: application/xml
Accept: text/html
```

이러한 헤더는 `request` 객체에 다음과 같이 표시됩니다.

```
"headers": {
    "accept": {
        "value": "application/json",
        "multiValue": [
            {
                "value": "application/json"
            },
            {
                "value": "application/xml"
            },
            {
                "value": "text/html"
            }
        ]
    }
}
```

> **Note**
>
> 첫 번째 헤더 값(이 경우 `application/json`)은 `value` 및 `multiValue` 속성 모두에서 반복됩니다. 이렇게 하면 `multiValue` 배열을 반복하여 모든 값에 액세스할 수 있습니다.

함수 코드가 `multiValue` 배열이 있는 쿼리 문자열, 헤더 또는 쿠키를 수정하는 경우 CloudFront 함수는 다음 규칙을 사용하여 변경 사항을 적용합니다.

1. `multiValue` 배열이 존재하고 수정이 있으면 해당 수정이 적용됩니다. `value` 속성의 첫 번째 요소는 무시됩니다.
2. 그렇지 않으면 `value` 속성에 대한 수정 사항이 적용되고 후속 값(있는 경우)은 변경되지 않습니다.

`multiValue` 속성은 앞의 예제와 같이 HTTP 요청 또는 응답에 동일한 이름의 중복 쿼리 문자열, 헤더 또는 쿠키가 포함된 경우에만 사용됩니다. 그러나 단일 쿼리 문자열, 헤더 또는 쿠키에 값이 여러 개 있으면 `multiValue` 속성이 사용되지 않습니다.

Example예제

예를 들어 세 개의 값을 포함하는 하나의 `Accept` 헤더가 있는 요청을 고려합니다.

```
Accept: application/json, application/xml, text/html
```

이 헤더는 `request` 객체에서 다음과 같이 표시됩니다.

```
"headers": {
    "accept": {
        "value": "application/json, application/xml, text/html"
    }
}
```

쿠키 속성

HTTP 응답의 `Set-Cookie` 헤더에서 헤더에는 쿠키의 이름-값 쌍과 세미콜론으로 구분된 속성 집합이 포함됩니다.

Example예제

```
Set-Cookie: cookie1=val1; Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021
 07:28:00 GMT
```

`response` 객체에서 이러한 속성은 쿠키 필드의 `attributes` 속성에 표시됩니다. 예를 들어, 앞의 `Set-Cookie` 헤더는 다음과 같이 표시됩니다.

```
"cookie1": {
    "value": "val1",
    "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021
 07:28:00 GMT"
}
```

예시 응답 객체

다음 예시는 본문이 뷰어 응답 함수로 대체된 `response` 객체(뷰어 응답 함수의 출력)를 보여줍니다.

```
{
  "response": {
    "statusCode": 200,
    "statusDescription": "OK",
```

```
    "headers": {
      "date": {
        "value": "Mon, 04 Apr 2021 18:57:56 GMT"
      },
      "server": {
        "value": "gunicorn/19.9.0"
      },
      "access-control-allow-origin": {
        "value": "*"
      },
      "access-control-allow-credentials": {
        "value": "true"
      },
      "content-type": {
        "value": "text/html"
      },
      "content-length": {
        "value": "86"
      }
    },
    "cookies": {
      "ID": {
        "value": "id1234",
        "attributes": "Expires=Wed, 05 Apr 2021 07:28:00 GMT"
      },
      "Cookie1": {
        "value": "val1",
        "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021
07:28:00 GMT",
        "multiValue": [
          {
            "value": "val1",
            "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021
07:28:00 GMT"
          },
          {
            "value": "val2",
            "attributes": "Path=/cat; Domain=example.com; Expires=Wed, 10 Jan 2021
07:28:00 GMT"
          }
        ]
      }
    },
```

```
    // Adding the body field is optional and it will not be present in the response
 object
    // unless you specify it in your function.
    // Your function does not have access to the original body returned by the
 CloudFront
    // cache or origin.
    // If you don't specify the body field in your viewer response function, the
 original
    // body returned by the CloudFront cache or origin is returned to viewer.

     "body": {
      "encoding": "text",
      "data": "<!DOCTYPE html><html><body><p>Here is your custom content.</p></body></
html>"
    }
  }
}
```

예시 이벤트 객체

다음 예는 완전한 `event` 객체를 보여줍니다. 이 예제는 다중 테넌트 배포가 아닌 표준 배포에서의 호출입니다. 다중 테넌트 배포의 경우 `distributionDomainName` 대신 `endpoint` 필드가 사용됩니다. `endpoint`의 값은 이벤트와 연결된 연결 그룹의 CloudFront 도메인 이름(예: d111111abcdef8.cloudfront.net)입니다.

> **Note**
>
> `event` 객체는 함수에 대한 입력입니다. 함수는 `request` 또는 `response` 객체만 반환하고 전체 `event` 객체는 반환하지 않습니다.

```
{
    "version": "1.0",
    "context": {
        "distributionDomainName": "d111111abcdef8.cloudfront.net",
        "distributionId": "EDFDVBD6EXAMPLE",
        "eventType": "viewer-response",
        "requestId": "EXAMPLEntjQpEXAMPLE_SG5Z-EXAMPLEPmPfEXAMPLEu3EqEXAMPLE=="
    },
    "viewer": {"ip": "198.51.100.11"},
    "request": {
        "method": "GET",
```

```
        "uri": "/media/index.mpd",
        "querystring": {
            "ID": {"value": "42"},
            "Exp": {"value": "1619740800"},
            "TTL": {"value": "1440"},
            "NoValue": {"value": ""},
            "querymv": {
                "value": "val1",
                "multiValue": [
                    {"value": "val1"},
                    {"value": "val2,val3"}
                ]
            }
        },
        "headers": {
            "host": {"value": "video.example.com"},
            "user-agent": {"value": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:83.0)
 Gecko/20100101 Firefox/83.0"},
            "accept": {
                "value": "application/json",
                "multiValue": [
                    {"value": "application/json"},
                    {"value": "application/xml"},
                    {"value": "text/html"}
                ]
            },
            "accept-language": {"value": "en-GB,en;q=0.5"},
            "accept-encoding": {"value": "gzip, deflate, br"},
            "origin": {"value": "https://website.example.com"},
            "referer": {"value": "https://website.example.com/videos/12345678?
action=play"},
            "cloudfront-viewer-country": {"value": "GB"}
        },
        "cookies": {
            "Cookie1": {"value": "value1"},
            "Cookie2": {"value": "value2"},
            "cookie_consent": {"value": "true"},
            "cookiemv": {
                "value": "value3",
                "multiValue": [
                    {"value": "value3"},
                    {"value": "value4"}
                ]
            }
```

```
        }
    },
    "response": {
        "statusCode": 200,
        "statusDescription": "OK",
        "headers": {
            "date": {"value": "Mon, 04 Apr 2021 18:57:56 GMT"},
            "server": {"value": "gunicorn/19.9.0"},
            "access-control-allow-origin": {"value": "*"},
            "access-control-allow-credentials": {"value": "true"},
            "content-type": {"value": "application/json"},
            "content-length": {"value": "701"}
        },
        "cookies": {
            "ID": {
                "value": "id1234",
                "attributes": "Expires=Wed, 05 Apr 2021 07:28:00 GMT"
            },
            "Cookie1": {
                "value": "val1",
                "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr
 2021 07:28:00 GMT",
                "multiValue": [
                    {
                        "value": "val1",
                        "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed,
 05 Apr 2021 07:28:00 GMT"
                    },
                    {
                        "value": "val2",
                        "attributes": "Path=/cat; Domain=example.com; Expires=Wed, 10
 Jan 2021 07:28:00 GMT"
                    }
                ]
            }
        }
    }
}
```

## CloudFront 함수에 대한 JavaScript 런타임 기능

CloudFront Functions JavaScript 런타임 환경은 ECMAScript(ES) 버전 5.1을 준수하며, ES 버전 6에서 12까지의 일부 기능을 지원합니다.

최신 기능을 이용하려면 JavaScript 런타임 2.0을 사용하는 것이 좋습니다.

JavaScript 1.0과 비교한 JavaScript 2.0의 변경 사항은 다음과 같습니다.

- 버퍼 모듈 메서드 사용 가능
- 다음과 같은 비표준 문자열 프로토타입 메서드는 사용할 수 없습니다.
  - `String.prototype.bytesFrom()`
  - `String.prototype.fromBytes()`
  - `String.prototype.fromUTF8()`
  - `String.prototype.toBytes()`
  - `String.prototype.toUTF8()`
- 암호화 모듈의 변경 사항은 다음과 같습니다.
  - `hash.digest()` - 인코딩이 제공되지 않은 경우 반환 유형이 `Buffer`로 변경됩니다.
  - `hmac.digest()` - 인코딩이 제공되지 않은 경우 반환 유형이 `Buffer`로 변경됩니다.
- 이 밖의 새로운 기능에 대한 자세한 내용은 CloudFront Functions를 위한 JavaScript 런타임 2.0 기능 섹션을 참조하세요.

주제

- CloudFront Functions를 위한 JavaScript 런타임 1.0 기능
- CloudFront Functions를 위한 JavaScript 런타임 2.0 기능

## CloudFront Functions를 위한 JavaScript 런타임 1.0 기능

CloudFront 함수 JavaScript 런타임 환경은 ECMAScript(ES) 버전 5.1을 준수하며, ES 버전 6에서 9까지의 일부 기능을 지원합니다. 또한 ES 사양의 일부가 아닌 일부 비표준 방법을 제공합니다.

다음 항목에는 지원되는 모든 언어 기능이 나열되어 있습니다.

주제

- 핵심 기능
- 기본 객체
- 기본 제공 객체
- 오류 유형
- Globals

- 기본 제공 모듈
- 제한된 기능

핵심 기능

ES의 다음과 같은 핵심 기능이 지원됩니다.

유형

모든 ES 5.1 유형이 지원됩니다. 여기에는 부울 값, 숫자, 문자열, 객체, 배열, 함수, 함수 생성자 및 정규 표현식이 포함됩니다.

연산자

모든 ES 5.1 연산자가 지원됩니다.

ES 7 거듭제곱 연산자(`**`)가 지원됩니다.

Statement

> **Note**
>
> `const` 및 `let` 문은 지원되지 않습니다.

다음 ES 5.1 문이 지원됩니다.

- `break`
- `catch`
- `continue`
- `do-while`
- `else`
- `finally`
- `for`
- `for-in`
- `if`
- `return`
- `switch`

- `throw`
- `try`
- `var`
- `while`
- 레이블이 지정된 명령문

리터럴

ES 6 템플릿 리터럴은 여러 줄 문자열, 표현식 보간 및 중첩 템플릿에 대해 지원됩니다..

함수

모든 ES 5.1 함수 기능이 지원됩니다.

ES 6 화살표 함수가 지원되며 ES 6 나머지 파라미터 구문이 지원됩니다.

유니코드

소스 텍스트 및 문자열 리터럴에는 유니코드로 인코딩된 문자가 포함될 수 있습니다. 6자의 유니코드 코드 포인트 이스케이프 시퀀스(예: \uXXXX)도 지원됩니다.

엄격 모드

함수는 기본적으로 엄격 모드에서 작동하므로 함수 코드에 `use strict` 명령문을 추가할 필요가 없습니다. 이것은 변경할 수 없습니다.

기본 객체

ES의 다음과 같은 기본 객체가 지원됩니다.

객체

객체에 대해 다음과 같은 ES 5.1 메서드가 지원됩니다.

- `create`(속성 목록 제외)
- `defineProperties`
- `defineProperty`
- `freeze`
- `getOwnPropertyDescriptor`
- `getOwnPropertyNames`

- getPrototypeOf
- hasOwnProperty
- isExtensible
- isFrozen
- prototype.isPrototypeOf
- isSealed
- keys
- preventExtensions
- prototype.propertyIsEnumerable
- seal
- prototype.toString
- prototype.valueOf

객체에 대해 다음 ES 6 메서드가 지원됩니다.

- assign
- is
- prototype.setPrototypeOf

객체에 대해 다음과 같은 ES 8 메서드가 지원됩니다.

- entries
- values

문자열

문자열에 대해 다음 ES 5.1 메서드가 지원됩니다.

- fromCharCode
- prototype.charAt
- prototype.concat
- prototype.indexOf
- prototype.lastIndexOf
- prototype.match
- prototype.replace

- `prototype.search`
- `prototype.slice`
- `prototype.split`
- `prototype.substr`
- `prototype.substring`
- `prototype.toLowerCase`
- `prototype.trim`
- `prototype.toUpperCase`

문자열에 대해 다음 ES 6 메서드가 지원됩니다.

- `fromCodePoint`
- `prototype.codePointAt`
- `prototype.endsWith`
- `prototype.includes`
- `prototype.repeat`
- `prototype.startsWith`

문자열에 대해 다음 ES 8 메서드가 지원됩니다.

- `prototype.padStart`
- `prototype.padEnd`

문자열에 대해 다음 ES 9 메서드가 지원됩니다.

- `prototype.trimStart`
- `prototype.trimEnd`

문자열에 대해 다음과 같은 비표준 메서드가 지원됩니다.

- `prototype.bytesFrom(`*`array | string, encoding`*`)`

  옥텟 배열 또는 인코딩된 문자열에서 바이트 문자열을 만듭니다. 문자열 인코딩 옵션은 `hex`, `base64` 및 `base64url`입니다.

- `prototype.fromBytes(`*`start`*`[, `*`end`*`])`

  각 바이트가 해당 유니코드 코드 포인트로 대체되는 바이트 문자열에서 유니코드 문자열을 만듭니다.

- `prototype.fromUTF8(start[, end])`

  UTF-8 인코딩된 바이트 문자열에서 유니코드 문자열을 만듭니다. 인코딩이 올바르지 않으면 `null`이(가) 반환됩니다.

- `prototype.toBytes(start[, end])`

  유니코드 문자열에서 바이트 문자열을 만듭니다. 모든 문자는 [0,255] 범위에 있어야 합니다. 그렇지 않은 경우 `null`이 반환됩니다.

- `prototype.toUTF8(start[, end])`

  유니코드 문자열에서 UTF-8 인코딩된 바이트 문자열을 만듭니다.

번호

숫자에 대한 모든 ES 5.1 메서드가 지원됩니다.

숫자에 대해 다음 ES 6 메서드가 지원됩니다.

- `isFinite`
- `isInteger`
- `isNaN`
- `isSafeInteger`
- `parseFloat`
- `parseInt`
- `prototype.toExponential`
- `prototype.toFixed`
- `prototype.toPrecision`
- `EPSILON`
- `MAX_SAFE_INTEGER`
- `MAX_VALUE`
- `MIN_SAFE_INTEGER`
- `MIN_VALUE`
- `NEGATIVE_INFINITY`
- `NaN`
- `POSITIVE_INFINITY`

## 기본 제공 객체

ES의 다음 내장 객체가 지원됩니다.

### 수학 연산

모든 ES 5.1 수학 메서드가 지원됩니다.

> **Note**
>
> CloudFront 함수 런타임 환경에서 `Math.random()` 구현은 함수가 실행될 때의 타임스탬프와 함께 시드된 OpenBSD `arc4random`을(를) 사용합니다.

다음 ES 6 수학 메서드가 지원됩니다.

- `acosh`
- `asinh`
- `atanh`
- `cbrt`
- `clz32`
- `cosh`
- `expm1`
- `fround`
- `hypot`
- `imul`
- `log10`
- `log1p`
- `log2`
- `sign`
- `sinh`
- `tanh`
- `trunc`
- `E`

- `LN10`
- `LN2`
- `LOG10E`
- `LOG2E`
- `PI`
- `SQRT1_2`
- `SQRT2`

날짜

모든 ES 5.1 `Date` 기능이 지원됩니다.

> **Note**
>
> 보안상의 이유로 단일 함수 실행의 수명 동안 `Date`은(는) 항상 동일한 값(함수의 시작 시간)을 반환합니다. 자세한 내용은 제한된 기능 단원을 참조하세요.

함수

`apply`, `bind` 및 `call` 메서드가 지원됩니다.

함수 생성자는 지원되지 않습니다.

정규식

모든 ES 5.1 정규 표현식 기능이 지원됩니다. 정규 표현식 언어는 Perl과 호환됩니다. ES 9 명명된 캡처 그룹이 지원됩니다.

JSON

`parse` 및 `stringify`을(를) 포함하여 모든 ES 5.1 JSON 기능이 지원됩니다.

배열

배열에 대해 다음 ES 5.1 메서드가 지원됩니다.

- `isArray`
- `prototype.concat`
- `prototype.every`
- `prototype.filter`

- `prototype.forEach`
- `prototype.indexOf`
- `prototype.join`
- `prototype.lastIndexOf`
- `prototype.map`
- `prototype.pop`
- `prototype.push`
- `prototype.reduce`
- `prototype.reduceRight`
- `prototype.reverse`
- `prototype.shift`
- `prototype.slice`
- `prototype.some`
- `prototype.sort`
- `prototype.splice`
- `prototype.unshift`

배열에 대해 다음 ES 6 메서드가 지원됩니다.

- `of`
- `prototype.copyWithin`
- `prototype.fill`
- `prototype.find`
- `prototype.findIndex`

배열에 대해 다음 ES 7 메서드가 지원됩니다.

- `prototype.includes`

형식화된 배열

다음 ES 6 형식화된 배열이 지원됩니다.

- `Int8Array`
- `Uint8Array`
- `Uint8ClampedArray`

- `Int16Array`
- `Uint16Array`
- `Int32Array`
- `Uint32Array`
- `Float32Array`
- `Float64Array`
- `prototype.copyWithin`
- `prototype.fill`
- `prototype.join`
- `prototype.set`
- `prototype.slice`
- `prototype.subarray`
- `prototype.toString`

배열 버퍼

`ArrayBuffer`에 대해 메서드가 지원됩니다.

- `prototype.isView`
- `prototype.slice`

Promise

Promise에 대해 다음 메서드가 지원됩니다.

- `reject`
- `resolve`
- `prototype.catch`
- `prototype.finally`
- `prototype.then`

crypto

암호화 모듈은 표준 해싱 및 HMAC(해시 기반 메시지 인증 코드) 헬퍼를 제공합니다. `require('crypto')`을(를) 사용하여 모듈을 로드할 수 있습니다. 모듈은 해당 Node.js 대응으로 정확하게 동작하는 다음과 같은 메서드를 노출합니다.

- `createHash(`*`algorithm`*`)`

- `hash.update(data)`
- `hash.digest([encoding])`
- `createHmac(algorithm, secret key)`
- `hmac.update(data)`
- `hmac.digest([encoding])`

자세한 내용은 기본 제공 모듈 단원의 암호화(해시 및 HMAC)를 참조하세요.

콘솔

디버깅을 위한 헬퍼 객체입니다. 로그 메시지를 기록하는 `log()` 메서드만 지원합니다.

> **Note**
>
> CloudFront 함수는 `console.log('a', 'b')`와 같은 쉼표 구문을 지원하지 않습니다. 대신 `console.log('a' + ' ' + 'b')` 형식을 사용합니다.

오류 유형

다음과 같은 오류 객체가 지원됩니다.

- `Error`
- `EvalError`
- `InternalError`
- `MemoryError`
- `RangeError`
- `ReferenceError`
- `SyntaxError`
- `TypeError`
- `URIError`

Globals

`globalThis` 객체가 지원됩니다.

다음 ES 5.1 전역 함수가 지원됩니다.

- `decodeURI`
- `decodeURIComponent`
- `encodeURI`
- `encodeURIComponent`
- `isFinite`
- `isNaN`
- `parseFloat`
- `parseInt`

다음과 같은 글로벌 제약이 지원됩니다.

- `NaN`
- `Infinity`
- `undefined`

기본 제공 모듈

다음 기본 제공 모듈이 지원됩니다.

모듈

- 암호화(해시 및 HMAC)
- 쿼리 문자열

암호화(해시 및 HMAC)

암호화 모듈(`crypto`)은 표준 해싱 및 HMAC(해시 기반 메시지 인증 코드) 헬퍼를 제공합니다. `require('crypto')`을(를) 사용하여 모듈을 로드할 수 있습니다. 이 모듈은 Node.js 대응물과 정확히 동일하게 동작하는 다음과 같은 메서드를 제공합니다.

해싱 메서드

`crypto.createHash(`*`algorithm`*`)`

지정된 알고리즘(`md5`, `sha1` 또는 `sha256`)을 사용하여 해시 다이제스트를 생성하는 데 사용할 수 있는 해시 객체를 생성하고 반환합니다.

`hash.update(data)`

지정된 `data`(으)로 해시 콘텐츠를 업데이트합니다.

`hash.digest([encoding])`

`hash.update()`을(를) 사용하여 전달된 모든 데이터의 다이제스트를 계산합니다. 인코딩은 `hex`, `base64` 또는 `base64url`일 수 있습니다.

### HMAC 메서드

`crypto.createHmac(algorithm, secret key)`

지정된 `algorithm` 및 `secret key`을(를) 사용하는 HMAC 객체를 생성 및 반환합니다. 알고리즘은 `md5`, `sha1` 또는 `sha256`일 수 있습니다.

`hmac.update(data)`

지정된 `data`(으)로 HMAC 콘텐츠를 업데이트합니다.

`hmac.digest([encoding])`

`hmac.update()`을(를) 사용하여 전달된 모든 데이터의 다이제스트를 계산합니다. 인코딩은 `hex`, `base64` 또는 `base64url`일 수 있습니다.

### 쿼리 문자열

> **Note**
>
> CloudFront 함수 이벤트 객체는 URL 쿼리 문자열을 자동으로 구문 분석합니다. 즉, 대부분의 경우 이 모듈을 사용할 필요가 없습니다.

쿼리 문자열 모듈(`querystring`)은 URL 쿼리 문자열을 구문 분석하고 서식을 지정하는 메서드를 제공합니다. `require('querystring')`을 사용하여 모듈을 로드할 수 있습니다. 이 모듈은 다음과 같은 방법을 제공합니다.

`querystring.escape(string)`

지정된 `string`을 URL-인코딩하여 이스케이프된 쿼리 문자열을 반환합니다. 이 방법은 `querystring.stringify()`에서 사용되며 직접 사용해서는 안 됩니다.

querystring.parse(*string*[, *separator*[, *equal*[, *options*]]])

쿼리 문자열(string)을 구문 분석하고 객체를 반환합니다.

separator 파라미터는 쿼리 문자열에서 키와 값 페어를 구분하기 위한 하위 문자열입니다. 기본 설정은 &입니다.

equal 파라미터는 쿼리 문자열에서 키와 값을 구분하기 위한 하위 문자열입니다. 기본 설정은 =입니다.

options 파라미터는 다음 키를 가진 객체입니다.

decodeURIComponent *function*

쿼리 문자열에서 백분율로 인코딩된 문자를 디코딩하는 함수입니다. 기본 설정은 querystring.unescape()입니다.

maxKeys *number*

구문 분석할 최대 키 수입니다. 기본 설정은 1000입니다. 0 값을 사용하여 키 카운트 제한을 제거합니다.

기본적으로 쿼리 문자열 내의 백분율로 인코딩된 문자는 UTF-8 인코딩을 사용하는 것으로 간주됩니다. 잘못된 UTF-8 시퀀스는 U+FFFD 대체 문자로 대체됩니다.

예를 들어 다음 쿼리 문자열의 경우:

```
'name=value&abc=xyz&abc=123'
```

querystring.parse()의 반환 값:

```
{
name: 'value',
abc: ['xyz', '123']
}
```

querystring.decode()는 querystring.parse()의 별칭입니다.

querystring.stringify(*object*[, *separator*[, *equal*[, *options*]]])

object를 직렬화하고 쿼리 문자열을 반환합니다.

separator 파라미터는 쿼리 문자열에서 키와 값 페어를 구분하기 위한 하위 문자열입니다. 기본 설정은 &입니다.

`equal` 파라미터는 쿼리 문자열에서 키와 값을 구분하기 위한 하위 문자열입니다. 기본 설정은 `=`입니다.

`options` 파라미터는 다음 키를 가진 객체입니다.

`encodeURIComponent` *function*

URL에 안전하지 않은 문자를 쿼리 문자열에서 백분율 인코딩으로 변환하는 데 사용할 함수입니다. 기본 설정은 `querystring.escape()`입니다.

기본적으로 쿼리 문자열 내에서 백분율 인코딩이 필요한 문자는 UTF-8로 인코딩됩니다. 다른 인코딩을 사용하려면 `encodeURIComponent` 옵션을 지정합니다.

예를 들어, 다음 코드의 경우:

```
querystring.stringify({ name: 'value', abc: ['xyz', '123'], anotherName: '' });
```

반환 값:

```
'name=value&abc=xyz&abc=123&anotherName='
```

`querystring.encode()`는 `querystring.stringify()`의 별칭입니다.

`querystring.unescape(`*`string`*`)`

지정된 `string`의 URL 백분율로 인코딩된 문자를 디코딩하여 이스케이프되지 않은 쿼리 문자열을 반환합니다. 이 방법은 `querystring.parse()`에서 사용되며 직접 사용해서는 안 됩니다.

### 제한된 기능

다음 JavaScript 언어 기능은 보안 문제로 인해 지원되지 않거나 제한됩니다.

#### 동적 코드 평가

동적 코드 평가는 지원되지 않습니다. 시도하면 `eval()` 및 `Function` 생성자 모두 오류가 발생합니다. 예를 들어 `const sum = new Function('a', 'b', 'return a + b')`에서 오류가 발생합니다.

#### 타이머

`setTimeout()`, `setImmediate()` 및 `clearTimeout()` 함수는 지원되지 않습니다. 함수 실행 내에서 연기하거나 출력할 프로비전이 없습니다. 함수가 완료되려면 동기적으로 실행되어야 합니다.

날짜 및 타임스탬프

보안상의 이유로 고해상도 타이머에 액세스할 수 없습니다. 현재 시간을 쿼리하는 모든 `Date` 메서드는 단일 함수 실행의 수명 동안 항상 동일한 값을 반환합니다. 반환된 타임 스탬프는 함수가 실행을 시작한 시간입니다. 따라서 함수에서 경과 시간을 측정할 수 없습니다.

파일 시스템 액세스

파일 시스템 액세스가 없습니다. 예를 들어 Node.js에서처럼 파일 시스템 액세스를 위한 `fs` 모듈이 없습니다.

프로세스 액세스

프로세스 액세스가 없습니다. 예를 들어 Node.js와는 달리 정보 액세스를 처리하기 위한 `process` 글로벌 객체가 없습니다.

환경 변수

환경 변수에 액세스할 수 없습니다.

대신 CloudFront KeyValueStore를 사용하여 CloudFront Functions에 대한 키 값 페어의 중앙 집중식 데이터 스토어를 생성할 수 있습니다. CloudFront KeyValueStore를 사용하면 코드 변경 사항을 배포할 필요 없이 구성 데이터를 동적으로 업데이트할 수 있습니다. CloudFront KeyValueStore를 사용하려면 JavaScript 런타임 2.0을 사용해야 합니다. 자세한 내용은 Amazon CloudFront KeyValueStore 단원을 참조하십시오.

네트워크 액세스

네트워크 통화에 대한 지원이 없습니다. 예를 들어 XHR, HTTP(S) 및 소켓은 지원되지 않습니다.

## CloudFront Functions를 위한 JavaScript 런타임 2.0 기능

CloudFront Functions JavaScript 런타임 환경은 ECMAScript(ES) 버전 5.1을 준수하며, ES 버전 6에서 12까지의 일부 기능을 지원합니다. 또한 ES 사양의 일부가 아닌 일부 비표준 방법을 제공합니다. 다음 주제에는 이 런타임에서 지원되는 모든 기능이 나열되어 있습니다.

주제


- 핵심 기능
- 기본 객체
- 기본 제공 객체
- 오류 유형

- Globals
- 기본 제공 모듈
- 제한된 기능

핵심 기능

ES의 다음과 같은 핵심 기능이 지원됩니다.

유형

모든 ES 5.1 유형이 지원됩니다. 여기에는 부울 값, 숫자, 문자열, 객체, 배열, 함수, 정규 표현식이 포함됩니다.

연산자

모든 ES 5.1 연산자가 지원됩니다.

ES 7 거듭제곱 연산자(`**`)가 지원됩니다.

Statement

다음 ES 5.1 문이 지원됩니다.

- `break`
- `catch`
- `continue`
- `do-while`
- `else`
- `finally`
- `for`
- `for-in`
- `if`
- `label`
- `return`
- `switch`
- `throw`
- `try`

- `var`
- `while`

다음 ES 6 문이 지원됩니다.

- `const`
- `let`

다음 ES 8 문이 지원됩니다.

- `async`
- `await`

> **Note**
>
> `async`, `await`, `const` 및 `let`가 JavaScript 런타임 2.0에서 지원됩니다.
> `await`는 `async` 함수 내에서만 사용할 수 있습니다. `async` 인수 및 종료는 지원되지 않습니다.

리터럴

ES 6 템플릿 리터럴은 여러 줄 문자열, 표현식 보간 및 중첩 템플릿에 대해 지원됩니다..

함수

모든 ES 5.1 함수 기능이 지원됩니다.

ES 6 화살표 함수가 지원되며 ES 6 나머지 파라미터 구문이 지원됩니다.

유니코드

소스 텍스트 및 문자열 리터럴에는 유니코드로 인코딩된 문자가 포함될 수 있습니다. 6자의 유니코드 코드 포인트 이스케이프 시퀀스(예: \uXXXX)도 지원됩니다.

엄격 모드

함수는 기본적으로 엄격 모드에서 작동하므로 함수 코드에 `use strict` 명령문을 추가할 필요가 없습니다. 이것은 변경할 수 없습니다.

기본 객체

ES의 다음과 같은 기본 객체가 지원됩니다.

## 객체

객체에 대해 다음과 같은 ES 5.1 메서드가 지원됩니다.

- `Object.create()`(속성 목록 제외)
- `Object.defineProperties()`
- `Object.defineProperty()`
- `Object.freeze()`
- `Object.getOwnPropertyDescriptor()`
- `Object.getOwnPropertyDescriptors()`
- `Object.getOwnPropertyNames()`
- `Object.getPrototypeOf()`
- `Object.isExtensible()`
- `Object.isFrozen()`
- `Object.isSealed()`
- `Object.keys()`
- `Object.preventExtensions()`
- `Object.seal()`

객체에 대해 다음 ES 6 메서드가 지원됩니다.

- `Object.assign()`

객체에 대해 다음과 같은 ES 8 메서드가 지원됩니다.

- `Object.entries()`
- `Object.values()`

객체에 대해 다음과 같은 ES 5.1 프로토타입 메서드가 지원됩니다.

- `Object.prototype.hasOwnProperty()`
- `Object.prototype.isPrototypeOf()`
- `Object.prototype.propertyIsEnumerable()`
- `Object.prototype.toString()`
- `Object.prototype.valueOf()`

객체에 대해 다음과 같은 ES 6 프로토타입 메서드가 지원됩니다.

- `Object.prototype.is()`
- `Object.prototype.setPrototypeOf()`

String

문자열에 대해 다음 ES 5.1 메서드가 지원됩니다.

- `String.fromCharCode()`

문자열에 대해 다음 ES 6 메서드가 지원됩니다.

- `String.fromCodePoint()`

문자열에 대해 다음 ES 5.1 프로토타입 메서드가 지원됩니다.

- `String.prototype.charAt()`
- `String.prototype.concat()`
- `String.prototype.indexOf()`
- `String.prototype.lastIndexOf()`
- `String.prototype.match()`
- `String.prototype.replace()`
- `String.prototype.search()`
- `String.prototype.slice()`
- `String.prototype.split()`
- `String.prototype.substr()`
- `String.prototype.substring()`
- `String.prototype.toLowerCase()`
- `String.prototype.trim()`
- `String.prototype.toUpperCase()`

문자열에 대해 다음 ES 6 프로토타입 메서드가 지원됩니다.

- `String.prototype.codePointAt()`
- `String.prototype.endsWith()`
- `String.prototype.includes()`
- `String.prototype.repeat()`
- `String.prototype.startsWith()`

문자열에 대해 다음 ES 8 프로토타입 메서드가 지원됩니다.

- `String.prototype.padStart()`
- `String.prototype.padEnd()`

문자열에 대해 다음 ES 9 프로토타입 메서드가 지원됩니다.

- `String.prototype.trimStart()`
- `String.prototype.trimEnd()`

문자열에 대해 다음 ES 12 프로토타입 메서드가 지원됩니다.

- `String.prototype.replaceAll()`

> **Note**
>
> `String.prototype.replaceAll()`이 JavaScript 런타임 2.0에 새로 추가되었습니다.

숫자

모든 ES 5.1 숫자가 지원됩니다.

숫자에 대해 다음 ES 6 속성이 지원됩니다.

- `Number.EPSILON`
- `Number.MAX_SAFE_INTEGER`
- `Number.MIN_SAFE_INTEGER`
- `Number.MAX_VALUE`
- `Number.MIN_VALUE`
- `Number.NaN`
- `Number.NEGATIVE_INFINITY`
- `Number.POSITIVE_INFINITY`

숫자에 대해 다음 ES 6 메서드가 지원됩니다.

- `Number.isFinite()`
- `Number.isInteger()`
- `Number.isNaN()`
- `Number.isSafeInteger()`

- `Number.parseInt()`
- `Number.parseFloat()`

숫자에 대해 다음 ES 5.1 프로토타입 메서드가 지원됩니다.

- `Number.prototype.toExponential()`
- `Number.prototype.toFixed()`
- `Number.prototype.toPrecision()`

ES 12 숫자 구분자가 지원됩니다.

> **Note**
> ES 12 숫자 구분자가 JavaScript 런타임 2.0에 새로 추가되었습니다.

기본 제공 객체

ES의 다음 내장 객체가 지원됩니다.

수학 연산

모든 ES 5.1 수학 메서드가 지원됩니다.

> **Note**
> CloudFront 함수 런타임 환경에서 `Math.random()` 구현은 함수가 실행될 때의 타임스탬프와 함께 시드된 OpenBSD `arc4random`을(를) 사용합니다.

다음 ES 6 수학 속성이 지원됩니다.

- `Math.E`
- `Math.LN10`
- `Math.LN2`
- `Math.LOG10E`
- `Math.LOG2E`
- `Math.PI`

- `Math.SQRT1_2`
- `Math.SQRT2`

다음 ES 6 수학 메서드가 지원됩니다.

- `Math.abs()`
- `Math.acos()`
- `Math.acosh()`
- `Math.asin()`
- `Math.asinh()`
- `Math.atan()`
- `Math.atan2()`
- `Math.atanh()`
- `Math.cbrt()`
- `Math.ceil()`
- `Math.clz32()`
- `Math.cos()`
- `Math.cosh()`
- `Math.exp()`
- `Math.expm1()`
- `Math.floor()`
- `Math.fround()`
- `Math.hypot()`
- `Math.imul()`
- `Math.log()`
- `Math.log1p()`
- `Math.log2()`
- `Math.log10()`
- `Math.max()`
- `Math.min()`
- `Math.pow()`

- `Math.random()`
- `Math.round()`
- `Math.sign()`
- `Math.sinh()`
- `Math.sin()`
- `Math.sqrt()`
- `Math.tan()`
- `Math.tanh()`
- `Math.trunc()`

날짜

모든 ES 5.1 `Date` 기능이 지원됩니다.

> **Note**
> 보안상의 이유로 단일 함수 실행의 수명 동안 `Date`은(는) 항상 동일한 값(함수의 시작 시간)을 반환합니다. 자세한 내용은 제한된 기능 단원을 참조하세요.

함수

다음 ES 5.1 프로토타입 메서드가 지원됩니다.

- `Function.prototype.apply()`
- `Function.prototype.bind()`
- `Function.prototype.call()`

함수 생성자는 지원되지 않습니다.

정규식

모든 ES 5.1 정규 표현식 기능이 지원됩니다. 정규 표현식 언어는 Perl과 호환됩니다.

다음 ES 5.1 프로토타입 접근자 속성이 지원됩니다.

- `RegExp.prototype.global`
- `RegExp.prototype.ignoreCase`
- `RegExp.protoype.multiline`

- `RegExp.protoype.source`
- `RegExp.prototype.sticky`
- `RegExp.prototype.flags`

> **Note**
>
> `RegExp.prototype.sticky`, `RegExp.prototype.flags`가 JavaScript 런타임 2.0에 새로 추가되었습니다.

다음 ES 5.1 프로토타입 메서드가 지원됩니다.

- `RegExp.prototype.exec()`
- `RegExp.prototype.test()`
- `RegExp.prototype.toString()`
- `RegExp.prototype[@@replace]()`
- `RegExp.prototype[@@split]()`

> **Note**
>
> `RegExp.prototype[@@split]()`이 JavaScript 런타임 2.0에 새로 추가되었습니다.

다음 ES 5.1 인스턴스 속성이 지원됩니다.

- `lastIndex`

ES 9 명명된 캡처 그룹이 지원됩니다.

JSON

다음 ES 5.1 메서드가 지원됩니다.

- `JSON.parse()`
- `JSON.stringify()`

배열

배열에 대해 다음 ES 5.1 메서드가 지원됩니다.

- `Array.isArray()`

배열에 대해 다음 ES 6 메서드가 지원됩니다.

- `Array.of()`

다음 ES 5.1 프로토타입 메서드가 지원됩니다.

- `Array.prototype.concat()`
- `Array.prototype.every()`
- `Array.prototype.filter()`
- `Array.prototype.forEach()`
- `Array.prototype.indexOf()`
- `Array.prototype.join()`
- `Array.prototype.lastIndexOf()`
- `Array.prototype.map()`
- `Array.prototype.pop()`
- `Array.prototype.push()`
- `Array.prototype.reduce()`
- `Array.prototype.reduceRight()`
- `Array.prototype.reverse()`
- `Array.prototype.shift()`
- `Array.prototype.slice()`
- `Array.prototype.some()`
- `Array.prototype.sort()`
- `Array.prototype.splice()`
- `Array.prototype.unshift()`

다음 ES 6 프로토타입 메서드가 지원됩니다.

- `Array.prototype.copyWithin()`
- `Array.prototype.fill()`
- `Array.prototype.find()`
- `Array.prototype.findIndex()`

다음 ES 7 프로토타입 메서드가 지원됩니다.

- `Array.prototype.includes()`

## 형식화된 배열

다음 ES 6 형식화된 배열 생성자가 지원됩니다.

- `Float32Array`
- `Float64Array`
- `Int8Array`
- `Int16Array`
- `Int32Array`
- `Uint8Array`
- `Uint8ClampedArray`
- `Uint16Array`
- `Uint32Array`

다음 ES 6 메서드가 지원됩니다.

- `TypedArray.from()`
- `TypedArray.of()`

> **Note**
>
> `TypedArray.from()`, `TypedArray.of()`가 JavaScript 런타임 2.0에 새로 추가되었습니다.

다음 ES 6 프로토타입 메서드가 지원됩니다.

- `TypedArray.prototype.copyWithin()`
- `TypedArray.prototype.every()`
- `TypedArray.prototype.fill()`
- `TypedArray.prototype.filter()`
- `TypedArray.prototype.find()`
- `TypedArray.prototype.findIndex()`
- `TypedArray.prototype.forEach()`
- `TypedArray.prototype.includes()`
- `TypedArray.prototype.indexOf()`

- `TypedArray.prototype.join()`
- `TypedArray.prototype.lastIndexOf()`
- `TypedArray.prototype.map()`
- `TypedArray.prototype.reduce()`
- `TypedArray.prototype.reduceRight()`
- `TypedArray.prototype.reverse()`
- `TypedArray.prototype.some()`
- `TypedArray.prototype.set()`
- `TypedArray.prototype.slice()`
- `TypedArray.prototype.sort()`
- `TypedArray.prototype.subarray()`
- `TypedArray.prototype.toString()`

> **Note**
>
> `TypedArray.prototype.every()`, `TypedArray.prototype.fill()`, `TypedArray.prototype.filter()`, `TypedArray.prototype.find()`, `TypedArray.prototype.findIndex()`, `TypedArray.prototype.forEach()`, `TypedArray.prototype.includes()`, `TypedArray.prototype.indexOf()`, `TypedArray.prototype.join()`, `TypedArray.prototype.lastIndexOf()`, `TypedArray.prototype.map()`, `TypedArray.prototype.reduce()`, `TypedArray.prototype.reduceRight()`, `TypedArray.prototype.reverse()`, `TypedArray.prototype.some()`이 JavaScript 런타임 2.0에 새로 추가되었습니다.

배열 버퍼

ArrayBuffer에 대해 다음 ES 6 메서드가 지원됩니다.

- `isView()`

ArrayBuffer에 대해 다음 ES 6 프로토타입 메서드가 지원됩니다.

- `ArrayBuffer.prototype.slice()`

Promise

프로미스에 대해 다음 ES 6 메서드가 지원됩니다.

- `Promise.all()`
- `Promise.allSettled()`
- `Promise.any()`
- `Promise.reject()`
- `Promise.resolve()`
- `Promise.race()`

> **Note**
>
> `Promise.all()`, `Promise.allSettled()`, `Promise.any()`, `Promise.race()`가 JavaScript 런타임 2.0에 새로 추가되었습니다.

프로미스에 대해 다음 ES 6 프로토타입 메서드가 지원됩니다.

- `Promise.prototype.catch()`
- `Promise.prototype.finally()`
- `Promise.prototype.then()`

DataView

다음 ES 6 프로토타입 메서드가 지원됩니다.

- `DataView.prototype.getFloat32()`
- `DataView.prototype.getFloat64()`
- `DataView.prototype.getInt16()`
- `DataView.prototype.getInt32()`
- `DataView.prototype.getInt8()`
- `DataView.prototype.getUint16()`
- `DataView.prototype.getUint32()`
- `DataView.prototype.getUint8()`
- `DataView.prototype.setFloat32()`
- `DataView.prototype.setFloat64()`
- `DataView.prototype.setInt16()`
- `DataView.prototype.setInt32()`
- `DataView.prototype.setInt8()`

- `DataView.prototype.setUint16()`
- `DataView.prototype.setUint32()`
- `DataView.prototype.setUint8()`

> **Note**
>
> 모든 Dataview ES 6 프로토타입 메서드가 JavaScript 런타임 2.0에 새로 추가되었습니다.

Symbol

다음 ES 6 메서드가 지원됩니다.

- `Symbol.for()`
- `Symbol.keyfor()`

> **Note**
>
> 모든 Symbol ES 6 메서드가 JavaScript 런타임 2.0에 새로 추가되었습니다.

텍스트 디코더

다음 프로토타입 메서드가 지원됩니다.

- `TextDecoder.prototype.decode()`

다음 프로토타입 접근자 속성이 지원됩니다.

- `TextDecoder.prototype.encoding`
- `TextDecoder.prototype.fatal`
- `TextDecoder.prototype.ignoreBOM`

텍스트 인코더

다음 프로토타입 메서드가 지원됩니다.

- `TextEncoder.prototype.encode()`
- `TextEncoder.prototype.encodeInto()`

## 오류 유형

다음과 같은 오류 객체가 지원됩니다.

- `Error`
- `EvalError`
- `InternalError`
- `RangeError`
- `ReferenceError`
- `SyntaxError`
- `TypeError`
- `URIError`

### Globals

`globalThis` 객체가 지원됩니다.

다음 ES 5.1 전역 함수가 지원됩니다.

- `decodeURI()`
- `decodeURIComponent()`
- `encodeURI()`
- `encodeURIComponent()`
- `isFinite()`
- `isNaN()`
- `parseFloat()`
- `parseInt()`

다음 ES 6 전역 함수가 지원됩니다.

- `atob()`
- `btoa()`

> **Note**
>
> `atob()`, `btoa()`가 JavaScript 런타임 2.0에 새로 추가되었습니다.

다음과 같은 글로벌 제약이 지원됩니다.

- `NaN`
- `Infinity`
- `undefined`
- `arguments`

### 기본 제공 모듈

다음 기본 제공 모듈이 지원됩니다.

모듈

- Buffer
- 쿼리 문자열
- crypto

### Buffer

이 모듈은 다음과 같은 메서드를 제공합니다.

- `Buffer.alloc(size[, fill[, encoding]])`

  `Buffer`를 할당합니다.

  - `size`: 버퍼 크기입니다. 정수를 입력합니다.
  - `fill`: 선택 사항입니다. 문자열, `Buffer`, Uint8Array 또는 정수를 입력합니다. 기본값은 `0`입니다.
  - `encoding`: 선택 사항입니다. `fill`가 문자열인 경우 `utf8`, `hex`, `base64`, `base64url` 중 하나를 입력합니다. 기본값은 `utf8`입니다.
- `Buffer.allocUnsafe(size)`

  초기화되지 않은 `Buffer` 값을 할당합니다.

  - `size`: 정수를 입력합니다.
- `Buffer.byteLength(value[, encoding])`

  값의 길이를 바이트 단위로 반환합니다.

  - `value`: 문자열, `Buffer`, TypedArray, Dataview 또는 Arraybuffer입니다.

  - encoding: 선택 사항입니다. value가 문자열인 경우 utf8, hex, base64, base64url 중 하나를 입력합니다. 기본값은 utf8입니다.
- Buffer.compare(buffer1, buffer2)

  두 Buffer를 비교하면 배열을 정렬하는 데 도움이 됩니다. 두 값이 같으면 0, buffer1이 먼저 오면 -1, buffer2가 먼저 오면 1을 반환합니다.
  - buffer1: Buffer를 입력합니다.
  - buffer2: 다른 Buffer를 입력합니다.
- Buffer.concat(list[, totalLength])

  여러 개의 Buffer를 연결합니다. 없으면 0을 반환합니다. totalLength까지 반환합니다.
  - list: Buffer 목록을 입력합니다. 이 항목은 totalLength 기준으로 잘려서 표시된다는 점에 유의하세요.
  - totalLength: 선택 사항입니다. 부호 없는 정수를 입력합니다. 비어 있는 경우 목록에 있는 Buffer 인스턴스의 합계를 사용하세요.
- Buffer.from(array)

  배열에서 Buffer를 생성합니다.
  - array: 0에서 255까지 바이트 배열을 입력합니다.
- Buffer.from(arrayBuffer, byteOffset[, length]))

  byteOffset 오프셋에서 시작하여 length 길이를 기준으로 arrayBuffer에서 뷰를 생성합니다.
  - arrayBuffer: Buffer 배열을 입력합니다.
  - byteOffset: 정수를 입력합니다.
  - length: 선택 사항입니다. 정수를 입력합니다.
- Buffer.from(buffer)

  Buffer의 사본을 생성합니다.
  - buffer: Buffer를 입력합니다.
- Buffer.from(object[, offsetOrEncoding[, length]])

  객체에서 Buffer를 생성합니다. valueOf() 값이 객체와 동일하지 않으면 Buffer.from(object.valueOf(), offsetOrEncoding, length) 값을 반환합니다.
  - object: 객체를 입력합니다.
  - offsetOrEncoding: 선택 사항입니다. 정수 또는 인코딩 문자열을 입력합니다.

  - `length`: 선택 사항입니다. 정수를 입력합니다.
- `Buffer.from(string[, encoding])`

  문자열에서 `Buffer`를 생성합니다.

  - `string`: 문자열을 입력합니다.
  - `encoding`: 선택 사항입니다. `utf8`, `hex`, `base64`, `base64url` 중 하나를 입력합니다. 기본값은 `utf8`입니다.
- `Buffer.isBuffer(object)`

  `object`가 버퍼인지 확인합니다. `true` 또는 `false`를 반환합니다.

  - `object`: 객체를 입력합니다.
- `Buffer.isEncoding(encoding)`

  `encoding`이 지원되는지 확인합니다. `true` 또는 `false`를 반환합니다.

  - `encoding`: 선택 사항입니다. `utf8`, `hex`, `base64`, `base64url` 중 하나를 입력합니다. 기본값은 `utf8`입니다.

이 모듈은 다음과 같은 버퍼 프로토타입 메서드를 제공합니다.

- `Buffer.prototype.compare(target[, targetStart[, targetEnd[, sourceStart[, sourceEnd]]]])`

  `Buffer`를 대상과 비교합니다. 두 값이 같으면 `0`, `buffer`이 먼저 오면 `1`, `target`가 먼저 오면 `-1`을 반환합니다.

  - `target`: `Buffer`를 입력합니다.
  - `targetStart`: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.
  - `targetEnd`: 선택 사항입니다. 정수를 입력합니다. 기본값은 `target` 길이입니다.
  - `sourceStart`: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.
  - `sourceEnd`: 선택 사항입니다. 정수를 입력합니다. 기본값은 `Buffer` 길이입니다.
- `Buffer.prototype.copy(target[, targetStart[, sourceStart[, sourceEnd]]])`

  버퍼를 `target`으로 복사합니다.

  - `target`: `Buffer` 또는 `Uint8Array`를 입력합니다.
  - `targetStart`: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.
  - `sourceStart`: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.

  - `sourceEnd`: 선택 사항입니다. 정수를 입력합니다. 기본값은 `Buffer` 길이입니다.
- `Buffer.prototype.equals(otherBuffer)`

  `Buffer`와 `otherBuffer`를 비교합니다. `true` 또는 `false`를 반환합니다.

  - `otherBuffer`: 문자열을 입력합니다.
- `Buffer.prototype.fill(value[, offset[, end][, encoding])`

  `Buffer`를 `value`로 채웁니다.

  - `value`: 문자열, `Buffer` 또는 정수를 입력합니다.
  - `offset`: 선택 사항입니다. 정수를 입력합니다.
  - `end`: 선택 사항입니다. 정수를 입력합니다.
  - `encoding`: 선택 사항입니다. `utf8`, `hex`, `base64`, `base64url` 중 하나를 입력합니다. 기본값은 `utf8`입니다.
- `Buffer.prototype.includes(value[, byteOffset][, encoding])`

  `Buffer`에서 `value`를 찾습니다. `true` 또는 `false`를 반환합니다.

  - `value`: 문자열, `Buffer`, `Uint8Array` 또는 정수를 입력합니다.
  - `byteOffset`: 선택 사항입니다. 정수를 입력합니다.
  - `encoding`: 선택 사항입니다. `utf8`, `hex`, `base64`, `base64url` 중 하나를 입력합니다. 기본값은 `utf8`입니다.
- `Buffer.prototype.indexOf(value[, byteOffset][, encoding])`

  `Buffer`에서 첫 번째 `value`를 찾습니다. 값이 발견되면 `index`, 발견되지 않으면 `-1`을 반환합니다.

  - `value`: 문자열, `Buffer`, Unit8Array 또는 0에서 255 사이의 정수를 입력합니다.
  - `byteOffset`: 선택 사항입니다. 정수를 입력합니다.
  - `encoding`: 선택 사항입니다. `value`가 문자열인 경우 `utf8`, `hex`, `base64`, `base64url` 중 하나를 입력합니다. 기본값은 `utf8`입니다.
- `Buffer.prototype.lastIndexOf(value[, byteOffset][, encoding])`

  `Buffer`에서 마지막 `value`를 찾습니다. 값이 발견되면 `index`, 발견되지 않으면 `-1`을 반환합니다.

  - `value`: 문자열, `Buffer`, Unit8Array 또는 0에서 255 사이의 정수를 입력합니다.
  - `byteOffset`: 선택 사항입니다. 정수를 입력합니다.
  - `encoding`: 선택 사항입니다. `value`가 문자열인 경우 `utf8`, `hex`, `base64`, `base64url` 중 하나를 입력합니다. 기본값은 `utf8`입니다.

- `Buffer.prototype.readInt8(offset)`

  `Buffer`의 `offset`에서 `Int8`을 읽습니다.

  - `offset`: 정수를 입력합니다.
- `Buffer.prototype.readIntBE(offset, byteLength)`

  `Buffer`의 `offset`에서 `Int`를 빅 엔디안으로 읽습니다.

  - `offset`: 정수를 입력합니다.
  - `byteLength`: 선택 사항입니다. 1에서 6까지의 정수를 입력합니다.
- `Buffer.prototype.readInt16BE(offset)`

  `Buffer`의 `offset`에서 `Int16`을 빅 엔디안으로 읽습니다.

  - `offset`: 정수를 입력합니다.
- `Buffer.prototype.readInt32BE(offset)`

  `Buffer`의 `offset`에서 `Int32`를 빅 엔디안으로 읽습니다.

  - `offset`: 정수를 입력합니다.
- `Buffer.prototype.readIntLE(offset, byteLength)`

  `Buffer`의 `offset`에서 `Int`을 리틀 엔디안으로 읽습니다.

  - `offset`: 정수를 입력합니다.
  - `byteLength`: 1에서 6까지의 정수를 입력합니다.
- `Buffer.prototype.readInt16LE(offset)`

  `Buffer`의 `offset`에서 `Int16`을 리틀 엔디안으로 읽습니다.

  - `offset`: 정수를 입력합니다.
- `Buffer.prototype.readInt32LE(offset)`

  `Buffer`의 `offset`에서 `Int32`을 리틀 엔디안으로 읽습니다.

  - `offset`: 정수를 입력합니다.
- `Buffer.prototype.readUInt8(offset)`

  `Buffer`의 `offset`에서 `UInt8`을 읽습니다.

  - `offset`: 정수를 입력합니다.
- `Buffer.prototype.readUIntBE(offset, byteLength)`

Buffer의 offset에서 UInt를 빅 엔디안으로 읽습니다.

- offset: 정수를 입력합니다.
- byteLength: 1에서 6까지의 정수를 입력합니다.

- Buffer.prototype.readUInt16BE(offset)

  Buffer의 offset에서 UInt16을 빅 엔디안으로 읽습니다.

- • offset: 정수를 입력합니다.
- Buffer.prototype.readUInt32BE(offset)

  Buffer의 offset에서 UInt32를 빅 엔디안으로 읽습니다.

  - offset: 정수를 입력합니다.

- Buffer.prototype.readUIntLE(offset, byteLength)

  Buffer의 offset에서 UInt을 리틀 엔디안으로 읽습니다.

  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.

- Buffer.prototype.readUInt16LE(offset)

  Buffer의 offset에서 UInt16을 리틀 엔디안으로 읽습니다.

  - offset: 정수를 입력합니다.

- Buffer.prototype.readUInt32LE(offset)

  Buffer의 offset에서 UInt32을 리틀 엔디안으로 읽습니다.

  - offset: 정수를 입력합니다.

- Buffer.prototype.readDoubleBE([offset])

  Buffer의 offset에서 64비트 배정도를 빅 엔디안으로 읽습니다.

  - offset: 선택 사항입니다. 정수를 입력합니다.

- Buffer.prototype.readDoubleLE([offset])

  Buffer의 offset에서 64비트 배정도를 리틀 엔디안으로 읽습니다.

  - offset: 선택 사항입니다. 정수를 입력합니다.

- Buffer.prototype.readFloatBE([offset])

  Buffer의 offset에서 32비트 부동소수점을 빅 엔디안으로 읽습니다.

  - offset: 선택 사항입니다. 정수를 입력합니다.
- Buffer.prototype.readFloatLE([offset])

  Buffer의 offset에서 32비트 부동소수점을 리틀 엔디안으로 읽습니다.

  - offset: 선택 사항입니다. 정수를 입력합니다.
- Buffer.prototype.subarray([start[, end]])

  오프셋되고 새 start 및 end로 크롭된 Buffer의 복사본을 반환합니다.

  - start: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.
  - end: 선택 사항입니다. 정수를 입력합니다. 기본값은 버퍼 길이입니다.
- Buffer.prototype.swap16()

  Buffer 배열을 16비트 숫자로 구성된 배열로 취급하여 바이트 순서를 바꿉니다. Buffer 길이는 2로 나눌 수 있어야 합니다. 그렇지 않으면 오류가 발생합니다.
- Buffer.prototype.swap32()

  Buffer 배열을 32비트 숫자로 구성된 배열로 취급하여 바이트 순서를 바꿉니다. Buffer 길이는 4로 나눌 수 있어야 합니다. 그렇지 않으면 오류가 발생합니다.
- Buffer.prototype.swap64()

  Buffer 배열을 64비트 숫자로 구성된 배열로 취급하여 바이트 순서를 바꿉니다. Buffer 길이는 8로 나눌 수 있어야 합니다. 그렇지 않으면 오류가 발생합니다.
- Buffer.prototype.toJSON()

  Buffer를 JSON으로 반환합니다.
- Buffer.prototype.toString([encoding[, start[, end]]])

  start에서 end로, 인코딩된 문자열로 Buffer를 변환합니다.

  - encoding: 선택 사항입니다. utf8, hex, base64, base64url 중 하나를 입력합니다. 기본값은 utf8입니다.
  - start: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.
  - end: 선택 사항입니다. 정수를 입력합니다. 기본값은 버퍼 길이입니다.
- Buffer.prototype.write(string[, offset[, length]][, encoding])

  공간이 있는 경우 인코딩된 string을 Buffer에 쓰고, 공간이 충분하지 않으면 잘라낸 string을 씁니다.

  - string: 문자열을 입력합니다.
  - offset: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.
  - length: 선택 사항입니다. 정수를 입력합니다. 기본값은 문자열 길이입니다.
  - encoding: 선택 사항입니다. 필요에 따라 utf8, hex, base64, base64url 중 하나를 입력합니다. 기본값은 utf8입니다.
- Buffer.prototype.writeInt8(value, offset, byteLength)

  offset에서 byteLength의 Int8 value를 Buffer에 씁니다.
  - value: 정수를 입력합니다.
  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeIntBE(value, offset, byteLength)

  빅 엔디안을 사용하여 offset에서 value를 Buffer에 씁니다.
  - value: 정수를 입력합니다.
  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeInt16BE(value, offset, byteLength)

  빅 엔디안을 사용하여 offset에서 value를 Buffer에 씁니다.
  - value: 정수를 입력합니다.
  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeInt32BE(value, offset, byteLength)

  빅 엔디안을 사용하여 offset에서 value를 Buffer에 씁니다.
  - value: 정수를 입력합니다.
  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeIntLE(offset, byteLength)

  리틀 엔디안을 사용하여 offset에서 value를 Buffer에 씁니다.
  - offset: 정수를 입력합니다.

  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeInt16LE(offset, byteLength)

  리틀 엔디안을 사용하여 offset에서 value를 Buffer에 씁니다.
  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeInt32LE(offset, byteLength)

  리틀 엔디안을 사용하여 offset에서 value를 Buffer에 씁니다.
  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeUInt8(value, offset, byteLength)

  offset에서 byteLength의 UInt8 value를 Buffer에 씁니다.
  - value: 정수를 입력합니다.
  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeUIntBE(value, offset, byteLength)

  빅 엔디안을 사용하여 offset에서 value를 Buffer에 씁니다.
  - value: 정수를 입력합니다.
  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeUInt16BE(value, offset, byteLength)

  빅 엔디안을 사용하여 offset에서 value를 Buffer에 씁니다.
  - value: 정수를 입력합니다.
  - offset: 정수를 입력합니다.
  - byteLength: 1에서 6까지의 정수를 입력합니다.
- Buffer.prototype.writeUInt32BE(value, offset, byteLength)

  빅 엔디안을 사용하여 offset에서 value를 Buffer에 씁니다.
  - value: 정수를 입력합니다.
  - offset: 정수를 입력합니다.

  - `byteLength`: 1에서 6까지의 정수를 입력합니다.
- `Buffer.prototype.writeUIntLE(value, offset, byteLength)`

  리틀 엔디안을 사용하여 `offset`에서 `value`를 `Buffer`에 씁니다.
  - `value`: 정수를 입력합니다.
  - `offset`: 정수를 입력합니다.
  - `byteLength`: 1에서 6까지의 정수를 입력합니다.
- `Buffer.prototype.writeUInt16LE(value, offset, byteLength)`

  리틀 엔디안을 사용하여 `offset`에서 `value`를 `Buffer`에 씁니다.
  - `value`: 정수를 입력합니다.
  - `offset`: 정수를 입력합니다.
  - `byteLength`: 1에서 6까지의 정수를 입력합니다.
- `Buffer.prototype.writeUInt32LE(value, offset, byteLength)`

  리틀 엔디안을 사용하여 `offset`에서 `value`를 `Buffer`에 씁니다.
  - `value`: 정수를 입력합니다.
  - `offset`: 정수를 입력합니다.
  - `byteLength`: 1에서 6까지의 정수를 입력합니다.
- `Buffer.prototype.writeDoubleBE(value, [offset])`

  빅 엔디안을 사용하여 `offset`에서 `value`를 `Buffer`에 씁니다.
  - `value`: 정수를 입력합니다.
  - `offset`: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.
- `Buffer.prototype.writeDoubleLE(value, [offset])`

  리틀 엔디안을 사용하여 `offset`에서 `value`를 `Buffer`에 씁니다.
  - `value`: 정수를 입력합니다.
  - `offset`: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.
- `Buffer.prototype.writeFloatBE(value, [offset])`

  빅 엔디안을 사용하여 `offset`에서 `value`를 `Buffer`에 씁니다.
  - `value`: 정수를 입력합니다.
  - `offset`: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.

- `Buffer.prototype.writeFloatLE(value, [offset])`

  리틀 엔디안을 사용하여 `offset`에서 `value`를 `Buffer`에 씁니다.

  - `value`: 정수를 입력합니다.
  - `offset`: 선택 사항입니다. 정수를 입력합니다. 기본값은 0.

다음 인스턴스 메서드가 지원됩니다.

- `buffer[index]`

  `Buffer`에 `index`의 8진수(바이트)를 가져와 설정합니다.

  - `0`에서 `255`의 숫자를 가져옵니다. 또는 `0`부터 `255`까지의 숫자를 설정할 수도 있습니다.

다음 인스턴스 속성이 지원됩니다.

- `buffer`

  버퍼의 `ArrayBuffer` 객체를 가져옵니다.
- `byteOffset`

  버퍼의 `Arraybuffer` 객체에 대한 `byteOffset`을 가져옵니다.
- `length`

  버퍼 바이트 수를 가져옵니다.

> **Note**
> 모든 버퍼 모듈 메서드가 JavaScript 런타임 2.0에 새로 추가되었습니다.

쿼리 문자열

> **Note**
> CloudFront 함수 이벤트 객체는 URL 쿼리 문자열을 자동으로 구문 분석합니다. 즉, 대부분의 경우 이 모듈을 사용할 필요가 없습니다.

쿼리 문자열 모듈(querystring)은 URL 쿼리 문자열을 구문 분석하고 서식을 지정하는 메서드를 제공합니다. `require('querystring')`을 사용하여 모듈을 로드할 수 있습니다. 이 모듈은 다음과 같은 방법을 제공합니다.

`querystring.escape(`*`string`*`)`

지정된 `string`을 URL-인코딩하여 이스케이프된 쿼리 문자열을 반환합니다. 이 방법은 `querystring.stringify()`에서 사용되며 직접 사용해서는 안 됩니다.

`querystring.parse(`*`string`*`[, `*`separator`*`[, `*`equal`*`[, `*`options`*`]]])`

쿼리 문자열(`string`)을 구문 분석하고 객체를 반환합니다.

`separator` 파라미터는 쿼리 문자열에서 키와 값 페어를 구분하기 위한 하위 문자열입니다. 기본 설정은 `&`입니다.

`equal` 파라미터는 쿼리 문자열에서 키와 값을 구분하기 위한 하위 문자열입니다. 기본 설정은 `=`입니다.

`options` 파라미터는 다음 키를 가진 객체입니다.

`decodeURIComponent` *`function`*

쿼리 문자열에서 백분율로 인코딩된 문자를 디코딩하는 함수입니다. 기본 설정은 `querystring.unescape()`입니다.

`maxKeys` *`number`*

구문 분석할 최대 키 수입니다. 기본 설정은 `1000`입니다. `0` 값을 사용하여 키 카운트 제한을 제거합니다.

기본적으로 쿼리 문자열 내의 백분율로 인코딩된 문자는 UTF-8 인코딩을 사용하는 것으로 간주됩니다. 잘못된 UTF-8 시퀀스는 `U+FFFD` 대체 문자로 대체됩니다.

예를 들어 다음 쿼리 문자열의 경우:

```
'name=value&abc=xyz&abc=123'
```

`querystring.parse()`의 반환 값:

```
{
name: 'value',
abc: ['xyz', '123']
}
```

`querystring.decode()`는 `querystring.parse()`의 별칭입니다.

`querystring.stringify(`*`object`*`[, `*`separator`*`[, `*`equal`*`[, `*`options`*`]]])`

`object`를 직렬화하고 쿼리 문자열을 반환합니다.

`separator` 파라미터는 쿼리 문자열에서 키와 값 페어를 구분하기 위한 하위 문자열입니다. 기본 설정은 `&`입니다.

`equal` 파라미터는 쿼리 문자열에서 키와 값을 구분하기 위한 하위 문자열입니다. 기본 설정은 `=`입니다.

`options` 파라미터는 다음 키를 가진 객체입니다.

`encodeURIComponent` *`function`*

URL에 안전하지 않은 문자를 쿼리 문자열에서 백분율 인코딩으로 변환하는 데 사용할 함수입니다. 기본 설정은 `querystring.escape()`입니다.

기본적으로 쿼리 문자열 내에서 백분율 인코딩이 필요한 문자는 UTF-8로 인코딩됩니다. 다른 인코딩을 사용하려면 `encodeURIComponent` 옵션을 지정합니다.

예를 들어, 다음 코드의 경우:

```
querystring.stringify({ name: 'value', abc: ['xyz', '123'], anotherName: '' });
```

반환 값:

```
'name=value&abc=xyz&abc=123&anotherName='
```

`querystring.encode()`는 `querystring.stringify()`의 별칭입니다.

`querystring.unescape(`*`string`*`)`

지정된 `string`의 URL 백분율로 인코딩된 문자를 디코딩하여 이스케이프되지 않은 쿼리 문자열을 반환합니다. 이 방법은 `querystring.parse()`에서 사용되며 직접 사용해서는 안 됩니다.

## crypto

암호화 모듈(`crypto`)은 표준 해싱 및 HMAC(해시 기반 메시지 인증 코드) 헬퍼를 제공합니다. `require('crypto')`을(를) 사용하여 모듈을 로드할 수 있습니다.

### 해싱 메서드

`crypto.createHash(algorithm)`

지정된 알고리즘(`md5`, `sha1` 또는 `sha256`)을 사용하여 해시 다이제스트를 생성하는 데 사용할 수 있는 해시 객체를 생성하고 반환합니다.

`hash.update(data)`

지정된 `data`(으)로 해시 콘텐츠를 업데이트합니다.

`hash.digest([encoding])`

`hash.update()`을(를) 사용하여 전달된 모든 데이터의 다이제스트를 계산합니다. 인코딩은 `hex`, `base64` 또는 `base64url`일 수 있습니다.

### HMAC 메서드

`crypto.createHmac(algorithm, secret key)`

지정된 `algorithm` 및 `secret key`을(를) 사용하는 HMAC 객체를 생성 및 반환합니다. 알고리즘은 `md5`, `sha1` 또는 `sha256`일 수 있습니다.

`hmac.update(data)`

지정된 `data`(으)로 HMAC 콘텐츠를 업데이트합니다.

`hmac.digest([encoding])`

`hmac.update()`을(를) 사용하여 전달된 모든 데이터의 다이제스트를 계산합니다. 인코딩은 `hex`, `base64` 또는 `base64url`일 수 있습니다.

### 제한된 기능

다음 JavaScript 언어 기능은 보안 문제로 인해 지원되지 않거나 제한됩니다.

동적 코드 평가

동적 코드 평가는 지원되지 않습니다. 시도하면 `eval()` 및 `Function` 생성자 모두 오류가 발생합니다. 예를 들어 `const sum = new Function('a', 'b', 'return a + b')`에서 오류가 발생합니다.

타이머

`setTimeout()`, `setImmediate()` 및 `clearTimeout()` 함수는 지원되지 않습니다. 함수 실행 내에서 연기하거나 출력할 프로비전이 없습니다. 함수가 완료되려면 동기적으로 실행되어야 합니다.

날짜 및 타임스탬프

보안상의 이유로 고해상도 타이머에 액세스할 수 없습니다. 현재 시간을 쿼리하는 모든 `Date` 메서드는 단일 함수 실행의 수명 동안 항상 동일한 값을 반환합니다. 반환된 타임 스탬프는 함수가 실행을 시작한 시간입니다. 따라서 함수에서 경과 시간을 측정할 수 없습니다.

파일 시스템 액세스

파일 시스템 액세스가 없습니다. 예를 들어 Node.js에서처럼 파일 시스템 액세스를 위한 `fs` 모듈이 없습니다.

프로세스 액세스

프로세스 액세스가 없습니다. 예를 들어 Node.js와는 달리 정보 액세스를 처리하기 위한 `process` 글로벌 객체가 없습니다.

환경 변수

환경 변수에 액세스할 수 없습니다. 대신 CloudFront KeyValueStore를 사용하여 CloudFront Functions에 대한 키 값 페어의 중앙 집중식 데이터 스토어를 생성할 수 있습니다. CloudFront KeyValueStore를 사용하면 코드 변경 사항을 배포할 필요 없이 구성 데이터를 동적으로 업데이트할 수 있습니다. 자세한 내용은 Amazon CloudFront KeyValueStore 섹션을 참조하세요.

네트워크 액세스

네트워크 통화에 대한 지원이 없습니다. 예를 들어 XHR, HTTP(S) 및 소켓은 지원되지 않습니다.

## 키 값 저장소를 위한 도우미 메서드

> **Note**
>
> CloudFront Functions의 키 값 저장소 헬퍼 메서드 호출은 AWS CloudTrail 데이터 이벤트를 트리거하지 않습니다. 이러한 이벤트는 CloudTrail 이벤트 기록에 로그되지 않습니다. 자세한 내용은 AWS CloudTrail을 사용하여 Amazon CloudFront API 직접 호출 로깅 섹션을 참조하세요.

이 섹션은 생성한 함수에 CloudFront 키 값 저장소를 사용하여 키 값을 포함하는 경우에 적용됩니다. CloudFront Functions에는 키 값 저장소에서 값을 읽을 수 있는 세 가지 도우미 메서드를 제공하는 모듈이 있습니다.

함수 코드에서 이 모듈을 사용하려면 키 값 저장소를 함수와 연결했는지 확인합니다.

그런 다음 함수 코드의 첫 줄에 다음 명령문을 포함시킵니다.

```
import cf from 'cloudfront';
const kvsHandle = cf.kvs();
```

### **get()** 메서드

이 메서드를 사용하여 지정한 키 이름의 키 값을 반환합니다.

요청

```
get("key", options);
```

- `key`: 값을 가져와야 하는 키의 이름입니다.
- `options`: 한 가지 옵션으로 `format`이 있습니다. 이 옵션을 사용하면 함수가 데이터를 올바르게 파싱합니다. 가능한 값은 다음과 같습니다.
  - `string`: (기본값) UTF8 인코딩
  - `json`
  - `bytes`: 원시 바이너리 데이터 버퍼

요청 예제

```
const value = await kvsHandle.get("myFunctionKey", { format: "string"});
```

응답

응답은 `options`를 사용하여 요청한 형식의 값으로 해석되는 `promise`입니다. 기본적으로 값은 문자열로 반환됩니다.

오류 처리

요청된 키가 관련 키 값 저장소에 존재하지 않는 경우 `get()` 메서드는 오류를 반환합니다. 이 사용 사례를 관리하기 위해 코드에 `try` 및 `catch` 블록을 추가할 수 있습니다.

> **Warning**
>
> promise 결합자(예: `Promise.all`, `Promise.any`) 및 promise 체인 메서드(예: `then` 및 `catch`)를 사용하려면 많은 함수 메모리 사용량이 필요할 수 있습니다. 함수가 최대 함수 메모

리 할당량을 초과하면 실행에 실패합니다. 이 오류를 방지하려면 `await` 구문을 순차적으로 사용하거나 루프에서 여러 값을 요청하는 것이 좋습니다.
예제

```
var value1 = await kvs.get('key1');
var value2 = await kvs.get('key2');
```

현재 다음 예제와 같이 promise 결합자를 사용하여 여러 값을 가져오더라도 성능이 향상되지 않습니다.

```
var values = await Promise.all([kvs.get('key1'), kvs.get('key2'),]);
```

### **exists()** 메서드

키가 키 값 저장소에 존재하는지 여부를 지정하려면 이 메서드를 사용합니다.

요청

```
exists("key");
```

요청 예제

```
const exist = await kvsHandle.exists("myFunctionkey");
```

응답

응답은 부울(`true` 또는 `false`)을 반환하는 `promise`입니다. 이 값은 키가 키 값 저장소에 존재하는지 여부를 지정합니다.

### **meta()** 메서드

이 메서드를 사용하여 키 값 저장소에 대한 메타데이터를 반환합니다.

요청

```
meta();
```

요청 예제

```
const meta = await kvsHandle.meta();
```

### 응답

응답은 다음과 같은 속성을 가진 객체로 확인되는 promise입니다.

- creationDateTime: 키 값 저장소가 생성된 ISO 8601 형식의 날짜 및 시간입니다.
- lastUpdatedDateTime: 키 값 저장소가 소스에서 마지막으로 동기화된 ISO 8601 형식의 날짜 및 시간입니다. 엣지로의 전파 시간은 값에 포함되지 않습니다.
- keyCount: 소스와의 마지막 동기화 이후 KVS에 있는 키의 총 개수입니다.

### 응답 예제

```
{keyCount:3,creationDateTime:2023-11-30T23:07:55.765Z,lastUpdatedDateTime:2023-12-15T03:57:52.4
```

## 오리진 수정용 헬퍼 메서드

이 섹션은 CloudFront Functions 코드 내에서 요청에 사용되는 오리진을 동적으로 업데이트하거나 변경하는 경우에 적용됩니다. 뷰어 요청 CloudFront Functions에서만 오리진을 업데이트할 수 있습니다. CloudFront Functions에는 오리진을 동적으로 업데이트하거나 변경하는 헬퍼 메서드를 제공하는 모듈이 있습니다.

이 모듈을 사용하려면 JavaScript 런타임 2.0을 사용하여 CloudFront Functions을 만들고 함수 코드의 첫 번째 줄에 다음 스테이트먼트를 포함합니다.

```
import cf from 'cloudfront';
```

자세한 내용은 CloudFront Functions를 위한 JavaScript 런타임 2.0 기능 섹션을 참조하세요.

> **Note**
> 테스트 API 및 테스트 콘솔 페이지에서는 오리진 수정이 발생했는지 여부를 테스트하지 않습니다. 그러나 테스트는 함수 코드가 오류 없이 실행되도록 합니다.

### CloudFront Functions과 Lambda@Edge 중에서 선택

CloudFront Functions 또는 Lambda@Edge를 사용하여 오리진을 업데이트할 수 있습니다.

CloudFront Functions을 사용하여 오리진을 업데이트할 때 뷰어 요청 이벤트 트리거를 사용합니다. 즉, 이 함수가 사용될 때 해당 로직이 모든 요청에서 실행됩니다. Lambda@Edge를 사용하는 경우 오리진 업데이트 기능은 오리진 요청 이벤트 트리거에 존재합니다. 즉, 이 로직은 캐시 누락에서만 실행됩니다.

주로 워크로드와 배포에서 CloudFront Functions 및 Lambda@Edge의 기존 사용량에 따라 선택이 달라집니다. 다음 고려 사항을 통해 CloudFront Functions 또는 Lambda@Edge를 사용하여 오리진을 업데이트할지 여부를 결정할 수 있습니다.

CloudFront Functions은 다음과 같은 상황에서 가장 유용합니다.

- 요청이 동적(즉, 캐싱할 수 없음)이고 항상 오리진으로 이동하는 경우입니다. CloudFront Functions은 더 나은 성능과 더 낮은 전체 비용을 제공합니다.
- 각 요청에서 실행되는 기존 뷰어 요청 CloudFront Functions이 이미 있는 경우 기존 함수에 오리진 업데이트 로직을 추가할 수 있습니다.

CloudFront Functions을 사용하여 오리진을 업데이트하려면 다음 주제의 헬퍼 메서드를 참조하시기 바랍니다.

Lambda@Edge는 다음과 같은 상황에서 가장 유용합니다.

- 캐싱 가능성이 높은 콘텐츠가 있으면 Lambda@Edge는 캐시 누락 시에만 실행되고 CloudFront Functions은 모든 요청 시 실행되므로 더 비용 효율적일 수 있습니다.
- 기존 오리진 요청 Lambda@Edge 함수가 이미 있는 경우 오리진 업데이트 로직을 기존 함수에 추가할 수 있습니다.
- 오리진 업데이트 로직에서 Amazon DynamoDB 또는 Amazon S3와 같은 서드 파티 데이터 소스에서 데이터를 가져와야 하는 경우입니다.

Lambda@Edge에 대한 자세한 내용은 Lambda@Edge를 사용하여 엣지에서 사용자 지정 섹션을 참조하시기 바랍니다.

updateRequestOrigin() 메서드

`updateRequestOrigin()` 메서드를 사용하여 요청의 오리진 설정을 업데이트합니다. 이 메서드를 사용하여 배포에 이미 정의된 오리진의 기존 오리진 속성을 업데이트하거나 요청에 대한 새 오리진을 정의할 수 있습니다. 이렇게 하려면 변경하려는 속성을 지정합니다.

> **Important**
>
> `updateRequestOrigin()`에서 지정하지 않은 설정은 기존 오리진의 구성에서 동일한 설정을 상속합니다.

`updateRequestOrigin()` 메서드에 따라 설정된 오리진은 HTTP 엔드포인트일 수 있으며 CloudFront 배포 내의 기존 오리진일 필요는 없습니다.

> **참고**
>
> - 오리진 그룹의 일부인 오리진을 업데이트하는 경우 오리진 그룹의 프라이머리 오리진만 업데이트됩니다. 보조 오리진은 변경되지 않습니다. 장애 조치 기준과 일치하는 수정된 오리진의 모든 응답 코드는 보조 오리진으로 장애 조치를 트리거합니다.
> - 오리진 유형을 변경하고 OAC를 사용 설정한 경우 `originAccessControlConfig`의 오리진 유형이 새 오리진 유형과 일치하는지 확인합니다.
> - `updateRequestOrigin()` 메서드를 사용하여 VPC 오리진을 업데이트할 수 없습니다. 요청이 완료되지 않습니다.

요청

```
updateRequestOrigin({origin properties})
```

`origin properties`에는 다음 사항이 포함될 수 있습니다.

domainName(선택 사항)

오리진의 도메인 이름입니다. 이를 제공하지 않으면 대신 할당된 오리진의 도메인 이름이 사용됩니다.

사용자 지정 오리진의 경우

DNS 도메인 이름(예: `www.example.com`)을 지정합니다. 도메인 이름에는 콜론(`:`)이 포함될 수 없으며 IP 주소가 될 수 없습니다. 도메인 이름은 최대 253자일 수 있습니다.

S3 오리진의 경우

Amazon S3 버킷의 DNS 도메인 이름(예: `amzn-s3-demo-bucket.s3.eu-west-1.amazonaws.com`)을 지정합니다. 이름은 최대 128자여야 하며, 모두 소문자여야 합니다.

hostHeader(선택 사항, S3가 아닌 사용자 지정 오리진의 경우)

오리진에 요청할 때 사용할 호스트 헤더입니다. 제공하지 않으면 domainName 파라미터의 값이 사용됩니다. 호스트 헤더 또는 도메인 이름 파라미터가 제공되지 않은 경우 할당된 오리진의 도메인 이름이 사용되고, 오리진으로 전달(FTO) 정책에 호스트가 포함된 경우 수신 요청의 호스트 헤더가 사용됩니다. 호스트 헤더에는 콜론(`:`)이 포함될 수 없으며 IP 주소가 될 수 없습니다. 호스트 헤더는 최대 253자까지 작성할 수 있습니다.

originPath(선택 사항)

요청이 콘텐츠를 찾아야 하는 오리진의 디렉터리 경로입니다. 경로는 슬래시(/)로 시작해야 하지만 슬래시 하나로 끝나지 않아야 합니다. 예를 들어 `example-path/`로 끝나지 않아야 합니다. 이를 제공하지 않으면 할당된 오리진의 오리진 경로가 사용됩니다.

사용자 지정 오리진의 경우

경로는 URL 인코딩되어야 하고 최대 255자여야 합니다.

customHeaders(선택 사항)

각 사용자 지정 헤더에 대한 헤더 이름 및 값 페어를 지정하여 요청을 포함한 사용자 지정 헤더를 포함할 수 있습니다. 형식은 이벤트 구조에서 요청 및 응답 헤더의 형식과 다릅니다. 다음의 키/값 쌍 구문을 사용합니다.

```
{"key1": "value1", "key2": "value2", ...}
```

허용되지 않는 헤더는 추가할 수 없으며 동일한 이름의 헤더가 수신되는 요청 `headers`에 존재할 수 없습니다. 헤더 이름은 함수 코드에서 소문자여야 합니다. CloudFront Functions이 이벤트 객체를 HTTP 요청으로 다시 변환하는 경우 헤더 이름에 있는 각 단어의 첫 글자가 대문자로 표시됩니다. 단어는 하이픈으로 구분됩니다.

예를 들어 함수 코드가 `example-header-name`이라는 이름의 헤더를 추가하는 경우, CloudFront는 이 헤더를 HTTP 요청에서 `Example-Header-Name`으로 변환합니다. 자세한 내용은 CloudFront에서 오리진 요청에 추가할 수 없는 사용자 지정 헤더 및 엣지 함수에 대한 제한 사항(을)를 참조하세요.

이를 제공하지 않으면 할당된 오리진의 사용자 지정 헤더가 사용됩니다.

connectionAttempts(선택 사항)

CloudFront가 오리진에 연결을 시도하는 횟수입니다. 최솟값은 1이고, 최댓값은 3입니다. 이를 제공하지 않으면 할당된 오리진을 통해 연결 시도가 사용됩니다.

originShield(선택 사항)

이렇게 하면 CloudFront Origin Shield가 사용 설정되거나 업데이트됩니다. Origin Shield를 사용하면 오리진의 부하를 줄일 수 있습니다. 자세한 내용은 Amazon CloudFront Origin Shield 사용 섹션을 참조하세요. 이를 제공하지 않으면 할당된 오리진을 통해 Origin Shield 설정이 사용됩니다.

enabled(필수)

Origin Shield를 사용 또는 사용 해제하는 부울 표현식입니다. `true` 또는 `false` 값을 허용합니다.

region(사용 설정된 경우 필수)

Origin Shield의 AWS 리전입니다. 오리진에 대한 지연 시간이 가장 낮은 AWS 리전을 지정합니다. 리전 이름이 아닌 리전 코드를 사용합니다. 예를 들어 `us-east-2`를 사용하여 미국 동부(오하이오) 리전을 지정합니다.

CloudFront Origin Shield를 사용하는 경우 이에 대한 AWS 리전을 지정해야 합니다. 사용 가능한 AWS 리전 목록과 오리진에 가장 적합한 리전을 선택하는 데 도움이 되는 목록은 Origin Shield의 AWS 리전 선택 섹션을 참조하시기 바랍니다.

originAccessControlConfig(선택 사항)

해당 오리진에 대한 원본 액세스 제어(OAC)의 고유한 식별자입니다. 이는 오리진이 Amazon S3, Lambda 함수 URL, MediaStore, MediaPackage V2 등 CloudFront OAC를 지원하는 경우에만 사용됩니다. 이를 제공하지 않으면 할당된 오리진의 OAC 설정이 사용됩니다.

이는 레거시 오리진 액세스 ID(OAI)를 지원하지 않습니다. 자세한 내용은 AWS 오리진에 대한 액세스 제한 섹션을 참조하세요.

enabled(필수)

OAC를 사용 또는 사용 해제하는 부울 표현식입니다. `true` 또는 `false` 값을 허용합니다.

signingBehavior(사용 설정된 경우 필수)

CloudFront가 서명(인증 정보 추가)하는 요청을 지정합니다. 대부분의 사용 사례에서 `always`를 지정합니다.. 자세한 내용은 오리진 액세스 제어를 위한 고급 설정 섹션을 참조하세요.

이 필드는 다음 값 중 하나를 가질 수 있습니다.

- `always` - CloudFront가 모든 오리진 요청을 서명하고, 최종 사용자 요청으로부터 `Authorization` 헤더를 덮어씁니다(있는 경우).
- `never` - CloudFront가 오리진 요청을 서명하지 않습니다. 이 값은 오리진에 대한 오리진 액세스 제어를 끕니다.
- `no-override` - 뷰어 요청에 `Authorization` 헤더가 포함되지 않은 경우 CloudFront가 오리진 요청을 서명합니다. 뷰어 요청에 `Authorization` 헤더가 포함된 경우 CloudFront가 오리진 요청에 서명하지 않고 최종 사용자 요청의 `Authorization` 헤더와 함께 전달합니다.

> **Warning**
>
> 뷰어 요청에서 `Authorization` 헤더를 전달하려면 이 오리진 액세스 제어와 연결된 오리진을 사용하는 모든 캐시 동작에 대한 해당 헤더를 오리진 요청 정책에 추가해야 합니다. 자세한 내용은 정책을 통한 오리진 요청 제어 섹션을 참조하세요.

signingProtocol(사용 설정된 경우 필수)

CloudFront가 요청을 서명(인증)하는 방법을 결정하는 OAC의 서명 프로토콜입니다. 유일한 유효 값은 `sigv4`입니다.

originType(사용 설정된 경우 필수)

이 OAC의 오리진 유형입니다. 유효한 값에는 `s3`, `mediapackagev2`, `mediastore` 및 `lambda` 등이 있습니다.

timeouts(선택 사항)

CloudFront가 오리진이 응답하거나 데이터를 전송할 때까지 대기해야 하는 기간에 대해 지정할 수 있는 제한 시간입니다. 이를 제공하지 않으면 할당된 오리진의 제한 시간 설정이 사용됩니다.

> **Note**
>
> 달리 명시되지 않은 한, 이러한 제한 시간은 사용자 지정 오리진과 Amazon S3 오리진을 모두 지원합니다.

readTimeout(선택 사항)

`readTimeout`은 다음 두 값 모두에 적용됩니다.

- CloudFront가 오리진에 요청을 전달한 후 응답을 기다리는 시간(초).

- CloudFront가 오리진으로부터 응답 패킷을 수신한 후 다음 패킷을 수신할 때까지 대기하는 시간(초).

최소 제한 시간은 1초이고, 최대 제한 시간은 120초입니다. 자세한 내용은 응답 제한 시간 섹션을 참조하세요.

responseCompletionTimeout(선택 사항)

CloudFront에서 오리진으로의 요청이 열려 있고 응답을 기다릴 수 있는 시간(초). 이 시간까지 오리진에서 완전한 응답을 받지 못하면 CloudFront는 연결을 종료합니다.

`responseCompletionTimeout` 값은 `readTimeout` 값과 같거나 이보다 커야 합니다. 자세한 내용은 응답 완료 제한 시간 섹션을 참조하세요.

keepAliveTimeout(선택 사항)

이 제한 시간은 Amazon S3 오리진이 아닌 사용자 지정 오리진에만 적용됩니다. (S3 오리진 구성은 이러한 설정을 무시합니다.)

`keepAliveTimeout`은 응답의 마지막 패킷을 수신한 후 CloudFront가 오리진에 대한 연결을 유지하려고 시도하는 시간을 지정합니다. 최소 제한 시간은 1초이고, 최대 제한 시간은 120초입니다. 자세한 내용은 연결 유지 제한 시간(사용자 지정 및 VPC 오리진만 해당) 섹션을 참조하세요.

connectionTimeout(선택 사항)

오리진에 대한 연결을 설정하려고 할 때 CloudFront가 대기하는 시간(초)입니다. 최소 제한 시간은 1초이고, 최대 제한 시간은 10초입니다. 자세한 내용은 연결 제한 시간 섹션을 참조하세요.

customOriginConfig(선택 사항)

`customOriginConfig`를 사용하여 Amazon S3 버킷이 아닌 오리진에 대한 연결 설정을 지정합니다. 한 가지 예외는 다음과 같습니다. S3 버킷이 고정 웹 사이트 호스팅으로 구성된 경우 이러한 설정을 지정할 수 있습니다. (다른 유형의 S3 버킷 구성은 이러한 설정을 무시합니다.) `customOriginConfig`가 제공되지 않으면 할당된 오리진의 설정이 사용됩니다.

port(필수)

CloudFront가 오리진에 연결하는 데 사용하는 HTTP 포트입니다. 오리진에서 수신 대기하는 HTTP 포트를 지정하십시오.

protocol(필수)

CloudFront가 오리진에 연결하는 데 사용하는 프로토콜(HTTP 또는 HTTPS)을 지정합니다. 유효한 값은 다음과 같습니다.

- `http` - CloudFront는 항상 HTTP를 사용하여 오리진에 연결합니다.
- `https` - CloudFront는 항상 HTTPS를 사용하여 오리진에 연결합니다.

sslProtocols(필수)

HTTPS를 통해 오리진에 연결할 때 CloudFront에서 사용하는 최소 SSL/TLS 프로토콜을 지정하는 목록입니다. 유효한 값에는 `SSLv3`, `TLSv1`, `TLSv1.1` 및 `TLSv1.2` 등이 있습니다. 자세한 내용은 최소 오리진 SSL 프로토콜 섹션을 참조하세요.

ipAddressType(선택 사항)

CloudFront가 오리진에 연결하는 데 사용하는 IP 주소 유형을 지정합니다. 유효값에는 `ipv4`, `ipv6` 및 `dualstack`이 있습니다. `ipAddressType` 변경은 `domainName` 속성도 변경 중인 경우에만 지원됩니다.

sni(선택 사항, S3가 아닌 사용자 지정 오리진의 경우)

서버 이름 표시(SNI)는 Transport Layer Security(TLS) 프로토콜의 확장으로, 클라이언트가 TLS 핸드셰이크 프로세스를 시작할 때 연결을 시도하는 호스트 이름을 나타냅니다. 이 값은 오리진 서버의 TLS 인증서에 있는 일반 이름과 일치해야 합니다. 그렇지 않으면 오리진 서버에서 오류가 발생할 수 있습니다.

제공하지 않으면 `hostHeader` 파라미터의 값이 사용됩니다. 호스트 헤더가 제공되지 않으면 `domainName` 파라미터의 값이 사용됩니다.

호스트 헤더 또는 도메인 이름 파라미터가 제공되지 않은 경우 할당된 오리진의 도메인 이름이 사용되고, 오리진으로 전달(FTO) 정책에 호스트가 포함된 경우 수신 요청의 호스트 헤더가 사용됩니다. SNI에는 콜론(`:`)이 포함될 수 없으며 IP 주소가 될 수 없습니다. SNI는 최대 253자까지 작성할 수 있습니다.

allowedCertificateNames(선택 사항, S3가 아닌 사용자 지정 오리진의 경우)

CloudFront에서 오리진 서버와의 TLS 핸드셰이크 중에 오리진 서버 TLS 인증서와 일치하는 도메인의 유효성을 검사하는 데 사용할 유효한 인증서 이름 목록을 포함할 수 있습니다. 이 필드에는 유효한 도메인 이름 배열이 필요하며 `*.example.com`과 같은 와일드카드 도메인이 포함될 수 있습니다.

허용되는 인증서 이름은 최대 20개까지 지정할 수 있습니다. 각 인증서의 이름은 최대 64자까지 가능합니다.

Example- Amazon S3 요청 오리진 업데이트

다음 예제에서는 뷰어 요청의 오리진을 S3 버킷으로 변경하고 OAC를 사용 설정하며 오리진으로 전송된 사용자 지정 헤더를 재설정합니다.

```
cf.updateRequestOrigin({
    "domainName" : "amzn-s3-demo-bucket-in-us-east-1.s3.us-east-1.amazonaws.com",
    "originAccessControlConfig": {
        "enabled": true,
        "signingBehavior": "always",
        "signingProtocol": "sigv4",
        "originType": "s3"
    },
    // Empty object resets any header configured on the assigned origin
    "customHeaders": {}
});
```

Example- Application Load Balancer 요청 오리진 업데이트

다음 예제에서는 뷰어 요청의 오리진을 Application Load Balancer 오리진으로 변경하고 사용자 지정 헤더와 제한 시간을 설정합니다.

```
cf.updateRequestOrigin({
    "domainName" : "example-1234567890.us-east-1.elb.amazonaws.com",
    "timeouts": {
        "readTimeout": 30,
        "connectionTimeout": 5
    },
    "customHeaders": {
        "x-stage": "production",
        "x-region": "us-east-1"
    }
});
```

Example- Origin Shield가 사용 설정된 오리진으로 업데이트

다음 예제에서는 배포의 오리진에 Origin Shield가 사용 설정되어 있습니다. 함수 코드는 오리진에 사용되는 도메인 이름만 업데이트하고 다른 모든 선택 사항 파라미터를 생략합니다. 이 경우 Origin Shield 파라미터가 업데이트되지 않았으므로 Origin Shield는 수정된 오리진 도메인 이름과 함께 계속 사용됩니다.

```
cf.updateRequestOrigin({
```

```
    "domainName" : "www.example.com"
});
```

Example- 호스트 헤더, SNI 및 허용된 인증서 이름 업데이트

> **Warning**
>
> 대부분의 사용 사례에서는 오리진으로 이동하는 요청에 이러한 유형의 수정을 사용할 필요가 없습니다. 이러한 값 변경의 영향을 이해하지 못한다면 이러한 파라미터를 사용해서는 안 됩니다.

다음 예시에서는 요청에 대한 도메인 이름, 호스트 헤더, SNI 및 허용된 인증서를 오리진으로 변경합니다.

```
cf.updateRequestOrigin({
    "domainName": "www.example.com",
    "hostHeader": "test.example.com",
    "sni": "test.example.net",
    "allowedCertificateNames": ["*.example.com", "*.example.net"],
});
```

selectRequestOriginById() 메서드

`selectRequestOriginById()`를 사용해 배포에 이미 구성된 다른 오리진을 선택하여 기존 오리진을 업데이트합니다. 이 메서드는 업데이트된 오리진에서 정의한 것과 동일한 설정을 모두 사용합니다.

이 메서드는 함수를 실행할 때 사용된 것과 동일한 배포에 이미 정의된 오리진만 허용합니다. 오리진은 오리진을 설정할 때 정의한 오리진 이름인 오리진 ID로 참조됩니다.

배포에 VPC 오리진이 구성된 경우 이 메서드를 사용하여 오리진을 VPC 오리진으로 업데이트할 수 있습니다. 자세한 내용은 VPC 오리진으로 액세스 제한 섹션을 참조하세요.

> **참고**
>
> - `selectRequestOriginById()` 함수는 상호 TLS(오리진)가 활성화된 오리진을 선택할 수 없습니다. 이 함수를 사용하여 상호 TLS(오리진)가 활성화된 오리진을 선택하려고 하면 검증 오류가 발생합니다.

- 사용 사례에 따라 상호 TLS(오리진)를 사용한 동적 오리진 선택이 필요한 경우, 모든 대상 오리진이 동일한 클라이언트 인증서를 사용하도록 `updateRequestOrigin()`을 대신 사용하세요.

### 요청

```
cf.selectRequestOriginById(origin_id, {origin_overrides})
```

이전 예시에서 `origin_id`는 함수를 실행하는 배포에서 오리진의 오리진 이름을 가리키는 문자열입니다. `origin_overrides` 파라미터에는 다음이 포함될 수 있습니다.

hostHeader(선택 사항, S3가 아닌 사용자 지정 오리진의 경우)

오리진에 요청할 때 사용할 호스트 헤더입니다. 제공하지 않으면 `domainName` 파라미터의 값이 사용됩니다.

호스트 헤더 또는 도메인 이름 파라미터가 제공되지 않은 경우 할당된 오리진의 도메인 이름이 사용되고, 오리진으로 전달(FTO) 정책에 호스트가 포함된 경우 수신 요청의 호스트 헤더가 사용됩니다. 호스트 헤더에는 콜론(`:`)이 포함될 수 없으며 IP 주소가 될 수 없습니다. 호스트 헤더는 최대 253자까지 작성할 수 있습니다.

sni(선택 사항, S3가 아닌 사용자 지정 오리진의 경우)

서버 이름 표시(SNI)는 Transport Layer Security(TLS) 프로토콜의 확장으로, 클라이언트가 TLS 핸드셰이크 프로세스를 시작할 때 연결을 시도하는 호스트 이름을 나타냅니다. 이 값은 오리진 서버의 TLS 인증서에 있는 일반 이름과 일치해야 합니다. 그렇지 않으면 오리진 서버에서 오류가 발생할 수 있습니다.

제공하지 않으면 `hostHeader` 파라미터의 값이 사용됩니다. 호스트 헤더가 제공되지 않으면 `domainName` 파라미터의 값이 사용됩니다.

호스트 헤더 또는 도메인 이름 파라미터가 제공되지 않은 경우 할당된 오리진의 도메인 이름이 사용되고, 오리진으로 전달(FTO) 정책에 호스트가 포함된 경우 수신 요청의 호스트 헤더가 사용됩니다. SNI에는 콜론(`:`)이 포함될 수 없으며 IP 주소가 될 수 없습니다. SNI는 최대 253자까지 작성할 수 있습니다.

allowedCertificateNames(선택 사항, S3가 아닌 사용자 지정 오리진의 경우)

CloudFront에서 오리진 서버와의 TLS 핸드셰이크 중에 오리진 서버 TLS 인증서와 일치하는 도메인의 유효성을 검사하는 데 사용할 유효한 인증서 이름 목록을 포함할 수 있습니다. 이 필드에는 유

효한 도메인 이름 배열이 필요하며 `*.example.com`과 같은 와일드카드 도메인이 포함될 수 있습니다.

허용되는 인증서 이름은 최대 20개까지 지정할 수 있습니다. 각 인증서의 이름은 최대 64자까지 가능합니다.

요청

```
selectRequestOriginById(origin_id)
```

앞의 예시에서 `origin_id`는 함수를 실행하는 배포에서 오리진의 오리진 이름을 가리키는 문자열입니다.

Example- Amazon S3 요청 오리진 선택

다음 예시에서는 배포와 연결된 오리진 목록에서 `amzn-s3-demo-bucket-in-us-east-1`라는 오리진을 선택하고 `amzn-s3-demo-bucket-in-us-east-1` 오리진의 구성 설정을 요청에 적용합니다.

```
cf.selectRequestOriginById("amzn-s3-demo-bucket-in-us-east-1");
```

Example- Application Load Balancer 요청 오리진 선택

다음 예시에서는 배포와 연결된 오리진 목록에서 `myALB-prod`라는 Application Load Balancer 오리진을 선택하고 `myALB-prod`의 구성 설정을 요청에 적용합니다.

```
cf.selectRequestOriginById("myALB-prod");
```

Example- Application Load Balancer 요청 오리진을 선택하고 호스트 헤더를 재정의합니다.

이전 예시와 같이 다음 예시에서는 배포와 연결된 오리진 목록에서 `myALB-prod`라는 Application Load Balancer 오리진을 선택하고 `myALB-prod`의 구성 설정을 요청에 적용합니다. 그러나 이 예제에서는 `origin_overrides`를 사용하여 호스트 헤더 값을 재정의합니다.

```
cf.overrideRequestOrigin("myALB-prod",{
        "hostHeader" : "test.example.com"
});
```

### createRequestOriginGroup() 메서드

`createRequestOriginGroup()`를 사용하여 고가용성이 필요한 시나리오에서 장애 조치용 오리진 그룹으로 사용할 두 개의 오리진을 정의합니다.

오리진 그룹에는 두 개의 오리진(프라이머리 및 보조)과 사용자가 지정하는 장애 조치 기준이 포함됩니다. CloudFront에서 오리진 장애 조치를 지원할 오리진 그룹을 생성합니다. 이 메서드를 사용하여 오리진 그룹을 생성하거나 업데이트할 때 단일 오리진 대신 오리진 그룹을 지정할 수 있습니다. CloudFront는 장애 조치 기준을 사용하여 프라이머리 오리진에서 보조 오리진으로 장애 조치를 수행합니다.

배포에 VPC 오리진이 구성된 경우 이 메서드를 통해 VPC 오리진을 사용하여 오리진 그룹을 생성할 수 있습니다. 자세한 내용은 VPC 오리진으로 액세스 제한 섹션을 참조하세요.

> **참고**
>
> - `createRequestOriginGroup()` 함수는 상호 TLS(오리진)가 활성화된 오리진을 포함하는 오리진 그룹 생성을 지원하지 않습니다. 상호 TLS(오리진) 오리진이 있는 오리진 그룹은 CloudFront Functions를 통해 동적으로 생성할 수 없습니다.
> - 상호 TLS(오리진)를 사용한 오리진 장애 조치 기능이 필요한 경우 함수에서 동적으로 생성하는 대신 CloudFront 배포 설정에서 오리진 그룹을 직접 구성하세요.

#### 요청

```
createRequestOriginGroup({origin_group_properties})
```

앞의 예시에서 `origin_group_properties`에는 다음이 포함될 수 있습니다.

originIds(필수)

`origin_ids` 배열이며, 여기서 `origin_id`는 함수를 실행하는 배포에서 오리진의 오리진 이름을 가리키는 문자열입니다. 배열의 일부로 두 개의 오리진을 제공해야 합니다. 목록의 첫 번째 오리진은 프라이머리 오리진이고 두 번째 오리진은 장애 조치를 위한 보조 오리진 역할을 합니다.

originOverrides(선택 사항)

`{origin_overrides}` 파라미터를 사용하여 몇 가지 고급 설정을 덮어쓸 수 있습니다. `origin overrides`에는 다음 사항이 포함될 수 있습니다.

hostHeader(선택 사항, S3가 아닌 사용자 지정 오리진의 경우)

오리진에 요청할 때 사용할 호스트 헤더입니다. 제공하지 않으면 `domainName` 파라미터의 값이 사용됩니다.

호스트 헤더 또는 도메인 이름 파라미터가 제공되지 않은 경우 할당된 오리진의 도메인 이름이 사용되고, 오리진으로 전달(FTO) 정책에 호스트가 포함된 경우 수신 요청의 호스트 헤더가 사용됩니다. 호스트 헤더에는 콜론(`:`)이 포함될 수 없으며 IP 주소가 될 수 없습니다. 호스트 헤더는 최대 253자까지 작성할 수 있습니다.

sni(선택 사항, S3가 아닌 사용자 지정 오리진의 경우)

서버 이름 표시(SNI)는 클라이언트가 TLS 핸드셰이크 프로세스를 시작할 때 연결을 시도하는 호스트 이름을 나타내는 Transport Layer Security(TLS) 프로토콜의 확장입니다. 이 값은 오리진 서버의 TLS 인증서에 있는 일반 이름과 일치해야 합니다. 그렇지 않으면 오리진 서버에서 오류가 발생할 수 있습니다.

제공하지 않으면 `hostHeader` 파라미터의 값이 사용됩니다. 호스트 헤더가 제공되지 않으면 `domainName` 파라미터의 값이 사용됩니다.

호스트 헤더 또는 도메인 이름 파라미터가 제공되지 않은 경우 할당된 오리진의 도메인 이름이 사용되고, 오리진으로 전달(FTO) 정책에 호스트가 포함된 경우 수신 요청의 호스트 헤더가 사용됩니다. SNI에는 콜론(`:`)이 포함될 수 없으며 IP 주소가 될 수 없습니다. SNI는 최대 253자까지 작성할 수 있습니다.

allowedCertificateNames(선택 사항, S3가 아닌 사용자 지정 오리진의 경우)

CloudFront에서 오리진 서버와의 TLS 핸드셰이크 중에 오리진 서버 TLS 인증서와 일치하는 도메인의 유효성을 검사하는 데 사용할 유효한 인증서 이름 목록을 포함할 수 있습니다. 이 필드에는 유효한 도메인 이름 배열이 필요하며 `*.example.com`과 같은 와일드카드 도메인이 포함될 수 있습니다.

허용되는 인증서 이름은 최대 20개까지 지정할 수 있습니다. 각 인증서의 이름은 최대 64자까지 가능합니다.

selectionCriteria(선택 사항)

`default` 오리진 장애 조치 기준을 사용할지 아니면 `media-quality-score` 기반 장애 조치 로직을 사용할지 선택합니다. 유효한 값은 다음과 같습니다.

- `default`는 `failoverCriteria`에 지정된 상태 코드를 기반으로 장애 조치 기준을 사용합니다. 함수에서 `selectionCriteria`를 설정하지 않으면 `default`가 사용됩니다.
- `media-quality-score`는 미디어 인식 라우팅 기능을 사용할 때 사용됩니다.

failoverCriteria(필수)

프라이머리 오리진에서 반환될 때 CloudFront가 보조 오리진으로 장애 조치하도록 트리거하는 상태 코드 배열입니다. 기존 오리진 그룹을 덮어쓰면 이 배열은 오리진 그룹의 원래 구성에 설정된 모든 장애 조치 상태 코드를 덮어씁니다.

`media-quality-score` `selectionCriteria`를 사용하는 경우 CloudFront는 미디어 품질 점수를 기반으로 요청을 라우팅하려고 시도합니다. 선택한 오리진이 이 배열에 설정된 오류 코드를 반환하면 CloudFront는 다른 오리진으로 장애 조치를 수행합니다.

Example- 요청 오리진 그룹 생성

다음 예시에서는 오리진 ID를 사용하여 요청에 대한 오리진 그룹을 생성합니다. 이러한 오리진 ID는 이 함수를 실행하는 데 사용되는 배포의 오리진 그룹 구성에서 가져옵니다.

선택적으로 `originOverrides`를 사용하여 `sni`, `hostHeader` 및 `allowedCertificateNames`에 대한 오리진 그룹 구성을 재정의할 수 있습니다.

```
import cf from 'cloudfront';

function handler(event) {
    cf.createRequestOriginGroup({
        "originIds": [
            {
                "originId": "origin-1",
                "originOverrides": {
                    "hostHeader": "hostHeader.example.com",
                    "sni": "sni.example.com",
                    "allowedCertificateNames": ["cert1.example.com",
 "cert2.example.com", "cert3.example.com"]
                }
            },
            {
                "originId": "origin-2",
                "originOverrides": {
                    "hostHeader": "hostHeader2.example.com",
                    "sni": "sni2.example.com",
                    "allowedCertificateNames": ["cert4.example.com",
 "cert5.example.com"]
                }
            }
        ],
```

```
        "failoverCriteria": {
            "statusCodes": [500]
        }
    });

    event.request.headers['x-hookx'] = { value: 'origin-overrides' };
    return event.request;
}
```

## CloudFront SaaS Manager 속성에 대한 헬퍼 메서드

CloudFront SaaS Manager에서 다음 헬퍼 함수를 사용하여 직접 만든 함수에서 다중 테넌트 배포의 값을 검색합니다. 이 페이지의 예제를 사용하려면 먼저 JavaScript 런타임 2.0을 사용하여 CloudFront 함수를 만들어야 합니다. 자세한 설명은 CloudFront Functions를 위한 JavaScript 런타임 2.0 기능 섹션을 참조하세요.

주제

- 연결 그룹
- 배포 테넌트

### 연결 그룹

배포 테넌트와 연결된 연결 그룹에는 도메인 이름이 있습니다.

이 값을 가져오려면 이벤트 객체의 `context` 하위 객체에서 `endpoint` 필드를 사용합니다.

요청

```
const value = event.context.endpoint;
```

응답

응답은 연결 그룹의 도메인 이름이 포함된 `string`(예: d111111abcdef8.cloudfront.net)입니다. `endpoint` 필드는 연결된 연결 그룹이 있는 다중 테넌트 배포에 대해 함수가 호출되는 경우에만 나타납니다. 자세한 내용은 컨텍스트 객체 섹션을 참조하세요.

### 배포 테넌트

CloudFront Functions에는 특정 배포 테넌트 값에 대한 액세스를 제공하는 모듈이 있습니다.

이 모듈을 사용하려면 다음 함수 코드의 첫 줄에 다음 스테이트먼트를 포함합니다.

```
import cf from 'cloudfront';
```

다음 예제는 handler 함수에서만 직접 또는 중첩 호출 함수를 통해 사용할 수 있습니다.

**distributionTenant.id** 필드

이 필드를 사용하여 배포 테넌트 ID의 값을 가져옵니다.

요청

```
const value = cf.distributionTenant.id;
```

응답

응답은 배포 테넌트 ID를 포함하는 string(예: dt_1a2b3c4d5e6f7)입니다.

오류 처리.

함수가 표준 배포에서 호출되는 경우 distributionTenant.id 필드를 지정하면 distributionTenant module is not available 유형 오류가 반환됩니다. 이 사용 사례를 처리하기 위해 코드에 try 및 catch 블록을 추가할 수 있습니다.

**distributionTenant.parameters.get()** 메서드

이 메서드를 사용하여 지정한 배포 테넌트 파라미터 이름의 값을 반환합니다.

```
distributionTenant.parameters.get("key");
```

key: 값을 가져오려는 배포 테넌트 파라미터 이름입니다.

요청

```
const value = distributionTenant.parameters.get("key");
```

응답

응답은 배포 테넌트 파라미터의 값을 포함하는 string입니다. 예를 들어 키 이름이 TenantPath인 경우 이 파라미터의 값은 tenant1일 수 있습니다.

오류 처리.

다음 오류가 표시될 수 있습니다.

- 함수가 표준 배포에서 호출되는 경우 `distributionTenant.parameters.get()` 메서드는 `distributionTenant module is not available` 유형 오류를 반환합니다.
- 지정한 배포 테넌트 파라미터가 존재하지 않으면 `DistributionTenantParameterKeyNotFound` 오류가 반환됩니다.

이 사용 사례를 관리하기 위해 코드에 `try` 및 `catch` 블록을 추가할 수 있습니다.

## async 및 await 사용

CloudFront Functions JavaScript 런타임 함수 2.0은 `Promise` 객체를 처리하기 위한 `async` 및 `await` 구문을 제공합니다. 프로미스는 `async`로 표시된 함수의 `await` 키워드를 통해 액세스할 수 있는 지연된 결과를 나타냅니다. 다양한 새 WebCrypto 함수가 프로미스를 사용합니다.

`Promise` 객체에 대한 자세한 내용은 Promise를 참조하세요.

> **Note**
>
> 다음 코드 샘플에는 JavaScript 런타임 2.0을 사용해야 합니다.
> `await`는 `async` 함수 내에서만 사용할 수 있습니다. `async` 인수 및 종료는 지원되지 않습니다.

```
async function answer() {
    return 42;
}

// Note: async, await can be used only inside an async function. async arguments and
 closures are not supported.

async function handler(event) {
    // var answer_value = answer(); // returns Promise, not a 42 value
    let answer_value = await answer(); // resolves Promise, 42
    console.log("Answer"+answer_value);
    event.request.headers['answer'] = { value : ""+answer_value };
    return event.request;
}
```

다음 예제 JavaScript 코드는 `then` 체인 메서드로 프로미스를 보는 방법을 보여줍니다. `catch`를 사용하여 오류를 확인할 수 있습니다.

> **Warning**
>
> promise 결합자(예: `Promise.all`, `Promise.any`) 및 promise 체인 메서드(예: `then` 및 `catch`)를 사용하려면 많은 함수 메모리 사용량이 필요할 수 있습니다. 함수가 최대 함수 메모리 할당량을 초과하면 실행에 실패합니다. 이 오류를 방지하도록 `promise` 메서드 대신 `await` 구문을 사용하는 것이 좋습니다.

```
async function answer() {
    return 42;
}

async function squared_answer() {
   return answer().then(value => value * value)
}
// Note: async, await can be used only inside an async function. async arguments and
 closures are not supported.
async function handler(event) {
    // var answer_value = answer(); // returns Promise, not a 42 value
    let answer_value = await squared_answer(); // resolves Promise, 42
    console.log("Answer"+answer_value);
    event.request.headers['answer'] = { value : ""+answer_value };
    return event.request;
}
```

## CloudFront Functions에 대한 CWT 지원

이 섹션에서는 CloudFront 엣지 로케이션에서 보안 토큰 기반 인증 및 권한 부여를 활성화하는 CloudFront Functions의 CBOR 웹 토큰(CWT) 지원에 대한 세부 정보를 제공합니다. 이 지원은 CloudFront Functions에서 액세스할 수 있는 모듈로 제공됩니다.

이 모듈을 사용하려면 JavaScript 런타임 2.0을 사용하여 CloudFront Functions을 만들고 함수 코드의 첫 번째 줄에 다음 스테이트먼트를 포함합니다.

```
import cf from 'cloudfront';
```

이 모듈과 연결된 메서드는 다음을 통해 액세스할 수 있습니다(여기서 *는 모듈에 있는 다양한 함수를 나타내는 와일드카드).

```
cf.cwt.*
```

자세한 내용은 CloudFront Functions를 위한 JavaScript 런타임 2.0 기능 섹션을 참조하세요.

현재 모듈은 최대 토큰 크기에 대해 한도가 1KB인 HS256(HMAC-SHA256) 알고리즘을 사용하는 MAC0 구조만 지원합니다.

토큰 구조

이 섹션에서는 CWT 모듈에서 예상되는 토큰 구조를 다룹니다. 모듈은 토큰에 태그를 올바르게 지정하고 식별할 수 있어야 합니다(예: COSE MAC0). 또한 토큰의 구조와 마찬가지로 모듈은 CBOR 객체 서명 및 암호화(COSE)[RFC 8152]에서 설정한 표준을 따릅니다.

```
( // CWT Tag (Tag value: 61) --- optional
    ( // COSE MAC0 Structure Tag (Tag value: 17) --- required
        [
            protectedHeaders,
            unprotectedHeaders,
            payload,
            tag,
        ]
    )
)
```

Example: COSE MAC0 구조를 사용한 CWT

```
61( // CWT tag
    17( // COSE_MAC0 tag
        [
            { // Protected Headers
                1: 4  // algorithm : HMAC-256-64
            },
            { // Unprotected Headers
                4: h'53796d6d6574726963323536' // kid : Symmetric key id
            },
            { // Payload
                1: "https://iss.example.com", // iss
                2: "exampleUser", // sub
                3: "https://aud.example.com", // aud
                4: 1444064944, // exp
                5: 1443944944, // nbf
                6: 1443944944, // iat
            },
            h'093101ef6d789200' // tag
        ]
```

```
    )
)
```

> **Note**
>
> CWT 태그는 토큰을 생성할 때 선택 사항입니다. 그러나 COSE 구조 태그는 필수입니다.

validateToken() 메서드

함수는 지정된 키를 사용하여 CWT 토큰을 디코딩하고 검증합니다. 검증에 성공하면 디코딩된 CWT 토큰을 반환합니다. 그렇지 않으면 오류가 발생합니다. 이 함수는 클레임 세트에 대한 검증을 수행하지 않습니다.

요청

```
cf.cwt.validateToken(token, handlerContext{key})
```

Parameters

토큰(필수)

검증을 위한 인코딩된 토큰입니다. JavaScript 버퍼여야 합니다.

handlerContext(필수)

validateToken 직접 호출에 대한 컨텍스트를 저장하는 JavaScript 객체입니다. 현재 키 속성만 지원됩니다.

키(필수)

메시지 다이제스트 계산을 위한 비밀 키입니다. 문자열 또는 JavaScript 버퍼로 제공할 수 있습니다.

응답

`validateToken()` 메서드가 성공적으로 검증된 토큰을 반환하면 함수의 응답은 다음 형식의 `CWTObject`입니다. 디코딩되면 모든 클레임 키가 문자열로 표시됩니다.

```
CWTObject {
    protectedHeaders,
    unprotectedHeaders,
```

```
    payload
}
```

### 예제 - 토큰의 일부로 전송된 KID로 토큰 검증

이 예제는 키드가 헤더에서 추출되는 CWT 토큰 검증을 보여줍니다. 그런 다음 키드가 CloudFront Functions KeyValueStore로 전달되어 토큰을 검증하는 데 사용되는 보안 키를 가져옵니다.

```
import cf from 'cloudfront'

const CwtClaims = {
   iss: 1,
   aud: 3,
   exp: 4
}

async function handler(event) {
    try {
        let request = event.request;
        let encodedToken = request.headers['x-cwt-token'].value;
        let kid = request.headers['x-cwt-kid'].value;

        // Retrieve the secret key from the kvs
        let secretKey = await cf.kvs().get(kid);

        // Now you can use the secretKey to decode & validate the token.
        let tokenBuffer = Buffer.from(encodedToken, 'base64url');

        let handlerContext = {
           key: secretKey,
        }

        try {
            let cwtObj = cf.cwt.validateToken(tokenBuffer, handlerContext);

            // Check if token is expired
            const currentTime = Math.floor(Date.now() / 1000); // Current time in
 seconds
            if (cwtObj[CwtClaims.exp] && cwtObj[CwtClaims.exp] < currentTime) {
                return {
                    statusCode: 401,
                    statusDescription: 'Token expired'
                };
```

```
            }
        } catch (error) {
            return {
               statusCode: 401,
               statusDescription: 'Invalid token'
            };
         }
    } catch (error) {
        return {
            statusCode: 402,
            statusDescription: 'Token processing failed'
        };
     }
    return request;
}
```

generateToken() 메서드

이 함수는 제공된 페이로드 및 컨텍스트 설정을 사용하여 새 CWT 토큰을 생성합니다.

요청

```
cf.cwt.generateToken(generatorContext, payload)
```

Parameters

generatorContext(필수)

이 JavaScript 객체는 토큰을 생성하기 위한 컨텍스트로 사용되며 다음 키 값 페어를 포함합니다.

cwtTag(선택 사항)

이 값은 부울로, `true`인 경우 `cwtTag`를 추가해야 함을 지정합니다.

coseTag(필수)

COSE 태그 유형을 지정합니다. 현재 `MAC0`만 지원됩니다.

키(필수)

메시지 다이제스트를 계산하는 비밀 키입니다. 이 값은 문자열 또는 JavaScript `Buffer`일 수 있습니다.

페이로드(필수)

인코딩을 위한 토큰 페이로드입니다. 페이로드는 `CWTObject` 형식이어야 합니다.

## 응답

인코딩된 토큰이 포함된 JavaScript 버퍼를 반환합니다.

### Example: CWT 토큰 생성

```
import cf from 'cloudfront';

const CwtClaims = {
    iss: 1,
    sub: 2,
    exp: 4
};

const CatClaims = {
    catu: 401,
    catnip: 402,
    catm: 403,
    catr: 404
};

const Catu = {
    host: 1,
    path: 2,
    ext: 3
};

const CatuMatchTypes = {
    prefix_match: 1,
    suffix_match: 2,
    exact_match: 3
};

const Catr = {
    renewal_method: 1,
    next_renewal_time: 2,
    max_uses: 3
};

async function handler(event) {
    try {
        const response = {
            statusCode: 200,
            statusDescription: 'OK',
```

```
            headers: {}
        };

        const commonAccessToken = {
            protected: {
                1: "5",
            },
            unprotected: {},
            payload: {
                [CwtClaims.iss]: "cloudfront-documentation",
                [CwtClaims.sub]: "cwt-support-on-cloudfront-functions",
                [CwtClaims.exp]: 1740000000,
                [CatClaims.catu]: {
                    [Catu.host]: {
                        [CatuMatchTypes.suffix_match]: ".cloudfront.net"
                    },
                    [Catu.path]: {
                        [CatuMatchTypes.prefix_match]: "/media/live-stream/cf-4k/"
                    },
                    [Catu.ext]: {
                        [CatuMatchTypes.exact_match]: [
                            ".m3u8",
                            ".ts",
                            ".mpd"
                        ]
                    }
                },
                [CatClaims.catnip]: [
                    "[IP_ADDRESS]",
                    "[IP_ADDRESS]"
                ],
                [CatClaims.catm]: [
                    "GET",
                    "HEAD"
                ],
                [CatClaims.catr]: {
                    [Catr.renewal_method]: "header_renewal",
                    [Catr.next_renewal_time]: 1750000000,
                    [Catr.max_uses]: 5
                }
            }
        };

        if (!request.headers['x-cwt-kid']) {
```

```
            throw new Error('Missing x-cwt-kid header');
        }

        const kid = request.headers['x-cwt-kid'].value;
        const secretKey = await cf.kvs().get(kid);

        if (!secretKey) {
            throw new Error('Secret key not found for provided kid');
        }

        try {
            const genContext = {
                cwtTag: true,
                coseTag: "MAC0",
                key: secretKey
            };

            const tokenBuffer = cf.cwt.generateToken(commonAccessToken, genContext);
            response.headers['x-generated-cwt-token'] = { value:
 tokenBuffer.toString('base64url') };

            return response;
        } catch (tokenError) {
            return {
                statusCode: 401,
                statusDescription: 'Could not generate the token'
            };
        }
    } catch (error) {
        return {
            statusCode: 402,
            statusDescription: 'Token processing failed'
        };
    }
}
```

## Example: 일부 로직을 기반으로 토큰 새로 고침

```
import cf from 'cloudfront'

const CwtClaims = {
   iss: 1,
   aud: 3,
```

```
   exp: 4
}

async function handler(event) {
    try {
        let request = event.request;
        let encodedToken = request.headers['x-cwt-token'].value;
        let kid = request.headers['x-cwt-kid'].value;
        let secretKey = await cf.kvs().get(kid); // Retrieve the secret key from the
 kvs

        // Now you can use the secretKey to decode & validate the token.
        let tokenBuffer = Buffer.from(encodedToken, 'base64url');

        let handlerContext = {
           key: secretKey,
        }

        try {
            let cwtJSON = cf.cwt.validateToken(tokenBuffer, handlerContext);

            // Check if token is expired
            const currentTime = Math.floor(Date.now() / 1000); // Current time in
 seconds
            if (cwtJSON[CwtClaims.exp] && cwtJSON[CwtClaims.exp] < currentTime) {
                // We can regnerate the token and add 8 hours to the expiry time
                cwtJSON[CwtClaims.exp] = Math.floor(Date.now() / 1000) + (8 * 60 * 60);

                let genContext = {
                  coseTag: "MAC0",
                  key: secretKey
                }

                let newTokenBuffer = cf.cwt.generateToken(cwtJSON, genContext);
                 request.headers['x-cwt-regenerated-token'] =
 newTokenBuffer.toString('base64url');
            }
        } catch (error) {
            return {
               statusCode: 401,
               statusDescription: 'Invalid token'
            };
         }
    }
```

```
    catch (error) {
        return {
            statusCode: 402,
            statusDescription: 'Token processing failed'
        };
     }
    return request;
}
```

## 일반 헬퍼 메서드

이 페이지에서는 CloudFront Functions 내의 추가 헬퍼 메서드를 제공합니다. 이러한 메서드를 사용하려면 JavaScript 런타임 2.0을 사용하여 CloudFront Functions를 생성합니다.

```
import cf from 'cloudfront';
```

자세한 내용은 CloudFront Functions를 위한 JavaScript 런타임 2.0 기능 섹션을 참조하세요.

### `edgeLocation` 메타데이터

이 메서드를 사용하려면 `cloudfront` 모듈을 사용해야 합니다.

> **Note**
>
> 뷰어 요청 함수에만 이 방법을 사용할 수 있습니다. 뷰어 응답 함수의 경우 이 메서드는 비어 있습니다.

이 JavaScript 객체를 사용하여 요청을 처리하는 데 사용되는 엣지 로케이션 공항 코드, 예상 리전 엣지 캐시 리전 또는 CloudFront 서버 IP 주소를 가져옵니다. 이 메타데이터는 뷰어 요청 이벤트 트리거만 사용할 수 있습니다.

```
cf.edgeLocation = {
    name: SEA
    serverIp: 1.2.3.4
    region: us-west-2
}
```

`cf.edgeLocation` 객체에는 다음 사항이 포함될 수 있습니다.

이름

요청을 처리한 엣지 로케이션의 세 글자 IATA 코드입니다.

serverIp

요청을 처리한 서버의 IPv4 또는 IPv6 주소입니다.

region

캐시 누락이 있는 경우 요청이 사용할 것으로 예상되는 CloudFront 리전 엣지 캐시(REC)입니다. 이 값은 예상 REC를 사용할 수 없고 백업 REC가 요청에 사용되는 경우 업데이트되지 않습니다. 여기에는 사용 중인 Origin Shield 위치가 포함되지 않습니다. 단, 기본 REC와 Origin Shield가 동일한 위치인 경우는 예외입니다.

> **Note**
>
> CloudFront가 오리진 장애 조치를 사용하도록 구성된 경우 CloudFront Functions는 두 번째로 간접 호출되지 않습니다. 자세한 내용은 CloudFront 오리진 장애 조치를 통한 고가용성 최적화 섹션을 참조하세요.

### `rawQueryString()` 메서드

이 메서드에는 `cloudFront` 모듈이 필요하지 않습니다.

`rawQueryString()` 메서드를 사용하여 구문 분석되지 않은 쿼리 문자열과 변경되지 않은 쿼리 문자열을 문자열로 검색합니다.

요청

```
function handler(event) {
    var request = event.request;
    const qs = request.rawQueryString();
}
```

응답

수신 요청의 전체 쿼리 문자열을 선행 ? 없이 문자열 값으로 반환합니다.

- 쿼리 문자열이 없지만 ?가 있는 경우 함수는 빈 문자열을 반환합니다.

- 쿼리 문자열이 없고 ?가 없는 경우 함수는 `undefined`를 반환합니다.

사례 1: 반환된 전체 쿼리 문자열(선행 **?** 없음)

수신 요청 URL: `https://example.com/page?name=John&age=25&city=Boston`

`rawQueryString()`에서 를 반환합니다.`"name=John&age=25&city=Boston"`

사례 2: 빈 문자열 반환됨(**?**가 있지만 파라미터가 없는 경우)

수신 요청 URL: `https://example.com/page?`

`rawQueryString()`에서 를 반환합니다.`""`

사례 3: **`undefined`** 반환됨(쿼리 문자열 및 **?** 없음)

수신 요청 URL: `https://example.com/page`

`rawQueryString()`에서 를 반환합니다.`undefined`

## 함수 생성

함수는 두 단계로 생성합니다.

1. 함수 코드를 JavaScript로 생성합니다. CloudFront 콘솔의 기본 예제를 사용하거나, 직접 작성할 수 있습니다. 자세한 정보는 다음의 주제를 참조하세요.
   - 함수 코드 작성
   - the section called “이벤트 구조”
   - CloudFront의 CloudFront Functions 예제
2. CloudFront를 사용하여 함수를 생성하고 코드를 포함합니다. 코드는 함수 내에 존재합니다(참조용이 아님).

Console

함수를 만들려면

1. CloudFront 콘솔(https://console.aws.amazon.com/cloudfront/v4/home#/functions)에 로그인하고 함수 페이지를 엽니다.
2. 함수 생성(Create function)을 선택합니다.

3. AWS 계정 내에서 고유한 함수 이름을 입력한 다음 JavaScript 버전을 선택하고 계속을 선택합니다. 새 함수의 세부 정보 페이지가 나타납니다.

> **Note**
>
> 함수에서 키 값 페어를 사용하려면 JavaScript 런타임 2.0을 선택해야 합니다.

4. 함수 코드 섹션에서 빌드 탭을 선택하고 함수 코드를 입력합니다. 빌드 탭에 포함된 샘플 코드는 함수 코드의 기본 구문을 보여줍니다.
5. Save changes(변경 사항 저장)를 선택합니다.
6. 함수 코드에서 키 값 페어를 사용하는 경우 키 값 저장소를 연결해야 합니다.

   함수를 처음 생성할 때 키 값 저장소를 연결할 수 있습니다. 또는 나중에 함수를 업데이트하여 연결할 수도 있습니다.

   지금 키 값 저장소를 연결하려면 다음 단계를 따르세요.

   - KeyValueStore 연결 섹션에서 기존 KeyValueStore 연결을 선택합니다.
   - 함수의 키 값 페어가 들어 있는 키 값 저장소를 선택한 다음 KeyValueStore 연결을 선택합니다.

   CloudFront는 저장소를 함수와 즉시 연결하므로 함수를 저장하지 않아도 됩니다.

## CLI

CLI를 사용하는 경우 일반적으로 먼저 파일에 함수 코드를 생성한 다음 AWS CLI로 함수를 생성합니다.

함수를 만들려면

1. 함수 코드를 파일로 만든 다음 컴퓨터가 연결할 수 있는 디렉터리에 저장합니다.
2. 다음 예제와 같이 명령을 실행합니다. 이 예제에서는 `fileb://` 표기법을 사용하여 파일을 전달합니다. 명령을 더 쉽게 읽을 수 있도록 줄 바꿈도 포함되어 있습니다.

```
aws cloudfront create-function \
    --name MaxAge \
    --function-config '{"Comment":"Max Age 2 years","Runtime":"cloudfront-
js-2.0","KeyValueStoreAssociations":{"Quantity":1,"Items":
[{"KeyValueStoreARN":"arn:aws:cloudfront::111122223333:key-value-store/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"}]}}' \
```

```
    --function-code fileb://function-max-age-v1.js
```

> **Notes**
>
> - `Runtime` - JavaScript의 버전입니다. 함수에 키 값 페어를 사용하려면 버전 2.0을 지정해야 합니다.
> - `KeyValueStoreAssociations` - 함수에서 키 값 페어를 사용하는 경우 함수를 처음 생성할 때 키 값 저장소를 연결할 수 있습니다. 또는 `update-function`을 사용하여 나중에 연결할 수도 있습니다. 각 함수에 키 값 저장소가 하나만 연결될 수 있기 때문에 `Quantity`는 항상 `1`입니다.

명령이 제대로 실행되면 다음과 비슷한 출력이 표시됩니다.

```
ETag: ETVABCEXAMPLE
FunctionSummary:
  FunctionConfig:
    Comment: Max Age 2 years
    Runtime: cloudfront-js-2.0
    KeyValueStoreAssociations= \
      {Quantity=1, \
      Items=[{KeyValueStoreARN='arn:aws:cloudfront::111122223333:key-value-
store/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'}]} \
  FunctionMetadata:
    CreatedTime: '2021-04-18T20:38:56.915000+00:00'
    FunctionARN: arn:aws:cloudfront::111122223333:function/MaxAge
    LastModifiedTime: '2023-11-19T20:38:56.915000+00:00'
    Stage: DEVELOPMENT
  Name: MaxAge
  Status: UNPUBLISHED
Location: https://cloudfront.amazonaws.com/2020-05-31/function/
arn:aws:cloudfront:::function/MaxAge
```

대부분의 정보는 요청에서 반복된 내용입니다. 기타 정보는 CloudFront에 의해 추가됩니다.

> **Notes**
>
> - `ETag` - 이 값은 키 값 저장소를 수정할 때마다 변경됩니다. 나중에 이 값과 함수 이름을 사용하여 함수를 참조할 수 있습니다. 항상 현재 `ETag`를 사용하도록 하세요.

- `FunctionARN` - CloudFront 함수의 ARN입니다.
- 111122223333 – AWS 계정입니다.
- `Stage` - 함수의 스테이지(`LIVE` 또는 `DEVELOPMENT`)입니다.
- `Status` - 함수의 상태(`PUBLISHED` 또는 `UNPUBLISHED`)입니다.

생성한 함수는 `DEVELOPMENT` 스테이지에 추가됩니다. 함수를 게시하기 전에 함수를 테스트하는 것이 좋습니다. 함수를 게시하고 나면 함수가 `LIVE` 스테이지로 변경됩니다.

## 함수 테스트

라이브 스테이지(프로덕션)에 함수를 배포하기 전에 함수를 테스트하여 의도한 대로 작동하는지 확인할 수 있습니다. 함수를 테스트하려면 CloudFront 배포가 프로덕션 환경에서 수신할 수 있는 HTTP 요청 또는 응답을 나타내는 이벤트 객체를 지정합니다.

CloudFront 함수는 다음을 수행합니다.

1. 제공된 이벤트 객체를 입력으로 사용하여 함수를 실행합니다.
2. 함수 로그 또는 오류 메시지 및 함수의 컴퓨팅 사용률과 함께 함수의 결과(수정된 이벤트 객체)를 반환합니다. 컴퓨팅 사용률에 대한 자세한 내용은 the section called “컴퓨팅 활용 이해” 섹션을 참조하세요.

> **Note**
>
> 함수를 테스트할 때 CloudFront는 함수 실행 오류에 대해서만 검증합니다. CloudFront는 요청이 게시된 후 성공적으로 전달되는지 여부를 검증하지 않습니다. 예를 들어 함수가 필수 헤더를 삭제하면 코드에 문제가 없기 때문에 테스트가 성공합니다. 그러나 함수를 게시하고 배포와 연결하면 CloudFront를 통해 요청이 이루어질 때 함수가 실패합니다.

목차


- 이벤트 객체 설정
- 함수 테스트
- 컴퓨팅 활용 이해

## 이벤트 객체 설정

함수를 테스트하기 전에 함수를 테스트할 이벤트 객체를 설정해야 합니다. 이때 여러 가지 선택지가 있습니다.

선택 1: 이벤트 객체를 저장하지 않고 설정

CloudFront 콘솔의 시각적 편집기에서 이벤트 객체를 설정한 후 저장하지 않아도 됩니다.

저장되지 않았더라도 이 이벤트 객체를 사용하여 CloudFront 콘솔에서 함수를 테스트할 수 있습니다.

선택 2: 시각적 편집기에서 이벤트 객체 생성

CloudFront 콘솔의 시각적 편집기에서 이벤트 객체를 설정한 후 저장하지 않아도 됩니다. 예를 들어, 가능한 여러 입력을 테스트할 수 있도록 각 함수에 대해 10개의 이벤트 객체를 생성할 수 있습니다.

이러한 방식으로 이벤트 객체를 생성하면 이벤트 객체를 사용하여 CloudFront 콘솔에서 함수를 테스트할 수 있습니다. AWS API 또는 SDK를 사용하여 함수를 테스트하는 데는 이벤트 객체를 사용할 수 없습니다.

선택 3: 텍스트 편집기를 사용하여 이벤트 객체 생성

텍스트 편집기를 사용하여 JSON 형식으로 이벤트 객체를 만들 수 있습니다. 이벤트 객체의 구조에 대한 자세한 내용은 이벤트 구조 섹션을 참조하세요.

이 이벤트 객체로 CLI를 사용하여 함수를 테스트할 수 있습니다. 하지만 CloudFront 콘솔에서 함수를 테스트하는 데는 사용할 수 없습니다.

이벤트 객체를 생성하려면(옵션 1 또는 2)

1. CloudFront 콘솔(https://console.aws.amazon.com/cloudfront/v4/home#/functions)에 로그인하고 함수 페이지를 엽니다.

   테스트할 함수를 선택합니다.
2. 함수 세부 정보 페이지에서 테스트 탭을 선택합니다.
3. 이벤트 유형에서 다음 옵션 중 하나를 선택합니다.

   - 함수가 HTTP 요청을 수정하거나 요청에 따라 응답을 생성하는 경우 뷰어 요청을 선택합니다. 요청 섹션이 나타납니다.

- 뷰어 응답을 선택합니다. 요청 및 응답 섹션이 나타납니다.

4. 이벤트에 포함할 필드를 작성합니다. JSON 편집을 선택하여 원시 JSON을 확인할 수 있습니다.
5. (선택 사항) 이벤트를 저장하려면 저장을 선택하고 테스트 이벤트 저장에서 이름을 입력한 다음 저장을 선택합니다.

   JSON 편집을 선택하고 원시 JSON을 복사한 다음 CloudFront 외부의 자체 파일에 저장할 수도 있습니다.

이벤트 객체를 생성하려면(옵션 3)

텍스트 편집기를 사용하여 이벤트 객체를 생성합니다. 컴퓨터가 연결할 수 있는 디렉터리에 파일을 저장합니다.

다음 가이드라인을 따르도록 합니다.

- `distributionDomainName`, `distributionId`, `requestId` 필드는 생략하세요.
- 헤더, 쿠키, 쿼리 문자열의 이름은 소문자여야 합니다.

이러한 방식으로 이벤트 객체를 생성하는 한 가지 방법은 시각적 편집기를 사용하여 샘플을 만드는 것입니다. 샘플 형식이 올바른지 확인할 수 있습니다. 그런 다음 원시 JSON을 복사해 텍스트 편집기에 붙여넣고 파일을 저장할 수 있습니다.

이벤트의 구조에 대한 자세한 내용은 이벤트 구조 섹션을 참조하세요.

## 함수 테스트

CloudFront 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 함수를 테스트할 수 있습니다.

Console

함수를 테스트하려면

1. CloudFront 콘솔(https://console.aws.amazon.com/cloudfront/v4/home#/functions)에 로그인하고 함수 페이지를 엽니다.
2. 테스트할 함수를 선택합니다.
3. 테스트 탭을 선택합니다.

4. 올바른 이벤트가 표시되는지 확인합니다. 현재 표시된 이벤트에서 다른 이벤트로 전환하려면 테스트 이벤트 선택 필드에서 다른 이벤트를 선택하세요.
5. 함수 테스트를 선택합니다. 콘솔에는 함수 로그와 컴퓨팅 활용을 비롯한 함수 출력이 표시됩니다.

## CLI

aws cloudfront test-function 명령을 사용하여 함수를 테스트할 수 있습니다.

함수를 테스트하려면

1. 명령줄 창을 엽니다.
2. 지정된 파일이 들어 있는 디렉터리에서 다음 명령을 실행합니다.

   이 예제에서는 `fileb://` 표기법을 사용하여 이벤트 객체 파일을 전달합니다. 명령을 더 쉽게 읽을 수 있도록 줄 바꿈도 포함되어 있습니다.

```
aws cloudfront test-function \
    --name MaxAge \
    --if-match ETVABCEXAMPLE \
    --event-object fileb://event-maxage-test01.json \
    --stage DEVELOPMENT
```

> **Notes**
>
> - 함수의 이름과 ETag(`if-match` 파라미터 내)로 함수를 참조합니다. 파일 시스템에서의 위치를 기준으로 이벤트 객체를 참조합니다.
> - 스테이지는 `DEVELOPMENT` 또는 `LIVE`일 수 있습니다.

   명령이 제대로 실행되면 다음과 비슷한 출력이 표시됩니다.

```
TestResult:
  ComputeUtilization: '21'
  FunctionErrorMessage: ''
  FunctionExecutionLogs: []
  FunctionOutput: '{"response":{"headers":{"cloudfront-functions":
{"value":"generated-by-CloudFront-Functions"},"location":{"value":"https://
```

```
aws.amazon.com/cloudfront/"}},"statusDescription":"Found","cookies":
{},"statusCode":302}}'
  FunctionSummary:
    FunctionConfig:
      Comment: MaxAge function
      Runtime: cloudfront-js-2.0
      KeyValueStoreAssociations= \
      {Quantity=1, \
      Items=[{KeyValueStoreARN='arn:aws:cloudfront::111122223333:key-value-
store/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'}]} \
    FunctionMetadata:
      CreatedTime: '2021-04-18T20:38:56.915000+00:00'
      FunctionARN: arn:aws:cloudfront::111122223333:function/MaxAge
      LastModifiedTime: '2023-17-20T10:38:57.057000+00:00'
      Stage: DEVELOPMENT
    Name: MaxAge
    Status: UNPUBLISHED
```

**Notes**

- `FunctionExecutionLogs`에는 해당 함수가 `console.log()` 명령문(있는 경우)에 작성한 로그 행 목록이 포함되어 있습니다.
- `ComputeUtilization`에는 함수 실행에 대한 정보가 들어 있습니다. the section called "컴퓨팅 활용 이해" 섹션을 참조하세요.
- `FunctionOutput`에는 함수가 반환한 이벤트 객체가 포함됩니다.

## 컴퓨팅 활용 이해

컴퓨팅 활용(Compute utilization)은 함수가 실행되는 데 걸린 시간의 최대 허용 시간의 백분율입니다. 예를 들어, 값이 35이면 함수가 최대 허용 시간의 35%에서 완료되었음을 의미합니다.

함수가 계속해서 최대 허용 시간을 초과할 경우 CloudFront는 해당 함수를 제한합니다. 다음 목록에서는 컴퓨팅 사용률 값에 따라 함수가 제한될 가능성을 설명합니다.

컴퓨팅 사용률 값:

- 1 ~ 50 - 함수가 최대 허용 시간에 도달하기까지 많이 남았으므로 제한 없이 실행됩니다.
- 51 ~ 70 - 함수가 최대 허용 시간에 근접하고 있습니다. 함수 코드를 최적화하는 것이 좋습니다.

- 71 ~ 100 - 함수가 최대 허용 시간에 매우 근접하거나 최대 허용 시간을 초과합니다. 배포와 연결할 경우 CloudFront에서 이 함수를 제한할 가능성이 있습니다.

## 함수 업데이트

함수는 언제든지 업데이트할 수 있습니다. `DEVELOPMENT` 스테이지에 있는 함수 버전만 변경됩니다. 업데이트를 `DEVELOPMENT` 스테이지에서 `LIVE`로 복사하려면 함수를 게시해야 합니다.

CloudFront 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 함수의 코드를 업데이트할 수 있습니다.

Console

함수 코드를 업데이트하려면

1. CloudFront 콘솔(https://console.aws.amazon.com/cloudfront/v4/home#/functions)에 로그인하고 함수 페이지를 엽니다.

   업데이트할 함수를 선택합니다.
2. 편집을 선택하고 다음과 같이 변경합니다.

   - 세부 정보 섹션의 모든 필드를 업데이트합니다.

   - 연결된 키 값 저장소를 변경하거나 제거합니다. 키 값 저장소에 대한 자세한 내용은 the section called “ CloudFront KeyValueStore” 섹션을 참조하세요.

   - 함수 코드를 변경합니다. 빌드 탭을 선택하고 필요한 부분을 변경한 다음 변경 사항 저장을 선택하여 코드의 변경 사항을 저장합니다.

CLI

함수 코드를 업데이트하려면

1. 명령줄 창을 엽니다.
2. 다음 명령을 실행합니다.

   이 예제에서는 `fileb://` 표기법을 사용하여 파일을 전달합니다. 명령을 더 쉽게 읽을 수 있도록 줄 바꿈도 포함되어 있습니다.

```
aws cloudfront update-function \
    --name MaxAge \
    --function-config '{"Comment":"Max Age 2 years","Runtime":"cloudfront-
js-2.0","KeyValueStoreAssociations":{"Quantity":1,"Items":
[{"KeyValueStoreARN":"arn:aws:cloudfront::111122223333:key-value-store/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"}]}}' \
    --function-code fileb://function-max-age-v1.js \
    --if-match ETVABCEXAMPLE
```

**Notes**

- 함수의 이름과 ETag(`if-match` 파라미터 내)로 함수를 식별할 수 있습니다. 현재 ETag를 사용하도록 하세요. DescribeFunction API 작업에서 이 값을 얻을 수 있습니다.
- 변경하지 않을 때에도 `function-code`가 포함되어야 합니다.
- `function-config`에 유의하세요. 구성에 유지하려는 모든 내용을 전달해야 합니다. 특히 키 값 저장소는 다음과 같이 처리하세요.
  - 기존 키 값 저장소 연결(있는 경우)을 유지하려면 기존 저장소의 이름을 지정합니다.
  - 연결을 변경하려면 새 키 값 저장소의 이름을 지정합니다.
  - 연결을 제거하려면 `KeyValueStoreAssociations` 파라미터를 생략하세요.

명령이 제대로 실행되면 다음과 비슷한 출력이 표시됩니다.

```
ETag: ETVXYZEXAMPLE
FunctionSummary:
  FunctionConfig:
    Comment: Max Age 2 years \
    Runtime: cloudfront-js-2.0 \
    KeyValueStoreAssociations= \
      {Quantity=1, \
      Items=[{KeyValueStoreARN='arn:aws:cloudfront::111122223333:key-value-
store/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'}]} \
  FunctionMetadata: \
    CreatedTime: '2021-04-18T20:38:56.915000+00:00' \
    FunctionARN: arn:aws:cloudfront::111122223333:function/MaxAge \
```

```
    LastModifiedTime: '2023-12-19T23:41:15.389000+00:00' \
    Stage: DEVELOPMENT \
  Name: MaxAge \
  Status: UNPUBLISHED
```

대부분의 정보는 요청에서 반복된 내용입니다. 기타 정보는 CloudFront에 의해 추가됩니다.

> **Notes**
>
> - `ETag` - 이 값은 키 값 저장소를 수정할 때마다 변경됩니다.
> - `FunctionARN` - CloudFront 함수의 ARN입니다.
> - `Stage` - 함수의 스테이지(`LIVE` 또는 `DEVELOPMENT`)입니다.
> - `Status` - 함수의 상태(`PUBLISHED` 또는 `UNPUBLISHED`)입니다.

## 함수 게시

함수를 게시하면 `DEVELOPMENT` 스테이지에서 `LIVE` 스테이지로 함수가 복사됩니다.

함수와 연결된 캐시 동작이 없는 경우 이를 게시하면 캐시 동작과 연결할 수 있습니다. 캐시 동작은 `LIVE` 스테이지에 있는 함수에만 연결할 수 있습니다.

> **Important**
>
> - 함수를 게시하기 전에 함수를 테스트하는 것이 좋습니다.
> - 함수를 게시하면 배포가 완료되는 즉시 해당 함수와 연결된 모든 캐시 동작이 새로 게시된 복사본을 사용하여 자동으로 시작됩니다.

CloudFront 콘솔 또는 AWS CLI를 사용하여 함수를 게시할 수 있습니다.

Console

함수를 게시하려면

1. CloudFront 콘솔(https://console.aws.amazon.com/cloudfront/v4/home#/functions)에 로그인하고 함수 페이지를 엽니다.

2. 업데이트할 함수를 선택합니다.
3. 게시 탭을 선택한 다음 게시를 선택합니다. 함수가 하나 이상의 캐시 동작에 이미 연결되어 있는 경우 게시 및 업데이트를 선택합니다.
4. (선택 사항) 해당 함수와 연결된 배포를 보려면 연결된 CloudFront 배포(Associated CloudFront distributions)를 선택하여 해당 섹션을 확장합니다.

성공하면 페이지 상단에 **## ##**이 성공적으로 게시되었다는 배너가 표시됩니다. 빌드(Build) 탭을 선택한 다음 라이브(Live)를 선택하여 함수 코드의 라이브 버전을 볼 수도 있습니다.

CLI

함수를 게시하려면

1. 명령줄 창을 엽니다.
2. 다음 aws cloudfront publish-function 명령을 실행합니다. 이 예에서는 예제를 보다 읽기 쉽도록 줄 바꿈이 제공됩니다.

```
aws cloudfront publish-function \
    --name MaxAge \
    --if-match ETVXYZEXAMPLE
```

명령이 제대로 실행되면 다음과 비슷한 출력이 표시됩니다.

```
FunctionSummary:
  FunctionConfig:
    Comment: Max Age 2 years
    Runtime: cloudfront-js-2.0
  FunctionMetadata:
    CreatedTime: '2021-04-18T21:24:21.314000+00:00'
    FunctionARN: arn:aws:cloudfront::111122223333:function/ExampleFunction
    LastModifiedTime: '2023-12-19T23:41:15.389000+00:00'
    Stage: LIVE
  Name: MaxAge
  Status: UNASSOCIATED
```

## 배포에 함수 연결

배포에 함수를 사용하려면 함수를 배포의 하나 이상의 캐시 동작과 연결합니다. 여러 배포에서 여러 캐시 동작과 함수를 연결할 수 있습니다.

함수는 다음 중 무엇과도 연결할 수 있습니다.

- 기존 캐시 동작
- 기존 배포의 새로운 캐시 동작
- 새 배포의 새로운 캐시 동작

함수를 캐시 동작과 연결할 때는 이벤트 유형(event type)을 선택해야 합니다. 이벤트 유형에 따라 CloudFront가 함수를 실행하는 시기가 결정됩니다.

다음과 같은 이벤트 유형을 선택할 수 있습니다.

- 최종 사용자 요청 – 이 함수는 CloudFront가 최종 사용자의 요청을 수신할 때 실행됩니다.
- 최종 사용자 응답 – 이 함수는 CloudFront가 최종 사용자에게 응답을 반환하기 전에 실행됩니다.

CloudFront Functions와 함께 오리진 관련 이벤트 유형(오리진 요청 및 오리진 응답)을 사용할 수 없습니다. 대신 Lambda@Edge를 사용할 수 있습니다. 자세한 내용은 Lambda@Edge 함수를 트리거할 수 있는 CloudFront 이벤트 단원을 참조하십시오.

> **Note**
>
> 함수를 연결하기 전에 `LIVE` 스테이지에 게시해야 합니다.

CloudFront 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 배포에 함수를 연결할 수 있습니다. 다음 절차에서는 함수를 기존 캐시 동작에 연결하는 방법을 보여 줍니다.

Console

기존 캐시 동작에 함수를 연결하려면

1. CloudFront 콘솔(https://console.aws.amazon.com/cloudfront/v4/home#/functions)에 로그인하고 함수 페이지를 엽니다.
2. 연결할 함수를 선택합니다.
3. 함수 페이지에서 게시 탭을 선택합니다.
4. Publish 함수를 선택합니다.
5. 연결 추가를 선택합니다. 표시되는 대화 상자에서 배포, 이벤트 유형 및/또는 캐시 동작을 선택합니다.

이벤트 유형에서 함수를 실행할 시기를 선택합니다.

- 뷰어 요청 - CloudFront가 요청을 수신할 때마다 이 함수를 실행합니다.
- 뷰어 응답 - CloudFront가 응답을 반환할 때마다 이 함수를 실행합니다.

6. 구성을 저장하려면 연결 추가를 선택합니다.

CloudFront가 배포를 함수와 연결합니다. 연결된 배포가 배포될 때까지 몇 분 정도 기다립니다. 함수 세부 정보 페이지에서 배포 보기를 선택하여 진행 상황을 확인할 수 있습니다.

CLI

기존 캐시 동작에 함수를 연결하려면

1. 명령줄 창을 엽니다.
2. 함수에 연결할 캐시 동작을 가진 배포에 대한 배포 구성을 저장하려면 다음 명령을 입력합니다. 이 명령은 배포 구성을 `dist-config.yaml`이라는 이름의 파일에 저장합니다. 이 명령을 사용하려면 다음을 수행합니다.

   - *`DistributionID`*를 해당 배포의 ID로 바꿉니다.
   - 한 줄로 명령을 실행합니다. 이 예에서는 예제를 보다 읽기 쉽도록 줄 바꿈이 제공됩니다.

```
aws cloudfront get-distribution-config \
    --id DistributionID \
    --output yaml > dist-config.yaml
```

   명령이 성공하면 AWS CLI는 출력을 반환하지 않습니다.

3. 생성한 `dist-config.yaml`이라는 파일을 엽니다. 파일을 편집하여 다음과 같이 변경합니다.

   a. `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

   b. 캐시 동작에서 `FunctionAssociations`(이)라는 이름의 객체를 찾습니다. 함수 연결을 추가하려면 이 객체를 업데이트합니다. 함수 연결에 대한 YAML 구문은 다음 예제와 같습니다.

      - 다음 예제에서는 최종 사용자 요청 이벤트 객체를 보여줍니다. 최종 사용자 응답 이벤트 유형을 사용하려면 `viewer-request`을(를)`viewer-response`(으)로 바꿉니다.

- *arn:aws:cloudfront::111122223333:function/ExampleFunction*을 이 캐시 동작에 연결하려는 함수의 Amazon 리소스 이름(ARN)으로 바꿉니다. 함수 ARN을 얻으려면 aws cloudfront list-functions 명령을 사용할 수 있습니다.

```
FunctionAssociations:
  Items:
    - EventType: viewer-request
      FunctionARN: arn:aws:cloudfront::111122223333:function/ExampleFunction
  Quantity: 1
```

c. 이러한 변경을 수행한 후 파일을 저장합니다.

4. 다음 명령을 사용하여 배포를 업데이트하고 함수 연결을 추가합니다. 이 명령을 사용하려면 다음을 수행합니다.

- *DistributionID*를 해당 배포의 ID로 바꿉니다.
- 한 줄로 명령을 실행합니다. 이 예에서는 예제를 보다 읽기 쉽도록 줄 바꿈이 제공됩니다.

```
aws cloudfront update-distribution \
    --id DistributionID \
    --cli-input-yaml file://dist-config.yaml
```

명령이 성공하면 함수 연결로 방금 업데이트된 배포를 설명하는 다음과 같은 출력이 표시됩니다. 다음 예제 출력은 가독성을 위해 잘립니다.

```
Distribution:
  ARN: arn:aws:cloudfront::111122223333:distribution/EBEDLT3BGRBBW
  ... truncated ...
  DistributionConfig:
    ... truncated ...
    DefaultCacheBehavior:
      ... truncated ...
      FunctionAssociations:
        Items:
        - EventType: viewer-request
          FunctionARN: arn:aws:cloudfront::111122223333:function/ExampleFunction
        Quantity: 1
      ... truncated ...
  DomainName: d111111abcdef8.cloudfront.net
  Id: EDFDVBD6EXAMPLE
```

```
  LastModifiedTime: '2021-04-19T22:39:09.158000+00:00'
  Status: InProgress
ETag: E2VJGGQEG1JT8S
```

배포가 재배포되는 동안 배포의 `Status`가 `InProgress`로 변경됩니다. 새 배포 구성이 CloudFront 엣지 로케이션에 도달하면 해당 엣지 로케이션에 연결된 함수를 사용하기 시작합니다. 배포가 완전히 완료되면 `Status`가 다시 `Deployed`로 바뀝니다. 이는 연결된 CloudFront 함수가 전 세계의 모든 CloudFront 엣지 로케이션에서 작동함을 나타냅니다. 이 작업은 일반적으로 몇 분 정도 걸립니다.

## Amazon CloudFront KeyValueStore

CloudFront KeyValueStore는 CloudFront 함수 내에서 읽기 액세스를 허용하는 안전한 글로벌 키 값 데이터 스토어로, CloudFront 엣지 로케이션에서 고급 사용자 지정 로직을 사용할 수 있습니다.

CloudFront KeyValueStore를 사용하면 함수 코드를 업데이트하고 함수와 관련된 데이터를 서로 독립적으로 업데이트할 수 있습니다. 이렇게 분리하면 함수 코드가 간소화되고 코드 변경 사항을 배포할 필요 없이 데이터를 쉽게 업데이트할 수 있습니다.

> **Note**
>
> CloudFront KeyValueStore를 사용하려면 CloudFront 함수가 JavaScript 런타임 2.0을 사용해야 합니다.

키 값 페어를 사용하는 일반적인 절차는 다음과 같습니다.

- 키 값 저장소를 만들고 키 값 페어 모음으로 채웁니다. Amazon S3 버킷에 키 값 스토어를 추가하거나 수동으로 입력할 수 있습니다.
- 키 값 스토어를 CloudFront 함수와 연결합니다.
- 함수 코드 내에서 키 이름을 사용하여 키와 관련된 값을 검색하거나 키가 존재하는지 알아볼 수 있습니다. 함수 코드에서 키 값 페어를 사용하는 방법과 도우미 메서드에 대한 자세한 내용은 the section called “키 값 저장소를 위한 도우미 메서드” 섹션을 참조하세요.

### 사용 사례

다음 예제에 키 값 페어를 사용할 수 있습니다.

- URL 재작성 또는 리디렉션 - 키 값 페어에는 재작성된 URL 또는 리디렉션 URL이 포함될 수 있습니다.
- A/B 테스트 및 기능 플래그 - 웹사이트의 특정 버전에 트래픽 비율을 할당하여 실험을 실행하는 함수를 만들 수 있습니다.
- 액세스 권한 부여 - 사용자가 정의한 기준과 KeyValueStore에 저장된 데이터에 따라 요청을 허용하거나 거부하는 액세스 제어를 구현할 수 있습니다.

## 지원되는 값 형식

키 값 페어의 값은 다음 형식 중 하나로 저장할 수 있습니다.

- String
- 바이트로 인코딩된 문자열
- JSON

## 보안

CloudFront Functions 및 모든 키 값 저장소 데이터는 다음과 같이 안전하게 처리됩니다.

- CloudFront는 저장 중이거나 전송 중(키 값 스토어를 읽거나 쓸 때)인 각 키 값 스토어를 CloudFront KeyValueStore API 작업을 직접 호출할 때 암호화합니다.
- 함수를 실행할 때 CloudFront는 CloudFront 엣지 로케이션의 메모리에 있는 각 키 값 페어를 복호화합니다.

CloudFront KeyValueStore를 시작하려면 다음 주제를 참조하세요.

주제

- 키 값 저장소 사용
- 키 값 데이터로 작업

- CloudFront 키 값 저장소 시작 방법에 대한 자세한 내용은 Introducing Amazon CloudFront KeyValueStore AWS 블로그 게시물을 참조하세요.

# 키 값 저장소 사용

CloudFront Functions에서 사용하려는 키 값 페어를 보관할 키 값 저장소를 생성해야 합니다.

키 값 저장소를 생성하고 키-값 페어를 추가한 후 CloudFront 함수 코드에서 키 값을 사용할 수 있습니다.

시작하려면 다음 주제를 참조하세요.

주제


- 키 값 저장소 생성
- 키 값 저장소를 함수와 연결
- 키 값 저장소 업데이트
- 키 값 저장소에 대한 참조 가져오기
- 키 값 저장소 삭제
- 키 값 페어의 파일 형식


> **Note**
>
> JavaScript 런타임 2.0에는 함수 코드의 키 값으로 작업하기 위한 몇 가지 도우미 메서드가 포함되어 있습니다. 자세한 내용은 the section called “키 값 저장소를 위한 도우미 메서드” 단원을 참조하십시오.

## 키 값 저장소 생성

키 값 저장소와 키 값 페어를 동시에 생성할 수 있습니다. 또한 비어 있는 키 값 저장소를 만든 다음 나중에 키 값 페어를 추가할 수 있습니다.

> **Note**
>
> Amazon S3 버킷에서 데이터 소스를 지정하는 경우 해당 버킷에 대한 `s3:GetObject` 및 `s3:GetBucketLocation` 권한이 있어야 합니다. 이러한 권한이 없는 경우 CloudFront는 키 값 저장소를 성공적으로 생성할 수 없습니다.

키 값 저장소를 생성하면서 키 값 페어를 동시에 추가할지 여부를 결정하세요. CloudFront 콘솔, CloudFront API 또는 AWS SDK를 사용하여 키 값 페어를 가져올 수 있습니다. 그러나 키 값 저장소를 처음 만들 때만 키 값 페어 파일을 가져올 수 있습니다.

키 값 페어로 구성된 파일을 생성하려면 키 값 페어의 파일 형식 섹션을 참조하세요.

Console

키 값 저장소를 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudfront/v4/home#/functions에 있는 CloudFront 콘솔에서 함수 페이지를 엽니다.
2. KeyValueStores 탭을 선택한 다음 KeyValueStore 생성을 선택합니다.
3. 키 값 저장소 이름을 입력하고 필요한 경우 설명을 입력합니다.
4. S3 URI 작성:
   - 키 값 페어 파일이 있는 경우 파일을 저장한 Amazon S3 버킷의 경로를 입력합니다.
   - 키 값 페어를 수동으로 입력하려는 경우 이 필드를 비워 두세요.
5. 생성(Create)을 선택합니다. 키 값 저장소가 생겼습니다.

   새로운 키 값 저장소의 세부 정보 페이지가 나타납니다. 페이지의 정보에는 키 값 저장소의 ID 및 ARN이 포함됩니다.

   - ID는 AWS 계정에서 고유한 임의의 문자열입니다.
   - ARN의 구문은 다음과 같습니다.

     ```
     AWS ##:key-value-store/# # ### ID
     ```
6. 키 값 페어 섹션을 살펴보세요. 파일을 가져온 경우 이 섹션에 일부 키 값 페어가 표시됩니다. 다음을 수행할 수 있습니다.
   - 파일을 가져온 경우 직접 값을 더 추가할 수도 있습니다.
   - Amazon S3 버킷에서 파일을 가져오지 않았고 지금 키 값 페어를 추가하려는 경우 다음 단계를 완료하면 됩니다.
   - 이 단계를 건너뛰고 나중에 키 값 페어를 추가할 수 있습니다.
7. 지금 페어를 추가하려면 다음 단계를 따르세요.

   a. 키 값 페어 추가를 선택합니다.

b. 페어 추가를 선택하고 이름과 값을 입력합니다. 태그를 더 추가하려면 이 단계를 반복합니다.

c. 작업을 마쳤으면 변경 사항 저장을 선택하여 모든 키 값 페어를 키 값 저장소에 저장합니다. 나타나는 대화 상자에서 완료를 선택합니다.

8. 지금 키 값 저장소를 함수와 연결하려면 연결된 함수 섹션을 완료하세요. 자세한 내용은 ??? 또는 ???을 참조하세요.

   나중에 이 키 값 저장소 세부 정보 페이지 또는 함수 세부 정보 페이지에서 함수를 연결할 수도 있습니다.

## AWS CLI

키 값 저장소를 생성하려면

- 다음 명령을 실행하여 키 값 저장소를 생성하고 Amazon S3 버킷에서 키 값 페어를 가져옵니다.

```
aws cloudfront create-key-value-store \
    --name=keyvaluestore1 \
    --comment="This is my key value store file" \
    --import-source=SourceType=S3,SourceARN=arn:aws:s3:::amzn-s3-demo-
bucket1/kvs-input.json
```

응답

```
{
    "ETag": "ETVABCEXAMPLE",
    "Location": "https://cloudfront.amazonaws.com/2020-05-31/key-value-store/
arn:aws:cloudfront::123456789012:key-value-store/8aa76c93-3198-462c-aaf6-
example",
    "KeyValueStore": {
        "Name": "keyvaluestore1",
        "Id": "8aa76c93-3198-462c-aaf6-example",
        "Comment": "This is my key value store file",
        "ARN": "arn:aws:cloudfront::123456789012:key-value-
store/8aa76c93-3198-462c-aaf6-example",
        "Status": "PROVISIONING",
        "LastModifiedTime": "2024-08-06T22:19:10.813000+00:00"
    }
```

```
}
```

API

키 값 저장소를 생성하려면

1. CloudFront CreateKeyValueStore 작업을 사용합니다. 이 작업에는 다음과 같은 몇 가지 파라미터가 사용됩니다.
   - 키 값 저장소의 `name`
   - 설명이 포함된 `comment` 파라미터
   - Amazon S3 버킷에 저장된 파일에서 키 값 페어를 가져올 수 있는 `import-source` 파라미터. 키 값 저장소를 처음 만들 때만 파일에서 키 값 페어를 가져올 수 있습니다. 파일 구조에 대한 자세한 내용은 the section called "키 값 페어의 파일 형식" 섹션을 참조하세요.

작업 응답에는 다음 정보가 포함됩니다.

- 지정한 이름을 포함하여 요청에 전달된 값
- 생성 시간 등의 데이터
- `ETag`(예: `ETVABCEXAMPLE`), 키 값 저장소의 이름을 포함하는 ARN(예: `arn:aws:cloudfront::123456789012:key-value-store/keyvaluestore1`)

  `ETag`, ARN 및 이름을 조합하여 키 값 저장소를 프로그래밍 방식으로 사용할 수 있습니다.

키 값 저장소 상태

키 값 저장소를 생성할 때 데이터 저장소는 다음과 같은 상태 값을 가질 수 있습니다.

| 값 | 설명 |
| --- | --- |
| 프로비저닝 | 키 값 저장소가 생성되었으며 CloudFront는 지정된 데이터 소스를 처리하고 있습니다. |
| 준비됨 | 키 값 저장소가 생성되고 CloudFront가 지정한 데이터 소스를 성공적으로 처리했습니다. |

| 값 | 설명 |
| --- | --- |
| 가져오기에 실패했습니다. | CloudFront는 지정한 데이터 소스를 처리하지 못했습니다. 파일 형식이 유효하지 않거나 크기 제한을 초과하는 경우 이 상태가 나타날 수 있습니다. 자세한 내용은 키 값 페어의 파일 형식 단원을 참조하십시오. |

키 값 저장소를 함수와 연결

키 값 저장소를 생성한 후 함수를 업데이트하여 키 값 저장소와 연결할 수 있습니다. 해당 함수에서 해당 저장소의 키 값 페어를 사용하려면 이 연결을 만들어야 합니다. 다음 규칙이 적용됩니다.

- 함수 하나에는 키 값 저장소가 하나만 있을 수 있습니다.
- 동일한 키 값 저장소를 여러 함수와 연결할 수 있습니다.

Console

키 값 저장소를 함수와 연결하려면

1. CloudFront 콘솔(https://console.aws.amazon.com/cloudfront/v4/home#/functions)에 로그인하고 함수 페이지를 엽니다.
2. 함수 이름을 선택합니다.
3. KeyValueStore 연결 섹션에서 기존 KeyValueStore 연결을 선택합니다.
4. 함수의 키 값 페어가 들어 있는 키 값 저장소를 선택한 다음 KeyValueStore 연결을 선택합니다.

   CloudFront는 저장소를 함수와 즉시 연결하므로 함수를 저장하지 않아도 됩니다.
5. 다른 키 값 저장소를 지정하려면 연결된 KeyValueStore 업데이트를 선택하고 다른 키 값 저장소 이름을 선택한 다음 KeyValueStore 연결을 선택합니다.

자세한 내용은 the section called “함수 업데이트” 단원을 참조하십시오.

AWS CLI

키 값 저장소를 함수와 연결하려면

- 다음 명령을 실행하여 *MaxAge* 함수를 업데이트하고 키 값 저장소 리소스를 연결합니다.

```
aws cloudfront update-function \
    --name MaxAge \
    --function-config '{"Comment":"Max Age 2 years","Runtime":"cloudfront-
js-2.0","KeyValueStoreAssociations":{"Quantity":1,"Items":
[{"KeyValueStoreARN":"arn:aws:cloudfront::123456789012:key-value-
store/8aa76c93-3198-462c-aaf6-example"}]}}' \
    --function-code fileb://function-max-age-v1.js \
    --if-match ETVABCEXAMPLE
```

- 키 값 저장소를 함수와 연결하려면 `KeyValueStoreAssociations` 파라미터 및 키 값 저장소 ARN을 지정합니다.
- 연결을 변경하려면 다른 키 값 저장소의 이름을 지정합니다.
- 연결을 제거하려면 `KeyValueStoreAssociations` 파라미터를 제거합니다.

자세한 내용은 the section called "함수 업데이트" 단원을 참조하십시오.

API

키 값 저장소를 함수와 연결하려면

- UpdateFunction API 작업을 사용합니다. 자세한 내용은 the section called "함수 업데이트" 단원을 참조하십시오.

> **Notes**
>
> - 키 값 페어를 변경하지 않고 키 값 저장소를 수정하거나 키 값 저장소에서 키 값 페어만 수정하는 경우 키 값 저장소를 다시 연결할 필요가 없습니다. 또한 함수를 다시 게시할 필요도 없습니다.
>
>  그러나 함수를 테스트하여 예상대로 작동하는지 확인하는 것이 좋습니다. 자세한 내용은 함수 테스트 단원을 참조하십시오.
>
> - 특정 키 값 저장소를 사용하는 모든 함수를 확인할 수 있습니다. CloudFront 콘솔에서 키 값 저장소 세부 정보 페이지를 선택합니다.

## 키 값 저장소 업데이트

키 값 저장소를 업데이트하는 경우 키 값 페어를 변경하거나 키 값 저장소와 함수 간의 연결을 변경할 수 있습니다.

### Console

키 값 저장소를 업데이트하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudfront/v4/home#/functions에 있는 CloudFront 콘솔에서 함수 페이지를 엽니다.
2. KeyValueStore 탭을 선택합니다.
3. 업데이트할 키 값 저장소를 선택합니다.
   - 키 값 페어를 업데이트하려면 키 값 페어 섹션에서 편집을 선택합니다. 키 값 페어를 추가하거나 삭제할 수 있습니다. 기존 키 값 페어의 값을 변경할 수도 있습니다. 작업을 마쳤으면 변경 내용 저장을 선택합니다.
   - 이 키 값 저장소에 대한 연결을 업데이트하려면 함수로 이동을 선택합니다. 자세한 내용은 the section called “키 값 저장소를 함수와 연결” 단원을 참조하십시오.

### AWS CLI

키 값 저장소를 업데이트하려면

1. 키 값 페어 변경 - 키 값 페어를 더 추가하고, 하나 이상의 키 값 페어를 삭제하고, 기존 키 값 페어의 값을 변경할 수 있습니다. 자세한 내용은 키 값 데이터로 작업 단원을 참조하십시오.
2. 키 값 저장소의 함수 연결 변경 - 키 값 저장소의 함수 연결을 업데이트하려면 키 값 저장소를 함수와 연결 섹션을 참조하세요.

> **Tip**
>
> 키 값 저장소의 ARN이 필요합니다. 자세한 내용은 the section called “키 값 저장소에 대한 참조 가져오기” 단원을 참조하십시오.

API

키 값 저장소를 업데이트하려면

1. 키 값 페어 변경 - 키 값 페어를 더 추가하고, 하나 이상의 키 값 페어를 삭제하고, 기존 키 값 페어의 값을 변경할 수 있습니다. 자세한 내용은 키 값 데이터로 작업 단원을 참조하십시오.
2. 키 값 저장소의 함수 연결 변경 - 키 값 저장소의 함수 연결을 업데이트하려면 UpdateFunction API 작업을 사용하세요. 자세한 내용은 the section called "함수 업데이트" 단원을 참조하십시오.

> **Tip**
>
> 키 값 저장소의 ARN이 필요합니다. 자세한 내용은 the section called "키 값 저장소에 대한 참조 가져오기" 단원을 참조하십시오.

키 값 저장소에 대한 참조 가져오기

프로그래밍 방식으로 키 값 저장소를 사용하려면 ETag와 키 값 저장소의 이름이 필요합니다.

두 값을 모두 가져오려면 AWS Command Line Interface(AWS CLI) 또는 CloudFront API를 사용할 수 있습니다.

AWS CLI

키 값 저장소 참조를 가져오려면

1. 키 값 저장소 목록을 반환하려면 다음 명령을 실행합니다. 변경할 키 값 저장소의 이름을 찾습니다.

```
aws cloudfront list-key-value-stores
```

2. 응답에서 원하는 키 값 저장소의 이름을 찾습니다.

   응답

```
{
    "KeyValueStoreList": {
        "Items": [
            {
                "Name": "keyvaluestore3",
```

```
                "Id": "37435e19-c205-4271-9e5c-example3",
                "ARN": "arn:aws:cloudfront::123456789012:key-value-
store/37435e19-c205-4271-9e5c-example3",
                "Status": "READY",
                "LastModifiedTime": "2024-05-08T14:50:18.876000+00:00"
            },
            {
                "Name": "keyvaluestore2",
                "Id": "47970d59-6408-474d-b850-example2",
                "ARN": "arn:aws:cloudfront::123456789012:key-value-
store/47970d59-6408-474d-b850-example2",
                "Status": "READY",
                "LastModifiedTime": "2024-05-30T21:06:22.113000+00:00"
            },
            {
                "Name": "keyvaluestore1",
                "Id": "8aa76c93-3198-462c-aaf6-example",
                "ARN": "arn:aws:cloudfront::123456789012:key-value-
store/8aa76c93-3198-462c-aaf6-example",
                "Status": "READY",
                "LastModifiedTime": "2024-08-06T22:19:30.510000+00:00"
            }
        ]
    }
}
```

3. 다음 명령을 실행하여 지정된 키 값 저장소의 ETag를 반환합니다.

```
aws cloudfront describe-key-value-store \
    --name=keyvaluestore1
```

응답

```
{
    "ETag": "E3UN6WX5RRO2AG",
    "KeyValueStore": {
        "Name": "keyvaluestore1",
        "Id": "8aa76c93-3198-462c-aaf6-example",
        "Comment": "This is an example KVS",
        "ARN": "arn:aws:cloudfront::123456789012:key-value-
store/8aa76c93-3198-462c-aaf6-example",
        "Status": "READY",
        "LastModifiedTime": "2024-08-06T22:19:30.510000+00:00"
```

```
    }
}
```

API

키 값 저장소 참조를 가져오려면

1. CloudFront ListKeyValueStores API 작업을 사용하여 키 값 저장소 목록을 반환합니다. 변경할 키 값 저장소의 이름을 찾습니다.
2. CloudFront DescribeKeyValueStore API 작업을 사용하여 이전 단계에서 반환한 키 값 저장소의 이름을 지정합니다.

응답에는 UUID, 키 값 저장소의 ARN, 키 값 저장소의 `ETag`가 포함됩니다.

- `ETag`(예: `E3UN6WX5RRO2AG`)
- UUID는 128비트임(예: `8aa76c93-3198-462c-aaf6-example`)
- 다음 예와 같이 ARN에는 AWS 계정 번호, 상수 `key-value-store`, UUID가 포함됩니다.

  `arn:aws:cloudfront::123456789012:key-value-store/8aa76c93-3198-462c-aaf6-example`

`DescribeKeyValueStore` 작업에 대한 자세한 내용은 the section called “CloudFront KeyValueStore 소개” 섹션을 참조하세요.

키 값 저장소 삭제

Amazon CloudFront 콘솔 또는 API를 사용하여 키 값 저장소를 삭제할 수 있습니다.

Console

키 값 저장소를 삭제하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudfront/v4/home#/functions에 있는 CloudFront 콘솔에서 함수 페이지를 엽니다.
2. 함수 이름을 선택합니다.
3. 연결된 KeyValueStore 섹션에서 키 값 저장소가 함수와 연결되어 있는지 확인합니다. 연결되어 있는 경우 KeyValueStore 연결 해제를 선택한 다음 연결 제거를 선택하여 연결을 제거합니다.

4. 탐색 창에서 함수 페이지를 선택한 다음 KeyValueStores 탭을 선택합니다.
5. 삭제하려는 키 값 저장소를 선택한 다음 삭제를 선택합니다.

### AWS CLI

키 값 저장소를 삭제하려면

1. ETag와 키 값 저장소의 이름을 가져옵니다. 자세한 내용은 the section called "키 값 저장소에 대한 참조 가져오기" 단원을 참조하십시오.
2. 키 값 저장소가 함수와 연결되어 있는지 확인합니다. 연결되어 있는 경우 연결이 삭제됩니다. 이 두 단계에 대한 자세한 내용은 ??? 섹션을 참조하세요.
3. 키 값 저장소의 이름과 ETag를 가져왔고 키 값 저장소가 함수와 더 이상 연결되지 않은 경우 해당 저장소를 삭제할 수 있습니다.

   다음 명령을 실행하여 지정된 키 값 저장소를 삭제합니다.

```
aws cloudfront delete-key-value-store \
    --name=keyvaluestore1 \
    --if-match=E3UN6WX5RRO2AG
```

### API

키 값 저장소를 삭제하려면

1. ETag와 키 값 저장소의 이름을 가져옵니다. 자세한 내용은 the section called "키 값 저장소에 대한 참조 가져오기" 단원을 참조하십시오.
2. 키 값 저장소가 함수와 연결되어 있는지 확인합니다. 연결되어 있는 경우 연결이 삭제됩니다. 이 두 단계에 대한 자세한 내용은 ??? 섹션을 참조하세요.
3. 키 값 저장소를 삭제하려면 CloudFront DeleteKeyValueStore API 작업을 사용하세요.

## 키 값 페어의 파일 형식

UTF-8 인코딩 파일을 만들 때는 다음 JSON 형식을 사용합니다.

```
{
  "data":[
    {
```

```
      "key":"key1",
      "value":"value"
    },
    {
      "key":"key2",
      "value":"value"
    }
  ]
}
```

파일에 중복 키를 포함할 수 없습니다. Amazon S3 버킷에 잘못된 파일을 지정한 경우 파일을 업데이트하여 중복 파일을 제거한 다음 키 값 저장소를 다시 생성해 볼 수 있습니다.

자세한 내용은 키 값 저장소 생성 단원을 참조하십시오.

> **Note**
>
> 데이터 소스 및 해당 키-값 페어의 파일에는 다음과 같은 한도가 있습니다.
>
> - 파일 크기 - 5MB
> - 키 크기 - 512자
> - 키 크기 - 1,024자

## 키 값 데이터로 작업

이 주제에서는 기존 키 값 저장소에 키 값 페어를 추가하는 방법을 설명합니다. 키 값 저장소를 처음 생성할 때 키 값 페어를 포함하려면 the section called “키 값 저장소 생성” 섹션을 참조하세요.

주제


- 키 값 페어 작업(콘솔)
- CloudFront KeyValueStore 소개
- 키 값 페어 작업(AWS CLI)
- 키 값 페어 작업(API)


### 키 값 페어 작업(콘솔)

CloudFront 콘솔을 사용하여 키 값 페어로 작업할 수 있습니다.

키-값 페어로 작업하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudfront/v4/home#/functions에 있는 CloudFront 콘솔에서 함수 페이지를 엽니다.
2. KeyValueStore 탭을 선택합니다.
3. 변경할 키 값 저장소를 선택합니다.
4. 키 값 페어 섹션에서 편집 버튼을 선택합니다.
5. 키 값 페어를 추가하거나, 키 값 페어를 삭제하거나, 기존 키 값 페어의 값을 변경할 수 있습니다.
6. 작업을 마쳤으면 변경 내용 저장을 선택합니다.

## CloudFront KeyValueStore 소개

> **Tip**
>
> CloudFront KeyValueStore API는 인증에 서명 버전 4A(SigV4A)를 사용하는 글로벌 서비스입니다. SigV4A에서 임시 자격 증명을 사용하려면 버전 2 세션 토큰이 필요합니다. 자세한 내용은 CloudFront KeyValueStore API에 임시 자격 증명 사용 단원을 참조하십시오.

AWS Command Line Interface(AWS CLI) 또는 자체 코드를 사용하여 CloudFront KeyValueStore API를 직접 호출하는 경우 다음 섹션을 참조하세요.

키 값 저장소와 해당 키 값 페어를 사용할 때 직접적으로 호출하는 서비스는 사용 사례에 따라 달라집니다.

- 기존 키 값 저장소에서 키 값 페어를 사용하려면 CloudFront KeyValueStore 서비스를 사용합니다.
- 키 값 저장소를 처음 생성할 때 키 값 저장소에 키 값 페어를 포함하려면 CloudFront 서비스를 사용합니다.

CloudFront API와 CloudFront KeyValueStore API에는 모두 `DescribeKeyValueStore` 작업이 있습니다. 직접 호출하는 이유는 여러 가지 입니다. 차이점을 이해하려면 다음 테이블을 참조하세요.

| | CloudFront DescribeKeyValueStore API | CloudFront KeyValueStore DescribeKeyValueStore API |
|---|---|---|
| 키 값 저장소 관련 데이터 | 키 값 저장소 자체가 마지막으로 수정된 상태 및 날짜와 같은 데이터를 반환합니다. | 스토리지 리소스의 콘텐츠(저장소의 키 값 페어, 콘텐츠 크기)에 대한 데이터를 제공합니다. |
| 키 값 저장소를 식별하는 데이터 | 키 값 저장소의 ETag, UUID, ARN을 반환합니다. | 키 값 저장소의 ETag, ARN을 반환합니다. |

Notes

- 각 DescribeKeyValueStore 작업은 서로 다른 ETag를 반환합니다. ETags는 서로 바꿔서 사용할 수 없습니다.
- API 작업을 직접적으로 호출하여 작업을 완료할 때는 적절한 API에서 ETag를 지정해야 합니다. 예를 들어 CloudFront KeyValueStore DeleteKey 작업에서는 CloudFront KeyValueStore DescribeKeyValueStore 작업에서 반환한 ETag를 지정합니다.
- CloudFront KeyValueStore를 사용하여 CloudFront Functions을 호출하는 경우 함수 호출 중에 키 값 저장소의 값이 업데이트되거나 변경되지 않습니다. 업데이트는 함수 호출 사이에 처리됩니다.

## 키 값 페어 작업(AWS CLI)

CloudFront KeyValueStore에 대해 다음 AWS Command Line Interface 명령을 실행할 수 있습니다.

목차


- 키 값 페어 나열
- 키 값 페어 가져오기
- 키 값 저장소 설명
- 키 값 페어 생성
- 키 값 페어 삭제

- 키 값 페어 업데이트

### 키 값 페어 나열

키 값 저장소의 키 값 페어를 나열하려면 다음 명령을 실행합니다.

```
aws cloudfront-keyvaluestore list-keys \
    --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-
example
```

### 응답

```
{
    "Items": [
        {
            "Key": "key1",
            "Value": "value1"
        }
    ]
}
```

### 키 값 페어 가져오기

키 값 저장소의 키 값 페어를 가져오려면 다음 명령을 실행합니다.

```
aws cloudfront-keyvaluestore get-key \
    --key=key1 \
    --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-
example
```

### 응답

```
{
    "Key": "key1",
    "Value": "value1",
    "ItemCount": 1,
    "TotalSizeInBytes": 11
}
```

### 키 값 저장소 설명

키 값 저장소를 설명하려면 다음 명령을 실행합니다.

```
aws cloudfront-keyvaluestore describe-key-value-store \
    --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-
example
```

응답

```
{
    "ETag": "KV1F83G8C2ARO7P",
    "ItemCount": 1,
    "TotalSizeInBytes": 11,
    "KvsARN": "arn:aws:cloudfront::123456789012:key-value-store/37435e19-
c205-4271-9e5c-example",
    "Created": "2024-05-08T07:48:45.381000-07:00",
    "LastModified": "2024-08-05T13:50:58.843000-07:00",
    "Status": "READY"
}
```

### 키 값 페어 생성

키 값 저장소에 키 값 페어를 생성하려면 다음 명령을 실행합니다.

```
aws cloudfront-keyvaluestore put-key \
    --if-match=KV1PA6795UKMFR9 \
    --key=key2 \
    --value=value2 \
    --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-
example
```

응답

```
{
    "ETag": "KV13V1IB3VIYZZH",
    "ItemCount": 3,
    "TotalSizeInBytes": 31
}
```

### 키 값 페어 삭제

키 값 페어를 삭제하려면 다음 명령을 실행합니다.

```
aws cloudfront-keyvaluestore delete-key \
    --if-match=KV13V1IB3VIYZZH \
    --key=key1 \
    --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-
example
```

출력

```
{
    "ETag": "KV1VC38T7YXB528",
    "ItemCount": 2,
    "TotalSizeInBytes": 22
}
```

### 키 값 페어 업데이트

`update-keys` 명령을 사용하여 둘 이상의 키 값 페어를 업데이트할 수 있습니다. 예를 들어 기존 키 값 페어를 삭제하고 다른 키 값 페어를 만들려면 다음 명령을 실행합니다.

```
aws cloudfront-keyvaluestore update-keys \
    --if-match=KV2EUQ1WTGCTBG2 \
    --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-
example \
    --deletes '[{"Key":"key2"}]' \
    --puts '[{"Key":"key3","Value":"value3"}]'
```

응답

```
{
    "ETag": "KV3AEGXETSR30VB",
    "ItemCount": 3,
    "TotalSizeInBytes": 28
}
```

### 키 값 페어 작업(API)

이 섹션을 따라 프로그래밍 방식으로 키 값 페어 작업

목차


- 키 값 저장소에 대한 참조 가져오기
- 키 값 저장소의 키 값 페어 변경
- CloudFront 키 값 저장소 예제 코드


## 키 값 저장소에 대한 참조 가져오기

CloudFront KeyValueStore API를 사용하여 쓰기 작업을 직접적으로 호출할 때는 키 값 저장소의 ARN과 `ETag`를 지정해야 합니다. 이 데이터를 가져오려면 다음 작업을 수행합니다.

키 값 저장소에 대한 참조를 가져오려면

1. CloudFront ListKeyValueStores API 작업을 사용하여 키 값 저장소 목록을 가져옵니다. 변경할 키 값 저장소를 찾습니다.
2. CloudFrontKeyValueStore DescribeKeyValueStore API 작업을 사용하여 이전 단계의 키 값 저장소를 지정합니다.

   응답에는 키 값 저장소의 ARN과 `ETag`가 포함됩니다.

   - 다음 예와 같이 ARN에는 AWS 계정 번호, 상수 `key-value-store`, UUID가 포함됩니다.

     ```
     arn:aws:cloudfront::123456789012:key-value-store/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
     ```

   - `ETag`는 다음 예와 같습니다.

     ```
     ETVABCEXAMPLE2
     ```

## 키 값 저장소의 키 값 페어 변경

업데이트하려는 키 값 페어가 들어 있는 키 값 저장소를 지정할 수 있습니다.

다음 CloudFront KeyValueStore API 작업을 참조하세요.

- CloudFrontKeyValueStore DeleteKey – 키 값 페어를 삭제합니다.
- CloudFrontKeyValueStore GetKey – 키 값 페어를 반환합니다.
- CloudFrontKeyValueStore ListKeys – 키 값 페어 목록을 반환합니다.
- CloudFrontKeyValueStore PutKey – 다음과 같은 작업을 수행할 수 있습니다.

  - 새 키 이름과 값을 지정하여 하나의 키 값 저장소에 키 값 페어를 생성합니다.
  - 기존 키 이름과 새 키 값을 지정하여 기존 키 값 페어에 다른 값을 설정합니다.
- CloudFrontKeyValueStore UpdateKeys - 전부 또는 전무 작업 하나로 다음 작업 중 하나 이상을 수행할 수 있습니다.
  - 하나 이상의 키 값 페어 삭제
  - 새 키 값 페어를 하나 이상 생성
  - 하나 이상의 기존 키 값 페어에 다른 값 설정

## CloudFront 키 값 저장소 예제 코드

### Example

다음 코드는 키 값 저장소에 대한 `DescribeKeyValueStore` API 작업을 직접적으로 호출하는 방법을 보여줍니다.

```
const {
  CloudFrontKeyValueStoreClient,
  DescribeKeyValueStoreCommand,
} = require("@aws-sdk/client-cloudfront-keyvaluestore");

require("@aws-sdk/signature-v4-crt");

(async () => {
  try {
    const client = new CloudFrontKeyValueStoreClient({
      region: "us-east-1"
    });
    const input = {
      KvsARN: "arn:aws:cloudfront::123456789012:key-value-store/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
    };
    const command = new DescribeKeyValueStoreCommand(input);

    const response = await client.send(command);
  } catch (e) {
    console.log(e);
  }
})();
```

# CloudFront 연결 함수를 사용하여 사용자 지정

CloudFront 연결 함수를 사용하면 mTLS 인증서 검증 및 사용자 지정 인증 로직을 위한 경량 JavaScript 함수를 작성할 수 있습니다. 연결 함수는 mTLS 연결 설정 중에 실행되어 클라이언트 인증서를 검증하고, 디바이스별 인증 규칙을 구현하고, 인증서 해지 시나리오를 처리합니다. 연결 함수 런타임 환경은 밀리초 미만의 시작 시간을 제공하고 초당 수백만 건의 연결을 처리할 수 있도록 즉시 확장되며 매우 안전합니다. 연결 함수는 CloudFront의 기본 기능입니다. 즉, CloudFront 내에서 완전히 코드를 빌드, 테스트 및 배포할 수 있습니다.

연결 함수를 mTLS 지원 CloudFront 배포와 연결하면 CloudFront는 CloudFront 엣지 로케이션에서 TLS 연결 요청을 가로채고 인증서 정보를 함수에 전달합니다. 다음 이벤트가 발생할 때 연결 함수를 간접 호출할 수 있습니다.

- TLS 연결 설정(연결 요청) 중 - 상호 TLS(mTLS) 연결용

연결 함수에 대한 자세한 내용은 다음 주제를 참조하세요.

주제


- 개요 및 워크플로
- 구성 및 제한 사항
- 상호 TLS(뷰어) 검증을 위한 CloudFront 연결 함수 생성
- 상호 TLS(뷰어) 검증을 위한 CloudFront 연결 함수 코드 쓰기
- 배포 전에 CloudFront 연결 함수 테스트
- 배포에 연결 함수 연결
- CloudFront Functions 및 KeyValueStore를 사용하여 상호 TLS(뷰어)에 대한 인증서 해지 구현


## 개요 및 워크플로

CloudFront 연결 함수는 클라이언트가 mTLS 연결을 설정하려고 할 때 TLS 핸드셰이크 중에 실행되는 특수한 유형의 CloudFront Functions입니다. 연결 함수는 클라이언트 인증서 정보, mTLS 구성 파라미터, 인증서 해지 검사 결과 및 클라이언트 IP 주소에 액세스할 수 있습니다.

CloudFront가 표준 인증서 검증(신뢰 체인, 만료, 서명 확인)을 수행한 후 연결 함수가 간접적으로 호출되지만 인증서 해지 확인에 실패하더라도 실행될 수 있습니다. 이렇게 하면 해지된 인증서를 처리하거나 추가 검증 기준을 추가하기 위한 사용자 지정 로직을 구현할 수 있습니다.

연결 함수를 생성하고 게시한 후에는 mTLS 지원 배포에 연결 요청 이벤트 유형에 대한 연결을 추가해야 합니다. 이렇게 하면 클라이언트가 CloudFront와 mTLS 연결을 설정하려고 할 때마다 함수가 실행됩니다.

CloudFront 연결 함수는 프로덕션에 배포하기 전에 함수를 개발하고 테스트할 수 있는 2단계 수명 주기를 따릅니다. 이 워크플로는 연결 함수가 라이브 트래픽에 영향을 미치기 전에 올바르게 작동하도록 합니다.

주제


- 함수 단계
- 개발 워크플로
- 다른 함수 유형과의 차이점


## 함수 단계

연결 함수는 다음 두 단계 중 하나로 존재합니다.

- 개발 - 이 단계의 함수를 수정, 테스트 및 업데이트할 수 있습니다. 함수 코드를 작성하고 디버깅하려면 이 단계를 사용합니다.
- 라이브 - 이 단계의 함수는 읽기 전용이며 프로덕션 트래픽을 처리합니다. 라이브 단계에서는 함수를 직접 수정할 수 없습니다.

새 연결 함수를 생성하면 개발 단계에서 시작됩니다. 테스트 및 검증 후 함수를 게시하여 라이브 단계로 이동합니다.

## 개발 워크플로

다음 워크플로에 따라 연결 함수를 개발하고 배포합니다.

1. 생성 - 초기 코드 및 구성을 사용하여 개발 단계에서 새 연결 함수를 생성합니다.
2. 테스트 - 테스트 기능을 사용하여 배포 전에 샘플 연결 이벤트로 함수를 검증합니다.
3. 업데이트 - 테스트 결과를 바탕으로 필요에 따라 함수 코드 및 구성을 수정합니다.
4. 게시 - 프로덕션 준비가 되면 함수를 게시하여 개발 단계에서 라이브 단계로 이동합니다.
5. 연결 - 게시된 함수를 mTLS 지원 배포와 연결하여 라이브 연결을 처리합니다.

라이브 함수를 변경하려면 개발 버전을 업데이트하고 다시 게시해야 합니다. 그러면 라이브 단계에서 새 버전이 생성됩니다.

### 다른 함수 유형과의 차이점

연결 함수는 다음과 같은 몇 가지 중요한 측면에서 뷰어 요청 및 뷰어 응답 함수와 다릅니다.

- 연결 함수는 mTLS 핸드셰이크 이후 HTTP 처리가 발생하기 전에 실행됩니다.
- 연결 함수는 HTTP 요청/응답 데이터 대신 TLS 인증서 정보에 액세스할 수 있습니다.
- 연결 함수는 HTTP 데이터를 수정하지 않고 연결만 허용하거나 거부할 수 있습니다.
- 연결 함수는 연결 재사용이 아닌 새 TLS 연결에 대해서만 간접적으로 호출됩니다.
- TLS 세션 재개는 모든 연결에서 인증서 검증이 이루어지도록 mTLS에서 지원되지 않습니다.
- 연결 함수는 표준 뷰어 요청 및 뷰어 응답 함수 외에도 실행됩니다.
- 연결 함수는 캐시 동작 수준이 아닌 배포 수준에서 연결합니다.
- 연결 함수는 JavaScript 런타임 2.0만 지원합니다.

## 구성 및 제한 사항

CloudFront 연결 함수에는 TLS 연결 검증에서의 특수한 역할과 엣지 컴퓨팅의 성능 요구 사항으로 인해 특정한 구성 요구 사항 및 서비스 제한이 있습니다.

주제

- 함수 코드 요구 사항
- 서비스 한도
- 함수 필터링 옵션

### 함수 코드 요구 사항

연결 함수에는 TLS 연결 이벤트를 처리하는 JavaScript 코드가 필요합니다. 함수 코드는 다음을 충족해야 합니다.

- JavaScript로 작성
- 연결 이벤트 처리 및 허용/거부 결정
- 제한 시간 내에 실행 완료

- 인증서 및 연결 검증 로직 처리

## 서비스 한도

연결 함수에는 다음 제한이 적용됩니다.

- 함수 크기 - 함수 코드 및 구성의 크기가 제한됩니다.
- 실행 시간 - 함수에는 TLS 연결 처리에 대한 엄격한 실행 시간 제한이 있습니다.
- 연결 제한 - 각 배포에는 연결 함수가 하나만 연결될 수 있습니다.
- 단계 제한 - 라이브 단계 함수만 배포와 연결할 수 있습니다.

## 함수 필터링 옵션

연결 함수를 나열할 때 다음 필터를 사용할 수 있습니다.

- 단계 필터 - 개발 또는 라이브 단계를 기준으로 필터링합니다.
- 연결 필터 - 배포 ID 또는 키-값 저장소 ID 연결을 기준으로 필터링합니다.

이러한 필터를 사용하면 다양한 환경 및 사용 사례에서 연결 함수를 구성하고 관리할 수 있습니다.

# 상호 TLS(뷰어) 검증을 위한 CloudFront 연결 함수 생성

CloudFront 연결 함수는 다음 두 단계로 생성합니다.

1. 함수 코드를 JavaScript로 생성합니다. CloudFront 콘솔의 기본 예제를 사용하거나, 직접 작성할 수 있습니다. 자세한 내용은 다음 항목을 참조하세요.
   - mTLS 검증을 위한 CloudFront 연결 함수 코드 작성
   - CloudFront 연결 함수 이벤트 구조 및 응답 형식
   - 연결 함수 코드 예제
2. CloudFront를 사용하여 연결 함수를 생성하고 코드를 포함합니다. 코드는 함수 내에 존재합니다(참조용이 아님).

주제

- CloudFront 콘솔
- AWS CLI

## CloudFront 콘솔

연결 함수를 생성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 함수 생성을 선택합니다.
3. AWS 계정 내에서 고유한 함수 이름을 입력하고 함수 유형으로 연결 함수를 선택한 다음 계속을 선택합니다.
4. 새 연결 함수의 세부 정보 페이지가 나타납니다.

> **Note**
>
> 연결 함수는 JavaScript 런타임 2.0만 지원합니다. 함수에서 CloudFront 연결 함수 KeyValueStore 통합을 사용하려면 이 런타임 버전을 사용해야 합니다.

5. 함수 코드 섹션에서 빌드 탭을 선택하고 연결 함수 코드를 입력합니다. 빌드 탭에 포함된 샘플 코드는 연결 함수 코드의 기본 구문을 보여줍니다.
6. 변경 사항 저장을 선택합니다.
7. 연결 함수 코드가 인증서 해지 확인 또는 디바이스 검증에 KeyValueStore를 사용하는 경우 KeyValueStore를 연결해야 합니다.

   함수를 처음 생성할 때 KeyValueStore를 연결할 수 있습니다. 또는 연결 함수를 연결하여 나중에 연결할 수 있습니다.

   지금 KeyValueStore를 연결하려면 다음 단계를 따르세요.

   - KeyValueStore 연결 섹션에서 기존 KeyValueStore 연결을 선택합니다.
   - 연결 함수의 인증서 데이터가 포함된 KeyValueStore를 선택한 다음 KeyValueStore 연결을 선택합니다.

   CloudFront는 저장소를 함수와 즉시 연결하므로 함수를 저장하지 않아도 됩니다.

## AWS CLI

AWS CLI를 사용하는 경우 일반적으로 먼저 파일에 연결 함수 코드를 생성한 다음 AWS CLI로 함수를 생성합니다.

연결 함수를 생성하려면

1. 연결 함수 코드를 파일로 만든 다음 컴퓨터가 연결할 수 있는 디렉터리에 저장합니다.
2. 다음 예제와 같이 명령을 실행합니다. 이 예제에서는 `fileb://` 표기법을 사용하여 파일을 전달합니다. 명령을 더 쉽게 읽을 수 있도록 줄 바꿈도 포함되어 있습니다.

```
aws cloudfront create-connection-function \
    --name CertificateValidator \
    --connection-function-config '{
        "Comment":"Device certificate validation",
        "Runtime":"cloudfront-js-2.0",
        "KeyValueStoreAssociations":{
            "Quantity":1,
            "Items":[{
                "KeyValueStoreARN":"arn:aws:cloudfront::111122223333:key-value-
store/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            }]
        }
    }' \
    --connection-function-code fileb://certificate-validator.js
```

> **Note**
>
> - 런타임 - 연결 함수는 JavaScript 런타임 2.0(cloudfront-js-2.0)만 지원합니다.
> - KeyValueStoreAssociations - 연결 함수가 인증서 검증에 KeyValueStore를 사용하는 경우, 함수를 처음 생성할 때 KeyValueStore를 연결할 수 있습니다. 또는 update-connection-function을 사용하여 나중에 연결할 수 있습니다. 각 연결 함수에 KeyValueStore가 하나만 연결될 수 있기 때문에 수량은 항상 1입니다.

3. 명령이 제대로 실행되면 다음과 비슷한 출력이 표시됩니다.

```
ETag: ETVABCEXAMPLE
ConnectionFunctionSummary:
  ConnectionFunctionConfig:
    Comment: Device certificate validation
    Runtime: cloudfront-js-2.0
    KeyValueStoreAssociations:
      Quantity: 1
      Items:
```

```
        - KeyValueStoreARN: arn:aws:cloudfront::111122223333:key-value-store/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
  ConnectionFunctionMetadata:
    CreatedTime: '2024-09-04T16:32:54.292000+00:00'
    ConnectionFunctionARN: arn:aws:cloudfront::111122223333:connection-function/
CertificateValidator
    LastModifiedTime: '2024-09-04T16:32:54.292000+00:00'
    Stage: DEVELOPMENT
  Name: CertificateValidator
  Status: UNPUBLISHED
Location: https://cloudfront.amazonaws.com/2020-05-31/connection-function/
arn:aws:cloudfront:::connection-function/CertificateValidator
```

대부분의 정보는 요청에서 반복된 내용입니다. 기타 정보는 CloudFront에 의해 추가됩니다.

> **Note**
>
> - ETag - 이 값은 연결 함수를 수정할 때마다 변경됩니다. 함수를 업데이트하거나 게시하려면 이 값이 필요합니다.
> - 단계 - 새 연결 함수는 개발 단계에서 시작됩니다. 함수를 배포와 연결하기 전에 함수를 게시하여 라이브 단계로 이동해야 합니다.
> - 상태 - 라이브 단계에 게시할 때까지 함수 상태는 '게시되지 않음'입니다.

## 상호 TLS(뷰어) 검증을 위한 CloudFront 연결 함수 코드 쓰기

CloudFront 연결 함수를 사용하면 mTLS 인증서 검증 및 사용자 지정 인증 로직을 위한 경량 JavaScript 함수를 작성할 수 있습니다. 연결 함수 코드는 클라이언트 인증서를 검증하고, 디바이스별 인증 규칙을 구현하고, 인증서 해지 시나리오를 처리하고, 전 세계 CloudFront 엣지 로케이션에서 TLS 연결에 대한 허용/거부 결정을 내릴 수 있습니다.

연결 함수는 자체 비즈니스 로직을 사용하여 CloudFront의 기본 제공 인증서 검증을 확장하는 강력한 방법을 제공합니다. HTTP 데이터를 처리하는 뷰어 요청 및 뷰어 응답 함수와 달리, 연결 함수는 TLS 계층에서 작동하며 인증서 정보, 클라이언트 IP 주소 및 TLS 연결 세부 정보에 액세스할 수 있습니다. 따라서 표준 PKI 검증을 넘어 제로 트러스트 보안 모델, 디바이스 인증 시스템 및 사용자 지정 인증서 검증 정책을 구현하는 데 이상적입니다.

연결 함수 코드는 시작 시간이 밀리초 미만인 안전하고 격리된 환경에서 실행되며 초당 수백만 개의 연결을 처리하도록 확장할 수 있습니다. 런타임은 인증서 검증 워크로드에 최적화되어 있으며 실시간 데

이터 조회 작업을 위해 CloudFront KeyValueStore와 기본적으로 통합되어 인증서 해지 목록 확인 및 디바이스 허용 목록 검증과 같은 정교한 인증 시나리오를 지원합니다.

효과적인 연결 함수 코드를 작성하는 데 도움이 되도록 다음 주제를 참조하세요. 전체 코드 예제와 단계별 자습서는 이 가이드의 자습서 섹션을 참조하고 CloudFront 콘솔에서 사용할 수 있는 연결 함수 예제를 살펴보세요.

주제


- CloudFront 연결 함수 사용 사례 및 용도
- CloudFront 연결 함수 이벤트 구조 및 응답 형식
- CloudFront 연결 함수 JavaScript 런타임 기능
- CloudFront 연결 함수 헬퍼 메서드 및 API
- CloudFront 연결 함수 KeyValueStore 통합
- async 및 await 사용
- 연결 함수 코드 예제


## CloudFront 연결 함수 사용 사례 및 용도

CloudFront 연결 함수를 작성하기 전에 구현해야 하는 인증서 검증 또는 인증 로직 유형을 신중하게 결정합니다. 연결 함수는 표준 PKI 인증서 검사 외에 사용자 지정 검증이 필요한 특정 사용 사례에 맞게 설계되었습니다. 사용 사례를 이해하면 최적의 성능을 유지하면서 보안 요구 사항을 충족하는 효율적인 코드를 설계할 수 있습니다.

일반적인 연결 함수 사용 사례는 다음과 같습니다.

- 인증서 해지 처리 - 인증서 교체 유예 기간, 내부 디바이스에 대한 신뢰할 수 있는 네트워크 예외 또는 해지된 인증서에 임시 액세스가 필요할 수 있는 긴급 액세스 시나리오를 포함하여 해지된 인증서를 처리하기 위한 사용자 지정 정책을 구현합니다.
- 선택적 mTLS 지원 - 서로 다른 인증 정책으로 mTLS 및 비mTLS 연결을 모두 처리하여 레거시 클라이언트와의 호환성을 유지하면서 인증서를 지원하는 클라이언트에 더 나은 보안을 제공할 수 있습니다.
- IP 기반 인증 - 클라이언트 IP 주소 확인과 인증서 검증을 결합하여 특정 지리적 리전, 회사 네트워크 또는 알려진 악성 IP 범위에서 액세스를 제한하는 등 보안을 강화합니다.
- 다중 테넌트 인증서 검증 - 클라이언트 인증서 발급자 또는 주체 속성에 따라 다른 인증 기관 또는 검증 기준이 적용되는 테넌트별 검증 규칙을 구현합니다.

- 시간 기반 액세스 제어 - 인증서 자체가 만료되지 않은 경우에도 특정 시간, 유지 관리 기간 또는 업무 기간 동안에만 인증서가 유효한 시간 기반 제한을 적용합니다.

연결 함수는 CloudFront가 표준 인증서 검증(신뢰 체인 확인, 만료 확인 및 서명 검증)을 수행한 후 TLS 연결이 설정되기 전에 실행됩니다. 이 타이밍을 사용하면 CloudFront의 내장 인증서 검증을 활용하면서 사용자 지정 검증 기준을 유연하게 추가할 수 있습니다. 함수는 표준 검증 결과를 수신하고 정보를 바탕으로 표준 및 사용자 지정 기준에 따라 연결을 허용할지 또는 거부할지를 결정할 수 있습니다.

연결 함수를 설계할 때는 검증 로직의 성능 영향을 고려하세요. 함수의 실행 한도는 5밀리초이므로 복잡한 작업은 속도에 맞게 최적화해야 합니다. 복잡한 계산 대신 빠른 데이터 조회에 KeyValueStore를 사용하고 잘못된 인증서는 빠르게 실패하도록 검증 로직을 구성합니다.

## CloudFront 연결 함수 이벤트 구조 및 응답 형식

CloudFront 연결 함수는 뷰어 요청 및 뷰어 응답 함수와 다른 이벤트 구조를 수신합니다. 연결 함수는 HTTP 요청/응답 데이터 대신 인증 결정에 사용할 수 있는 인증서 및 연결 정보를 수신합니다.

주제

- 연결 함수의 이벤트 구조
- 연결 함수 응답 형식

### 연결 함수의 이벤트 구조

연결 함수는 인증서 및 연결 정보가 포함된 이벤트 객체를 수신합니다. 함수의 이벤트 구조는 다음과 같습니다.

```
{
  "clientCertificate": {
    "certificates": {
      "leaf": {
        "serialNumber": "string",
        "issuer": "string",
        "subject": "string",
        "validity": {
          "notBefore": "string",
          "notAfter": "string",
        },
        "sha256Fingerprint": "string"
      }
```

```
    }
  },
  "clientIp": "string",
  "endpoint": "string",
  "distributionId": "string",
  "connectionId": "string"
}
```

다음은 이벤트 객체 구조의 예입니다.

```
{
  "clientCertificate": {
    "certificates": {
      "leaf": {
        "serialNumber": "00:9e:2a:af:16:56:e5:47:25:7d:2e:38:c3:f9:9d:57:fa",
        "issuer": "C=US, O=Ram, OU=Edge, ST=WA, CN=mTLS-CA, L=Snoqualmie",
        "subject": "C=US, O=Ram, OU=Edge, ST=WA, CN=mTLS-CA, L=Snoqualmie",
        "validity": {
          "notBefore": "2025-09-10T23:43:10Z",
          "notAfter": "2055-09-11T00:43:02Z"
        },
        "sha256Fingerprint": "_w6bJ7aOAlGOj7NUhJxTfsfee-ONg_xop3_PTgTJpqs="
      }
    }
  },
  "clientIp": "127.0.0.1",
  "endpoint": "d3lch071jze0cb.cloudfront.net",
  "distributionId": "E1NXS4MQZH501R",
  "connectionId": "NpvTe1925xfj24a67sPQr7ae42BIq03FGhJJKfrQYWZcWZFp96SIIg=="
}
```

연결 함수 응답 형식

연결 함수는 연결을 허용할지 또는 거부할지를 나타내는 응답 객체를 반환해야 합니다. 다음과 같이 헬퍼 메서드를 사용하여 연결 결정을 내립니다.

```
function connectionHandler(connection) {
    // Helper methods to allow or deny connections
    if (/* some logic to determine if function should allow connection */) {
        connection.allow();
    } else {
        connection.deny();
```

```
    }
}
```

뷰어 요청 및 뷰어 응답 함수와 달리 연결 함수는 HTTP 요청 또는 응답을 수정할 수 없습니다. TLS 연결만 허용하거나 거부할 수 있습니다.

## CloudFront 연결 함수 JavaScript 런타임 기능

CloudFront 연결 함수는 인증서 검증 워크로드에 특별히 최적화된 안전한 고성능 환경을 제공하는 CloudFront Functions JavaScript 런타임 2.0을 사용합니다. 런타임은 수 밀리초 이내에 시작하고 CloudFront의 글로벌 엣지 네트워크에서 수백만 개의 동시 실행을 처리하도록 설계되었습니다.

런타임 환경에는 다음과 같이 포괄적인 JavaScript 언어 지원이 포함됩니다.

- ECMAScript 2020(ES11) 지원 - 선택적 연결(?.), nullish 병합(??) 및 대용량 인증서 일련 번호 처리를 위한 BigInt를 포함한 최신 JavaScript 기능
- 기본 제공 객체 - 객체, 배열, JSON, 수학 및 날짜와 같은 표준 JavaScript 객체
- 콘솔 로깅 - console.log()를 사용하여 인증서 검증 결정을 디버깅하고 모니터링합니다. 테스트 중에 로그를 실시간으로 사용할 수 있으며 개발 중인 검증 로직 문제를 해결하는 데 도움이 될 수 있습니다.
- KeyValueStore 통합 - 초고속 데이터 조회 작업을 위한 CloudFront KeyValueStore에 대한 기본 액세스로 실시간 인증서 해지 확인, 디바이스 허용 목록 검증 및 테넌트별 구성 검색 활성화

연결 함수는 인증서 검증 시나리오를 위한 고성능에 최적화되어 있습니다. 런타임은 메모리 관리, 폐영역 회수 및 리소스 정리를 자동으로 처리하여 수백만 개의 동시 연결에서 일관된 성능을 보장합니다. 모든 작업은 결정적이고 빠르게 진행되도록 설계되었으며 KeyValueStore 조회는 일반적으로 마이크로초 단위로 완료됩니다.

런타임 환경은 함수 실행 간에 완전히 격리되므로 서로 다른 클라이언트 연결 간에 데이터가 유출되지 않습니다. 각 함수 실행은 정상 상태로 시작되며 이전 실행 결과 또는 다른 연결의 클라이언트 데이터에 액세스할 수 없습니다.

## CloudFront 연결 함수 헬퍼 메서드 및 API

CloudFront 연결 함수는 인증서 검증 결정을 간소화하고 관찰성을 향상하도록 설계된 특수 헬퍼 메서드를 제공합니다. 이러한 방법은 연결 검증 워크플로에 최적화되어 있으며 CloudFront의 연결 로깅 및 모니터링 시스템과 원활하게 통합됩니다.

- connection.allow() - TLS 연결이 진행되도록 허용합니다. 이 방법은 CloudFront에 신호를 보내어 TLS 핸드셰이크를 완료하고 클라이언트가 연결을 설정할 수 있도록 합니다. 인증서 검증을 통과하고 사용자 지정 인증 로직이 충족될 때 사용
- connection.deny() - TLS 연결을 거부하고 핸드셰이크를 종료합니다. 이 방법은 연결을 즉시 닫고 HTTP 트래픽이 흐르지 않도록 합니다. 클라이언트에 TLS 연결 오류가 발생합니다. 잘못된 인증서, 인증 실패 또는 정책 위반에 사용
- connection.logCustomData() - 연결 로그에 사용자 지정 데이터를 추가합니다(최대 800바이트의 UTF-8 텍스트). 이 방법을 사용하면 보안 모니터링, 규정 준수 감사 및 문제 해결을 위해 CloudFront 연결 로그에 검증 결과, 인증서 세부 정보 또는 결정 근거를 포함할 수 있습니다.

이러한 메서드는 연결 결정을 내리고 모니터링 및 디버깅과 관련된 정보를 로깅하는 데 필요한 깔끔하고 선언적인 인터페이스를 제공합니다. 허용/거부 패턴은 함수의 의도가 명확하고 CloudFront가 결정에 따라 연결 처리를 최적화할 수 있도록 합니다. 사용자 지정 로깅 데이터는 CloudFront 연결 로그에서 즉시 사용할 수 있으며, 보안 모니터링 및 운영 인사이트를 위한 로그 분석 도구와 함께 사용할 수 있습니다.

함수가 완료되기 전에 항상 connection.allow() 또는 connection.deny()를 직접 호출합니다. 두 메서드가 모두 직접 호출되지 않으면 CloudFront는 보안 예방 조치로 기본적으로 연결을 거부합니다.

## CloudFront 연결 함수 KeyValueStore 통합

CloudFront 연결 함수는 CloudFront KeyValueStore를 사용하여 인증서 검증 시나리오에서 초고속 데이터 조회를 수행할 수 있습니다. KeyValueStore는 모든 CloudFront 엣지 로케이션에서 최종적으로 일관된 전역 데이터 액세스를 마이크로초 수준의 조회 시간으로 제공하므로 연결 함수에 특히 유용합니다. 따라서 TLS 핸드셰이크 중에 액세스해야 하는 인증서 해지 목록, 디바이스 허용 목록, 테넌트 구성 및 기타 검증 데이터를 유지하는 데 이상적입니다.

KeyValueStore 통합은 고성능 연결 검증 워크플로를 위해 특별히 설계되었습니다.

- kvsHandle.exists(key) - 값을 검색하지 않고 KeyValueStore에 키가 있는지 확인합니다. 인증서 해지 확인과 같은 이진 검증 시나리오에서 인증서 일련 번호가 해지 목록에 있는지 여부만 알면 되는 가장 효율적인 방법입니다.
- kvsHandle.get(key) - 더 복잡한 검증 시나리오를 위해 KeyValueStore에서 값을 검색합니다. 인증서 또는 디바이스 식별자와 연결된 구성 데이터, 검증 규칙 또는 메타데이터에 액세스해야 할 때 사용합니다.

KeyValueStore 작업은 비동기식이며 비동기/대기 구문과 함께 사용해야 합니다. KeyValueStore는 총 크기 제한이 10MB이며 최대 1천만 개의 키-값 페어를 지원합니다. KeyValueStore 데이터는 모든 엣지 로케이션에서 최종적으로 일관되며 업데이트는 일반적으로 몇 초 이내에 전파됩니다.

최적의 성능을 위해 KeyValueStore 키를 구성하여 조회 작업을 최소화합니다. 인증서 일련 번호를 간단한 해지 확인을 위한 키로 사용하거나 다중 CA 환경의 발급자 해시와 일련 번호를 결합한 복합 키를 생성합니다. 데이터 구조를 설계할 때 키 복잡성과 KeyValueStore 용량 간의 장단점을 고려합니다.

## async 및 await 사용

연결 함수는 KeyValueStore 작업 또는 기타 비동기 작업으로 작업할 때 필수적인 비동기/대기 구문을 사용하는 비동기 작업을 지원합니다. 비동기/대기 패턴을 사용하면 TLS 핸드셰이크 처리에 필요한 고성능 특성을 유지하면서 연결 결정을 내리기에 앞서 함수가 KeyValueStore 조회가 완료될 때까지 대기할 수 있습니다.

KeyValueStore 작업은 매우 빠르지만 CloudFront의 분산 인프라 전반에서 조정이 필요한 네트워크 작업이므로 연결 함수에는 적절한 비동기/대기 사용이 중요합니다. 런타임은 promise 해석을 자동으로 처리하고 함수가 5밀리초 실행 한도 내에서 완료되도록 합니다.

Example: KeyValueStore와 비동기 연결 함수

```
import cf from 'cloudfront';

async function connectionHandler(connection) {
    const kvsHandle = cf.kvs();

    // Async operation to check KeyValueStore for certificate revocation
    const isRevoked = await
 kvsHandle.exists(connection.clientCertificate.certificates.leaf.serialNumber);

    if (isRevoked) {
        // Log the revocation decision with certificate details
        connection.logCustomData(`REVOKED_CERT:
${connection.clientCertificate.certificates.leaf.serialNumber}:
${connection.clientCertificate.certificates.leaf.issuer}`);
        console.log(`Denying connection for revoked certificate:
 ${connection.clientCertificate.certificates.leaf.serialNumber}`);
        return connection.deny();
    }

    // Log successful validation for monitoring
```

```
    connection.logCustomData(`VALID_CERT:
${connection.clientCertificate.certificates.leaf.serialNumber}`);
    console.log(`Allowing connection for valid certificate:
 ${connection.clientCertificate.certificates.leaf.serialNumber}`);
    return connection.allow();
}
```

KeyValueStore 메서드 또는 기타 비동기 작업을 직접 호출할 때는 항상 비동기/대기를 사용합니다. 연결 함수 런타임은 promise 해석을 자동으로 처리하고 TLS 핸드셰이크 처리의 엄격한 타이밍 제약 내에서 적절한 실행 흐름을 보장합니다. async/await는 연결 함수 환경에서 더 명확한 오류 처리와 더 나은 성능을 제공하므로 .then() 또는 콜백 패턴을 사용하지 마세요.

비동기 연결 함수를 설계할 때 KeyValueStore 작업 수를 최소화하고 검증 로직에서 가능한 한 빨리 수행하도록 코드를 구성합니다. 이렇게 하면 성능을 극대화하고 트래픽이 많은 기간 동안 제한 시간 문제의 위험을 줄일 수 있습니다. 관련 검증 검사를 일괄 처리하고 사용 사례에 가장 효율적인 KeyValueStore 메서드(exists() vs get())를 사용하는 것이 좋습니다.

## 연결 함수 코드 예제

다음 예제는 다양한 검증 시나리오에 일반적인 연결 함수 패턴을 보여줍니다. 다음 예제를 자체 연결 함수 구현의 시작점으로 사용합니다.

Example: 디바이스 인증서 검증

이 예제에서는 IoT 디바이스, 게임 콘솔 및 기타 클라이언트 인증 시나리오에 대한 디바이스 일련 번호 및 인증서 제목 필드를 검증합니다.

```
async function connectionHandler(connection) {
    // Custom validation: check device serial number format
    const serialNumber = connection.clientCertificate.certificates.leaf.serialNumber;
    if (!serialNumber.startsWith("DEV")) {
        connection.logCustomData(`INVALID_SERIAL:${serialNumber}`);
        return connection.deny();
    }

    // Validate certificate subject contains required organizational unit
    const subject = connection.clientCertificate.certificates.leaf.subject;
    if (!subject.includes("OU=AuthorizedDevices")) {
        connection.logCustomData(`INVALID_OU:${subject}`);
        return connection.deny();
    }
```

```
    // Allow connection for valid devices
    connection.logCustomData(`VALID_DEVICE:${serialNumber}`);
    return connection.allow();
}
```

이 함수는 디바이스 일련 번호 형식 및 조직 단위 확인을 포함하여 표준 인증서 검증 외에도 여러 검증 검사를 수행합니다.

Example: 혼합 인증을 사용하는 선택적 mTLS

이 예제에서는 서로 다른 인증 정책을 사용하여 mTLS 연결과 비mTLS 연결을 모두 처리합니다.

```
async function connectionHandler(connection) {
    if (connection.clientCertificate) {
        // mTLS connection - enhanced validation for certificate holders
        const subject = connection.clientCertificate.certificates.leaf.subject;
        connection.logCustomData(`MTLS_SUCCESS:${subject}:${connection.clientIp}`);
        console.log(`mTLS connection from: ${subject}`);
        return connection.allow();
    } else {
        // Non-mTLS connection - apply IP-based restrictions
        const clientIp = connection.clientIp;

        // Only allow non-mTLS from specific IP ranges
        if (clientIp.startsWith("203.0.113.") || clientIp.startsWith("198.51.100.")) {
            connection.logCustomData(`NON_MTLS_ALLOWED:${clientIp}`);
            console.log(`Non-mTLS connection allowed from: ${clientIp}`);
            return connection.allow();
        }

        connection.logCustomData(`NON_MTLS_DENIED:${clientIp}`);
        return connection.deny();
    }
}
```

이 함수는 신뢰할 수 있는 IP 범위의 레거시 클라이언트와의 호환성을 유지하면서 인증서를 통해 클라이언트에 향상된 보안을 제공합니다.

# 배포 전에 CloudFront 연결 함수 테스트

TestConnectionFunction API 작업을 사용하여 개발 단계에서 CloudFront 연결 함수를 테스트할 수 있습니다. 테스트를 통해 라이브 단계에 게시하기 전에 샘플 연결 이벤트로 함수 로직을 검증할 수 있습니다.

주제


- 테스트 프로세스
- 테스트 결과
- 연결 테스트 객체


## 테스트 프로세스

연결 함수를 테스트하려면:

1. 개발 단계에서 연결 함수 생성
2. TLS 연결 이벤트를 나타내는 테스트 연결 객체 준비
3. TestConnectionFunction API 작업을 사용하여 테스트 데이터로 함수 실행
4. 함수 출력, 실행 로그 및 오류 메시지를 포함한 테스트 결과 검토
5. 필요에 따라 함수 코드를 업데이트하고 테스트 프로세스를 반복합니다.

## 테스트 결과

연결 함수를 테스트할 때 결과는 다음과 같습니다.

- 함수 요약 - 테스트된 함수에 대한 메타데이터
- 컴퓨팅 사용률 - 리소스 사용량을 보여주는 성능 지표
- 실행 로그 - 로깅 문을 포함하여 함수의 콘솔 출력
- 함수 출력 - 함수가 반환한 결과
- 오류 메시지 - 실행 중에 발생한 모든 런타임 오류 또는 예외

## 연결 테스트 객체

연결 테스트 객체는 함수가 처리할 TLS 연결 이벤트를 나타내는 이진 BLOB(최대 40KB)입니다. 이 객체에는 함수가 인증 결정을 내리는 데 사용하는 인증서 및 연결 정보가 포함되어 있습니다.

> **Note**
>
> 연결 테스트 객체의 특정 구조 및 형식은 CloudFront 연결 함수 런타임에 의해 정의됩니다. 사용 사례에 적합한 테스트 객체를 생성하는 방법에 대한 자세한 내용은 CloudFront Functions 설명서를 참조하거나 AWS Support에 문의하세요.

연결 함수를 생성한 후 다음을 수행할 수 있습니다.

- 함수 테스트 - 콘솔 또는 CLI의 테스트 기능을 사용하여 샘플 연결 이벤트로 함수를 검증합니다. 자세한 정보는 연결 함수 테스트를 참조하세요.
- 함수 업데이트 - 필요에 따라 함수 코드 및 구성을 수정합니다. 개발 단계의 연결 함수는 언제든지 업데이트할 수 있습니다.
- 함수 게시 - 프로덕션 준비가 되면 함수를 게시하여 개발 단계에서 라이브 단계로 이동합니다. 자세한 내용은 연결 함수 연결을 참조하세요.
- 배포와 연결 - 게시된 함수를 mTLS 지원 배포와 연결하여 라이브 연결을 처리합니다. 자세한 내용은 연결 함수 연결을 참조하세요.

## 배포에 연결 함수 연결

연결 함수를 라이브 단계에 게시한 후에는 이를 mTLS 지원 배포와 연결하여 라이브 연결을 처리해야 합니다. 연결 함수는 캐시 동작과 연결된 뷰어 요청 및 뷰어 응답 함수와 달리 배포 수준에서 연결됩니다.

주제


- 연결 요구 사항
- 필터를 사용하여 함수 구성
- 배포 고려 사항


### 연결 요구 사항

연결 함수를 배포와 연결하려면:

- 함수는 라이브 단계에 있어야 함
- 배포에 mTLS가 활성화되어 있어야 함

- 배포에 유효한 트러스트 스토어가 구성되어 있어야 함
- 배포당 하나의 연결 함수만 연결할 수 있음

## 필터를 사용하여 함수 구성

CloudFront는 연결 함수를 구성하고 관리하는 데 도움이 되는 필터링 기능을 제공합니다.

- 배포 ID 필터 - 특정 배포와 연결된 함수 찾기
- 키-값 저장소 필터 - 데이터 조회에 특정 키-값 저장소를 사용하는 함수 찾기
- 단계 필터 - 개발 또는 라이브 단계의 함수 나열

이러한 필터는 다양한 배포 또는 개발 환경에서 여러 연결 함수를 관리할 때 사용합니다.

## 배포 고려 사항

연결 함수를 배포할 때는 다음 요소를 고려하세요.

- 글로벌 배포 - 연결 함수가 전 세계 모든 CloudFront 엣지 로케이션에 배포되므로 몇 분 정도 걸릴 수 있습니다.
- 버전 관리 - 게시된 각 버전은 이전 버전을 대체하는 새 라이브 함수를 생성합니다.
- 롤백 전략 - 함수 코드의 이전 작동 버전을 유지하여 롤백을 계획합니다.
- 프로덕션 환경에서 테스트 - 스테이징 및 프로덕션 환경에 별도의 배포를 사용하는 것이 좋습니다.

# CloudFront Functions 및 KeyValueStore를 사용하여 상호 TLS(뷰어)에 대한 인증서 해지 구현

CloudFront 연결 함수를 KeyValueStore와 함께 사용하여 인증서 해지 확인을 구현할 수 있습니다. 이렇게 하면 해지된 인증서 일련 번호 목록을 유지하고 TLS 핸드셰이크 중에 이 목록과 비교하여 클라이언트 인증서를 확인할 수 있습니다.

인증서 해지를 구현하려면 다음 구성 요소가 필요합니다.

- 뷰어 mTLS로 구성된 배포
- 해지된 인증서 일련 번호가 포함된 KeyValueStore
- KeyValueStore를 쿼리하여 인증서 상태를 확인하는 연결 함수

클라이언트가 연결되면 CloudFront는 트러스트 스토어에 대해 인증서를 검증한 다음 연결 함수를 실행합니다. 함수는 KeyValueStore에 대해 인증서 일련 번호를 확인하고 연결을 허용하거나 거부합니다.

주제


- 1단계: 해지된 인증서에 대한 KeyValueStore 생성
- 2단계: 해지 연결 함수 생성
- 3단계: 해지 함수 테스트
- 4단계: 배포에 함수 연결
- 고급 해지 시나리오


## 1단계: 해지된 인증서에 대한 KeyValueStore 생성

KeyValueStore를 생성하여 mTLS 연결 중에 연결 함수가 확인할 수 있는 해지된 인증서 일련 번호를 저장합니다.

먼저 해지된 인증서 일련 번호를 JSON 형식으로 준비합니다.

```
{
  "data": [
    {
      "key": "ABC123DEF456",
      "value": ""
    },
    {
      "key": "789XYZ012GHI",
      "value": ""
    }
  ]
}
```

이 JSON 파일을 S3 버킷에 업로드한 다음 KeyValueStore를 생성합니다.

```
aws s3 cp revoked-serials.json s3://your-bucket-name/revoked-serials.json
aws cloudfront create-key-value-store \
  --name revoked-serials-kvs \
  --import-source '{
    "SourceType": "S3",
    "SourceARN": "arn:aws:s3:::your-bucket-name/revoked-serials.json"
  }'
```

KeyValueStore가 프로비저닝을 완료할 때까지 기다립니다. 다음을 통해 상태를 확인합니다.

```
aws cloudfront get-key-value-store --name "revoked-serials-kvs"
```

## 2단계: 해지 연결 함수 생성

KeyValueStore에 대한 인증서 일련 번호를 확인하여 인증서가 해지되었는지 확인하는 연결 함수를 생성합니다.

KeyValueStore에 대해 인증서 일련 번호를 확인하는 연결 함수를 생성합니다.

```
aws cloudfront create-connection-function \
  --name "revocation-control" \
  --connection-function-config file://connection-function-config.json \
  --connection-function-code file://connection-function-code.txt
```

구성 파일은 KeyValueStore 연결을 지정합니다.

```
{
  "Runtime": "cloudfront-js-2.0",
  "Comment": "A function that implements revocation control via KVS",
  "KeyValueStoreAssociations": {
    "Quantity": 1,
    "Items": [
      {
        "KeyValueStoreArn": "arn:aws:cloudfront::account-id:key-value-store/kvs-id"
      }
    ]
  }
}
```

연결 함수 코드는 KeyValueStore에서 해지된 인증서를 확인합니다.

```
import cf from 'cloudfront';

async function connectionHandler(connection) {
    const kvsHandle = cf.kvs();

    // Get parsed client serial number from client certificate
    const clientSerialNumber = connection.clientCertInfo.serialNumber;
```

```
    // Check KVS to see if serial number exists as a key
    const serialNumberExistsInKvs = await kvsHandle.exists(clientSerialNumber);

    // Deny connection if serial number exists in KVS
    if (serialNumberExistsInKvs) {
        console.log("Connection denied - certificate revoked");
        return connection.deny();
    }

    // Allow connections that don't exist in kvs
    console.log("Connection allowed");
    return connection.allow();
}
```

## 3단계: 해지 함수 테스트

CloudFront 콘솔을 사용하여 샘플 인증서로 연결 함수를 테스트합니다. 콘솔에서 연결 함수로 이동하여 테스트 탭을 사용합니다.

샘플 인증서로 테스트

1. 테스트 인터페이스에 PEM 형식의 샘플 인증서 붙여넣기
2. 선택적으로 IP 기반 로직을 테스트하기 위한 클라이언트 IP 주소 지정
3. 함수 테스트를 선택하여 실행 결과를 확인합니다.
4. 실행 로그를 검토하여 함수 로직 확인

유효한 인증서와 해지된 인증서를 모두 사용해 테스트하여 함수가 두 시나리오를 올바르게 처리하는지 확인합니다. 실행 로그에는 console.log 출력과 함수 실행 중에 발생하는 모든 오류가 표시됩니다.

## 4단계: 배포에 함수 연결

연결 함수를 게시한 후 mTLS 지원 배포와 연결하여 인증서 해지 확인을 활성화합니다.

배포 설정 페이지 또는 연결 함수의 연결된 배포 테이블에서 함수를 연결할 수 있습니다. 배포 설정으로 이동하여 뷰어 상호 인증(mTLS) 섹션으로 스크롤한 후 연결 함수를 선택하고 변경 사항을 저장합니다.

## 고급 해지 시나리오

더 복잡한 인증서 해지 요구 사항을 보려면 다음과 같은 추가 구성을 고려하세요.

주제


- 인증서 해지 목록(CRL)을 KeyValueStore 형식으로 변환
- 여러 인증 기관 처리
- 연결 로그에 사용자 지정 데이터 추가
- CRL 업데이트 관리
- KeyValueStore 용량 계획


## 인증서 해지 목록(CRL)을 KeyValueStore 형식으로 변환

인증서 해지 목록(CRL) 파일이 있는 경우 OpenSSL 및 jq를 사용하여 KeyValueStore JSON 형식으로 변환할 수 있습니다.

### CRL을 KeyValueStore 형식으로 변환

다음과 같이 CRL 파일에서 일련 번호를 추출합니다.

```
openssl crl -text -noout -in rfc5280_CRL.crl | \
  awk '/Serial Number:/ {print $3}' | \
  cut -d'=' -f2 | \
  sed 's/../&:/g;s/:$//' >> serialnumbers.txt
```

일련 번호를 KeyValueStore JSON 형식으로 변환합니다.

```
jq -R -s 'split("\n") | map(select(length > 0)) | {data: map({"key": ., "value": ""})}'
 \
  serialnumbers.txt >> serialnumbers_kvs.json
```

형식이 지정된 파일을 S3에 업로드하고 1단계에 설명된 대로 KeyValueStore를 생성합니다.

### 여러 인증 기관 처리

TrustStore에 여러 인증 기관(CA)이 포함된 경우 KeyValueStore 키에 발급자 정보를 포함하여 일련 번호가 같을 수 있는 다른 CA의 인증서 간 충돌을 방지합니다.

다중 CA 시나리오의 경우 발급자의 SHA1 해시와 일련 번호를 키로 조합하여 사용합니다.

```
import cf from 'cloudfront';
```

```
async function connectionHandler(connection) {
    const kvsHandle = cf.kvs();
    const clientCert = connection.clientCertInfo;

    // Create composite key with issuer hash and serial number
    const issuer = clientCert.issuer.replace(/[^a-zA-Z0-9]/g, '').substring(0, 20);
    const serialno = clientCert.serialNumber;
    const compositeKey = `${issuer}_${serialno}`;

    const cert_revoked = await kvsHandle.exists(compositeKey);

    if (cert_revoked) {
        console.log(`Blocking revoked cert: ${serialno} from issuer: ${issuer}`);
        connection.deny();
    } else {
        connection.allow();
    }
}
```

> **Note**
>
> 발급자 식별자 + 일련 번호를 사용하면 키가 더 길어져 KeyValueStore에 저장할 수 있는 총 항목 수가 줄어들 수 있습니다.

### 연결 로그에 사용자 지정 데이터 추가

연결 함수는 logCustomData 메서드를 사용하여 CloudFront 연결 로그에 사용자 지정 데이터를 추가할 수 있습니다. 이렇게 하면 해지 확인 결과, 인증서 정보 또는 기타 관련 데이터를 로그에 포함할 수 있습니다.

```
async function connectionHandler(connection) {
    const kvsHandle = cf.kvs();
    const clientSerialNumber = connection.clientCertInfo.serialNumber;
    const serialNumberExistsInKvs = await kvsHandle.exists(clientSerialNumber);

    if (serialNumberExistsInKvs) {
        // Log revocation details to connection logs
        connection.logCustomData(`REVOKED:${clientSerialNumber}:DENIED`);
        console.log("Connection denied - certificate revoked");
        return connection.deny();
    }
```

```
    // Log successful validation
    connection.logCustomData(`VALID:${clientSerialNumber}:ALLOWED`);
    console.log("Connection allowed");
    return connection.allow();
}
```

사용자 지정 데이터는 800바이트의 유효한 UTF-8 텍스트로 제한됩니다. 이 제한을 초과하면 CloudFront는 데이터를 가장 가까운 유효한 UTF-8 경계로 잘라냅니다.

> **Note**
>
> 사용자 지정 데이터 로깅은 배포에 연결 로그가 활성화된 경우에만 작동합니다. 연결 로그가 구성되지 않은 경우 logCustomData 메서드는 no-op입니다.

### CRL 업데이트 관리

인증 기관은 두 가지 유형의 CRL을 발급할 수 있습니다.

- 전체 CRL: 해지된 모든 인증서의 전체 목록 포함
- 델타 CRL: 마지막 전체 CRL 이후 해지된 인증서만 나열

전체 CRL 업데이트의 경우 업데이트된 데이터로 새 KeyValueStore를 생성하고 연결 함수 연결을 새 KeyValueStore로 리디렉션합니다. 이 접근 방식은 차이를 계산하고 증분 업데이트를 수행하는 것보다 더 간단합니다.

델타 CRL 업데이트의 경우 update-keys 명령을 사용하여 해지된 새 인증서를 기존 KeyValueStore에 추가합니다.

```
aws cloudfront update-key-value-store \
  --name "revoked-serials-kvs" \
  --if-match "current-etag" \
  --put file://delta-revoked-serials.json
```

### KeyValueStore 용량 계획

KeyValueStore는 크기 제한이 5MB이며 최대 1천만 개의 키-값 페어를 지원합니다. 키 형식 및 데이터 크기에 따라 취소 목록 용량을 계획합니다.

- 일련 번호만 해당: 간단한 해지 확인을 위한 효율적인 스토리지
- 발급자 식별자 + 일련 번호: 다중 CA 환경을 위한 더 긴 키

대규모 해지 목록의 경우 다양한 인증서 범주 또는 기간에 대해 별도의 KeyValueStores를 유지하는 계층형 접근 방식을 구현하는 것이 좋습니다.

# Lambda@Edge를 사용하여 엣지에서 사용자 지정

Lambda@Edge는 AWS Lambda의 확장된 기능입니다. Lambda@Edge는 Amazon CloudFront를 통해 전달되는 콘텐츠를 사용자 지정하는 함수를 실행할 수 있게 해 주는 컴퓨팅 서비스입니다. 미국 동부(버지니아 북부) AWS 리전 한 곳에서 Lambda 콘솔을 사용하여 Node.js 또는 Python 함수를 작성할 수 있습니다.

함수를 만든 후에 Lambda 콘솔 또는 CloudFront 콘솔에 트리거를 추가하여 서버를 프로비저닝하거나 관리하지 않고도 뷰어에 더 가까운 AWS 위치에서 함수를 실행합니다. 선택적으로 Lambda 및 CloudFront API 작업을 사용하여 프로그래밍 방식으로 함수와 트리거를 설정할 수 있습니다.

Lambda@Edge는 하루 몇 번의 요청에서 초당 수천 개의 요청으로 자동 확장됩니다. 오리진 서버가 아니라 최종 사용자에게 가까운 AWS 위치에서 요청을 처리하므로 지연 시간이 크게 단축되고 사용자 경험이 상당히 개선됩니다.

> **Note**
>
> Lambda@Edge는 gRPC 요청에서 지원되지 않습니다. 자세한 내용은 CloudFront 배포에 gRPC 사용 섹션을 참조하시기 바랍니다.

주제


- Lambda@Edge가 요청 및 응답을 처리하는 방법
- Lambda@Edge 사용 방법
- Lambda@Edge 함수 시작(콘솔)
- Lambda@Edge에 대한 IAM 권한 및 역할 설정
- Lambda@Edge 함수 작성 및 생성
- Lambda@Edge 함수에 대한 트리거 추가
- Lambda@Edge 함수 테스트 및 디버깅

- Lambda@Edge 함수 및 복제본 삭제
- Lambda@Edge 이벤트 구조
- 요청 및 응답 작업 수행
- Lambda@Edge 예제 함수

## Lambda@Edge가 요청 및 응답을 처리하는 방법

CloudFront 배포를 Lambda@Edge 함수와 연결하면 CloudFront가 CloudFront 엣지 로케이션에서 요청 및 응답을 가로챕니다. 다음과 같은 CloudFront 이벤트가 발생할 때 Lambda 함수를 실행할 수 있습니다.

- CloudFront가 최종 사용자의 요청을 수신할 때(최종 사용자 요청)
- CloudFront가 오리진에 요청을 전달하기 전(오리진 요청)
- CloudFront가 오리진의 응답을 수신할 때(오리진 응답)
- CloudFront가 최종 사용자에게 응답을 반환하기 전(최종 사용자 응답)

AWS WAF를 사용하는 경우, Lambda@Edge 최종 사용자 요청은 AWS WAF 규칙이 적용된 후에 실행됩니다.

자세한 내용은 요청 및 응답 작업 수행 및 Lambda@Edge 이벤트 구조(을)를 참조하세요.

## Lambda@Edge 사용 방법

다음 예와 같이 Amazon CloudFront 배포에서 Lambda@Edge 처리는 여러 용도로 사용됩니다.

- Lambda 함수는 사용자가 A/B 테스트를 위해 사이트의 다양한 버전을 볼 수 있도록 쿠키를 검사하고 URL을 다시 작성합니다.
- CloudFront는 `User-Agent` 헤더를 확인하여 사용 중인 디바이스를 기반으로 디바이스에 대한 정보를 포함하여 최종 사용자에게 다양한 객체를 반환할 수 있습니다. 예를 들어, CloudFront는 디바이스의 화면 크기에 따라 다른 이미지를 반환할 수 있습니다. 마찬가지로 함수는 `Referer` 헤더 값을 고려하여 CloudFront가 사용 가능한 최저 해상도의 이미지를 봇에 반환하게 할 수 있습니다.
- 아니면 다른 기준으로 쿠키를 검사할 수 있습니다. 예를 들어, 의류를 판매하는 소매 웹 사이트에서 사용자가 재킷에 대해 선택한 색상을 표시하기 위해 쿠키를 사용하는 경우 Lambda 함수는 CloudFront가 선택된 색상으로 재킷 이미지를 반환하도록 요청을 변경할 수 있습니다.
- Lambda 함수는 CloudFront 최종 사용자 요청 또는 오리진 요청 이벤트가 발생할 때 HTTP 응답을 생성할 수 있습니다.

- 함수는 헤더 또는 권한 부여 토큰을 검사하고, CloudFront가 오리진으로 요청을 전달하기 전에 헤더를 삽입하여 콘텐츠에 대한 액세스 권한을 제어할 수 있습니다.
- 또한 Lambda 함수는 외부 리소스에 대한 네트워크 호출을 생성하여 사용자 자격 증명을 확인하거나 추가 콘텐츠를 가져와 응답을 사용자 지정할 수 있습니다.

예제 코드를 포함한 자세한 내용은 Lambda@Edge 예제 함수 섹션을 참조하세요.

콘솔의 Lambda@Edge 설정에 대한 자세한 내용은 자습서: 기본 Lambda@Edge 함수 만들기(콘솔) 섹션을 참조하세요.

## Lambda@Edge 함수 시작(콘솔)

Lambda@Edge에서 CloudFront 트리거를 사용하여 Lambda 함수를 간접 호출할 수 있습니다. CloudFront 배포를 Lambda 함수와 연결하면 CloudFront가 CloudFront 엣지 로케이션에서 요청 및 응답을 가로채고 함수를 실행합니다. Lambda 함수로 보안을 강화하거나 뷰어와 더 관련성 있는 정보가 표시되도록 사용자 지정하여 성능을 높일 수 있습니다.

다음 목록은 CloudFront에서 Lambda 함수를 생성 및 사용하는 방법에 대한 기본적인 개요를 제공합니다.

개요: CloudFront를 통한 Lambda 함수 생성 및 사용

1. 미국 동부(버지니아 북부) 리전에 Lambda 함수를 생성합니다.
2. 번호가 지정된 함수 버전을 저장하고 게시합니다.

   함수를 변경하려면 미국 동부(버지니아 북부)에서 $LATEST 버전의 함수를 편집해야 합니다. 그런 다음 번호가 지정된 새 버전을 게시하고 CloudFront와 연동되도록 설정합니다.
3. 함수를 CloudFront 배포 및 캐시 동작에 연결합니다. 그런 다음 함수를 실행시키는 하나 이상의 CloudFront 이벤트(트리거)를 지정합니다. 예를 들어, CloudFront가 최종 사용자의 요청을 받으면 함수가 실행되도록 하는 트리거를 생성할 수 있습니다.
4. 트리거를 생성하면 Lambda가 해당 함수를 전 세계의 AWS 위치에 복제를 생성합니다.

> **Tip**
>
> 자세한 내용은 함수 생성 및 업데이트, 이벤트 구조와 CloudFront 트리거 추가를 참조하세요. 또한 Lambda@Edge 예제 함수에서 다양한 아이디어와 코드 샘플도 다룹니다.

단계별 자습서는 다음 주제를 참조하세요.

주제

- 자습서: 기본 Lambda@Edge 함수 만들기(콘솔)

## 자습서: 기본 Lambda@Edge 함수 만들기(콘솔)

이 자습서에서는 CloudFront에서 실행되는 예제 Node.js 함수를 만들고 구성하여 Lambda@Edge를 시작하는 방법을 보여줍니다. 이 예시에서는 CloudFront가 파일을 검색할 때 HTTP 보안 헤더를 응답에 추가합니다. (이를 통해 웹 사이트의 보안 및 개인 정보 보호를 개선할 수 있음)

이 자습서에는 자체 웹 사이트가 필요하지 않습니다. 그러나 자체 Lambda@Edge 솔루션을 생성하는 경우 비슷한 단계를 따르고 동일한 옵션 중에서 선택합니다.

주제


- 1단계: AWS 계정 가입
- 2단계: CloudFront 배포 생성
- 3단계: 함수 생성
- 4단계: 함수를 실행할 CloudFront 트리거 추가
- 5단계: 함수 실행 확인
- 6단계: 문제 해결
- 7단계: 예제 리소스 정리
- 관련 정보


### 1단계: AWS 계정 가입

아직 계정에 가입하지 않았다면 AWS 계정에 가입하세요. 자세한 내용은 AWS 계정에 가입 단원을 참조합니다.

### 2단계: CloudFront 배포 생성

예제 Lambda@Edge 함수를 만들려면 먼저 콘텐츠를 제공할 오리진이 있는 CloudFront 환경부터 갖춰야 합니다.

이 예제에서는 Amazon S3 버킷을 배포의 오리진으로 사용하는 CloudFront 배포를 만듭니다. 사용할 환경이 이미 있는 경우 이 단계를 건너뛸 수 있습니다.

Amazon S3 오리진을 사용하여 CloudFront 배포를 생성하려면

1. 이미지 파일 등 샘플 콘텐츠가 될 파일 한두 개로 Amazon S3 버킷을 만듭니다. Amazon S3에 콘텐츠 업로드의 단계를 따르면 됩니다. 버킷의 객체에 대한 퍼블릭 읽기 액세스를 허용하는 권한을 설정해야 합니다.
2. CloudFront 웹 배포 생성의 단계에 따라 CloudFront 배포를 만들고 오리진으로 S3 버킷을 추가합니다. 이미 배포가 있다면 버킷을 그 배포의 오리진으로 추가할 수 있습니다.

> **Tip**
> 배포 ID를 기록해 둡니다. 이 자습서 후반부에서 함수의 CloudFront 트리거를 추가할 때 드롭다운 목록에서 배포 ID(예: `E653W22221KDDL`)를 선택해야 합니다.

3단계: 함수 생성

이 단계에서는 Lambda 콘솔에 있는 청사진 템플릿에서 Lambda 함수를 생성합니다. 이 함수는 CloudFront 배포의 보안 헤더를 업데이트하는 코드를 추가합니다.

Lambda 함수를 생성하려면

1. AWS Management Console에 로그인한 다음 AWS Lambda에서 https://console.aws.amazon.com/lambda/ 콘솔을 엽니다.

> **Important**
> 현재 US-East-1(버지니아 북부) AWS 리전(us-east-1)에 있는지 확인합니다. 이 리전에 있어야 Lambda@Edge 함수를 만들 수 있습니다.

2. 함수 생성을 선택합니다.
3. 함수 생성 페이지에서 청사진 사용을 선택한 다음 검색 필드에 **`cloudfront`**를 입력하여 CloudFront 청사진을 필터링합니다.

> **Note**
> CloudFront 청사진은 US-East-1(버지니아 북부) 리전(us-east-1)에서만 사용 가능합니다.

4. 함수에 대한 템플릿으로 HTTP 응답 헤더 수정 청사진을 선택합니다.
5. 다음과 같이 함수에 대한 정보를 입력합니다.

- 함수 이름 - 함수의 이름을 입력합니다.
- 실행 역할 - 함수에 대한 권한 설정 방법을 선택합니다. 권장되는 기본 Lambda@Edge 권한 정책 템플릿을 사용하려면 AWS 정책 템플릿에서 새 역할 생성을 선택합니다.
- 역할 이름 - 정책 템플릿이 생성하는 역할 이름을 입력합니다.
- 정책 템플릿 - CloudFront 청사진을 함수의 기본으로 선택했으므로 Lambda는 정책 템플릿 기본 Lambda@Edge 권한을 자동으로 추가합니다. 이 정책 템플릿은 CloudFront가 전 세계 CloudFront 로케이션에서 Lambda 함수를 실행할 수 있도록 하는 실행 역할 권한을 추가합니다. 자세한 내용은 Lambda@Edge에 대한 IAM 권한 및 역할 설정 섹션을 참조하세요.

6. 페이지 하단에서 함수 생성을 선택합니다.
7. Lambda@Edge에 배포 창이 나타나면 취소를 선택합니다. (이 자습서에서는 함수를 Lambda@Edge에 배포하기 전에 함수 코드를 수정해야 합니다.)
8. 아래로 스크롤하여 페이지의 코드 소스 섹션으로 이동합니다.
9. 템플릿 코드를 오리진에서 반환하는 보안 헤더를 수정하는 함수로 바꿉니다. 예를 들면 다음과 비슷한 코드를 사용할 수 있습니다.

```
'use strict';
export const handler = (event, context, callback) => {

    //Get contents of response
    const response = event.Records[0].cf.response;
    const headers = response.headers;

    //Set new headers
    headers['strict-transport-security'] = [{key: 'Strict-Transport-Security',
 value: 'max-age= 63072000; includeSubdomains; preload'}];
    headers['content-security-policy'] = [{key: 'Content-Security-Policy', value:
 "default-src 'none'; img-src 'self'; script-src 'self'; style-src 'self'; object-
src 'none'"}];
    headers['x-content-type-options'] = [{key: 'X-Content-Type-Options', value:
 'nosniff'}];
    headers['x-frame-options'] = [{key: 'X-Frame-Options', value: 'DENY'}];
    headers['x-xss-protection'] = [{key: 'X-XSS-Protection', value: '1;
 mode=block'}];
    headers['referrer-policy'] = [{key: 'Referrer-Policy', value: 'same-origin'}];

    //Return modified response
    callback(null, response);
};
```

10. 파일을 선택하여 저장하고 업데이트된 코드를 저장합니다.
11. 배포(Deploy)를 선택합니다.

다음 섹션으로 이동하여 함수를 실행할 CloudFront 트리거를 추가합니다.

4단계: 함수를 실행할 CloudFront 트리거 추가

이제 보안 헤더를 업데이트할 Lambda 함수가 생성되었으므로, 그 함수를 실행하여 오리진에서 해당 배포에 대해 CloudFront로 보내는 모든 응답에 헤더를 추가할 CloudFront 트리거를 구성해야 합니다.

함수의 CloudFront 트리거를 구성하려면

1. Lambda 콘솔의 함수 개요 페이지에서 해당 함수에 대한 함수 개요 페이지에서 트리거 추가를 선택합니다.
2. 트리거 구성에서 CloudFront를 선택합니다.
3. Lambda@Edge에 배포를 선택합니다.
4. Lambda@Edge 배포 창에서 CloudFront 트리거 구성 아래에 다음 정보를 입력합니다.
   - 배포 - 함수와 연결할 CloudFront 배포 ID입니다. 드롭다운 목록에서 배포 ID를 선택합니다.
   - 캐시 동작 - 트리거에 사용할 캐시 동작입니다. 이 예제에서는 *로 설정된 값을 그대로 둡니다. 이 값은 해당 배포의 기본 캐시 동작을 의미합니다. 자세한 내용은 캐시 동작 설정 주제에서 모든 배포 설정 참조 섹션을 참조하세요.
   - CloudFront 이벤트 - 언제 함수를 실행할지 지정하는 트리거입니다. 여기서는 CloudFront가 오리진에서 응답을 반환할 때마다 보안 헤더 함수가 실행되도록 하겠습니다. 드롭다운 목록에서 오리진 응답을 선택합니다. 자세한 내용은 Lambda@Edge 함수에 대한 트리거 추가 섹션을 참조하세요.
5. Lambda@Edge로의 배포 확인의 확인란을 선택합니다.
6. 배포를 선택하여 트리거를 추가하고 함수를 전 세계 AWS 위치에 복제합니다.
7. 함수가 복제될 때까지 기다립니다. 일반적으로 몇 분 정도 걸립니다.

   CloudFront 콘솔로 이동하여 해당 배포를 보고 복제가 완료되었는지 확인할 수 있습니다. 배포 상태가 배포 중에서 날짜 및 시간으로 변경될 때까지 기다립니다. 이는 함수가 복제되었음을 의미합니다. 함수가 작동하는지 확인하려면 다음 단원의 단계를 따릅니다.

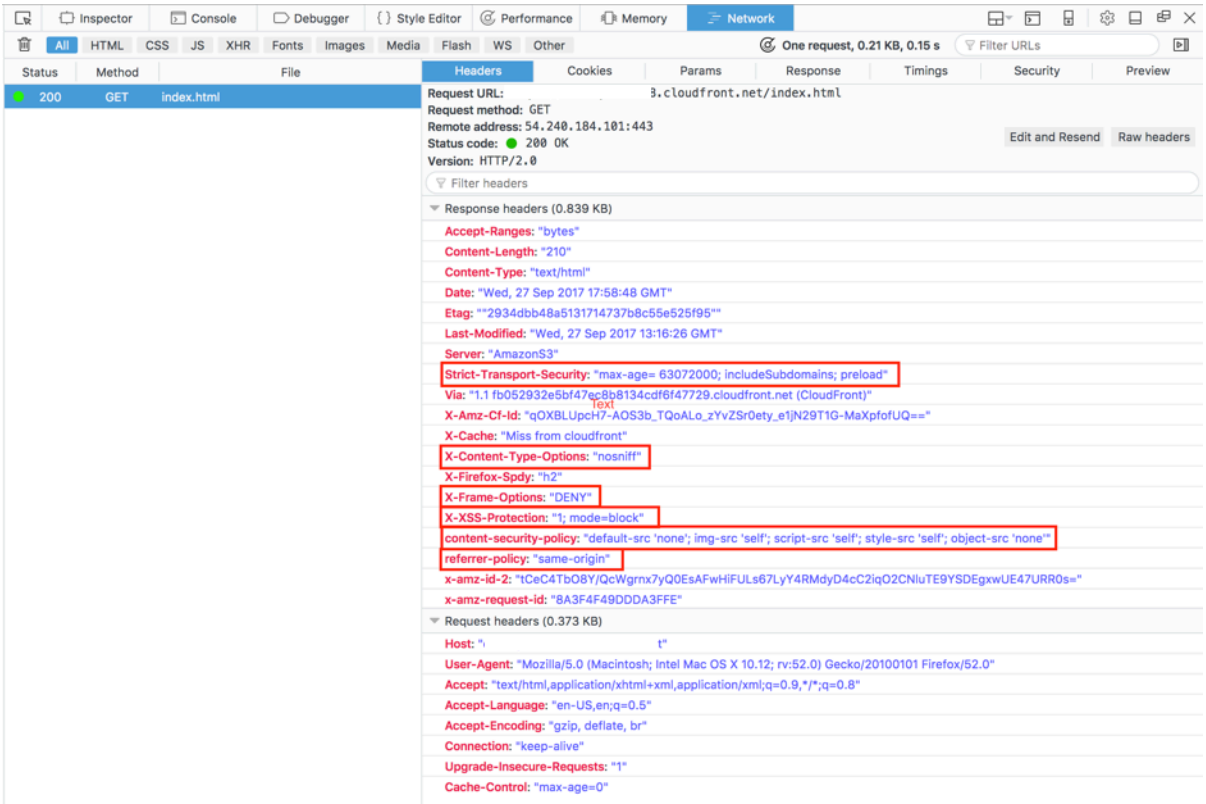
## 5단계: 함수 실행 확인

Lambda 함수를 만들고 CloudFront 배포에 대해 이를 실행하는 트리거를 구성했으니 이제 이 함수가 원하는 결과를 달성하는지 확인해야 합니다. 이 예제에서는 CloudFront가 반환하는 HTTP 헤더를 확인하여 보안 헤더가 추가되었는지 알아봅니다.

Lambda@Edge 함수가 보안 헤더를 추가하는지 확인하려면

1. 브라우저에서 S3 버킷에 있는 파일의 URL을 입력합니다. 예를 들면 `https://d111111abcdef8.cloudfront.net/image.jpg`와 비슷한 URL을 사용할 수 있습니다.

   파일 URL에 사용할 CloudFront 도메인 이름에 대한 자세한 내용은 CloudFront에서 파일에 대한 URL 형식 사용자 지정 단원을 참조하세요.
2. 브라우저에서 웹 개발자 도구 모음을 엽니다. 예를 들어 Chrome 브라우저 창에서는 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 연 다음 Inspect(검사)를 선택합니다.
3. 네트워크 탭을 선택합니다.
4. 페이지를 새로 고쳐 이미지를 확인한 다음 왼쪽 창에서 HTTP 요청을 선택합니다. HTTP 헤더가 별도의 창에 표시됩니다.
5. HTTP 헤더 목록을 살펴보면서 원하는 보안 헤더가 목록에 포함되어 있는지 확인합니다. 예를 들어 다음 스크린샷에 표시된 것과 비슷한 헤더가 보일 것입니다.



추가한 보안 헤더가 헤더 목록에 있으면 완료된 것입니다. 첫 번째 Lambda@Edge 함수가 생성되었습니다. CloudFront에서 오류를 반환하거나 다른 문제가 있는 경우에는 계속해서 다음 단계로 넘어가 문제를 해결하세요.

6단계: 문제 해결

CloudFront가 오류를 반환하거나 원하는 보안 헤더를 추가하지 않는 경우, CloudWatch Logs를 조사하여 함수의 실행 상태를 확인할 수 있습니다. 함수를 실행한 위치와 가장 가까운 AWS 위치에 저장된 로그를 사용해야 합니다.

예를 들어, 런던에서 파일을 볼 때는 CloudWatch 콘솔에서 리전을 유럽(런던)으로 변경해야 합니다.

Lambda@Edge 함수에 대한 CloudWatch 로그를 검사하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 리전을 브라우저에서 파일을 볼 때 표시되는 위치로 변경합니다. 이는 함수가 실행되고 있는 리전입니다.
3. 왼쪽 창에서 로그를 선택하여 해당 배포의 로그를 봅니다.

자세한 내용은 Amazon CloudWatch를 사용한 CloudFront 지표 모니터링 단원을 참조하세요.

7단계: 예제 리소스 정리

이 자습서에 사용할 용도로만 Amazon S3 버킷과 CloudFront 배포를 만들었다면 할당한 AWS 리소스를 삭제하여 더 이상 요금이 발생하지 않도록 하세요. AWS 리소스를 삭제한 뒤에는 추가한 콘텐츠를 이용할 수 없게 됩니다.

작업

- S3 버킷 삭제
- Lambda 함수 삭제
- CloudFront 배포 삭제

S3 버킷 삭제

Amazon S3 버킷을 삭제하기 전에 버킷에 대한 로깅이 비활성화되어 있는지 확인하세요. 그렇지 않으면 삭제해도 AWS가 해당 버킷에 계속 로그를 기록합니다.

버킷에 대한 로깅 사용 중지

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 버킷을 선택한 다음, 속성를 선택합니다.
3. 속성에서 로깅을 선택합니다.
4. 사용(Enabled) 확인란의 선택을 취소합니다.
5. 저장을 선택합니다.

이제 버킷을 삭제할 수 있습니다. 자세한 내용은 Amazon Simple Storage Service Console 사용 설명서의 버킷 삭제를 참조하십시오.

Lambda 함수 삭제

Lambda 함수 연결 및 선택적으로 함수 자체를 삭제하는 지침은 Lambda@Edge 함수 및 복제본 삭제 단원을 참조합니다.

CloudFront 배포 삭제

CloudFront 배포를 삭제하기 전에 반드시 배포를 비활성화해야 합니다. 비활성화된 배포가 작동하지 않아 요금이 발생하지 않습니다. 언제든지 비활성화된 배포를 활성화할 수 있습니다. 비활성화된 배포를 삭제한 뒤에는 사용할 수 없습니다.

CloudFront 배포를 사용하지 않도록 설정하고 삭제하려면

1. 에서 CloudFront 콘솔을 엽니다https://console.aws.amazon.com/cloudfront/v4/home
2. 사용 중지하려는 배포를 선택한 후 사용 중지를 선택합니다.
3. 확인 메시지가 표시되면 예, 사용 중지를 선택합니다.
4. 사용 중지된 배포를 선택한 후 삭제를 선택합니다.
5. 확인 메시지가 나타나면 예, 삭제합니다를 선택합니다.

관련 정보

Lambda@Edge 함수의 작동 방식에 대한 기본적인 내용을 배웠습니다. 이제 다음 자료를 읽고 자세히 알아보십시오.

- Lambda@Edge 예제 함수

- Lambda@Edge 설계 모범 사례
- Lambda@Edge로 대기 시간 감소 및 컴퓨팅 작업을 엣지로 이전

## Lambda@Edge에 대한 IAM 권한 및 역할 설정

Lambda@Edge를 구성하려면 AWS Lambda에 대한 다음 IAM 권한과 역할이 있어야 합니다.

- IAM 권한 - 이러한 권한을 통해 Lambda 함수를 생성하고 CloudFront 배포와 연결할 수 있습니다.
- Lambda 함수 실행 역할(IAM 역할) – Lambda 서비스 보안 주체가 이 역할을 맡아 함수를 실행합니다.
- Lambda@Edge 서비스 연결 역할 - 서비스 연결 역할은 특정 AWS 서비스가 Lambda 함수를 AWS 리전에 복제하고 CloudWatch가 CloudFront 로그 파일을 사용할 수 있도록 합니다.

### Lambda@Edge 함수를 CloudFront 배포와 연결하는 데 필요한 IAM 권한

사용자가 Lambda 함수를 CloudFront 배포와 연결하려면 Lambda에 필요한 IAM 권한 외에 다음과 같은 권한이 필요합니다.

- `lambda:GetFunction` – 함수가 포함된 `.zip` 파일을 다운로드하기 위한 미리 서명된 URL과 Lambda 함수에 대한 구성 정보를 가져오기 위한 권한을 부여합니다.
- `lambda:EnableReplication*` – Lambda 복제 서비스가 함수 코드 및 구성을 가져올 수 있도록 리소스 정책에 대한 권한을 부여합니다.
- `lambda:DisableReplication*` – Lambda 복제 서비스가 함수를 삭제할 수 있도록 리소스 정책에 대한 권한을 부여합니다.

> **Important**
>
> `lambda:EnableReplication*` 및 `lambda:DisableReplication*` 작업 끝에 별표(`*`)를 추가해야 합니다.

- 리소스의 경우, 다음 예시와 같이 CloudFront 이벤트가 발생할 때 실행할 함수 버전의 ARN을 지정합니다.

  ```
  arn:aws:lambda:us-east-1:123456789012:function:TestFunction:2
  ```

- `iam:CreateServiceLinkedRole` – Lambda@Edge가 CloudFront에서 Lambda 함수를 복제하는 데 사용하는 서비스 연결 역할을 생성하기 위한 권한을 부여합니다. Lambda@Edge를 처음으로 구

성한 후 서비스 연결 역할이 자동으로 생성됩니다. Lambda@Edge를 사용하는 다른 배포에는 이 권한을 추가할 필요가 없습니다.

- `cloudfront:UpdateDistribution` 또는 `cloudfront:CreateDistribution` – 배포를 업데이트 또는 생성하기 위한 권한을 부여합니다.

자세한 내용은 다음 항목을 참조하세요.

- Amazon CloudFront용 Identity and Access Management
- AWS Lambda 개발자 안내서의 Lambda 리소스 액세스 권한

## 서비스 보안 주체에 대한 함수 실행 역할

함수를 실행할 때 `lambda.amazonaws.com` 및 `edgelambda.amazonaws.com` 서비스 보안 주체가 맡을 수 있는 IAM 역할을 생성해야 합니다.

> **Tip**
>
> Lambda 콘솔에서 함수를 생성할 때 AWS 정책 템플릿을 사용하여 새로운 실행 역할을 생성할 수 있습니다. 이 단계는 함수를 실행하는 데 필요한 Lambda@Edge 권한을 자동으로 추가합니다. 자습서의 5단계: 간단한 Lambda@Edge 함수 생성을 참조하세요.

IAM 역할 수동 생성에 대한 자세한 내용은 IAM 사용 설명서의 역할 생성 및 정책 연결(콘솔)을 참조하세요.

Example예시: 역할 신뢰 정책

IAM 콘솔의 신뢰 관계 탭에서 이 역할을 추가할 수 있습니다. 권한 탭 아래에 이 정책을 추가하지 마세요.

JSON

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

```
            "Service": [
               "lambda.amazonaws.com",
               "edgelambda.amazonaws.com"
            ]
         },
         "Action": "sts:AssumeRole"
      }
   ]
}
```

실행 역할에 부여해야 하는 권한에 대한 자세한 내용은 AWS Lambda 개발자 안내서의 Lambda 리소스 액세스 권한을 참조하세요.

> **참고**
>
> - 기본적으로 CloudFront 이벤트가 Lambda 함수를 트리거할 때마다 데이터는 CloudWatch Logs에 기록됩니다. 이러한 로그를 사용하려면 실행 역할에 CloudWatch Logs에 데이터를 기록할 권한이 있어야 합니다. 사전 정의된 AWSLambdaBasicExecutionRole을 사용하여 실행 역할에 대한 권한을 부여할 수 있습니다.
>
>  CloudWatch Logs에 대한 자세한 내용은 the section called “엣지 함수 로그” 섹션을 참조하세요.
>
> - Lambda 함수 코드가 S3 버킷에서 객체를 읽는 것처럼 다른 AWS 리소스에 액세스하는 경우 실행 역할에는 해당 작업을 수행할 수 있는 권한이 필요합니다.

## Lambda@Edge의 서비스 연결 역할

Lambda@Edge는 IAM 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 서비스에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 해당 서비스에서 사전 정의하며 서비스에서 사용자를 대신하여 다른 AWS 서비스를 호출하기 위해 필요한 모든 권한을 포함합니다.

Lambda@Edge는 다음 IAM 서비스 연결 역할을 사용합니다.

- AWSServiceRoleForLambdaReplicator - Lambda@Edge에서는 이 역할을 사용해 Lambda@Edge가 AWS 리전에 함수를 복제하도록 합니다.

  CloudFront에 Lambda@Edge 트리거를 처음 추가할 때 AWSServiceRoleForLambdaReplicator 라는 역할이 자동으로 생성되어 Lambda@Edge가 AWS 리전에 함수를 복제할 수 있도록 합니다.

Lambda@Edge 함수를 사용하려면 이 역할이 필요합니다. AWSServiceRoleForLambdaReplicator 역할에 대한 ARN은 다음 예시와 같을 수 있습니다.

```
arn:aws:iam::123456789012:role/aws-service-role/
replicator.lambda.amazonaws.com/AWSServiceRoleForLambdaReplicator
```

- AWSServiceRoleForCloudFrontLogger – CloudFront는 이 역할을 사용하여 CloudWatch에 로그 파일을 푸시합니다. 로그 파일을 사용하여 Lambda@Edge 검증 오류를 디버깅할 수 있습니다.

  AWSServiceRoleForCloudFrontLogger 역할은 CloudFront에서 Lambda@Edge 오류 로그 파일을 CloudWatch로 푸시하도록 허용하기 위해 Lambda@Edge 함수 연결을 추가한 경우 자동으로 생성됩니다. AWSServiceRoleForCloudFrontLogger 역할의 ARN의 모양은 다음과 같습니다.

  ```
  arn:aws:iam::account_number:role/aws-service-role/
  logger.cloudfront.amazonaws.com/AWSServiceRoleForCloudFrontLogger
  ```

서비스 연결 역할이 있으면 필요한 권한을 수동으로 추가할 필요가 없으므로 Lambda@Edge를 설정 및 사용하기가 쉽습니다. 서비스 연결 역할의 권한은 Lambda@Edge에서 정의하며, Lambda@Edge만이 그 역할을 맡을 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함됩니다. 권한 정책은 다른 어떤 IAM 엔터티에도 연결할 수 없습니다.

서비스 연결 역할을 삭제하려면 먼저 연결된 CloudFront 또는 Lambda@Edge 리소스를 제거해야 합니다. 그러면 활성 리소스에 액세스하는 데 필요한 서비스 연결 역할을 제거하지 않고 Lambda@Edge 리소스를 보호할 수 있습니다.

서비스 연결 역할에 대한 자세한 내용은 CloudFront 서비스 연결 역할를 참조하세요.

Lambda@Edge의 서비스 연결 역할 권한

Lambda@Edge는 AWSServiceRoleForLambdaReplicator 및 AWSServiceRoleForCloudFrontLogger이라는 서비스 연결 역할을 사용합니다. 다음 단원에서는 이러한 각 역할에 대한 권한을 설명합니다.

목차


- Lambda Replicator의 서비스 연결 역할 권한
- CloudFront Logger에 대한 서비스 연결 역할 권한


Lambda Replicator의 서비스 연결 역할 권한

이 서비스 연결 역할을 통해 Lambda는 Lambda@Edge 함수를 AWS 리전에 복제할 수 있습니다.

AWSServiceRoleForLambdaReplicator 서비스 연결 역할은 역할을 수임하기 위해 `replicator.lambda.amazonaws.com`서비스를 신뢰합니다.

역할 권한 정책에서는 Lambda@Edge가 지정된 리소스에서 다음 작업을 완료할 수 있도록 허용합니다.

- `lambda:CreateFunction`의 `arn:aws:lambda:*:*:function:*`
- `lambda:DeleteFunction`의 `arn:aws:lambda:*:*:function:*`
- `lambda:DisableReplication`의 `arn:aws:lambda:*:*:function:*`
- `iam:PassRole`의 `all AWS resources`
- `cloudfront:ListDistributionsByLambdaFunction`의 `all AWS resources`

CloudFront Logger에 대한 서비스 연결 역할 권한

이 서비스 연결 역할을 사용하면 Lambda@Edge 검증 오류를 디버깅하기 위해 CloudFront에서 CloudWatch로 로그 파일을 푸시할 수 있습니다.

AWSServiceRoleForCloudFrontLogger 서비스 연결 역할은 역할을 수임하기 위해 `logger.cloudfront.amazonaws.com`서비스를 신뢰합니다.

역할 권한 정책에서는 Lambda@Edge가 지정된 `arn:aws:logs:*:*:log-group:/aws/cloudfront/*` 리소스에서 다음 작업을 완료할 수 있도록 허용합니다.

- `logs:CreateLogGroup`
- `logs:CreateLogStream`
- `logs:PutLogEvents`

IAM 엔터티(예: 사용자, 그룹 또는 역할)가 Lambda@Edge 서비스 연결 역할을 삭제할 수 있도록 권한을 구성해야 합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

Lambda@Edge의 서비스 연결 역할 생성

일반적으로 Lambda@Edge에 대한 서비스 연결 역할은 수동으로 생성하지 않습니다. 다음 시나리오에서 서비스는 역할을 자동으로 생성합니다.

- 트리거를 처음으로 생성할 때, 이 서비스에서는 AWSServiceRoleForLambdaReplicator 역할이 없는 경우 해당 역할을 생성합니다. 이 역할을 통해 Lambda는 Lambda@Edge 함수를 AWS 리전에 복제할 수 있습니다.

서비스 연결 역할을 삭제하는 경우, 배포에서 Lambda@Edge의 새 트리거를 추가할 때 이 역할이 다시 생성됩니다.

- Lambda@Edge 연결이 있는 CloudFront 배포를 업데이트 또는 생성할 때, 이 서비스에서는 AWSServiceRoleForCloudFrontLogger라는 역할이 없는 경우 해당 역할을 생성합니다. 이 정책은 CloudFront에서 로그 파일을 CloudFront로 푸시할 수 있도록 허용합니다.

  서비스 연결 역할을 삭제했다 하더라도, Lambda@Edge 연결이 있는 CloudFront 배포를 업데이트 또는 생성하는 경우 해당 역할이 다시 생성됩니다.

이러한 서비스 연결 역할을 수동으로 만들어야 하는 경우 다음 AWS Command Line Interface(AWS CLI) 명령을 실행합니다.

AWSServiceRoleForLambdaReplicator 역할을 만들려면

- 다음 명령을 실행합니다.

```
aws iam create-service-linked-role --aws-service-name
 replicator.lambda.amazonaws.com
```

AWSServiceRoleForCloudFrontLogger 역할을 만들려면

- 다음 명령을 실행합니다.

```
aws iam create-service-linked-role --aws-service-name
 logger.cloudfront.amazonaws.com
```

Lambda@Edge 서비스 연결 역할 편집

Lambda@Edge에서는 AWSServiceRoleForLambdaReplicator 또는 AWSServiceRoleForCloudFrontLogger 서비스 연결 역할을 편집할 수 없습니다. 서비스에서 서비스 연결 역할을 만든 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 그러나 IAM을 사용하여 역할 설명을 편집할 수는 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

Lambda@Edge 서비스 연결 역할에 대해 지원되는 AWS 리전

CloudFront는 다음 AWS 리전에서 Lambda@Edge에 대한 서비스 연결 역할 사용을 지원합니다.

- 미국 동부(버지니아 북부) – `us-east-1`
- 미국 동부(오하이오) – `us-east-2`
- 미국 서부(캘리포니아 북부) – `us-west-1`
- 미국 서부(오리건) – `us-west-2`
- 아시아 태평양(뭄바이) – `ap-south-1`
- 아시아 태평양(서울) – `ap-northeast-2`
- 아시아 태평양(싱가포르) – `ap-southeast-1`
- 아시아 태평양(시드니) – `ap-southeast-2`
- 아시아 태평양(도쿄) – `ap-northeast-1`
- 유럽(프랑크푸르트) – `eu-central-1`
- 유럽(아일랜드) – `eu-west-1`
- 유럽(런던) – `eu-west-2`
- 남아메리카(상파울루) – `sa-east-1`

## Lambda@Edge 함수 작성 및 생성

Lambda@Edge를 사용하려면 AWS Lambda 함수에 대한 코드를 작성해야 합니다. Lambda@Edge 함수를 작성하는 데 도움이 되도록 다음 리소스를 참조하세요.

- Lambda@Edge 이벤트 구조 – Lambda@Edge와 함께 사용할 이벤트 구조를 이해합니다.
- Lambda@Edge 예제 함수 – A/B 테스트 및 HTTP 리디렉션 생성 등의 예시 함수입니다.

Lambda@Edge에서 Node.js 또는 Python을 사용하는 프로그래밍 AWS 리전에서 Lambda를 사용하는 것과 동일합니다. 자세한 내용은 AWS Lambda 개발자 안내서의 Node.js를 사용하여 Lambda 함수 작성 또는 Python을 사용하여 Lambda 함수 작성을 참조하세요.

Lambda@Edge 함수에 `callback` 파라미터를 포함하고 요청 또는 응답 이벤트에 해당되는 객체를 반환합니다.

- 요청 이벤트 - 응답에 `cf.request` 객체를 포함합니다.

  응답을 생성하는 경우 응답에 `cf.response` 객체를 포함합니다. 자세한 내용은 요청 트리거에서 HTTP 응답 생성 단원을 참조하세요.
- 응답 이벤트 - 응답에 `cf.response` 객체를 포함합니다.

코드를 직접 작성하거나 예제 중 하나를 사용한 후 Lambda에서 함수를 생성합니다. 함수를 생성하거나 기존 함수를 편집하려면 다음 주제를 참조하세요.

주제

- Lambda@Edge 함수 생성
- Lambda 함수 편집

Lambda에서 함수를 생성한 후에는 triggers라고 하는 특정 CloudFront 이벤트를 기반으로 함수를 실행하도록 Lambda를 설정합니다. 자세한 내용은 Lambda@Edge 함수에 대한 트리거 추가 섹션을 참조하세요.

## Lambda@Edge 함수 생성

CloudFront 이벤트를 기반으로 한 Lambda 함수를 실행하도록 AWS Lambda를 설정하려면 다음 절차를 수행합니다.

Lambda@Edge 함수를 생성하려면

1. AWS Management Console에 로그인하고 AWS Lambdahttps://console.aws.amazon.com/lambda/에서 콘솔을 엽니다.
2. 이미 하나 이상의 Lambda 함수가 있는 경우에는 함수 생성을 선택합니다.

   아무 함수도 없는 경우 Get Started Now(지금 시작하기)를 선택합니다.
3. 페이지 상단의 리전 목록에서 미국 동부(버지니아 북부)(US East (N. Virginia))를 선택합니다.
4. 자체 코드를 사용하여 함수를 생성하거나 CloudFront 청사진으로 시작하는 함수를 생성합니다.
   - 자체 코드를 사용하여 함수를 생성하려면 새로 작성을 선택합니다.
   - CloudFront용 청사진 목록을 표시하려면 필터 필드에 cloudfront를 입력한 다음, Enter 키를 누릅니다.

     사용할 청사진을 찾은 경우 청사진 이름을 선택합니다.
5. 기본 정보 섹션에서 다음 값을 지정합니다.
   a. 이름 – 함수의 이름을 입력합니다.
   b. 역할 - 빠르게 시작하려면 템플릿에서 새 역할 만들기를 선택합니다. 기존 역할 선택 또는 사용자 지정 역할 생성을 선택한 다음 지시에 따라 이 섹션의 정보를 작성할 수도 있습니다.
   c. 역할 이름 – 역할의 이름을 입력합니다.

d. 정책 템플릿 – 기본 Edge Lambda 권한을 선택합니다.

6. 4단계에서 새로 작성을 선택한 경우 7단계로 건너뜁니다.

   4단계에서 청사진을 선택한 경우 cloudfront 단원에서 이 함수를 CloudFront 배포 및 CloudFront 이벤트의 캐시와 연결하는 트리거를 하나 생성할 수 있습니다. 이때 함수 생성 시 트리거가 없도록 제거를 선택하는 것이 좋습니다. 그런 다음, 나중에 트리거를 추가할 수 있습니다.

   > **Tip**
   >
   > 트리거를 추가하기 전에 함수를 테스트하고 디버깅하는 것이 좋습니다. 지금 트리거를 추가하면 함수가 생성되어 전 세계의 AWS 위치로 복제를 완료하고 난 후 해당 배포가 배포되는 즉시 함수가 실행됩니다.

7. 함수 생성을 선택합니다.

   Lambda는 함수의 두 버전인 $LATEST와 Version 1을 생성합니다. $LATEST 버전만 편집할 수 있지만 콘솔에 처음에는 Version 1이 표시됩니다.

8. 함수를 편집하려면 페이지 위쪽의, 함수 ARN 아래에서 Version 1을 선택합니다. 그런 다음, 버전 탭에서 $LATEST를 선택합니다. 함수에서 나간 후 다시 돌아오면 버튼 레이블이 한정자로 바뀌어 있습니다.
9. 구성 탭에서 해당하는 코드 입력 유형을 선택합니다. 그런 다음, 프롬프트의 메시지를 따라 코드를 편집하거나 업로드합니다.
10. 실행 시간에서 함수 코드를 기반으로 값을 선택합니다.
11. 태그 섹션에서 해당하는 태그를 추가합니다.
12. 작업을 선택한 다음, Publish new version(새 버전 게시)을 선택합니다.
13. 새 함수 버전의 설명을 입력합니다.
14. [Publish]를 선택합니다.
15. 함수를 테스트하고 디버깅합니다. Lambda 콘솔 테스트에 대한 자세한 내용은 AWS Lambda 개발자 안내서에서 콘솔을 사용하여 Lambda 함수 호출을 참조하세요.
16. CloudFront 이벤트에 대해 함수를 실행할 준비가 되면 다른 버전을 게시하고 함수를 편집하여 트리거를 추가합니다. 자세한 내용은 Lambda@Edge 함수에 대한 트리거 추가 섹션을 참조하세요.

## Lambda 함수 편집

Lambda@Edge 함수를 생성한 후 Lambda 콘솔을 사용하여 편집할 수 있습니다.

참고

- 원래 버전은 $LATEST라고 표시되어 있습니다.
- $LATEST 버전만 편집할 수 있습니다.
- $LATEST 버전을 편집할 때마다 번호가 지정된 새 버전을 게시해야 합니다.
- $LATEST에 대한 트리거는 생성할 수 없습니다.
- 새 버전의 함수를 게시하면 Lambda는 이전 버전의 트리거를 새 버전으로 자동 복사하지 않습니다. 새 버전에 대한 트리거를 다시 생성해야 합니다.
- CloudFront 이벤트에 대한 트리거를 함수에 추가할 때 동일한 함수의 이전 버전에 대해 동일한 배포, 캐시 동작 및 이벤트에 대한 트리거가 이미 존재하면 Lambda는 이전 버전에서 트리거를 삭제합니다.
- 트리거를 추가하는 등 CloudFront 배포를 업데이트한 후에는 변경 사항이 엣지 로케이션으로 전파될 때까지 기다려야만 트리거에서 지정한 함수가 작동합니다.

Lambda 함수를 편집하려면

1. AWS Management Console에 로그인하고 AWS Lambdahttps://console.aws.amazon.com/lambda/에서 콘솔을 엽니다.
2. 페이지 상단의 리전 목록에서 미국 동부(버지니아 북부)(US East (N. Virginia))를 선택합니다.
3. 함수 목록에서 함수의 이름을 선택합니다.

   기본 설정 사용 시, $LATEST 버전이 콘솔에 표시됩니다. 이전 버전을 볼 수 있지만(한정자 선택) $LATEST만 편집할 수 있습니다.
4. 코드(Code) 탭의 코드 입력 유형(Code entry type)에서 브라우저에서 코드를 편집하거나, .zip 파일을 업로드하거나, Amazon S3에서 파일을 업로드하도록 선택합니다.
5. 저장 또는 저장 및 테스트를 선택합니다.
6. 작업 및 Publish new version(새 버전 게시)을 차례대로 선택합니다.
7. $LATEST의 새 버전 게시 대화 상자에서 새 버전 설명을 입력합니다. 이 설명은 자동으로 생성된 버전 번호와 함께 버전 목록에 표시됩니다.
8. [Publish]를 선택합니다.

   새 버전은 자동으로 최신 버전이 됩니다. 버전 번호는 페이지 왼쪽 상단 모서리의 버전에 표시됩니다.

> **Note**
>
> 함수에 트리거를 아직 추가하지 않았다면 Lambda@Edge 함수에 대한 트리거 추가 섹션을 참조하세요.

9. 트리거 탭을 선택합니다.
10. 트리거 추가를 선택합니다.
11. 트리거 추가(Add trigger) 대화 상자에서 점선으로 된 상자를 선택한 다음, CloudFront를 선택합니다.

> **Note**
>
> 함수에 대해 하나 이상의 트리거를 이미 만든 경우 CloudFront가 기본 서비스입니다.

12. 다음 값을 지정하여 Lambda 함수를 실행할 시기를 나타냅니다.
    a. 배포 ID – 트리거를 추가할 배포의 ID를 선택합니다.
    b. 캐시 동작 – 함수를 실행할 객체를 지정하는 캐시 동작을 선택합니다.
    c. CloudFront 이벤트 – 함수를 실행시키는 CloudFront 이벤트를 선택합니다.
    d. 트리거 활성화 및 복제 – 이 확인란을 선택하면 Lambda가 전 세계 AWS 리전으로 함수를 복제합니다.
13. 제출을 선택합니다.
14. 이 함수에 대한 트리거를 추가하려면 10~13단계를 반복합니다.

Lambda 콘솔에서 함수를 테스트하고 디버깅하는 방법에 대한 자세한 내용은 AWS Lambda 개발자 안내서에서 콘솔을 사용하여 Lambda 함수 호출을 참조하세요.

CloudFront 이벤트에 대해 함수를 실행할 준비가 되면 다른 버전을 게시하고 함수를 편집하여 트리거를 추가합니다. 자세한 내용은 Lambda@Edge 함수에 대한 트리거 추가 섹션을 참조하세요.

## Lambda@Edge 함수에 대한 트리거 추가

Lambda@Edge 트리거는 CloudFront 배포, 캐시 동작, 그리고 함수 실행을 유도하는 이벤트를 하나로 조합한 것입니다. 예를 들어, 최종 사용자가 해당 배포에 설정된 특정 캐시 동작을 CloudFront에 요청하면 함수가 실행되도록 하는 트리거를 생성할 수 있습니다. 하나 이상의 CloudFront 트리거를 지정할 수 있습니다.

> **Tip**
>
> CloudFront 배포를 생성할 때 다른 요청을 수신할 때 응답하는 방법을 CloudFront에게 알려주는 설정을 지정합니다. 기본 설정을 배포에 대한 기본 캐시 동작이라고 합니다. CloudFront가 특정한 상황(예를 들어, 특정 파일 유형에 대한 요청을 수신할 때)에서 응답하는 방법을 정의하는 추가 캐시 동작을 설정할 수 있습니다. 자세한 내용은 캐시 동작 설정 섹션을 참조하세요.

Lambda 함수를 처음 생성할 때는 트리거를 하나만 지정할 수 있습니다. 나중에 Lambda 콘솔을 사용하거나 CloudFront 콘솔에서 배포를 편집하여 동일한 함수에 더 많은 트리거를 추가할 수 있습니다.

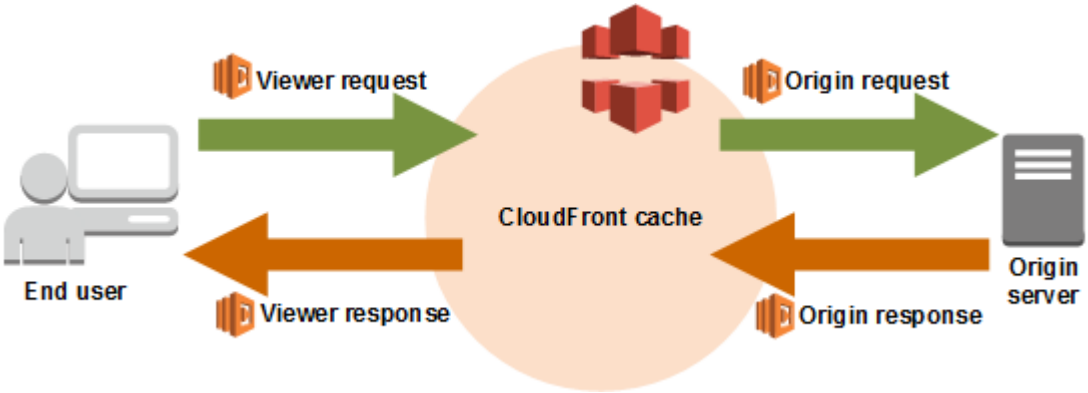
- Lambda 콘솔을 사용하는 방법은 동일한 CloudFront 배포에 대한 함수에 트리거를 더 추가하려는 경우에 적합합니다.
- CloudFront 콘솔을 사용하는 방법은 업데이트하려는 배포를 쉽게 찾을 수 있기 때문에 여러 배포에 대한 트리거를 추가하려는 경우에 더 적합할 수 있습니다. 또한 동시에 다른 CloudFront 설정을 업데이트할 수 있습니다.

주제


- Lambda@Edge 함수를 트리거할 수 있는 CloudFront 이벤트
- 함수를 트리거할 이벤트를 선택합니다.
- Lambda@Edge 함수에 트리거 추가


## Lambda@Edge 함수를 트리거할 수 있는 CloudFront 이벤트

특정한 CloudFront 이벤트가 발생하면 Lambda 함수가 실행되도록 하는 트리거(연결)를 Amazon CloudFront 배포의 각 캐시 동작에 대해 4개까지 추가할 수 있습니다. CloudFront 트리거는 아래 그림의 네 가지 CloudFront 이벤트 중 하나를 토대로 합니다.



Lambda@Edge 함수를 트리거하는 데 사용할 수 있는 CloudFront 이벤트는 다음과 같습니다.

최종 사용자 요청

이 함수는 CloudFront가 최종 사용자로부터 요청을 수신하면 실행되며, 요청된 객체가 CloudFront 캐시에 있는지를 확인합니다.

다음과 같은 경우에는 함수가 실행되지 않습니다.

- 사용자 지정 오류 페이지를 가져오는 경우.
- CloudFront가 자동으로 HTTP 요청을 HTTPS로 리디렉션하는 경우(뷰어 프로토콜 정책 값이 Redirect HTTP to HTTPS일 경우).

오리진 요청

이 함수는 CloudFront가 오리진으로 요청을 전달할 경우에만 실행됩니다. 요청된 객체가 CloudFront 캐시에 저장되어 있으면 함수는 실행되지 않습니다.

오리진 응답

이 함수는 CloudFront가 오리진으로부터 응답을 수신한 후 실행되어 응답의 객체를 캐싱합니다. 오리진에서 오류가 반환되는 경우에도 함수는 실행됩니다.

다음과 같은 경우에는 함수가 실행되지 않습니다.

- 요청된 파일이 CloudFront 캐시에 있고 만료되지 않은 경우.
- 오리진 요청 이벤트가 트리거한 함수로부터 응답이 생성되는 경우.

최종 사용자 응답

요청된 파일을 최종 사용자에게 반환하기 전에 함수가 실행됩니다. 이때 함수는 해당 파일이 이미 CloudFront 캐시에 있는지 여부와 상관없이 실행됩니다.

다음과 같은 경우에는 함수가 실행되지 않습니다.

- 오리진이 400 이상의 HTTP 상태 코드 반환하는 경우
- 사용자 지정 오류 페이지가 반환될 때.
- 최종 사용자 요청 이벤트가 트리거한 함수로부터 응답이 생성되는 경우.
- CloudFront가 자동으로 HTTP 요청을 HTTPS로 리디렉션하는 경우(뷰어 프로토콜 정책 값이 Redirect HTTP to HTTPS일 경우).

한 캐시 동작에 여러 트리거를 추가하는 경우, 이를 사용하여 동일한 함수를 실행하거나 트리거마다 다른 함수를 실행할 수 있습니다. 또한 복수의 배포에 동일한 함수를 연결할 수도 있습니다.

> **Note**
>
> CloudFront 이벤트가 Lambda 함수의 실행을 트리거하면 CloudFront가 계속되기 전에 함수가 완료되어야 합니다.
> 예를 들어, CloudFront 최종 사용자 요청 이벤트에 따라 Lambda 함수가 트리거되면 CloudFront는 Lambda 함수 실행이 완료될 때까지 최종 사용자에게 응답을 반환하거나 오리진에 요청을 전달하지 않습니다.
> 다시 말해, Lambda 함수를 트리거하는 각 요청은 요청 지연 시간을 증가시키므로 최대한 빨리 함수가 실행되도록 해야 합니다.

## 함수를 트리거할 이벤트를 선택합니다.

Lambda 함수를 트리거하는 데 사용할 CloudFront 이벤트를 결정할 때 다음 사항을 고려해야 합니다.

CloudFront에서 Lambda 함수가 변경한 객체를 캐싱하고자 함

CloudFront가 Lambda 함수로 수정된 객체를 캐싱하여 다음에 해당 객체가 요청될 경우 엣지 로케이션의 객체에 CloudFront가 서비스를 제공할 수 있게 하려면 오리진 요청 또는 오리진 응답 이벤트를 사용합니다.

그러면 오리진에 대한 로드가 감소하고, 이후 요청의 지연 시간이 단축되며, 이후 요청에서 Lambda@Edge를 호출하는 비용이 절감됩니다.

예를 들어, 오리진에 의해 반환되는 객체의 헤더를 추가, 제거 또는 변경하고 CloudFront에서 해당 결과를 캐싱하게 하려는 경우 오리진 응답 이벤트를 사용합니다.

모든 요청에 대해 함수를 실행하고자 함

CloudFront가 배포에 대해 수신하는 모든 요청에서 함수를 실행하려면 뷰어 요청 또는 뷰어 응답 이벤트를 사용합니다.

오리진 요청 및 오리진 응답 이벤트는 요청된 객체가 엣지 로케이션에 캐싱되지 않고 CloudFront가 오리진으로 요청을 전달하는 경우에만 발생합니다.

함수가 캐시 키를 변경하고자 함

함수가 캐싱의 기준으로 사용되는 값을 변경하게 하려면 뷰어 요청 이벤트를 사용합니다.

예를 들어, 함수가 URL을 변경하여 경로에 언어 약자를 포함시키는 경우(예를 들어, 사용자가 드롭다운 목록에서 언어를 선택했기 때문) 최종 사용자 요청 이벤트를 사용합니다.

- 최종 사용자 요청 내 URL - https://example.com/en/index.html

- 요청이 독일 내 IP 주소로부터 온 경우의 URL - https://example.com/de/index.html

쿠키 또는 요청 헤더를 기준으로 캐싱하는 경우에도 최종 사용자 요청 이벤트를 사용합니다.

> **Note**
>
> 함수가 쿠키 또는 헤더를 변경하는 경우 CloudFront가 요청의 해당 부분을 오리진으로 전달하도록 구성합니다. 자세한 내용은 다음 항목을 참조하세요.
>
> - 쿠키 기반의 콘텐츠 캐싱
> - 요청 헤더 기반의 콘텐츠 캐싱

함수가 오리진의 응답에 영향을 미침

함수가 오리진의 응답에 영향을 미치는 방식으로 요청을 변경하려면 오리진 요청 이벤트를 사용합니다.

일반적으로 대부분의 뷰어 요청 이벤트는 오리진으로 전달되지 않습니다. CloudFront는 이미 엣지 캐시에 저장된 객체를 사용하여 요청에 응답합니다. 함수가 오리진 요청 이벤트를 기준으로 요청을 변경하는 경우 CloudFront가 변경된 오리진 요청에 대한 응답을 캐싱합니다.

## Lambda@Edge 함수에 트리거 추가

Lambda@Edge 함수에 트리거를 추가하기 위해 AWS Lambda 콘솔 또는 Amazon CloudFront 콘솔을 사용할 수 있습니다.

> **Important**
>
> 번호가 매겨진 함수 버전에 대해서만 트리거를 생성할 수 있습니다($LATEST 제외).

Lambda console

Lambda@Edge 함수에 CloudFront 이벤트에 대한 트리거를 추가하려면

1. AWS Management Console에 로그인하고 AWS Lambdahttps://console.aws.amazon.com/lambda/에서 콘솔을 엽니다.
2. 페이지 상단의 리전 목록에서 미국 동부(버지니아 북부)(US East (N. Virginia))를 선택합니다.
3. 함수 페이지에서 트리거를 추가할 함수의 이름을 선택합니다.

4. 함수 개요 페이지에서 버전 탭을 선택합니다.
5. 트리거를 추가할 버전을 선택합니다.

   버전을 선택한 후 버튼의 이름이 버전: $LATEST 또는 버전: 버전 번호로 변경됩니다.
6. 트리거 탭을 선택합니다.
7. 트리거 추가를 선택합니다.
8. 트리거 구성에서 소스 선택을 선택하고 **`cloudfront`**를 입력한 다음 CloudFront를 선택합니다.

   > **Note**
   > 하나 이상의 트리거를 이미 만든 경우 CloudFront가 기본 서비스입니다.

9. 다음 값을 지정하여 Lambda 함수를 실행할 시기를 나타냅니다.

   a. 배포 – 트리거를 추가할 배포를 선택합니다.
   b. 캐시 동작 – 함수를 실행할 객체를 지정하는 캐시 동작을 선택합니다.

      > **Note**
      > 캐시 동작에 *를 지정하는 경우 Lambda 함수는 기본 캐시 동작으로 배포됩니다.

   c. CloudFront 이벤트 – 함수를 실행시키는 CloudFront 이벤트를 선택합니다.
   d. 본문 포함 – 함수의 요청 본문에 액세스하려는 경우 이 확인란을 선택합니다.
   e. Lambda@Edge로 배포 확인 - 이 확인란을 선택하면 AWS Lambda가 전 세계 AWS 리전으로 함수를 복제합니다.
10. 추가를 선택합니다.

    업데이트된 CloudFront 배포가 배포되면 함수가 지정된 CloudFront 이벤트에 대한 요청을 처리하기 시작합니다. 배포가 배포되었는지 확인하려면 탐색 창에서 배포를 선택합니다. 배포가 되면 배포에 대한 상태 열의 값이 배포 중에서 배포 날짜 및 시간으로 변경됩니다.

CloudFront console

Lambda@Edge 함수에 CloudFront 이벤트에 대한 트리거를 추가하려면

1. 트리거를 추가하려는 Lambda 함수의 ARN을 얻습니다.

a. AWS Management Console에 로그인하고 AWS Lambdahttps://console.aws.amazon.com/lambda/에서 콘솔을 엽니다.

b. 페이지 상단의 리전 목록에서 미국 동부(버지니아 북부)(US East (N. Virginia))를 선택합니다.

c. 함수 목록에서 트리거를 추가하려는 함수의 이름을 선택합니다.

d. 함수 개요 페이지에서 버전 탭을 선택하고 트리거를 추가할 번호가 매겨진 버전을 선택합니다.

e. ARN 복사 버튼을 선택하여 ARN을 클립보드에 복사합니다. Lambda 함수의 ARN은 다음과 같은 형태입니다.

```
arn:aws:lambda:us-east-1:123456789012:function:TestFunction:2
```

끝부분의 숫자(이 예제에서는 2)가 함수의 버전 번호입니다.

2. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
3. 배포 목록에서 트리거를 추가하려는 배포의 ID를 선택합니다.
4. 동작 탭을 선택합니다.
5. 트리거를 추가하려는 캐시 동작을 선택한 다음 편집을 클릭합니다.
6. 함수 연결에서, 함수 유형 목록에서 함수를 실행할 시점, 즉 최종 사용자 요청, 최종 사용자 응답, 오리진 요청 또는 오리진 응답 중에 Lambda@Edge를 선택합니다.

   자세한 내용은 함수를 트리거할 이벤트를 선택합니다. 섹션을 참조하세요.
7. 함수 ARN/이름 텍스트 상자에 선택한 이벤트가 발생할 때 실행하려는 Lambda 함수의 ARN을 붙여넣습니다. Lambda 콘솔에서 복사한 값입니다.
8. 함수의 요청 본문에 액세스하려는 경우 본문 포함을 선택합니다.

   요청 본문을 바꾸기만 하려는 경우에는 이 옵션을 선택할 필요가 없습니다.
9. 더 많은 이벤트 유형에 대해 동일한 함수를 실행하려면 6 및 7단계를 반복합니다.
10. 변경 사항 저장을 선택합니다.
11. 이 배포에 대한 다른 캐시 동작에 트리거를 추가하려면 5~10단계를 반복합니다.

    업데이트된 CloudFront 배포가 배포되면 함수가 지정된 CloudFront 이벤트에 대한 요청을 처리하기 시작합니다. 배포가 배포되었는지 확인하려면 탐색 창에서 배포를 선택합니다. 배포판이 배포되면 배포에 대한 상태 열의 값이 배포 중에서 배포 시간 및 날짜로 변경됩니다.

# Lambda@Edge 함수 테스트 및 디버깅

의도한 작업을 완료하는지 확인하기 위해서는 Lambda@Edge 함수 코드를 단독으로 테스트해야 하고, 함수가 CloudFront와 함께 잘 작동하는지 확인하기 위해서는 통합 테스트를 수행해야 합니다.

통합 테스트 중 또는 함수 배포 완료 후 HTTP 5xx 오류 등과 같은 CloudFront 오류를 디버깅해야 할 수 있습니다. Lambda 함수에서 반환되는 잘못된 응답, 함수 실행 시 발생하는 실행 오류 또는 Lambda 서비스의 실행 조절로 인한 오류 등이 있을 수 있습니다. 이 주제의 단원에서는 이러한 문제가 어떤 유형의 실패인지 확인한 다음 각 문제를 해결하기 위해 수행할 수 있는 단계에 대해 설명합니다.

> **Note**
>
> 오류를 해결하기 위해 CloudWatch 로그 파일 또는 지표를 검토할 때 함수가 실행된 위치에 가장 가까운 AWS 리전에 로그 파일 또는 지표가 표시 또는 저장된다는 점에 유의하세요. 예를 들어, 영국 내 사용자가 있는 웹 사이트 또는 웹 애플리케이션이 있고 배포와 연결된 Lambda 함수가 있는 경우 런던 AWS 리전 리전에 대한 CloudWatch 지표 또는 로그 파일을 볼 수 있도록 리전을 변경해야 합니다. 자세한 내용은 the section called “Lambda@Edge 리전 확인” 섹션을 참조하세요.

주제


- Lambda@Edge 함수 테스트
- CloudFront에서 Lambda@Edge 함수 오류 식별
- 잘못된 Lambda@Edge 함수 응답(검증 오류) 문제 해결
- Lambda@Edge 함수 실행 오류 문제 해결
- Lambda@Edge 리전 확인
- 계정이 CloudWatch로 로그를 푸시하는지 확인


## Lambda@Edge 함수 테스트

Lambda 함수 테스트에는 독립 실행형 테스트와 통합 테스트, 이렇게 두 가지 단계가 있습니다.

### 독립 실행형 기능 테스트

CloudFront에 Lambda 함수를 추가하기 전에 Lambda 콘솔의 테스트 기능을 사용하거나 다른 방법을 사용하여 기능을 먼저 테스트해야 합니다. Lambda 콘솔 테스트에 대한 자세한 내용은 AWS Lambda 개발자 안내서에서 콘솔을 사용하여 Lambda 함수 호출을 참조하세요.

### CloudFront에서 함수의 작동 상태를 보기 위한 테스트

함수가 배포와 연결되어 있고 CloudFront 이벤트를 기반으로 실행되는 경우에는 통합 테스트를 반드시 완료해야 합니다. 함수가 올바른 이벤트 발생 시 실행되고 CloudFront에 대해 올바른 응답을 반환하는지 확인합니다. 예를 들어, 이벤트 구조가 올바른지, 유효한 헤더만 포함되어 있는지 등을 확인합니다.

Lambda 콘솔에서 함수의 통합 테스트를 반복하는 것처럼 코드를 수정하거나 함수를 호출하는 CloudFront 트리거를 변경할 때 Lambda@Edge 자습서의 단계를 참조하세요. 예를 들어, 본 자습서의 단계(4단계: 함수를 실행할 CloudFront 트리거 추가)에서 설명하는 것처럼 다양한 버전의 함수에서 작동하는지 확인합니다.

함수를 변경하고 배포할 때 업데이트된 함수 및 CloudFront 트리거가 모든 리전에서 복제되는 데 몇 분 정도 걸릴 수 있습니다. 일반적으로 몇 분이면 되지만 최대 15분까지 걸릴 수 있습니다.

CloudFront 콘솔로 이동하여 해당 배포를 보고 복제가 완료되었는지 확인할 수 있습니다.

복제본 배포가 완료되었는지 확인하려면

1. https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 배포 이름을 선택합니다.
3. 배포 상태가 진행 중에서 다시 배포 완료로 바뀌었는지 확인합니다. 이것은 함수가 복제되었다는 의미입니다. 이제 다음 단원의 단계에 따라 함수가 작동하는지 확인합니다.

콘솔의 테스트는 함수의 로직만 확인하며, Lambda@Edge에 고유한 모든 서비스 할당량(이전에는 제한이라고 함)은 적용하지 않습니다.

## CloudFront에서 Lambda@Edge 함수 오류 식별

함수 로직이 올바르게 작동하는지 확인한 후 CloudFront에서 함수 실행 시 HTTP 5xx 오류가 나타날 수 있습니다. HTTP 5xx 오류는 여러 가지 이유로 반환될 수 있습니다. 여기에는 Lambda 함수 오류 또는 CloudFront의 다른 문제가 포함될 수 있습니다.

- Lambda@Edge 함수를 사용하는 경우 CloudFront 콘솔의 그래프를 사용하여 오류의 원인을 추적하고 문제를 해결할 수 있습니다. 예를 들어, HTTP 5xx 오류가 CloudFront 또는 Lambda 함수에 의해 발생한 것인지를 확인한 다음 특정 함수에 대해 관련 로그 파일을 검토하여 문제를 조사할 수 있습니다.

- CloudFront에서 일반적인 HTTP 오류 문제를 해결하려면 CloudFront의 오류 응답 상태 코드 문제 해결 주제의 문제 해결 단계를 참조하세요.

### CloudFront에서 Lambda@Edge 함수 오류를 일으키는 원인

Lambda 함수가 HTTP 5xx 오류를 일으키는 원인에는 여러 가지가 있으며, 수행해야 할 문제 해결 단계는 오류 유형에 따라 달라집니다. 오류는 다음과 같이 분류할 수 있습니다.

Lambda 함수 실행 오류

함수에 처리되지 않은 예외가 있거나 코드에 오류가 있기 때문에 CloudFront가 Lambda에서 응답을 받지 못한 경우 실행 오류가 발생합니다. 예: 코드에 콜백이 포함된 경우(오류)

잘못된 Lambda 함수 응답이 CloudFront로 반환됨

함수 실행 후 CloudFront가 Lambda에서 응답을 수신했습니다. 응답의 객체 구조가 Lambda@Edge 이벤트 구조를 따르지 않거나 응답에 잘못된 헤더 또는 기타 잘못된 필드가 포함되어 있는 경우 오류가 반환됩니다.

Lambda 서비스 할당량(이전에는 제한이라고 함)으로 인해 CloudFront에서의 실행이 제한됨

Lambda 서비스는 각 리전에서 실행을 조절하는데 할당량을 초과하면 오류를 반환합니다. 자세한 내용은 Lambda@Edge에 대한 할당량 섹션을 참조하세요.

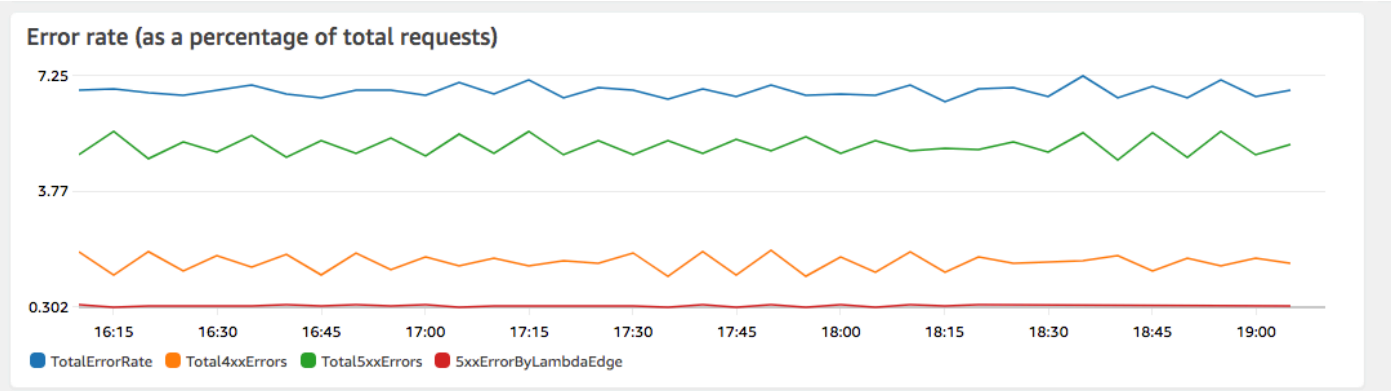
### 실패 유형을 결정하는 방법

디버깅할 때 집중할 위치를 결정하고 CloudFront에서 반환한 오류를 해결하려면 CloudFront에서 HTTP 오류가 반환된 이유를 파악하는 것이 좋습니다. 시작하려면 AWS Management Console의 CloudFront 콘솔에 있는 모니터링 섹션에 제공된 그래프를 사용합니다. CloudFront 콘솔의 모니터링(Monitoring) 섹션에서 그래프를 보는 방법에 대한 자세한 내용은 Amazon CloudWatch를 사용한 CloudFront 지표 모니터링 단원을 참조하세요.

다음 그래프는 오리진 또는 Lambda 함수로 오류를 반환하는지 여부를 추적하고 Lambda 함수의 오류일 경우 문제 유형의 범위를 좁히려 할 때 특히 유용합니다.

오류 발생율 그래프

각 배포의 개요 탭에서 볼 수 있는 그래프 중에는 오류 발생율 그래프가 있습니다. 이 그래프는 배포에 들어오는 총 요청의 백분율로 오류 발생율을 표시합니다. 그래프는 총 오류 발생율, 총 4xx 오류, 총 5xx 오류 및 Lambda 함수의 총 5xx 오류를 보여줍니다. 오류 유형 및 볼륨에 따라 원인을 조사하고 문제를 해결할 수 있는 단계를 수행할 수 있습니다.




- Lambda 오류가 표시되면 함수가 반환하는 특정 유형의 오류를 확인하여 추가 조사를 수행할 수 있습니다. Lambda@Edge errors(Lambda@Edge 오류) 탭에는 함수 오류를 유형별로 분류하여 특정 함수에 대한 문제를 찾아낼 수 있는 그래프가 포함되어 있습니다.
- CloudFront 오류가 표시되면 문제를 해결하고 오리진 오류를 수정하거나 CloudFront 구성을 변경할 수 있습니다. 자세한 내용은 CloudFront의 오류 응답 상태 코드 문제 해결 단원을 참조하세요.

실행 오류 및 잘못된 함수 응답 그래프

Lambda@Edge errors(Lambda@Edge 오류) 탭에는 특정 배포에 대한 Lambda@Edge 오류를 유형별로 분류하는 그래프가 포함되어 있습니다. 예를 들어, 한 그래프는 모든 실행 오류를 AWS 리전별로 표시합니다.

문제를 쉽게 해결할 수 있도록 특정 함수에 대한 로그 파일을 열고 리전별로 검토하여 특정 문제를 찾을 수 있습니다.

리전별로 특정 함수의 로그 파일을 보려면

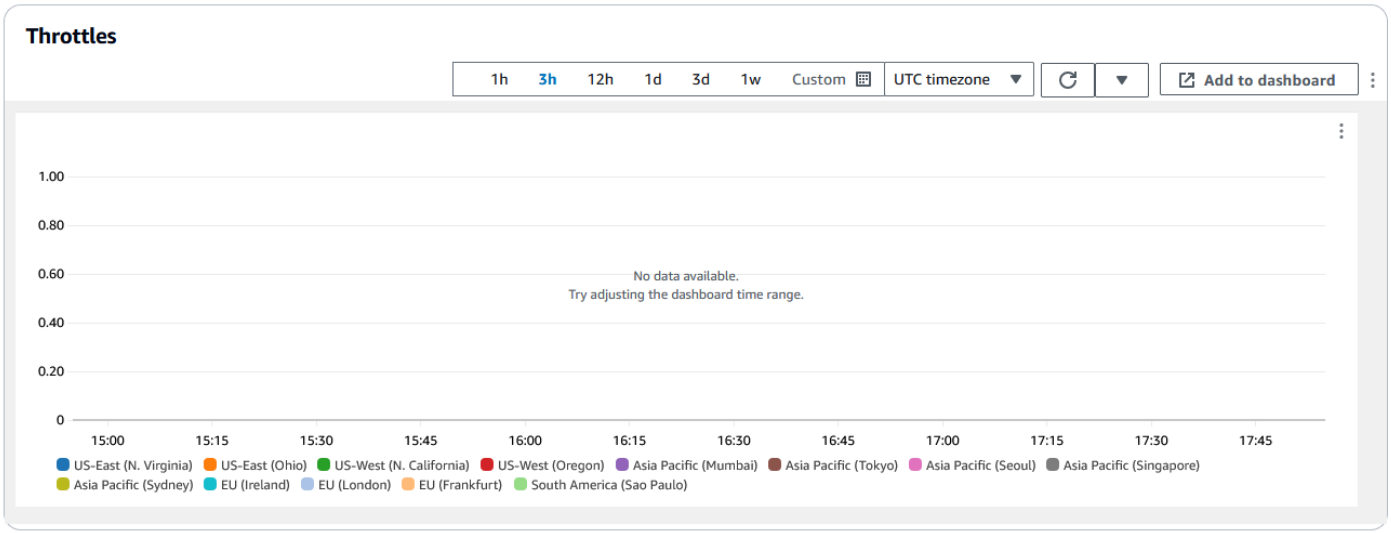
1. Lambda@Edge 오류 탭의 관련 Lambda@Edge 함수에서 함수 이름을 선택한 다음 지표 보기를 선택합니다.
2. 그런 다음, 함수 이름이 있는 페이지의 오른쪽 상단 모서리에서 함수 로그 보기를 선택한 다음 리전을 선택합니다.

   예를 들어 미국 서부(오레곤) 리전에 대한 오류 그래프에 문제가 있는 경우 드롭다운 목록에서 해당 리전을 선택합니다. 그러면 Amazon CloudWatch 콘솔이 열립니다.
3. 해당 리전의 CloudWatch 콘솔에서 로그 스트림 아래의 로그 스트림을 선택하여 함수에 대한 이벤트를 확인합니다.

또한 이 장의 다음 단원에서 오류 해결 및 수정에 대한 추가 권장 사항을 읽으십시오.

### 제한 그래프

Lambda@Edge errors(Lambda@Edge 오류) 탭에는 제한 그래프도 있습니다. 경우에 따라 리전 동시성 할당량(이전에는 제한이라고 함)에 도달하면 Lambda 서비스가 리전별로 함수 호출을 제한합니다. 제한 초과 오류가 발생한 경우는 리전에서의 실행에 대해 Lambda 서비스가 부과한 할당량에 도달한 것입니다. 이러한 할당량을 늘리기 위한 요청 방법을 비롯한 자세한 내용은 Lambda@Edge에 대한 할당량 단원을 참조하세요.




이 정보를 사용하여 HTTP 오류 문제를 해결하는 방법에 대한 예제는 AWS에서 콘텐츠 전송을 디버깅하기 위한 네 가지 단계를 참조하세요.

## 잘못된 Lambda@Edge 함수 응답(검증 오류) 문제 해결

Lambda 확인 오류가 문제라고 파악한 경우는 Lambda 함수가 CloudFront에 잘못된 응답을 반환하고 있다는 의미입니다. 이 단원의 지침에 따라 함수를 검토하는 단계를 수행한 다음 응답이 CloudFront 요구 사항을 따르는지 확인하세요.

CloudFront는 다음 두 가지 방법으로 Lambda 함수의 응답을 확인합니다.

- Lambda 응답은 필수 객체 구조를 따라야 합니다. 잘못된 객체 구조의 예에는 구문 분석이 불가능한 JSON, 필수 필드 누락 및 응답에 포함된 잘못된 객체 등이 있습니다. 자세한 내용은 Lambda@Edge 이벤트 구조 단원을 참조하세요.
- 응답에는 올바른 객체만 포함되어 있어야 합니다. 응답에 올바른 객체가 포함되어 있는데 그 값이 지원되지 않는 경우에는 오류가 발생합니다. 등록된 헤더 또는 읽기 전용 헤더를 추가 또는 업데이트하는 경우(엣지 함수에 대한 제한 사항 참조), 최대 본문 크기 초과(Lambda@Edge 오류 주제에서 생성

된 응답 크기 제한 참조) 및 잘못된 문자 또는 값(Lambda@Edge 이벤트 구조 참조) 등을 예로 들 수 있습니다.

Lambda가 CloudFront에 대해 유효하지 않은 응답을 반환하면 CloudFront가 Lambda 함수가 실행된 리전의 CloudWatch에 푸시하는 로그 파일에 오류 메시지가 기록됩니다. 잘못된 응답이 있는 경우 CloudWatch로 로그 파일을 보내는 것은 기본 동작입니다. 그러나 Lambda 함수를 릴리스하기 전에 CloudFront와 연결한 경우에는 함수에 대해 활성화되지 않을 수 있습니다. 자세한 내용은 이 주제 뒷부분에 나오는 계정이 CloudWatch로 로그를 푸시하는지 확인을 참조하세요.

CloudFront는 배포와 연결된 로그 그룹 내에서 함수가 실행되는 위치에 해당하는 리전으로 로그 파일을 푸시합니다. 로그 그룹의 형식은 `/aws/cloudfront/LambdaEdge/`*`DistributionId`*인데, 여기서 *`DistributionId`*는 배포 ID입니다. CloudWatch 로그 파일을 찾을 수 있는 리전을 확인하려면 이 주제 뒷부분에 나오는 Lambda@Edge 리전 확인을 참조하세요.

오류를 재현할 수 있는 경우 오류가 발생하는 새로운 요청을 생성한 다음 실패한 CloudFront 응답(`X-Amz-Cf-Id` 헤더)에서 해당 요청 ID를 찾아 로그 파일에서 단일 실패를 확인할 수 있습니다. 로그 파일 항목에는 오류가 반환된 이유를 파악하는 데 도움이 되는 정보가 포함되어 있고 해당하는 Lambda 요청 ID가 나열되어 있기 때문에 단일 요청의 컨텍스트 내에서 근본 원인을 분석할 수 있습니다.

오류가 간헐적으로 발생하는 경우에는 CloudFront 액세스 로그를 사용하여 실패한 요청의 요청 ID를 찾은 다음 CloudWatch 로그에서 해당하는 오류 메시지를 검색할 수 있습니다. 자세한 내용은 앞의 실패 유형 확인 단원을 참조하십시오.

## Lambda@Edge 함수 실행 오류 문제 해결

Lambda 실행 오류가 문제인 경우에는 Lambda 함수에 대한 로깅 문을 생성해 CloudWatch 로그 파일에 CloudFront에서 함수의 실행을 모니터링하는 메시지를 작성한 다음 예상대로 작동하는지 확인하면 도움이 될 수 있습니다. 그런 다음 CloudWatch 로그 파일에서 해당 문을 검색해 함수가 작동 중인지 확인할 수 있습니다.

> **Note**
>
> Lambda@Edge 함수를 변경하지 않은 경우에도 Lambda 함수 실행 환경에 대한 업데이트가 영향을 줄 수 있으며, 실행 오류가 반환될 수 있습니다. 테스트 및 이후 버전으로의 마이그레이션에 대한 자세한 내용은 AWS Lambda 및 AWS Lambda@Edge 실행 환경에 대해 예정된 업데이트를 참조하세요.

### Lambda@Edge 리전 확인

Lambda@Edge 함수가 트래픽을 수신하는 리전을 보려면 에서 CloudFront 콘솔의 함수에 대한 지표를 확인하세요AWS Management Console 지표는 각 AWS 리전별로 표시됩니다. 동일한 페이지에서 리전을 선택하고 해당 리전의 로그 파일을 확인하여 문제를 조사할 수 있습니다. CloudFront에서 Lambda 함수를 실행할 때 생성된 로그 파일을 확인하려면 올바른 AWS 리전의 CloudWatch Logs 파일을 검토해야 합니다.

CloudFront 콘솔의 모니터링(Monitoring) 섹션에서 그래프를 보는 방법에 대한 자세한 내용은 Amazon CloudWatch를 사용한 CloudFront 지표 모니터링 단원을 참조하세요.

### 계정이 CloudWatch로 로그를 푸시하는지 확인

기본적으로 CloudFront는 잘못된 Lambda 함수 응답 로깅을 활성화하고 Lambda@Edge의 서비스 연결 역할 중 하나를 수행해 로그 파일을 CloudWatch로 푸시합니다. 잘못된 Lambda 함수 응답 로그 기능이 활성화되기 전에 CloudFront에 Lambda@Edge 함수를 추가한 경우에는 예를 들어, CloudFront 트리거 추가 등과 같이 다음에 Lambda@Edge 구성을 업데이트하는 경우 로깅이 활성화됩니다.

다음을 수행해 계정에 대해 CloudWatch로 로그 파일 푸시 기능이 활성화되어 있는지 확인할 수 있습니다.

- CloudWatch에 로그가 나타나는지 확인 - Lambda @Edge 함수가 실행된 리전에서 로그 파일을 확인해야 합니다. 자세한 내용은 Lambda@Edge 리전 확인 섹션을 참조하세요.
- IAM의 계정에 관련 서비스 연결 역할이 있는지 확인 - 계정에 IAM 역할 `AWSServiceRoleForCloudFrontLogger`가 있어야 합니다. 이에 대한 자세한 내용은 Lambda@Edge의 서비스 연결 역할 섹션을 참조하세요.

## Lambda@Edge 함수 및 복제본 삭제

CloudFront에서 함수의 복제본이 삭제된 경우에만 Lambda@Edge 함수를 삭제할 수 있습니다. 다음과 같은 상황에서 Lambda 함수 복제본이 자동으로 삭제됩니다.

- 함수의 복제본은 함수에 대한 마지막 연결을 모든 CloudFront 배포에서 제거한 후에 자동으로 삭제됩니다. 둘 이상의 배포에서 함수를 사용할 경우 마지막 배포에서 함수 연결이 제거된 후에만 복제본이 삭제됩니다.
- 함수에 연결된 마지막 배포를 삭제한 후에도 복제본이 제거됩니다.

복제본은 일반적으로 몇 시간 내에 삭제됩니다. Lambda@Edge 함수 복제본은 수동으로 삭제할 수 없습니다. 따라서 아직 사용 중인 복제본이 삭제되어 오류가 발생하는 상황을 방지할 수 있습니다.

> **Warning**
>
> CloudFront 외부에서 Lambda@Edge 함수 복제본을 사용하는 애플리케이션을 빌드하지 마세요. 이러한 복제본은 배포와의 연결이 제거되거나 배포 자체가 삭제될 때 삭제됩니다. 이 경우 외부 애플리케이션에서 사용하는 복제본이 경고도 없이 제거되어 오류가 발생할 수 있습니다.

CloudFront 배포에서 Lambda@Edge 함수 연결을 삭제하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 삭제할 Lambda@Edge 함수 연결이 있는 배포의 ID를 선택합니다.
3. 동작 탭을 선택합니다.
4. 삭제할 Lambda@Edge 함수 연결이 있는 캐시 동작을 선택한 다음 편집을 선택합니다.
5. 함수 연결, 함수 유형에서, 연결 없음을 선택하여 Lambda@Edge 함수 연결을 삭제합니다.
6. 변경 사항 저장를 선택합니다.

CloudFront 배포에서 Lambda@Edge 함수 연결을 삭제한 후 선택적으로 에서 Lambda 함수 또는 함수 버전을 삭제할 수 있습니다AWS Lambda Lambda@Edge 함수 복제본을 정리할 수 있도록 함수 연결을 삭제한 후 몇 시간 정도 기다립니다. 그런 다음 Lambda 콘솔, AWS CLI, Lambda API 또는 AWS SDK를 사용하여 함수를 삭제할 수 있습니다.

버전에 연결된 CloudFront 배포가 없는 경우에는 특정 버전의 Lambda 함수를 삭제할 수도 있습니다. Lambda 함수 버전에서 모든 연결을 제거한 뒤 몇 시간 정도 기다려 주세요. 그러면 해당 함수 버전을 삭제할 수 있습니다.

## Lambda@Edge 이벤트 구조

다음 주제에서는 트리거될 때 CloudFront가 Lambda@Edge 함수에 전달하는 요청 및 응답 이벤트 객체에 대해 설명합니다.

주제


- 동적 원본 선택
- 이벤트 요청

- 응답 이벤트

## 동적 원본 선택

캐시 동작의 경로 패턴을 사용하여 `images/*.jpg`와 같이 요청된 객체의 경로와 이름을 기반으로 오리진에 요청을 라우팅할 수 있습니다. 또한 Lambda@Edge를 사용하여 요청 헤더의 값과 같은 기타 특성을 기반으로 오리진에 요청을 라우팅할 수 있습니다.

여러 경우에 이러한 동적 오리진 선택이 유용할 수 있습니다. 예를 들어, 각기 다른 지리 영역에 있는 오리진에 걸쳐 요청을 배포하여 전역 로드 밸런싱에 도움이 될 수 있습니다. 또는 봇 처리, SEO 최적화, 인증 등과 같은 특정 기능을 서비스하는 각기 다른 오리진에 요청을 선택적으로 라우팅할 수 있습니다. 이 기능을 사용하는 방법을 보여주는 코드 예제는 콘텐츠 기반 동적 원본 선택 - 예제 단원을 참조하세요.

CloudFront 오리진 요청 이벤트에서 이벤트 구조의 오리진 `origin` 객체에는 요청이 경로 패턴을 기반으로 라우팅될 오리진에 대한 정보가 들어 있습니다. `origin` 객체의 값을 업데이트하여 요청을 다른 오리진으로 라우팅할 수 있습니다. `origin` 객체를 업데이트할 때 배포에서 오리진을 정의할 필요가 없습니다. 또한 Amazon S3 오리진 객체를 사용자 지정 오리진 객체로 바꿀 수 있으며, 그 반대의 경우도 마찬가지입니다. 요청당 하나의 오리진만 지정할 수 있습니다. 즉, 사용자 지정 오리진 또는 Amazon S3 오리진 중 하나를 지정할 수 있지만 둘 모두 지정할 수는 없습니다.

## 이벤트 요청

다음 주제에서는 CloudFront가 최종 사용자 및 오리진 요청 이벤트에 대한 Lambda 함수에 전달하는 객체의 구조를 보여줍니다. 이 예제에서는 본문이 없는 `GET` 요청을 보여줍니다. 다음 예제는 최종 사용자 및 오리진 요청 이벤트에서 가능한 모든 필드의 목록입니다.

주제

- 뷰어 요청 예제
- 원본 요청 예제
- 요청 이벤트 필드

### 뷰어 요청 예제

다음 예제에서는 최종 사용자 요청 이벤트 객체를 보여줍니다.

```
{
  "Records": [
    {
```

```
      "cf": {
        "config": {
          "distributionDomainName": "d111111abcdef8.cloudfront.net",
          "distributionId": "EDFDVBD6EXAMPLE",
          "eventType": "viewer-request",
          "requestId": "4TyzHTaYWb1GX1qTfsHhEqV6HUDd_BzoBZnwfnvQc_1oF26ClkoUSEQ=="
        },
        "request": {
          "clientIp": "203.0.113.178",
          "headers": {
            "host": [
              {
                "key": "Host",
                "value": "d111111abcdef8.cloudfront.net"
              }
            ],
            "user-agent": [
              {
                "key": "User-Agent",
                "value": "curl/7.66.0"
              }
            ],
            "accept": [
              {
                "key": "accept",
                "value": "*/*"
              }
            ]
          },
          "method": "GET",
          "querystring": "",
          "uri": "/"
        }
      }
    }
  ]
}
```

## 원본 요청 예제

다음 예제에서는 오리진 요청 이벤트 객체를 보여줍니다.

```
{
  "Records": [
```

```
    {
      "cf": {
        "config": {
          "distributionDomainName": "d111111abcdef8.cloudfront.net",
          "distributionId": "EDFDVBD6EXAMPLE",
          "eventType": "origin-request",
          "requestId": "4TyzHTaYWb1GX1qTfsHhEqV6HUDd_BzoBZnwfnvQc_1oF26ClkoUSEQ=="
        },
        "request": {
          "clientIp": "203.0.113.178",
          "headers": {
            "x-forwarded-for": [
              {
                "key": "X-Forwarded-For",
                "value": "203.0.113.178"
              }
            ],
            "user-agent": [
              {
                "key": "User-Agent",
                "value": "Amazon CloudFront"
              }
            ],
            "via": [
              {
                "key": "Via",
                "value": "2.0 2afae0d44e2540f472c0635ab62c232b.cloudfront.net
(CloudFront)"
              }
            ],
            "host": [
              {
                "key": "Host",
                "value": "example.org"
              }
            ],
            "cache-control": [
              {
                "key": "Cache-Control",
                "value": "no-cache"
              }
            ]
          },
          "method": "GET",
```

```
          "origin": {
            "custom": {
              "customHeaders": {},
              "domainName": "example.org",
              "keepaliveTimeout": 5,
              "path": "",
              "port": 443,
              "protocol": "https",
              "readTimeout": 30,
              "responseCompletionTimeout": 30,
              "sslProtocols": [
                "TLSv1",
                "TLSv1.1",
                "TLSv1.2"
              ]
            }
          },
          "querystring": "",
          "uri": "/"
        }
      }
    }
  ]
}
```

요청 이벤트 필드

요청 이벤트 객체 데이터는 두 개의 하위 객체(`config`(`Records.cf.config`) 및 `request`(`Records.cf.request`))에 포함되어 있습니다. 다음 목록에서는 각 하위 객체의 필드에 대해 설명합니다.

구성 객체의 필드

다음 표에서는 `config` 객체(`Records.cf.config`)의 필드에 대해 설명합니다.

**`distributionDomainName`**(읽기 전용)

요청과 연결된 배포의 도메인 이름입니다.

**`distributionID`**(읽기 전용)

요청과 관련이 있는 배포의 ID입니다.

**eventType**(읽기 전용)

요청(`viewer-request` 또는 `origin-request`)과 관련된 트리거의 유형입니다.

**requestId**(읽기 전용)

최종 사용자 요청/CloudFront 요청을 고유하게 식별하는 암호화된 문자열입니다. 또한 `requestId` 값은 CloudFront 액세스 로그에 `x-edge-request-id`로 표시됩니다. 자세한 내용은 액세스 로그 (표준 로그) 및 로그 파일 필드 단원을 참조하세요.

요청 객체의 필드

다음 표에서는 `request` 객체(`Records.cf.request`)의 필드에 대해 설명합니다.

**clientIp**(읽기 전용)

요청을 생성한 최종 사용자의 IP 주소입니다. 최종 사용자가 HTTP 프록시 또는 로드 밸런서를 사용하여 요청을 전송한 경우 이 값은 프록시 또는 로드 밸런서의 IP 주소입니다.

headers(읽기/쓰기)

요청의 헤더입니다. 다음을 참조하십시오.

- `headers` 객체 내 키는 표준 HTTP 헤더 이름의 소문자 버전입니다. 소문자 키를 사용하여 대/소문자를 구분하지 않고 헤더 값에 액세스할 수 있습니다.
- 각 헤더 객체(예: `headers["accept"]` 또는 `headers["host"]`)는 키–값 페어의 어레이입니다. 해당 헤더에 대해 어레이에는 요청의 각 값에 대한 하나의 키–값 페어를 포함합니다.
- `key`에는 HTTP 요청에 표시되는 헤더의 이름(대/소문자 구분)이 포함됩니다(예: `Host`, `User-Agent`, `X-Forwarded-For`, `Cookie` 등).
- `value`에는 HTTP 요청에 표시되는 헤더 값을 포함합니다.
- Lambda 함수가 요청 헤더를 추가하거나 수정하고 사용자가 헤더 `key` 필드를 포함시키지 않으면, Lambda@Edge는 사용자가 입력한 헤더 이름을 사용하여 헤더 `key`를 자동으로 삽입합니다. 헤더 이름의 형식을 어떻게 지정했든 상관없이, 자동으로 삽입된 헤더 키는 각 부분의 첫 문자가 대문자로 지정되고 각 부분이 하이픈(-)으로 구분됩니다.

  예를 들어 헤더를 다음과 같이 헤더 `key` 없이 추가할 수 있습니다.

```
"user-agent": [
  {
    "value": "ExampleCustomUserAgent/1.X.0"
```

```
  }
]
```

이 예제에서는 Lambda@Edge가 자동으로 `"key": "User-Agent"`을 삽입합니다.

헤더 사용 시 제한 사항에 대한 자세한 내용은 엣지 함수에 대한 제한 사항 단원을 참조하십시오.

**method**(읽기 전용)

요청의 HTTP 메서드.

**querystring**(읽기/쓰기)

요청의 쿼리 문자열입니다(있는 경우). 요청에 쿼리 문자열이 포함되지 않더라도 이벤트 객체에는 여전히 빈 값과 함께 `querystring`이 포함됩니다. 쿼리 문자열에 대한 자세한 내용은 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱 단원을 참조하십시오.

**uri**(읽기/쓰기)

요청된 객체의 상대 경로입니다. Lambda 함수가 `uri` 값을 수정하는 경우 다음 사항에 유의하십시오.

- 새 `uri` 값은 슬래시(/) 로 시작해야 합니다.
- 함수가 `uri` 값을 변경하는 경우 이로 인해 최종 사용자가 요청 중인 객체가 변경됩니다.
- 함수가 `uri` 값을 변경하는 경우 이로 인해 요청 또는 요청이 전송되는 오리진에 대한 캐시 동작이 변경되지 않습니다.

**body**(읽기/쓰기)

HTTP 요청의 본문입니다. body 구조에는 다음과 같은 필드가 포함될 수 있습니다.

**inputTruncated**(읽기 전용)

Lambda@Edge가 본문을 잘랐는지 여부를 나타내는 부울 플래그. 자세한 내용은 본문 포함 옵션이 적용된 요청 본문에 대한 제한 사항 단원을 참조하세요.

**action**(읽기/쓰기)

본문을 사용해 수행하고자 한 작업. `action`의 옵션은 다음과 같습니다.

- `read-only`: 이 값이 기본값입니다. Lambda 함수에서 응답이 반환되었는데 `action`이 read-only이면, Lambda@Edge는 `encoding` 또는 `data`에 대한 모든 변경 사항을 무시합니다.
- `replace`: 오리진으로 전송된 본문을 바꾸려는 경우 이 옵션을 지정합니다.

**encoding**(읽기/쓰기)

본문에 대한 인코딩. Lambda@Edge가 Lambda 함수에 본문을 노출시키는 경우 먼저 본문을 base64-encoding으로 변환합니다. 본문을 바꾸기 위해 `replace`에 `action`를 선택한 경우 `base64`(기본값) 또는 `text` 인코딩을 사용하도록 선택할 수 있습니다. `encoding`을 `base64`로 지정했지만 본문이 유효한 base64가 아닌 경우 CloudFront는 오류를 반환합니다.

**data**(읽기/쓰기)

요청 본문의 내용

**origin**(읽기/쓰기) (오리진 이벤트만 해당)

요청을 전송할 오리진입니다. `origin` 구조에는 정확히 하나의 오리진만 포함되어야 합니다. 즉, 오리진은 사용자 지정 오리진 또는 Amazon S3 오리진일 수 있습니다.

지정하는 오리진 유형(사용자 지정 또는 Amazon S3 오리진)에 따라 요청에 다음 필드를 지정해야 합니다.

**customHeaders**(읽기/쓰기)(사용자 지정 및 Amazon S3 오리진)

(선택 사항) 각 사용자 지정 헤더에 대한 헤더 이름 및 값 페어를 지정하여 요청을 포함한 사용자 지정 헤더를 포함할 수 있습니다. 허용되지 않는 헤더는 추가할 수 없으며 동일한 이름의 헤더가 `Records.cf.request.headers`에 존재할 수 없습니다. 요청 헤더에 대한 노트는 사용자 지정 헤더에도 적용됩니다. 자세한 내용은 CloudFront에서 오리진 요청에 추가할 수 없는 사용자 지정 헤더 및 엣지 함수에 대한 제한 사항 단원을 참조하세요.

**domainName**(읽기/쓰기)(사용자 지정 및 Amazon S3 오리진)

오리진의 도메인 이름입니다. 도메인 이름은 비워둘 수 없습니다.

- 사용자 지정 오리진의 경우 – DNS 도메인 이름을 지정합니다(예: `www.example.com`). 도메인 이름에는 콜론(:)이 포함될 수 없으며 IP 주소가 될 수 없습니다. 도메인 이름은 최대 253자일 수 있습니다.
- Amazon S3 오리진의 경우 – Amazon S3 버킷의 DNS 도메인 이름을 지정합니다(예: `amzn-s3-demo-bucket.s3.eu-west-1.amazonaws.com`). 이름은 최대 128자여야 하며, 모두 소문자여야 합니다.

**path**(읽기/쓰기)(사용자 지정 및 Amazon S3 오리진)

요청이 콘텐츠를 찾아야 하는 오리진의 디렉터리 경로입니다. 경로는 슬래시(/)로 시작해야 하지만 슬래시 하나로 끝나지 않아야 합니다(예: `example-path/`와 같이 끝나면 안 됨). 사용자 지정 오리진의 경우만, 경로는 URL로 인코딩되어야 하고 최대 길이는 255자여야 합니다.

**keepaliveTimeout**(읽기/쓰기) (사용자 지정 오리진만 해당)

CloudFront가 응답의 마지막 패킷을 수신한 후 오리진 연결을 유지하기 위해 시도해야 하는 시간(초)입니다. 값은 1~120(경계값 포함) 사이의 숫자여야 합니다.

**port**(읽기/쓰기) (사용자 지정 오리진만 해당)

CloudFront가 사용자 지정 오리진에서 연결해야 하는 포트입니다. 포트는 80, 443 또는 1024~65535(경계값 포함) 사이의 숫자여야 합니다.

**protocol**(읽기/쓰기) (사용자 지정 오리진만 해당)

오리진에 연결할 때 CloudFront가 사용해야 하는 연결 프로토콜입니다. `http` 또는 `https` 값을 가질 수 있습니다.

**readTimeout**(읽기/쓰기)(사용자 지정 및 Amazon S3 오리진)

요청을 오리진으로 전송한 후 CloudFront가 응답을 대기해야 하는 시간(초)입니다. 또한 다음 패킷을 수신하기 전에 응답 패킷을 수신한 후 CloudFront가 대기해야 하는 시간도 지정합니다. 값은 1~120(경계값 포함) 사이의 숫자여야 합니다.

더 높은 할당량이 필요한 경우 오리진당 응답 제한 시간을 참조하세요.

**responseCompletionTimeout**(읽기/쓰기)(사용자 지정 및 Amazon S3 오리진)

CloudFront에서 오리진으로의 요청이 열려 있고 응답을 기다릴 수 있는 시간(초). 이 시간까지 오리진에서 완전한 응답을 받지 못하면 CloudFront는 연결을 종료합니다.

`responseCompletionTimeout` 값은 `readTimeout` 값과 같거나 이보다 커야 합니다. 이 값을 0으로 설정하면 이전에 설정한 모든 값이 제거되고 기본값으로 돌아갑니다. 이벤트 요청에서 `responseCompletionTimeout` 필드를 삭제하여 이 작업을 수행할 수도 있습니다.

**sslProtocols**(읽기/쓰기) (사용자 지정 오리진만 해당)

오리진과의 HTTPS 연결을 설정할 때 CloudFront가 사용할 수 있는 최소 SSL/TLS 프로토콜입니다. 값은 `TLSv1.2`, `TLSv1.1`, `TLSv1` 또는 `SSLv3` 중 하나일 수 있습니다.

**authMethod**(읽기/쓰기)(Amazon S3 오리진만 해당)

오리진 액세스 ID(OAI)를 사용하는 경우 이 필드를 `origin-access-identity`로 설정합니다. OAI를 사용하지 않는 경우 `none`으로 설정합니다. `authMethod`를 `origin-access-identity`로 설정하면 다음과 같은 몇 가지 요구 사항이 있습니다.

- `region`을 지정해야 합니다(다음 필드 참조).
- 하나의 Amazon S3 오리진에서 다른 오리진으로 요청을 변경할 때 동일한 OAI를 사용해야 합니다.

- 사용자 지정 오리진에서 Amazon S3 오리진으로 요청을 변경할 때 OAI를 사용할 수 없습니다.

> **Note**
>
> 이 필드는 오리진 액세스 제어(OAC)를 지원하지 않습니다.

**`region`**(읽기/쓰기)(Amazon S3 오리진만 해당)

Amazon S3 버킷의 AWS 리전입니다. 이 작업은 `authMethod`를 `origin-access-identity`로 설정한 경우에만 필요합니다.

## 응답 이벤트

다음 주제에서는 CloudFront가 최종 사용자 및 오리진 응답 이벤트에 대한 Lambda 함수에 전달하는 객체의 구조를 보여줍니다. 다음 예제는 최종 사용자 및 오리진 응답 이벤트에서 가능한 모든 필드의 목록입니다.

주제

- 원본 응답 예제
- 뷰어 응답 예제
- 응답 이벤트 필드

### 원본 응답 예제

다음 예제에서는 오리진 응답 이벤트 객체를 보여줍니다.

```
{
  "Records": [
    {
      "cf": {
        "config": {
          "distributionDomainName": "d111111abcdef8.cloudfront.net",
          "distributionId": "EDFDVBD6EXAMPLE",
          "eventType": "origin-response",
          "requestId": "4TyzHTaYWb1GX1qTfsHhEqV6HUDd_BzoBZnwfnvQc_1oF26ClkoUSEQ=="
        },
        "request": {
```

```
          "clientIp": "203.0.113.178",
          "headers": {
            "x-forwarded-for": [
              {
                "key": "X-Forwarded-For",
                "value": "203.0.113.178"
              }
            ],
            "user-agent": [
              {
                "key": "User-Agent",
                "value": "Amazon CloudFront"
              }
            ],
            "via": [
              {
                "key": "Via",
                "value": "2.0 8f22423015641505b8c857a37450d6c0.cloudfront.net
(CloudFront)"
              }
            ],
            "host": [
              {
                "key": "Host",
                "value": "example.org"
              }
            ],
            "cache-control": [
              {
                "key": "Cache-Control",
                "value": "no-cache"
              }
            ]
          },
          "method": "GET",
          "origin": {
            "custom": {
              "customHeaders": {},
              "domainName": "example.org",
              "keepaliveTimeout": 5,
              "path": "",
              "port": 443,
              "protocol": "https",
              "readTimeout": 30,
```

```
              "responseCompletionTimeout": 30,
              "sslProtocols": [
                "TLSv1",
                "TLSv1.1",
                "TLSv1.2"
              ]
            }
          },
          "querystring": "",
          "uri": "/"
        },
        "response": {
          "headers": {
            "access-control-allow-credentials": [
              {
                "key": "Access-Control-Allow-Credentials",
                "value": "true"
              }
            ],
            "access-control-allow-origin": [
              {
                "key": "Access-Control-Allow-Origin",
                "value": "*"
              }
            ],
            "date": [
              {
                "key": "Date",
                "value": "Mon, 13 Jan 2020 20:12:38 GMT"
              }
            ],
            "referrer-policy": [
              {
                "key": "Referrer-Policy",
                "value": "no-referrer-when-downgrade"
              }
            ],
            "server": [
              {
                "key": "Server",
                "value": "ExampleCustomOriginServer"
              }
            ],
            "x-content-type-options": [
```

```
              {
                "key": "X-Content-Type-Options",
                "value": "nosniff"
              }
            ],
            "x-frame-options": [
              {
                "key": "X-Frame-Options",
                "value": "DENY"
              }
            ],
            "x-xss-protection": [
              {
                "key": "X-XSS-Protection",
                "value": "1; mode=block"
              }
            ],
            "content-type": [
              {
                "key": "Content-Type",
                "value": "text/html; charset=utf-8"
              }
            ],
            "content-length": [
              {
                "key": "Content-Length",
                "value": "9593"
              }
            ]
          },
          "status": "200",
          "statusDescription": "OK"
        }
      }
    }
  ]
}
```

## 뷰어 응답 예제

다음 예제에서는 최종 사용자 응답 이벤트 객체를 보여줍니다.

```
{
  "Records": [
```

```
    {
      "cf": {
        "config": {
          "distributionDomainName": "d111111abcdef8.cloudfront.net",
          "distributionId": "EDFDVBD6EXAMPLE",
          "eventType": "viewer-response",
          "requestId": "4TyzHTaYWb1GX1qTfsHhEqV6HUDd_BzoBZnwfnvQc_1oF26ClkoUSEQ=="
        },
        "request": {
          "clientIp": "203.0.113.178",
          "headers": {
            "host": [
              {
                "key": "Host",
                "value": "d111111abcdef8.cloudfront.net"
              }
            ],
            "user-agent": [
              {
                "key": "User-Agent",
                "value": "curl/7.66.0"
              }
            ],
            "accept": [
              {
                "key": "accept",
                "value": "*/*"
              }
            ]
          },
          "method": "GET",
          "querystring": "",
          "uri": "/"
        },
        "response": {
          "headers": {
            "access-control-allow-credentials": [
              {
                "key": "Access-Control-Allow-Credentials",
                "value": "true"
              }
            ],
            "access-control-allow-origin": [
              {
```

```
                "key": "Access-Control-Allow-Origin",
                "value": "*"
              }
            ],
            "date": [
              {
                "key": "Date",
                "value": "Mon, 13 Jan 2020 20:14:56 GMT"
              }
            ],
            "referrer-policy": [
              {
                "key": "Referrer-Policy",
                "value": "no-referrer-when-downgrade"
              }
            ],
            "server": [
              {
                "key": "Server",
                "value": "ExampleCustomOriginServer"
              }
            ],
            "x-content-type-options": [
              {
                "key": "X-Content-Type-Options",
                "value": "nosniff"
              }
            ],
            "x-frame-options": [
              {
                "key": "X-Frame-Options",
                "value": "DENY"
              }
            ],
            "x-xss-protection": [
              {
                "key": "X-XSS-Protection",
                "value": "1; mode=block"
              }
            ],
            "age": [
              {
                "key": "Age",
                "value": "2402"
```

```
              }
            ],
            "content-type": [
              {
                "key": "Content-Type",
                "value": "text/html; charset=utf-8"
              }
            ],
            "content-length": [
              {
                "key": "Content-Length",
                "value": "9593"
              }
            ]
          },
          "status": "200",
          "statusDescription": "OK"
        }
      }
    }
  ]
}
```

응답 이벤트 필드

응답 이벤트 객체 데이터는 세 개의 하위 객체(`config`(`Records.cf.config`), `request`(`Records.cf.request`) 및 `response`(`Records.cf.response`))에 포함되어 있습니다. 요청 객체의 필드에 대한 자세한 내용은 요청 객체의 필드 단원을 참조하세요. 다음 목록에서는 `config` 및 `response` 하위 객체의 필드에 대해 설명합니다.

구성 객체의 필드

다음 표에서는 `config` 객체(`Records.cf.config`)의 필드에 대해 설명합니다.

**`distributionDomainName`**(읽기 전용)

응답과 연결된 배포의 도메인 이름입니다.

**`distributionID`**(읽기 전용)

응답과 관련이 있는 배포의 ID입니다.

**`eventType`**(읽기 전용)

응답과 관련된 트리거의 유형입니다(`origin-response` 또는 `viewer-response`).

**requestId**(읽기 전용)

이 응답이 연결된 최종 사용자 요청/CloudFront 요청을 고유하게 식별하는 암호화된 문자열입니다. 또한 `requestId` 값은 CloudFront 액세스 로그에 `x-edge-request-id`로 표시됩니다. 자세한 내용은 액세스 로그(표준 로그) 및 로그 파일 필드 단원을 참조하세요.

응답 객체의 필드

다음 표에서는 `response` 객체(`Records.cf.response`)의 필드에 대해 설명합니다. Lambda@Edge 함수를 사용하여 HTTP 응답을 생성하는 방법에 대한 자세한 내용은 요청 트리거에서 HTTP 응답 생성 단원을 참조하세요.

**headers**(읽기/쓰기)

응답의 헤더입니다. 다음을 참조하십시오.

- `headers` 객체 내 키는 표준 HTTP 헤더 이름의 소문자 버전입니다. 소문자 키를 사용하여 대/소문자를 구분하지 않고 헤더 값에 액세스할 수 있습니다.
- 각 헤더 객체(예: `headers["content-type"]` 또는 `headers["content-length"]`)는 키–값 페어의 어레이입니다. 해당 헤더에 대해 어레이에는 응답의 각 값에 대한 하나의 키–값 페어를 포함합니다.
- `key`에는 HTTP 응답에 표시되는 헤더의 대/소문자 구분 이름을 포함합니다(예: `Content-Type`, `Content-Length`, `Cookie`, 등).
- `value`에는 HTTP 응답에 표시되는 헤더 값을 포함합니다.
- Lambda 함수가 응답 헤더를 추가하거나 수정하고 사용자가 헤더 `key` 필드를 포함시키지 않으면, Lambda@Edge는 사용자가 입력한 헤더 이름을 사용하여 헤더 `key`를 자동으로 삽입합니다. 헤더 이름의 형식을 어떻게 지정했든 상관없이, 자동으로 삽입된 헤더 키는 각 부분의 첫 문자가 대문자로 지정되고 각 부분이 하이픈(-)으로 구분됩니다.

  예를 들어 헤더를 다음과 같이 헤더 `key` 없이 추가할 수 있습니다.

  ```
  "content-type": [
    {
      "value": "text/html;charset=UTF-8"
    }
  ]
  ```

  이 예제에서는 Lambda@Edge가 자동으로 `"key": "Content-Type"`을 삽입합니다.

헤더 사용 시 제한 사항에 대한 자세한 내용은 엣지 함수에 대한 제한 사항 단원을 참조하십시오.

**status**

응답의 HTTP 상태 코드입니다.

**statusDescription**

응답의 HTTP 상태 설명입니다.

## 요청 및 응답 작업 수행

Lambda@Edge 요청 및 응답을 사용하려면 다음 주제를 참조하세요.

주제

- 오리진 장애 조치와 함께 Lambda@Edge 함수 사용
- 요청 트리거에서 HTTP 응답 생성
- 오리진 응답 트리거에서 HTTP 응답 업데이트
- 본문 포함 옵션을 선택해 요청 본문에 액세스

### 오리진 장애 조치와 함께 Lambda@Edge 함수 사용

예를 들어 고가용성 확보를 위해 구성하는 오리진 장애 조치에 대하여 오리진 그룹과 함께 설정한 CloudFront 배포와 함께 Lambda@Edge 함수를 사용할 수 있습니다. 오리진 그룹과 함께 Lambda 함수를 사용하려면, 캐시 동작을 생성할 때 오리진 그룹에 대한 오리진 요청 또는 오리진 응답 트리거에 함수를 지정합니다.

자세한 내용은 다음 자료를 참조하세요.

- 오리진 그룹 만들기: 오리진 그룹 생성
- 오리진 장애 조치가 Lambda@Edge와 함께 작동하는 방식: Lambda@Edge 함수와 함께 오리진 장애 조치 사용

### 요청 트리거에서 HTTP 응답 생성

CloudFront 요청을 수신할 때 Lambda 함수를 사용하여 CloudFront가 오리진에 응답을 전달하지 않고 최종 사용자에게 직접 반환하는 HTTP 응답을 생성할 수 있습니다. HTTP 응답을 생성하면 오리진에 대한 로드가 감소하고 일반적으로 최종 사용자에 대한 지연 시간 또한 감소합니다.

HTTP 응답 생성의 몇 가지 일반 시나리오는 다음을 포함합니다.

- 최종 사용자에게 소규모 웹 페이지 반환
- HTTP 301 또는 302 상태 코드를 반환하여 다른 웹 페이지로 사용자를 리디렉션
- 사용자가 인증되지 않았을 때 최종 사용자에게 HTTP 401 상태 코드 반환

Lambda@Edge 함수는 다음 CloudFront 이벤트가 발생할 때 HTTP 응답을 생성할 수 있습니다.

최종 사용자 요청 이벤트

최종 사용자 요청 이벤트에 의해 함수가 트리거될 때 CloudFront는 최종 사용자에게 응답을 반환하고 이를 캐싱하지 않습니다.

오리진 요청 이벤트

오리진 요청 이벤트에 의해 함수가 트리거될 때 CloudFront는 함수에 의해 이전에 생성된 응답에 대한 엣지 캐시를 확인합니다.

- 응답이 캐시에 있는 경우 함수는 실행되지 않고 CloudFront는 캐싱된 응답을 최종 사용자에게 반환합니다.
- 응답이 캐시에 없는 경우 함수는 실행되고 CloudFront는 캐싱된 응답을 최종 사용자에게 반환하며 이를 캐싱합니다.

HTTP 응답을 생성하는 몇 가지 샘플 코드를 보려면 Lambda@Edge 예제 함수 단원을 참조합니다. 응답 트리거에서 HTTP 응답을 바꿀 수도 있습니다. 자세한 내용은 오리진 응답 트리거에서 HTTP 응답 업데이트 단원을 참조합니다.

프로그래밍 모델

이 단원에서는 Lambda@Edge를 사용하여 HTTP 응답을 생성하기 위한 프로그래밍 모델을 설명합니다.

주제

- 응답 객체
- 오류
- 필수 필드

### 응답 객체

`result` 메서드의 `callback`파라미터로 반환한 응답은 다음 구조를 가져야 합니다(`status` 필드만이 필요함).

```
const response = {
    body: 'content',
    bodyEncoding: 'text' | 'base64',
    headers: {
        'header name in lowercase': [{
            key: 'header name in standard case',
            value: 'header value'
         }],
         ...
    },
    status: 'HTTP status code (string)',
    statusDescription: 'status description'
};
```

응답 객체에는 다음 값이 포함될 수 있습니다.

**`body`**

CloudFront가 생성된 응답에서 반환하고자 하는 본문입니다.

**`bodyEncoding`**

body에서 지정한 값에 대한 인코딩입니다. 유일하게 유효한 인코딩은 `text`과 `base64`입니다. body 객체에 `response`를 포함했지만 `bodyEncoding`을 누락한 경우 CloudFront는 본문을 텍스트로 처리합니다.

`bodyEncoding`을 `base64`로 지정하지만 본문이 유효한 base64가 아닌 경우 CloudFront는 오류를 반환합니다.

**`headers`**

CloudFront가 생성된 응답에서 반환하고자 하는 헤더입니다. 다음을 참조하십시오.

- `headers` 객체 내 키는 표준 HTTP 헤더 이름의 소문자 버전입니다. 소문자 키를 사용하여 대/소문자를 구분하지 않고 헤더 값에 액세스할 수 있습니다.
- 각 헤더(예: `headers["accept"]` 또는 `headers["host"]`)는 키-값 페어의 어레이입니다. 해당 헤더에 대해 어레이에는 생성된 응답의 각 값에 대한 하나의 키-값 페어를 포함합니다.

- `key`(선택사항)는 HTTP 요청에 표시되는 헤더의 대/소문자 구분 이름입니다(예: `accept` 또는 `host`).
- `value`를 헤더 값으로 지정합니다.
- 키-값 페어의 헤더 키 부분을 포함시키지 않으면 Lambda@Edge는 사용자가 입력한 헤더 이름을 사용하여 헤더 키를 자동으로 삽입합니다. 헤더 이름의 형식을 어떻게 지정했든 상관없이, 삽입된 헤더 키는 하이픈(-)으로 구분된 각 부분의 첫 문자가 자동으로 대문자로 지정됩니다.

  예를 들어 헤더를 다음과 같이 헤더 키 없이 추가할 수 있습니다. `'content-type': [{ value: 'text/html;charset=UTF-8' }]`

  이 예제에서 Lambda@Edge는 다음과 같은 헤더 키를 생성합니다. `Content-Type`

헤더 사용 시 제한 사항에 대한 자세한 내용은 엣지 함수에 대한 제한 사항 단원을 참조하십시오.

**`status`**

HTTP 상태 코드. 상태 코드를 문자열로 제공합니다. CloudFront는 다음에 대해 제공된 상태 코드를 사용합니다.

- 응답 시 반환
- 오리진 요청 이벤트에 의해 트리거된 함수에 의해 응답이 생성되었을 때의 CloudFront 엣지 캐시에 있는 캐시
- CloudFront 액세스 로그(표준 로그)에 로그인

`status` 값이 200~599가 아닌 경우 CloudFront는 최종 사용자에게 오류를 반환합니다.

**`statusDescription`**

CloudFront가 응답에서 반환하고 HTTP 상태 코드를 포함하고자 하는 설명입니다. HTTP 상태 코드 200에 대해 OK와 같은 표준 설명을 사용할 필요는 없습니다.

## 오류

다음은 생성된 HTTP 응답에서 발생할 수 있는 오류입니다.

### 본문을 포함하고 상태에 대해 204(콘텐츠 없음)을 지정한 응답

최종 사용자 요청 이벤트에 의해 함수가 트리거될 때 CloudFront는 다음 두 가지가 모두 true일 경우 HTTP 502 상태 코드(잘못된 게이트웨이)를 반환합니다.

- `status`의 기본값이 204(콘텐츠 없음)임
- 응답에 body에 대한 값이 포함

이는 Lambda@Edge가 HTTP 204 응답에 메시지 본문이 포함될 필요가 없다고 명시한 RFC 2616의 선택적인 제한을 부과하기 때문입니다.

생성된 응답의 크기 제한

Lambda 함수가 생성하는 응답의 최대 크기는 함수를 트리거한 이벤트에 따라 다릅니다.

- 최종 사용자 요청 이벤트 - 40KB
- 오리진 요청 이벤트 - 1MB

응답이 허용된 크기보다 큰 경우 CloudFront는 최종 사용자에게 HTTP 502 상태 코드(잘못된 게이트웨이)를 반환합니다.

필수 필드

`status` 필드는 필수입니다.

다른 모든 필드는 선택 사항입니다.

## 오리진 응답 트리거에서 HTTP 응답 업데이트

CloudFront가 오리진 서버에서 HTTP 응답을 수신할 때 오리진 응답 트리거가 캐시 동작과 연결된 경우 HTTP 응답을 수정하여 오리진에서 반환된 것을 재정의할 수 있습니다.

HTTP 응답 업데이트의 몇 가지 일반 시나리오는 다음을 포함합니다.

- 오리진이 오류 상태 코드(4xx 또는 5xx)를 반환할 때 상태를 변경하여 HTTP 200 상태 코드를 설정하고 정적 본문 콘텐츠를 생성하여 최종 사용자에게 반환합니다. 샘플 코드에 대한 내용은 예시: 오리진 응답 트리거를 사용하여 오류 상태 코드를 200으로 업데이트를 참조하십시오.
- 오리진이 오류 상태 코드(4xx 또는 5xx)를 반환할 때 상태를 변경하여 HTTP 301 또는 HTTP 302 상태 코드를 설정하고 사용자를 다른 웹 사이트로 리디렉션합니다. 샘플 코드에 대한 내용은 예시: 오리진 응답 트리거를 사용하여 오류 상태 코드를 302로 업데이트를 참조하십시오.

> **Note**
>
> 함수는 200~599(경계값 포함) 사이의 상태 값을 반환해야 합니다. 그렇지 않으면 CloudFront가 최종 사용자에게 오류를 반환합니다.

최종 사용자 및 오리진 요청 이벤트에서 HTTP 응답을 바꿀 수도 있습니다. 자세한 내용은 요청 트리거에서 HTTP 응답 생성 섹션을 참조하세요.

HTTP 응답을 사용하여 작업 중일 때 Lambda@Edge는 오리진 서버에 의해 오리진 응답 트리거로 반환되는 본문을 공개하지 않습니다. 원하는 값으로 설정하여 정적 콘텐츠 본문을 생성하거나 값을 비어 있음으로 설정하여 함수 내부의 본문을 제거할 수 있습니다. 함수의 본문 필드를 업데이트하지 않은 경우 오리진 서버에서 반환된 원래 본문이 최종 사용자에게 반환됩니다.

### 본문 포함 옵션을 선택해 요청 본문에 액세스

Lambda 함수에서 액세스할 수 있도록 Lambda@Edge가 본문에 쓰기 가능한 HTTP 메서드(POST, PUT, DELETE 등)에 대한 요청을 노출하도록 선택할 수 있습니다. 읽기 전용 액세스 권한을 선택하거나 본문을 바꾸도록 지정할 수 있습니다.

이 옵션을 활성화하려면 함수에 대해 최종 사용자 요청 또는 오리진 요청 이벤트에 대해 실행되는 CloudFront 트리거를 생성할 때 본문 포함을 선택합니다. 자세한 내용은 Lambda@Edge 함수에 대한 트리거 추가 단원을 참조하고, 함수에서 본문 포함을 사용하는 방법을 알아보려면 Lambda@Edge 이벤트 구조 단원을 참조하세요.

이 기능을 사용하려는 시나리오는 다음과 같을 수 있습니다.

- 오리진 서버로 고객 입력 데이터를 다시 보내지 않는 웹 양식(예: "문의처") 처리
- 최종 사용자의 브라우저가 보낸 웹 비콘 데이터를 수집해 엣지에서 처리

샘플 코드에 대한 내용은 Lambda@Edge 예제 함수를 참조하십시오.

> **Note**
>
> 요청 본문이 크면 Lambda@Edge가 자릅니다. 최대 크기 및 잘림에 대한 자세한 내용은 본문 포함 옵션이 적용된 요청 본문에 대한 제한 사항 단원을 참조하세요.

## Lambda@Edge 예제 함수

Amazon CloudFront에서 Lambda 함수를 사용하려면 다음 예시를 참조하세요.

> **Note**
>
> Lambda@Edge 함수에 대해 런타임 Node.js 18 이상을 선택하면 `index.mjs` 파일이 자동으로 생성됩니다. 다음 코드 예제를 사용하려면 `index.mjs` 파일의 이름을 `index.js`로 바꿉니다.

주제

- 일반 예제
- 응답 생성 - 예시
- 쿼리 문자열 - 예시
- 국가 또는 디바이스 유형 헤더별 콘텐츠 개인화 - 예제
- 콘텐츠 기반 동적 원본 선택 - 예제
- 오류 상태 업데이트 - 예시
- 요청 본문 액세스 - 예시

## 일반 예제

다음 예시에서는 CloudFront에서 Lambda@Edge 를 사용하는 일반적인 방법을 보여줍니다.

주제

- 예: A/B 테스트
- 예시: 응답 헤더 재정의

### 예: A/B 테스트

다음 예제를 사용하여 리디렉션을 만들거나 URL을 변경하지 않고 두 가지 다른 버전의 이미지를 테스트할 수 있습니다. 이 예제는 최종 사용자 요청의 쿠키를 읽고 그에 따라 요청 URL을 수정합니다. 최종 사용자가 예상 값 중 하나와 함께 쿠키를 보내지 않는 경우, 이 예제에서는 최종 사용자를 URL 중 하나에 임의로 할당합니다.

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const headers = request.headers;

    if (request.uri !== '/experiment-pixel.jpg') {
        // do not process if this is not an A-B test request
        callback(null, request);
        return;
    }
```

```
    const cookieExperimentA = 'X-Experiment-Name=A';
    const cookieExperimentB = 'X-Experiment-Name=B';
    const pathExperimentA = '/experiment-group/control-pixel.jpg';
    const pathExperimentB = '/experiment-group/treatment-pixel.jpg';

    /*
     * Lambda at the Edge headers are array objects.
     *
     * Client may send multiple Cookie headers, i.e.:
     * > GET /viewerRes/test HTTP/1.1
     * > User-Agent: curl/7.18.1 (x86_64-unknown-linux-gnu) libcurl/7.18.1
OpenSSL/1.0.1u zlib/1.2.3
     * > Cookie: First=1; Second=2
     * > Cookie: ClientCode=abc
     * > Host: example.com
     *
     * You can access the first Cookie header at headers["cookie"][0].value
     * and the second at headers["cookie"][1].value.
     *
     * Header values are not parsed. In the example above,
     * headers["cookie"][0].value is equal to "First=1; Second=2"
     */
    let experimentUri;
    if (headers.cookie) {
        for (let i = 0; i < headers.cookie.length; i++) {
            if (headers.cookie[i].value.indexOf(cookieExperimentA) >= 0) {
                console.log('Experiment A cookie found');
                experimentUri = pathExperimentA;
                break;
            } else if (headers.cookie[i].value.indexOf(cookieExperimentB) >= 0) {
                console.log('Experiment B cookie found');
                experimentUri = pathExperimentB;
                break;
            }
        }
    }

    if (!experimentUri) {
        console.log('Experiment cookie has not been found. Throwing dice...');
        if (Math.random() < 0.75) {
            experimentUri = pathExperimentA;
        } else {
            experimentUri = pathExperimentB;
        }
```

```
    }

    request.uri = experimentUri;
    console.log(`Request uri set to "${request.uri}"`);
    callback(null, request);
};
```

## Python

```
import json
import random

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    headers = request['headers']

    if request['uri'] != '/experiment-pixel.jpg':
        # Not an A/B Test
        return request

    cookieExperimentA, cookieExperimentB = 'X-Experiment-Name=A', 'X-Experiment-
Name=B'
    pathExperimentA, pathExperimentB = '/experiment-group/control-pixel.jpg', '/
experiment-group/treatment-pixel.jpg'

    '''
    Lambda at the Edge headers are array objects.

    Client may send multiple cookie headers. For example:
    > GET /viewerRes/test HTTP/1.1
    > User-Agent: curl/7.18.1 (x86_64-unknown-linux-gnu) libcurl/7.18.1
 OpenSSL/1.0.1u zlib/1.2.3
    > Cookie: First=1; Second=2
    > Cookie: ClientCode=abc
    > Host: example.com

    You can access the first Cookie header at headers["cookie"][0].value
    and the second at headers["cookie"][1].value.

    Header values are not parsed. In the example above,
    headers["cookie"][0].value is equal to "First=1; Second=2"
    '''
```

```
    experimentUri = ""

    for cookie in headers.get('cookie', []):
        if cookieExperimentA in cookie['value']:
            print("Experiment A cookie found")
            experimentUri = pathExperimentA
            break
        elif cookieExperimentB in cookie['value']:
            print("Experiment B cookie found")
            experimentUri = pathExperimentB
            break

    if not experimentUri:
        print("Experiment cookie has not been found. Throwing dice...")
        if random.random() < 0.75:
            experimentUri = pathExperimentA
        else:
            experimentUri = pathExperimentB

    request['uri'] = experimentUri
    print(f"Request uri set to {experimentUri}")
    return request
```

### 예시: 응답 헤더 재정의

다음 예제는 다른 헤더의 값을 기준으로 응답 헤더의 값을 변경하는 방법을 보여줍니다.

#### Node.js

```
export const handler = async (event) => {
    const response = event.Records[0].cf.response;
    const headers = response.headers;

    const headerNameSrc = 'X-Amz-Meta-Last-Modified';
    const headerNameDst = 'Last-Modified';

    if (headers[headerNameSrc.toLowerCase()]) {
        headers[headerNameDst.toLowerCase()] = [{
            key: headerNameDst,
            value: headers[headerNameSrc.toLowerCase()][0].value,
        }];
        console.log(`Response header "${headerNameDst}" was set to ` +
                    `"${headers[headerNameDst.toLowerCase()][0].value}"`);
```

```
    }

    return response;
};
```

Python

```
import json

def lambda_handler(event, context):
    response = event['Records'][0]['cf']['response']
    headers = response['headers']

    header_name_src = 'X-Amz-Meta-Last-Modified'
    header_name_dst = 'Last-Modified'

    if headers.get(header_name_src.lower()):
        headers[header_name_dst.lower()] = [{
            'key': header_name_dst,
            'value': headers[header_name_src.lower()][0]['value']
        }]
        print(f'Response header "{header_name_dst}" was set to '
              f'"{headers[header_name_dst.lower()][0]["value"]}"')

    return response
```

## 응답 생성 - 예시

다음 예시에서는 Lambda@Edge를 사용하여 응답을 생성하는 방법을 보여줍니다.

주제

- 예시: 정적 콘텐츠 제공(생성된 응답)
- 예: HTTP 리디렉션 생성(생성된 응답)

### 예시: 정적 콘텐츠 제공(생성된 응답)

다음 예제는 Lambda 함수를 사용하여 정적 웹사이트 콘텐츠를 서비스하는 방법을 보여줍니다. 이는 오리진 서버에 대한 로드를 줄이고 전체 지연 시간을 단축합니다.

> **Note**
>
> 최종 사용자 요청 및 오리진 요청 이벤트에 대해 HTTP 응답을 만들 수 있습니다. 자세한 내용은 the section called "요청 트리거에서 HTTP 응답 생성" 섹션을 참조하세요.
> 원본 응답 이벤트에서 HTTP 응답의 본문을 바꾸거나 제거할 수도 있습니다. 자세한 내용은 the section called "오리진 응답 트리거에서 HTTP 응답 업데이트" 섹션을 참조하세요.

Node.js

```
'use strict';

const content = `
<\!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Simple Lambda@Edge Static Content Response</title>
  </head>
  <body>
    <p>Hello from Lambda@Edge!</p>
  </body>
</html>
`;

exports.handler = (event, context, callback) => {
    /*
     * Generate HTTP OK response using 200 status code with HTML body.
     */
    const response = {
        status: '200',
        statusDescription: 'OK',
        headers: {
            'cache-control': [{
                key: 'Cache-Control',
                value: 'max-age=100'
            }],
            'content-type': [{
                key: 'Content-Type',
                value: 'text/html'
            }]
        },
```

```
        body: content,
    };
    callback(null, response);
};
```

## Python

```
import json

CONTENT = """
<\!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>Simple Lambda@Edge Static Content Response</title>
</head>
<body>
    <p>Hello from Lambda@Edge!</p>
</body>
</html>
"""

def lambda_handler(event, context):
    # Generate HTTP OK response using 200 status code with HTML body.
    response = {
        'status': '200',
        'statusDescription': 'OK',
        'headers': {
            'cache-control': [
                {
                    'key': 'Cache-Control',
                    'value': 'max-age=100'
                }
            ],
            "content-type": [
                {
                    'key': 'Content-Type',
                    'value': 'text/html'
                }
            ]
        },
        'body': CONTENT
    }
```

```
    return response
```

### 예: HTTP 리디렉션 생성(생성된 응답)

다음 예제는 HTTP 리디렉션을 만드는 방법을 보여줍니다.

> **Note**
> 최종 사용자 요청 및 오리진 요청 이벤트에 대해 HTTP 응답을 만들 수 있습니다. 자세한 내용은 요청 트리거에서 HTTP 응답 생성 섹션을 참조하세요.

#### Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
    /*
     * Generate HTTP redirect response with 302 status code and Location header.
     */
    const response = {
        status: '302',
        statusDescription: 'Found',
        headers: {
            location: [{
                key: 'Location',
                value: 'https://docs.aws.amazon.com/lambda/latest/dg/lambda-
edge.html',
            }],
        },
    };
    callback(null, response);
};
```

#### Python

```
def lambda_handler(event, context):

    # Generate HTTP redirect response with 302 status code and Location header.

    response = {
```

```
        'status': '302',
        'statusDescription': 'Found',
        'headers': {
            'location': [{
                'key': 'Location',
                'value': 'https://docs.aws.amazon.com/lambda/latest/dg/lambda-
edge.html'
            }]
        }
    }

    return response
```

## 쿼리 문자열 - 예시

다음 예시에서는 Lambda@Edge를 쿼리 문자열과 함께 사용할 수 있는 방법이 포함되어 있습니다.

주제

- 예시: 쿼리 문자열 파라미터를 기반으로 헤더 추가
- 예시: 쿼리 문자열 파라미터를 정규화하여 캐시 적중률 향상
- 예시: 인증되지 않은 사용자를 로그인 페이지로 리디렉션

### 예시: 쿼리 문자열 파라미터를 기반으로 헤더 추가

다음 예제에서는 쿼리 문자열 파라미터의 키-값 페어를 가져오고 그 값을 토대로 헤더를 추가하는 방법을 보여줍니다.

Node.js

```
'use strict';

const querystring = require('querystring');
exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    /* When a request contains a query string key-value pair but the origin server
     * expects the value in a header, you can use this Lambda function to
     * convert the key-value pair to a header. Here's what the function does:
     * 1. Parses the query string and gets the key-value pair.
```

```
     * 2. Adds a header to the request using the key-value pair that the function
 got in step 1.
     */

    /* Parse request querystring to get javascript object */
    const params = querystring.parse(request.querystring);

    /* Move auth param from querystring to headers */
    const headerName = 'Auth-Header';
    request.headers[headerName.toLowerCase()] = [{ key: headerName, value:
 params.auth }];
    delete params.auth;

    /* Update request querystring */
    request.querystring = querystring.stringify(params);

    callback(null, request);
};
```

## Python

```
from urllib.parse import parse_qs, urlencode

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']

    '''
    When a request contains a query string key-value pair but the origin server
    expects the value in a header, you can use this Lambda function to
    convert the key-value pair to a header. Here's what the function does:
        1. Parses the query string and gets the key-value pair.
        2. Adds a header to the request using the key-value pair that the function
 got in step 1.
    '''

    # Parse request querystring to get dictionary/json
    params = {k : v[0] for k, v in parse_qs(request['querystring']).items()}

    # Move auth param from querystring to headers
    headerName = 'Auth-Header'
    request['headers'][headerName.lower()] = [{'key': headerName, 'value':
 params['auth']}]
    del params['auth']
```

```
    # Update request querystring
    request['querystring'] = urlencode(params)

    return request
```

예시: 쿼리 문자열 파라미터를 정규화하여 캐시 적중률 향상

다음 예제에서는 CloudFront가 요청을 오리진에 전달하기 전에 쿼리 문자열을 다음과 같이 변경하여 캐시 적중률을 향상하는 방법을 보여 줍니다.

- 파라미터 이름을 기준으로 키-값 페어를 알파벳순으로 정렬
- 키-값 페어의 대/소문자를 소문자로 변경

자세한 내용은 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱 섹션을 참조하세요.

Node.js

```
'use strict';

const querystring = require('querystring');

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    /* When you configure a distribution to forward query strings to the origin and
     * to cache based on an allowlist of query string parameters, we recommend
     * the following to improve the cache-hit ratio:
     * - Always list parameters in the same order.
     * - Use the same case for parameter names and values.
     *
     * This function normalizes query strings so that parameter names and values
     * are lowercase and parameter names are in alphabetical order.
     *
     * For more information, see:
     * https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/
QueryStringParameters.html
     */

    console.log('Query String: ', request.querystring);

    /* Parse request query string to get javascript object */
```

```
    const params = querystring.parse(request.querystring.toLowerCase());
    const sortedParams = {};

    /* Sort param keys */
    Object.keys(params).sort().forEach(key => {
        sortedParams[key] = params[key];
    });

    /* Update request querystring with normalized  */
    request.querystring = querystring.stringify(sortedParams);

    callback(null, request);
};
```

Python

```
from urllib.parse import parse_qs, urlencode

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    '''
    When you configure a distribution to forward query strings to the origin and
    to cache based on an allowlist of query string parameters, we recommend
    the following to improve the cache-hit ratio:
    Always list parameters in the same order.
    - Use the same case for parameter names and values.

    This function normalizes query strings so that parameter names and values
    are lowercase and parameter names are in alphabetical order.

    For more information, see:
    https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/
QueryStringParameters.html
    '''
    print("Query string: ", request["querystring"])

    # Parse request query string to get js object
    params = {k : v[0] for k, v in parse_qs(request['querystring'].lower()).items()}

    # Sort param keys
    sortedParams = sorted(params.items(), key=lambda x: x[0])

    # Update request querystring with normalized
```

```
    request['querystring'] = urlencode(sortedParams)

    return request
```

예시: 인증되지 않은 사용자를 로그인 페이지로 리디렉션

다음 예제에서는 사용자가 자격 증명을 입력하지 않은 경우 사용자를 로그인 페이지로 리디렉션하는 방법을 보여 줍니다.

Node.js

```
'use strict';

function parseCookies(headers) {
    const parsedCookie = {};
    if (headers.cookie) {
        headers.cookie[0].value.split(';').forEach((cookie) => {
            if (cookie) {
                const parts = cookie.split('=');
                parsedCookie[parts[0].trim()] = parts[1].trim();
            }
        });
    }
    return parsedCookie;
}

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const headers = request.headers;

    /* Check for session-id in request cookie in viewer-request event,
     * if session-id is absent, redirect the user to sign in page with original
     * request sent as redirect_url in query params.
     */

    /* Check for session-id in cookie, if present then proceed with request */
    const parsedCookies = parseCookies(headers);
    if (parsedCookies && parsedCookies['session-id']) {
        callback(null, request);
        return;
    }
```

```
    /* URI encode the original request to be sent as redirect_url in query params */
    const encodedRedirectUrl = encodeURIComponent(`https://
${headers.host[0].value}${request.uri}?${request.querystring}`);
    const response = {
        status: '302',
        statusDescription: 'Found',
        headers: {
            location: [{
                key: 'Location',
                value: `https://www.example.com/signin?redirect_url=
${encodedRedirectUrl}`,
            }],
        },
    };
    callback(null, response);
};
```

## Python

```
import urllib

def parseCookies(headers):
    parsedCookie = {}
    if headers.get('cookie'):
        for cookie in headers['cookie'][0]['value'].split(';'):
            if cookie:
                parts = cookie.split('=')
                parsedCookie[parts[0].strip()] = parts[1].strip()
    return parsedCookie

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    headers = request['headers']

    '''
    Check for session-id in request cookie in viewer-request event,
    if session-id is absent, redirect the user to sign in page with original
    request sent as redirect_url in query params.
    '''

    # Check for session-id in cookie, if present, then proceed with request
    parsedCookies = parseCookies(headers)
```

```
    if parsedCookies and parsedCookies['session-id']:
        return request

    # URI encode the original request to be sent as redirect_url in query params
    redirectUrl = "https://%s%s?%s" % (headers['host'][0]['value'], request['uri'],
request['querystring'])
    encodedRedirectUrl = urllib.parse.quote_plus(redirectUrl.encode('utf-8'))

    response = {
        'status': '302',
        'statusDescription': 'Found',
        'headers': {
            'location': [{
                'key': 'Location',
                'value': 'https://www.example.com/signin?redirect_url=%s' %
encodedRedirectUrl
            }]
        }
    }
    return response
```

## 국가 또는 디바이스 유형 헤더별 콘텐츠 개인화 - 예제

다음 예시에서는 Lambda@Edge를 사용하여 뷰어가 사용하는 위치나 장치 유형에 따라 동작을 사용자 정의하는 방법을 보여줍니다.

주제

- 예시: 최종 사용자 요청을 국가별 URL로 리디렉션
- 예시: 디바이스를 기반으로 객체의 다양한 버전 제공

### 예시: 최종 사용자 요청을 국가별 URL로 리디렉션

다음 예제에서는 국가별 URL이 포함된 HTTP 리디렉션 응답을 생성하고 해당 응답을 최종 사용자에게 반환하는 방법을 보여 줍니다. 이 작업은 국가별 응답을 제공하려는 경우에 유용합니다. 예:

- 국가별 하위 도메인이 있는 경우(예: us.example.com 및 tw.example.com) 최종 사용자가 example.com을 요청할 때 리디렉션 응답을 생성할 수 있습니다.
- 비디오를 스트리밍하지만 특정 국가에서 콘텐츠를 스트리밍할 권한이 없는 경우 비디오를 볼 수 없는 이유를 설명하는 페이지로 해당 국가의 사용자를 리디렉션할 수 있습니다.

다음을 참조하십시오.

- `CloudFront-Viewer-Country` 헤더 값을 기반으로 캐시하도록 배포를 구성해야 합니다. 자세한 내용은 선택한 요청 헤더 기반의 캐시 단원을 참조하세요.
- CloudFront는 최종 사용자 요청 이벤트 뒤에 `CloudFront-Viewer-Country` 헤더를 추가합니다. 이 예제를 사용하려면 오리진 요청 이벤트에 대한 트리거를 생성해야 합니다.

Node.js

```
'use strict';

/* This is an origin request function */
exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const headers = request.headers;

    /*
     * Based on the value of the CloudFront-Viewer-Country header, generate an
     * HTTP status code 302 (Redirect) response, and return a country-specific
     * URL in the Location header.
     * NOTE: 1. You must configure your distribution to cache based on the
     *          CloudFront-Viewer-Country header. For more information, see
     *          https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-
headers
     *       2. CloudFront adds the CloudFront-Viewer-Country header after the
 viewer
     *          request event. To use this example, you must create a trigger for
 the
     *          origin request event.
     */

    let url = 'https://example.com/';
    if (headers['cloudfront-viewer-country']) {
        const countryCode = headers['cloudfront-viewer-country'][0].value;
        if (countryCode === 'TW') {
            url = 'https://tw.example.com/';
        } else if (countryCode === 'US') {
            url = 'https://us.example.com/';
        }
    }

    const response = {
```

```
        status: '302',
        statusDescription: 'Found',
        headers: {
            location: [{
                key: 'Location',
                value: url,
            }],
        },
    };
    callback(null, response);
};
```

Python

```
# This is an origin request function

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    headers = request['headers']

    '''
    Based on the value of the CloudFront-Viewer-Country header, generate an
    HTTP status code 302 (Redirect) response, and return a country-specific
    URL in the Location header.
    NOTE: 1. You must configure your distribution to cache based on the
            CloudFront-Viewer-Country header. For more information, see
            https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-headers
          2. CloudFront adds the CloudFront-Viewer-Country header after the viewer
            request event. To use this example, you must create a trigger for the
            origin request event.
    '''

    url = 'https://example.com/'
    viewerCountry = headers.get('cloudfront-viewer-country')
    if viewerCountry:
        countryCode = viewerCountry[0]['value']
        if countryCode == 'TW':
            url = 'https://tw.example.com/'
        elif countryCode == 'US':
            url = 'https://us.example.com/'

    response = {
        'status': '302',
```

```
        'statusDescription': 'Found',
        'headers': {
            'location': [{
                'key': 'Location',
                'value': url
            }]
        }
    }

    return response
```

예시: 디바이스를 기반으로 객체의 다양한 버전 제공

다음 예제에서는 사용자가 사용 중인 디바이스 유형(예: 모바일 디바이스 또는 태블릿)을 기반으로 객체의 다양한 버전을 서비스하는 방법을 보여 줍니다. 다음을 참조하십시오.

- `CloudFront-Is-*-Viewer` 헤더 값을 기반으로 캐시하도록 배포를 구성해야 합니다. 자세한 내용은 선택한 요청 헤더 기반의 캐시 단원을 참조하세요.
- CloudFront는 최종 사용자 요청 이벤트 뒤에 `CloudFront-Is-*-Viewer` 헤더를 추가합니다. 이 예제를 사용하려면 오리진 요청 이벤트에 대한 트리거를 생성해야 합니다.

Node.js

```
'use strict';

/* This is an origin request function */
exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const headers = request.headers;

    /*
     * Serve different versions of an object based on the device type.
     * NOTE: 1. You must configure your distribution to cache based on the
     *          CloudFront-Is-*-Viewer headers. For more information, see
     *          the following documentation:
     *          https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-
headers
     *          https://docs.aws.amazon.com/console/cloudfront/cache-on-device-type
     *       2. CloudFront adds the CloudFront-Is-*-Viewer headers after the viewer
     *          request event. To use this example, you must create a trigger for
 the
```

```
     *          origin request event.
     */

    const desktopPath = '/desktop';
    const mobilePath = '/mobile';
    const tabletPath = '/tablet';
    const smarttvPath = '/smarttv';

    if (headers['cloudfront-is-desktop-viewer']
        && headers['cloudfront-is-desktop-viewer'][0].value === 'true') {
        request.uri = desktopPath + request.uri;
    } else if (headers['cloudfront-is-mobile-viewer']
                && headers['cloudfront-is-mobile-viewer'][0].value === 'true') {
        request.uri = mobilePath + request.uri;
    } else if (headers['cloudfront-is-tablet-viewer']
                && headers['cloudfront-is-tablet-viewer'][0].value === 'true') {
        request.uri = tabletPath + request.uri;
    } else if (headers['cloudfront-is-smarttv-viewer']
                && headers['cloudfront-is-smarttv-viewer'][0].value === 'true') {
        request.uri = smarttvPath + request.uri;
    }
    console.log(`Request uri set to "${request.uri}"`);

    callback(null, request);
};
```

## Python

```
# This is an origin request function
def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    headers = request['headers']

    '''
    Serve different versions of an object based on the device type.
    NOTE: 1. You must configure your distribution to cache based on the
            CloudFront-Is-*-Viewer headers. For more information, see
            the following documentation:
            https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-headers
            https://docs.aws.amazon.com/console/cloudfront/cache-on-device-type
          2. CloudFront adds the CloudFront-Is-*-Viewer headers after the viewer
            request event. To use this example, you must create a trigger for the
            origin request event.
```

```
    '''

    desktopPath = '/desktop';
    mobilePath = '/mobile';
    tabletPath = '/tablet';
    smarttvPath = '/smarttv';

    if 'cloudfront-is-desktop-viewer' in headers and headers['cloudfront-is-desktop-
viewer'][0]['value'] == 'true':
        request['uri'] = desktopPath + request['uri']
    elif 'cloudfront-is-mobile-viewer' in headers and headers['cloudfront-is-mobile-
viewer'][0]['value'] == 'true':
        request['uri'] = mobilePath + request['uri']
    elif 'cloudfront-is-tablet-viewer' in headers and headers['cloudfront-is-tablet-
viewer'][0]['value'] == 'true':
        request['uri'] = tabletPath + request['uri']
    elif 'cloudfront-is-smarttv-viewer' in headers and headers['cloudfront-is-
smarttv-viewer'][0]['value'] == 'true':
        request['uri'] = smarttvPath + request['uri']

    print("Request uri set to %s" % request['uri'])

    return request
```

## 콘텐츠 기반 동적 원본 선택 - 예제

다음 예시에서는 Lambda@Edge를 사용하여 요청 정보를 기반으로 다른 오리진으로 라우팅하는 방법을 보여줍니다.

주제

- 예시: 오리진 요청 트리거를 사용하여 사용자 지정 오리진에서 Amazon S3 오리진으로 변경
- 예시: 오리진 요청 트리거를 사용하여 Amazon S3 오리진 리전 변경
- 예시: 오리진 요청 트리거를 사용하여 Amazon S3 오리진에서 사용자 지정 오리진으로 변경
- 예시: 오리진 요청 트리거를 사용하여 하나의 Amazon S3 버킷에서 다른 버킷으로 점진적으로 트래픽 전송
- 예시: 오리진 요청 트리거를 사용하여 국가 헤더를 기반으로 오리진 도메인 이름 변경

### 예시: 오리진 요청 트리거를 사용하여 사용자 지정 오리진에서 Amazon S3 오리진으로 변경

이 함수는 오리진 요청 트리거를 사용하여 사용자 지정 오리진에서 콘텐츠를 요청 속성을 기반으로 가져오는 Amazon S3 오리진으로 변경하는 방법을 보여줍니다.

Node.js

```
'use strict';

 const querystring = require('querystring');

 exports.handler = (event, context, callback) => {
     const request = event.Records[0].cf.request;

     /**
      * Reads query string to check if S3 origin should be used, and
      * if true, sets S3 origin properties.
      */

     const params = querystring.parse(request.querystring);

     if (params['useS3Origin']) {
         if (params['useS3Origin'] === 'true') {
             const s3DomainName = 'amzn-s3-demo-bucket.s3.amazonaws.com';

             /* Set S3 origin fields */
             request.origin = {
                 s3: {
                     domainName: s3DomainName,
                     region: '',
                     authMethod: 'origin-access-identity',
                     path: '',
                     customHeaders: {}
                 }
             };
             request.headers['host'] = [{ key: 'host', value: s3DomainName}];
         }
     }

    callback(null, request);
};
```

Python

```
from urllib.parse import parse_qs

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    '''
    Reads query string to check if S3 origin should be used, and
    if true, sets S3 origin properties
    '''
    params = {k: v[0] for k, v in parse_qs(request['querystring']).items()}
    if params.get('useS3Origin') == 'true':
        s3DomainName = 'amzn-s3-demo-bucket.s3.amazonaws.com'

        # Set S3 origin fields
        request['origin'] = {
            's3': {
                'domainName': s3DomainName,
                'region': '',
                'authMethod': 'origin-access-identity',
                'path': '',
                'customHeaders': {}
            }
        }
        request['headers']['host'] = [{'key': 'host', 'value': s3DomainName}]
    return request
```

예시: 오리진 요청 트리거를 사용하여 Amazon S3 오리진 리전 변경

이 함수는 오리진 요청 트리거를 사용하여 콘텐츠를 요청 속성을 기반으로 가져오는 Amazon S3 오리진을 변경하는 방법을 보여줍니다.

이 예제에서는 `CloudFront-Viewer-Country` 헤더 값을 사용하여 뷰어에게 더 가까운 리전의 버킷으로 S3 버킷 도메인 이름을 업데이트합니다. 이는 여러 방법에서 유용할 수 있습니다.

- 지정된 리전이 최종 사용자의 국가와 더욱 근접할 때 지연 시간을 감소합니다.
- 데이터가 요청이 온 곳과 동일한 국가에 있는 리전에서 서비스되는지 확인함으로써 데이터 주권을 제공합니다.

이 예제를 활용하려면 다음과 같이 해야 합니다.

- `CloudFront-Viewer-Country` 헤더를 기반으로 캐시하도록 배포를 구성합니다. 자세한 내용은 선택한 요청 헤더 기반의 캐시 섹션을 참조하세요.
- 오리진 요청 이벤트에서 이 함수에 대한 트리거를 생성합니다. CloudFront는 뷰어 요청 이벤트 뒤에 `CloudFront-Viewer-Country` 헤더를 추가하므로 이 예제를 사용하려면 함수가 원본 요청에 대해 실행되는지 확인해야 합니다.

> **Note**
>
> 다음 예제 코드는 오리진에 사용 중인 모든 S3 버킷에 대해 동일한 오리진 액세스 ID(OAI)를 사용합니다. 자세한 내용은 오리진 액세스 ID 섹션을 참조하세요.

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    /**
     * This blueprint demonstrates how an origin-request trigger can be used to
     * change the origin from which the content is fetched, based on request
 properties.
     * In this example, we use the value of the CloudFront-Viewer-Country header
     * to update the S3 bucket domain name to a bucket in a Region that is closer to
     * the viewer.
     *
     * This can be useful in several ways:
     *      1) Reduces latencies when the Region specified is nearer to the viewer's
     *         country.
     *      2) Provides data sovereignty by making sure that data is served from an
     *         origin that's in the same country that the request came from.
     *
     * NOTE: 1. You must configure your distribution to cache based on the
     *          CloudFront-Viewer-Country header. For more information, see
     *          https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-
headers
     *       2. CloudFront adds the CloudFront-Viewer-Country header after the
 viewer
     *          request event. To use this example, you must create a trigger for
 the
```

```
     *          origin request event.
     */

    const countryToRegion = {
        'DE': 'eu-central-1',
        'IE': 'eu-west-1',
        'GB': 'eu-west-2',
        'FR': 'eu-west-3',
        'JP': 'ap-northeast-1',
        'IN': 'ap-south-1'
    };

    if (request.headers['cloudfront-viewer-country']) {
        const countryCode = request.headers['cloudfront-viewer-country'][0].value;
        const region = countryToRegion[countryCode];

        /**
         * If the viewer's country is not in the list you specify, the request
         * goes to the default S3 bucket you've configured.
         */
        if (region) {
            /**
             * If you've set up OAI, the bucket policy in the destination bucket
             * should allow the OAI GetObject operation, as configured by default
             * for an S3 origin with OAI. Another requirement with OAI is to provide
             * the Region so it can be used for the SIGV4 signature. Otherwise, the
             * Region is not required.
             */
            request.origin.s3.region = region;
            const domainName = `amzn-s3-demo-bucket-in-${region}.s3.
${region}.amazonaws.com`;
            request.origin.s3.domainName = domainName;
            request.headers['host'] = [{ key: 'host', value: domainName }];
        }
    }

    callback(null, request);
};
```

## Python

```
def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
```

```
    '''
    This blueprint demonstrates how an origin-request trigger can be used to
    change the origin from which the content is fetched, based on request
properties.
    In this example, we use the value of the CloudFront-Viewer-Country header
    to update the S3 bucket domain name to a bucket in a Region that is closer to
    the viewer.

    This can be useful in several ways:
        1) Reduces latencies when the Region specified is nearer to the viewer's
            country.
        2) Provides data sovereignty by making sure that data is served from an
            origin that's in the same country that the request came from.

    NOTE: 1. You must configure your distribution to cache based on the
            CloudFront-Viewer-Country header. For more information, see
            https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-headers
          2. CloudFront adds the CloudFront-Viewer-Country header after the viewer
            request event. To use this example, you must create a trigger for the
            origin request event.
    '''

    countryToRegion = {
        'DE': 'eu-central-1',
        'IE': 'eu-west-1',
        'GB': 'eu-west-2',
        'FR': 'eu-west-3',
        'JP': 'ap-northeast-1',
        'IN': 'ap-south-1'
    }

    viewerCountry = request['headers'].get('cloudfront-viewer-country')
    if viewerCountry:
        countryCode = viewerCountry[0]['value']
        region = countryToRegion.get(countryCode)

        # If the viewer's country in not in the list you specify, the request
        # goes to the default S3 bucket you've configured
        if region:
            '''
            If you've set up OAI, the bucket policy in the destination bucket
            should allow the OAI GetObject operation, as configured by default
            for an S3 origin with OAI. Another requirement with OAI is to provide
```

```
            the Region so it can be used for the SIGV4 signature. Otherwise, the
            Region is not required.
            '''
            request['origin']['s3']['region'] = region
            domainName = 'amzn-s3-demo-bucket-in-{0}.s3.
{0}.amazonaws.com'.format(region)
            request['origin']['s3']['domainName'] = domainName
            request['headers']['host'] = [{'key': 'host', 'value': domainName}]

    return request
```

예시: 오리진 요청 트리거를 사용하여 Amazon S3 오리진에서 사용자 지정 오리진으로 변경

이 함수는 오리진 요청 트리거를 사용하여 콘텐츠를 요청 속성을 기반으로 가져오는 사용자 지정 오리진을 변경하는 방법을 보여줍니다.

Node.js

```
'use strict';

const querystring = require('querystring');

 exports.handler = (event, context, callback) => {
     const request = event.Records[0].cf.request;

     /**
      * Reads query string to check if custom origin should be used, and
      * if true, sets custom origin properties.
      */

     const params = querystring.parse(request.querystring);

     if (params['useCustomOrigin']) {
         if (params['useCustomOrigin'] === 'true') {

             /* Set custom origin fields*/
             request.origin = {
                 custom: {
                     domainName: 'www.example.com',
                     port: 443,
                     protocol: 'https',
                     path: '',
```

```
                    sslProtocols: ['TLSv1', 'TLSv1.1'],
                    readTimeout: 5,
                    keepaliveTimeout: 5,
                    customHeaders: {}
                }
            };
            request.headers['host'] = [{ key: 'host', value: 'www.example.com'}];
        }
    }
    callback(null, request);
};
```

Python

```
from urllib.parse import parse_qs

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']

    # Reads query string to check if custom origin should be used, and
    # if true, sets custom origin properties

    params = {k: v[0] for k, v in parse_qs(request['querystring']).items()}

    if params.get('useCustomOrigin') == 'true':
            # Set custom origin fields
            request['origin'] = {
                'custom': {
                    'domainName': 'www.example.com',
                    'port': 443,
                    'protocol': 'https',
                    'path': '',
                    'sslProtocols': ['TLSv1', 'TLSv1.1'],
                    'readTimeout': 5,
                    'keepaliveTimeout': 5,
                    'customHeaders': {}
                }
            }
            request['headers']['host'] = [{'key': 'host', 'value':
 'www.example.com'}]

    return request
```

### 예시: 오리진 요청 트리거를 사용하여 하나의 Amazon S3 버킷에서 다른 버킷으로 점진적으로 트래픽 전송

이 함수는 통제된 방법을 통해 하나의 Amazon S3 버킷에서 다른 버킷으로 점진적으로 트래픽을 전송하는 방법을 보여 줍니다.

#### Node.js

```
'use strict';

    function getRandomInt(min, max) {
        /* Random number is inclusive of min and max*/
        return Math.floor(Math.random() * (max - min + 1)) + min;
 }

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const BLUE_TRAFFIC_PERCENTAGE = 80;

    /**
      * This Lambda function demonstrates how to gradually transfer traffic from
      * one S3 bucket to another in a controlled way.
      * We define a variable BLUE_TRAFFIC_PERCENTAGE which can take values from
      * 1 to 100. If the generated randomNumber less than or equal to
 BLUE_TRAFFIC_PERCENTAGE, traffic
      * is re-directed to blue-bucket. If not, the default bucket that we've
 configured
      * is used.
      */

    const randomNumber = getRandomInt(1, 100);

if (randomNumber <= BLUE_TRAFFIC_PERCENTAGE) {
         const domainName = 'blue-bucket.s3.amazonaws.com';
         request.origin.s3.domainName = domainName;
         request.headers['host'] = [{ key: 'host', value: domainName}];
     }
    callback(null, request);
};
```

#### Python

```
import math
```

```
import random

def getRandomInt(min, max):
    # Random number is inclusive of min and max
    return math.floor(random.random() * (max - min + 1)) + min

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    BLUE_TRAFFIC_PERCENTAGE = 80

    '''
    This Lambda function demonstrates how to gradually transfer traffic from
    one S3 bucket to another in a controlled way.
    We define a variable BLUE_TRAFFIC_PERCENTAGE which can take values from
    1 to 100. If the generated randomNumber less than or equal to
 BLUE_TRAFFIC_PERCENTAGE, traffic
    is re-directed to blue-bucket. If not, the default bucket that we've configured
    is used.
    '''

    randomNumber = getRandomInt(1, 100)

    if randomNumber <= BLUE_TRAFFIC_PERCENTAGE:
        domainName = 'blue-bucket.s3.amazonaws.com'
        request['origin']['s3']['domainName'] = domainName
        request['headers']['host'] = [{'key': 'host', 'value': domainName}]

    return request
```

예시: 오리진 요청 트리거를 사용하여 국가 헤더를 기반으로 오리진 도메인 이름 변경

이 함수는 `CloudFront-Viewer-Country` 헤더를 기반으로 원본 도메인 이름을 변경하여 뷰어의 국가에 더 가까운 원본에서 콘텐츠를 제공하는 방법을 보여줍니다.

배포에서 이 기능을 구현하면 다음과 같은 이점을 누릴 수 있습니다.

- 지정된 리전이 최종 사용자의 국가와 더욱 근접할 때 지연 시간이 감소
- 데이터가 요청이 온 곳과 동일한 국가에 있는 오리진에서 서비스되는지 확인함으로써 데이터 주권을 제공

이 함수를 사용하려면 `CloudFront-Viewer-Country` 헤더를 기반으로 캐시하도록 배포를 구성해야 합니다. 자세한 내용은 the section called "선택한 요청 헤더 기반의 캐시" 섹션을 참조하세요.

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
     const request = event.Records[0].cf.request;

  if (request.headers['cloudfront-viewer-country']) {
         const countryCode = request.headers['cloudfront-viewer-country'][0].value;
         if (countryCode === 'GB' || countryCode === 'DE' || countryCode === 'IE' )
 {
             const domainName = 'eu.example.com';
             request.origin.custom.domainName = domainName;
             request.headers['host'] = [{key: 'host', value: domainName}];
         }
     }

    callback(null, request);
};
```

Python

```
def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']

    viewerCountry = request['headers'].get('cloudfront-viewer-country')
    if viewerCountry:
        countryCode = viewerCountry[0]['value']
        if countryCode == 'GB' or countryCode == 'DE' or countryCode == 'IE':
            domainName = 'eu.example.com'
            request['origin']['custom']['domainName'] = domainName
            request['headers']['host'] = [{'key': 'host', 'value': domainName}]
    return request
```

## 오류 상태 업데이트 - 예시

다음 예시에서는 Lambda@Edge를 사용하여 사용자에게 반환되는 오류의 상태를 변경하는 방법에 대한 지침을 제공합니다.

주제

- 예시: 오리진 응답 트리거를 사용하여 오류 상태 코드를 200으로 업데이트
- 예시: 오리진 응답 트리거를 사용하여 오류 상태 코드를 302로 업데이트

## 예시: 오리진 응답 트리거를 사용하여 오류 상태 코드를 200으로 업데이트

이 함수는 응답 상태를 200으로 업데이트하고 정적 본문 콘텐츠를 생성하여 다음 시나리오에서 최종 사용자에게 반환하는 방법을 보여줍니다.

- 함수가 오리진 응답에서 트리거됩니다.
- 오리진 서버의 응답 상태는 오류 상태 코드(4xx 또는 5xx)입니다.

### Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
    const response = event.Records[0].cf.response;

    /**
     * This function updates the response status to 200 and generates static
     * body content to return to the viewer in the following scenario:
     * 1. The function is triggered in an origin response
     * 2. The response status from the origin server is an error status code (4xx or
 5xx)
     */

    if (response.status >= 400 && response.status <= 599) {
        response.status = 200;
        response.statusDescription = 'OK';
        response.body = 'Body generation example';
    }

    callback(null, response);
};
```

### Python

```
def lambda_handler(event, context):
    response = event['Records'][0]['cf']['response']
```

```
    '''
    This function updates the response status to 200 and generates static
    body content to return to the viewer in the following scenario:
    1. The function is triggered in an origin response
    2. The response status from the origin server is an error status code (4xx or
 5xx)
    '''

    if int(response['status']) >= 400 and int(response['status']) <= 599:
        response['status'] = 200
        response['statusDescription'] = 'OK'
        response['body'] = 'Body generation example'
    return response
```

예시: 오리진 응답 트리거를 사용하여 오류 상태 코드를 302로 업데이트

이 함수는 HTTP 상태 코드를 302로 업데이트하여 다른 오리진이 구성된 다른 경로(캐시 동작)로 리디렉션하는 방법을 보여줍니다. 다음을 참조하십시오.

- 함수가 오리진 응답에서 트리거됩니다.
- 오리진 서버의 응답 상태는 오류 상태 코드(4xx 또는 5xx)입니다.

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
    const response = event.Records[0].cf.response;
    const request = event.Records[0].cf.request;

    /**
     * This function updates the HTTP status code in the response to 302, to
 redirect to another
     * path (cache behavior) that has a different origin configured. Note the
 following:
     * 1. The function is triggered in an origin response
     * 2. The response status from the origin server is an error status code (4xx or
 5xx)
     */
```

```
    if (response.status >= 400 && response.status <= 599) {
        const redirect_path = `/plan-b/path?${request.querystring}`;

        response.status = 302;
        response.statusDescription = 'Found';

        /* Drop the body, as it is not required for redirects */
        response.body = '';
        response.headers['location'] = [{ key: 'Location', value: redirect_path }];
    }

    callback(null, response);
};
```

Python

```
def lambda_handler(event, context):
    response = event['Records'][0]['cf']['response']
    request = event['Records'][0]['cf']['request']

    '''
    This function updates the HTTP status code in the response to 302, to redirect
 to another
    path (cache behavior) that has a different origin configured. Note the
 following:
    1. The function is triggered in an origin response
    2. The response status from the origin server is an error status code (4xx or
 5xx)
    '''

    if int(response['status']) >= 400 and int(response['status']) <= 599:
        redirect_path = '/plan-b/path?%s' % request['querystring']

        response['status'] = 302
        response['statusDescription'] = 'Found'

        # Drop the body as it is not required for redirects
        response['body'] = ''
        response['headers']['location'] = [{'key': 'Location', 'value':
 redirect_path}]

    return response
```

## 요청 본문 액세스 - 예시

다음 예시에서는 Lambda@Edge를 사용하여 POST 요청을 처리하는 방법을 보여줍니다.

> **Note**
>
> 이러한 예제를 사용하려면 배포의 Lambda 함수 연결에서 본문 포함 옵션을 활성화해야 합니다. 기본적으로 활성화되어 있지 않습니다.
>
> - CloudFront 콘솔에서 이 설정을 활성화하려면 Lambda 함수 연결(Lambda Function Association)에서 본문 포함(Include Body) 확인란을 선택하세요.
> - CloudFront API 또는 CloudFormation에서 이 설정을 활성화하려면 `LambdaFunctionAssociation`에서 `IncludeBody` 필드를 `true`로 설정합니다.

주제


- 예시: 요청 트리거를 사용하여 HTML 양식 읽기
- 예시: 요청 트리거를 사용하여 HTML 양식 수정


### 예시: 요청 트리거를 사용하여 HTML 양식 읽기

이 기능은 "문의처" 양식 등과 같은 HTML 양식(웹 양식)으로 인해 생성된 POST 요청의 본문을 처리할 수 있는 방법을 보여줍니다. 예를 들어, 다음과 같은 HTML 양식이 있을 수 있습니다.

```
<html>
  <form action="https://example.com" method="post">
    Param 1: <input type="text" name="name1"><br>
    Param 2: <input type="text" name="name2"><br>
    input type="submit" value="Submit">
  </form>
</html>
```

예를 들어, 다음 함수는 CloudFront 최종 사용자 요청 또는 오리진 요청 시 실행되어야 합니다.

Node.js

```
'use strict';

const querystring = require('querystring');
```

```
/**
 * This function demonstrates how you can read the body of a POST request
 * generated by an HTML form (web form). The function is triggered in a
 * CloudFront viewer request or origin request event type.
 */

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    if (request.method === 'POST') {
        /* HTTP body is always passed as base64-encoded string. Decode it. */
        const body = Buffer.from(request.body.data, 'base64').toString();

        /* HTML forms send the data in query string format. Parse it. */
        const params = querystring.parse(body);

        /* For demonstration purposes, we only log the form fields here.
         * You can put your custom logic here. For example, you can store the
         * fields in a database, such as Amazon DynamoDB, and generate a response
         * right from your Lambda@Edge function.
         */
        for (let param in params) {
            console.log(`For "${param}" user submitted "${params[param]}".\n`);
        }
    }
    return callback(null, request);
};
```

## Python

```
import base64
from urllib.parse import parse_qs

'''
Say there is a POST request body generated by an HTML such as:

<html>
<form action="https://example.com" method="post">
    Param 1: <input type="text" name="name1"><br>
    Param 2: <input type="text" name="name2"><br>
    input type="submit" value="Submit">
</form>
```

```
</html>

'''

'''
This function demonstrates how you can read the body of a POST request
generated by an HTML form (web form). The function is triggered in a
CloudFront viewer request or origin request event type.
'''

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']

    if request['method'] == 'POST':
        # HTTP body is always passed as base64-encoded string. Decode it
        body = base64.b64decode(request['body']['data'])

        # HTML forms send the data in query string format. Parse it
        params = {k: v[0] for k, v in parse_qs(body).items()}

        '''
        For demonstration purposes, we only log the form fields here.
        You can put your custom logic here. For example, you can store the
        fields in a database, such as Amazon DynamoDB, and generate a response
        right from your Lambda@Edge function.
        '''
        for key, value in params.items():
            print("For %s use submitted %s" % (key, value))

    return request
```

### 예시: 요청 트리거를 사용하여 HTML 양식 수정

이 기능은 "문의처" 양식 등과 같은 HTML 양식(웹 양식)으로 인해 생성된 POST 요청의 본문을 수정할 수 있는 방법을 보여줍니다. 이 함수는 CloudFront 최종 사용자 요청 또는 오리진 요청 시 실행됩니다.

#### Node.js

```
'use strict';

const querystring = require('querystring');
```

```
exports.handler = (event, context, callback) => {
    var request = event.Records[0].cf.request;
    if (request.method === 'POST') {
        /* Request body is being replaced. To do this, update the following
        /* three fields:
         *    1) body.action to 'replace'
         *    2) body.encoding to the encoding of the new data.
         *
         *       Set to one of the following values:
         *
         *           text - denotes that the generated body is in text format.
         *               Lambda@Edge will propagate this as is.
         *           base64 - denotes that the generated body is base64 encoded.
         *               Lambda@Edge will base64 decode the data before sending
         *               it to the origin.
         *    3) body.data to the new body.
         */
        request.body.action = 'replace';
        request.body.encoding = 'text';
        request.body.data = getUpdatedBody(request);
    }
    callback(null, request);
};

function getUpdatedBody(request) {
    /* HTTP body is always passed as base64-encoded string. Decode it. */
    const body = Buffer.from(request.body.data, 'base64').toString();

    /* HTML forms send data in query string format. Parse it. */
    const params = querystring.parse(body);

    /* For demonstration purposes, we're adding one more param.
     *
     * You can put your custom logic here. For example, you can truncate long
     * bodies from malicious requests.
     */
    params['new-param-name'] = 'new-param-value';
    return querystring.stringify(params);
}
```

## Python

```
import base64
```

```
from urllib.parse import parse_qs, urlencode

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    if request['method'] == 'POST':
        '''
        Request body is being replaced. To do this, update the following
        three fields:
            1) body.action to 'replace'
            2) body.encoding to the encoding of the new data.

            Set to one of the following values:

                text - denotes that the generated body is in text format.
                    Lambda@Edge will propagate this as is.
                base64 - denotes that the generated body is base64 encoded.
                    Lambda@Edge will base64 decode the data before sending
                    it to the origin.
            3) body.data to the new body.
        '''
        request['body']['action'] = 'replace'
        request['body']['encoding'] = 'text'
        request['body']['data'] = getUpdatedBody(request)
    return request

def getUpdatedBody(request):
    # HTTP body is always passed as base64-encoded string. Decode it
    body = base64.b64decode(request['body']['data'])

    # HTML forms send data in query string format. Parse it
    params = {k: v[0] for k, v in parse_qs(body).items()}

    # For demonstration purposes, we're adding one more param

    # You can put your custom logic here. For example, you can truncate long
    # bodies from malicious requests
    params['new-param-name'] = 'new-param-value'
    return urlencode(params)
```

# 엣지 함수에 대한 제한 사항

다음 항목에서는 CloudFront 함수 및 Lambda@Edge에 적용되는 제한 사항에 대해 설명합니다. 일부 제한 사항은 모든 엣지 함수에 적용되지만 다른 제한은 CloudFront 함수 또는 Lambda@Edge에만 적용됩니다.

각 주제에서는 CloudFront를 사용하여 엣지 함수를 개발하고 배포할 때 고려해야 하는 제한 사항 및 제약 조건에 대한 자세한 정보를 제공합니다.

이러한 제약 조건을 이해하면 엣지 함수가 예상대로 작동하고 지원되는 기능을 따르도록 하는 데 도움이 됩니다.

주제

- 모든 엣지 함수에 대한 제한 사항
- CloudFront 함수에 대한 제한
- Lambda@Edge에 대한 제한 사항

할당량(한도라고도 함)에 대한 자세한 내용은 CloudFront 함수의 할당량 및 Lambda@Edge에 대한 할당량 섹션을 참조하세요.

## 모든 엣지 함수에 대한 제한 사항

다음 제한 사항은 CloudFront 함수와 Lambda@Edge의 모든 엣지 함수에 적용됩니다.

주제

- AWS 계정 소유권
- CloudFront 함수와 Lambda@Edge 조합
- HTTP 상태 코드
- HTTP 헤더
- 쿼리 문자열
- URI
- URI, 쿼리 문자열 및 헤더 인코딩
- Microsoft Smooth Streaming
- 태그 지정

## AWS 계정 소유권

엣지 함수를 CloudFront 배포와 연결하려면 동일한 AWS 계정에서 함수와 배포를 소유해야 합니다.

## CloudFront 함수와 Lambda@Edge 조합

지정된 캐시 동작에 대해 다음과 같은 제한이 적용됩니다.

- 각 이벤트 유형(뷰어 요청, 오리진 요청, 오리진 응답 및 뷰어 응답)은 하나의 엣지 함수 연결만 가질 수 있습니다.
- 뷰어 이벤트(뷰어 요청 및 뷰어 응답)에는 CloudFront Functions와 Lambda@Edge를 조합할 수 없습니다.

엣지 함수의 다른 모든 조합이 허용됩니다. 다음 표에서는 허용되는 조합에 대해 설명합니다.

| | | CloudFront 함수 | |
|---|---|---|---|
| | | 뷰어 요청 | 뷰어 응답 |
| Lambda@Edge | 뷰어 요청 | 허용되지 않음 | 허용되지 않음 |
| | 원본 요청 | Allowed | Allowed |
| | 원본 응답 | Allowed | Allowed |
| | 뷰어 응답 | 허용되지 않음 | 허용되지 않음 |

## HTTP 상태 코드

CloudFront는 오리진이 HTTP 상태 코드 400 이상을 반환할 때 뷰어 응답 이벤트에 대해 엣지 함수를 간접적으로 호출하지 않습니다.

오리진 응답 이벤트에 대한 Lambda@Edge 함수는 오리진이 HTTP 상태 코드 400 이상의 코드를 반환하는 경우를 포함한 모든 오리진 응답에 대해 호출됩니다. 자세한 내용은 오리진 응답 트리거에서 HTTP 응답 업데이트 단원을 참조하세요.

## HTTP 헤더

특정 HTTP 헤더는 허용되지 않습니다. 즉, 엣지 함수에 노출되지 않고 함수는 이를 추가할 수 없습니다. 다른 헤더는 읽기 전용이므로 함수가 읽을 수는 있지만 추가, 수정 또는 삭제할 수 없습니다.

주제


- 허용되지 않는 헤더
- 읽기 전용 헤더


## 허용되지 않는 헤더

다음 HTTP 헤더는 엣지 함수에 노출되지 않으며 함수는 이를 추가할 수 없습니다. 함수가 이러한 헤더 중 하나를 추가하면 CloudFront 유효성 검사에 실패하고 CloudFront가 HTTP 상태 코드 502(Bad Gateway)를 뷰어에게 반환합니다.

- `Connection`
- `Expect`
- `Keep-Alive`
- `Proxy-Authenticate`
- `Proxy-Authorization`
- `Proxy-Connection`
- `Trailer`
- `Upgrade`
- `X-Accel-Buffering`
- `X-Accel-Charset`
- `X-Accel-Limit-Rate`
- `X-Accel-Redirect`
- `X-Amz-Cf-*`
- `X-Amzn-Auth`
- `X-Amzn-Cf-Billing`
- `X-Amzn-Cf-Id`
- `X-Amzn-Cf-Xff`
- `X-Amzn-Errortype`
- `X-Amzn-Fle-Profile`
- `X-Amzn-Header-Count`
- `X-Amzn-Header-Order`

- `X-Amzn-Lambda-Integration-Tag`
- `X-Amzn-RequestId`
- `X-Cache`
- `X-Edge-*`
- `X-Forwarded-Proto`
- `X-Real-IP`

읽기 전용 헤더

다음 헤더는 읽기 전용입니다. 함수는 이러한 값을 읽고 함수 논리에 대한 입력으로 사용할 수 있지만 값을 변경할 수는 없습니다. 함수가 읽기 전용 헤더를 추가 또는 편집하는 경우 요청은 CloudFront 유효성 검사에 실패하고 CloudFront는 HTTP 상태 코드 502(잘못된 게이트웨이)를 뷰어에게 반환합니다.

뷰어 요청 이벤트의 읽기 전용 헤더

다음 헤더는 뷰어 요청 이벤트의 읽기 전용 헤더입니다.

- `Content-Length`
- `Host`
- `Transfer-Encoding`
- `Via`

오리진 요청 이벤트의 읽기 전용 헤더(Lambda@Edge 전용)

다음 헤더는 Lambda@Edge에만 있는 오리진 요청 이벤트에서만 읽을 수 있습니다.

- `Accept-Encoding`
- `Content-Length`
- `If-Modified-Since`
- `If-None-Match`
- `If-Range`
- `If-Unmodified-Since`
- `Transfer-Encoding`
- `Via`

### 오리진 응답 이벤트의 읽기 전용 헤더(Lambda@Edge 전용)

다음 헤더는 Lambda@Edge에만 있는 오리진 응답 이벤트에서만 읽을 수 있습니다.

- `Transfer-Encoding`
- `Via`

### 뷰어 응답 이벤트의 읽기 전용 헤더

다음 헤더는 CloudFront Functions와 Lambda@Edge에 대한 뷰어 응답 이벤트에서 읽기 전용입니다.

- `Warning`
- `Via`

다음 헤더는 Lambda@Edge에 대한 뷰어 응답 이벤트에서 읽기 전용입니다.

- `Content-Length`
- `Content-Encoding`
- `Transfer-Encoding`

## 쿼리 문자열

요청 URI에서 쿼리 문자열을 읽거나 업데이트하거나 만드는 함수에는 다음 제한 사항이 적용됩니다.

- (Lambda@Edge 전용) 오리진 요청 또는 오리진 응답 함수에서 쿼리 문자열에 액세스하려면 캐시 정책 또는 오리진 요청 정책을 쿼리 문자열에 대해 모두(ALL)로 설정해야 합니다.
- 함수는 뷰어 요청 및 오리진 요청 이벤트에 대한 쿼리 문자열을 만들거나 업데이트할 수 있습니다(오리진 요청 이벤트는 Lambda@Edge에만 있음).
- 함수는 쿼리 문자열을 읽을 수 있지만 오리진 응답 및 뷰어 응답 이벤트에 대해 쿼리 문자열을 만들거나 업데이트할 수는 없습니다(오리진 응답 이벤트는 Lambda@Edge에만 있음).
- 함수가 쿼리 문자열을 만들거나 업데이트하는 경우 다음과 같은 제한이 적용됩니다.
  - 쿼리 문자열에는 공백, 제어 문자 또는 조각 ID(#)가 포함될 수 없습니다.
  - 쿼리 문자열을 포함한 URI 및 총 크기는 8,192자 미만이어야 합니다.
  - URI 및 쿼리 문자열에 퍼센트 인코딩을 사용하는 것이 좋습니다. 자세한 내용은 URI, 쿼리 문자열 및 헤더 인코딩 섹션을 참조하세요.

## URI

함수가 요청의 URI를 변경하는 경우 이로 인해 요청 또는 요청이 전달되는 오리진에 대한 캐시 동작이 변경되지 않습니다.

쿼리 문자열을 포함한 URI 및 총 크기는 8,192자 미만이어야 합니다.

## URI, 쿼리 문자열 및 헤더 인코딩

엣지 함수에 전달된 URI, 쿼리 문자열 및 헤더 값은 UTF-8로 인코딩됩니다. 함수는 반환하는 URI, 쿼리 문자열 및 헤더 값에 대해 UTF-8 인코딩을 사용해야 합니다. 퍼센트 인코딩은 UTF-8 인코딩과 호환됩니다.

다음 목록에서는 CloudFront가 URI, 쿼리 문자열 및 헤더 값 인코딩을 처리하는 방법을 설명합니다.

- 요청의 값이 UTF-8로 인코딩되면 CloudFront는 값을 변경하지 않고 해당 함수로 바로 전달합니다.
- 요청의 값이 ISO 8859-1로 인코딩되면 CloudFront는 값을 UTF-8 인코딩으로 변환한 후 함수에 전달합니다.
- 다른 문자 인코딩을 사용하여 요청의 값이 인코딩되는 경우 CloudFront는 값이 ISO-8859-1로 인코딩된 것으로 가정하고 ISO-8859-1에서 UTF-8로 변환하려고 시도합니다.

> **Important**
>
> 변환된 문자는 원래 요청에 있는 값의 부정확한 해석일 수 있습니다. 이 경우 해당 함수 또는 오리진에서 의도하지 않은 결과가 생성될 수 있습니다.

CloudFront가 오리진에 전달하는 URI, 쿼리 문자열 및 헤더 값은 함수가 값을 변경하는지 여부에 따라 달라집니다.

- 함수가 URI, 쿼리 문자열 또는 헤더를 변경하지 않는 경우 CloudFront는 CloudFront가 요청에서 수신한 값을 오리진으로 전달합니다.
- 함수가 URI, 쿼리 문자열 또는 헤더를 변경하는 경우 CloudFront는 UTF-8로 인코딩된 값을 전달합니다.

## Microsoft Smooth Streaming

Microsoft Smooth Streaming 형식으로 변환한 미디어 파일을 스트리밍하는 데 사용하는 CloudFront 배포에는 엣지 함수를 사용할 수 없습니다.

## 태그 지정

엣지 함수에 태그를 추가할 수 없습니다. CloudFront의 태그 지정에 대한 자세한 내용은 배포 태깅 섹션을 참조하세요.

# CloudFront 함수에 대한 제한

다음 제한은 CloudFront 함수에만 적용됩니다.

목차


- 로그
- 요청 본문
- CloudFront KeyValueStore API에 임시 자격 증명 사용
- 런타임
- 컴퓨팅 사용률


할당량(한도라고도 함)에 대한 자세한 내용은 CloudFront 함수의 할당량 섹션을 참조하세요.

## 로그

CloudFront 함수의 함수 로그는 10KB로 잘립니다.

## 요청 본문

CloudFront Functions가 HTTP 요청의 본문에 액세스할 수 없습니다.

## CloudFront KeyValueStore API에 임시 자격 증명 사용

AWS Security Token Service(AWS STS)를 사용하여 임시 보안 자격 증명(세션 토큰이라고도 함)을 생성할 수 있습니다. 세션 토큰을 사용하면 일시적으로 AWS Identity and Access Management(IAM) 역할을 맡아 AWS 서비스에 액세스할 수 있습니다.

CloudFront KeyValueStore API를 직접 호출하려면 AWS STS에서 리전 엔드포인트를 사용하여 버전 2 세션 토큰을 반환하세요. AWS STS(`sts.amazonaws.com`)에 글로벌 엔드포인트를 사용하면 AWS STS는 서명 버전 4A(SigV4A)에서 지원하지 않는 버전 1 세션 토큰을 생성합니다. 따라서 인증 오류가 발생할 수 있습니다.

CloudFront KeyValueStore API를 직접 호출하려면 다음 옵션을 사용할 수 있습니다.

AWS CLI 및 AWS SDK

리전 AWS STS 엔드포인트를 사용하도록 AWS CLI 또는 AWS SDK를 구성할 수 있습니다. 자세한 내용은 AWS SDK 및 도구 참조 가이드의 AWS STS Regionalized endpoints를 참조하세요.

사용 가능한 AWS STS 엔드포인트에 대한 자세한 내용은 IAM 사용 설명서의 Regions and endpoints를 참조하세요.

SAML

리전 AWS STS 엔드포인트를 사용하도록 SAML을 구성할 수 있습니다. 자세한 내용은 How to use regional SAML endpoints for failover 블로그 게시물을 참조하세요.

**`SetSecurityTokenServicePreferences`** API

리전 AWS STS 엔드포인트를 사용하는 대신 버전 2 세션 토큰을 반환하도록 AWS STS에 대한 글로벌 엔드포인트를 구성할 수 있습니다. 이렇게 하려면 SetSecurityTokenServicePreferences API 작업을 사용하여 AWS 계정을 구성하세요.

Example예: IAM CLI 명령

```
aws iam set-security-token-service-preferences --global-endpoint-token-version
 v2Token
```

> **Tip**
>
> 이 옵션 대신 AWS STS 리전 엔드포인트를 사용하는 것이 좋습니다. 리전 엔드포인트는 더 높은 가용성 및 장애 조치 시나리오를 제공합니다.

사용자 지정 ID 제공업체

페더레이션을 수행하고 역할을 맡는 사용자 지정 ID 제공업체를 사용하는 경우 세션 토큰 생성을 담당하는 상위 ID 제공업체 시스템에 대한 이전 옵션 중 하나를 사용하세요.

## 런타임

CloudFront Functions 런타임 환경은 동적 코드 평가를 지원하지 않으며 네트워크, 파일 시스템, 환경 변수, 타이머에 대한 액세스를 제한합니다. 자세한 내용은 제한된 기능 섹션을 참조하세요.

> **Note**
>
> CloudFront KeyValueStore를 사용하려면 CloudFront 함수가 JavaScript 런타임 2.0을 사용해야 합니다.

### 컴퓨팅 사용률

CloudFront 함수는 컴퓨팅 사용률로 측정되는 실행 시간에 제한이 있습니다. 컴퓨팅 사용률은 함수가 최대 허용 시간의 백분율로 실행되는 데 걸린 시간을 나타내는 0에서 100 사이의 숫자입니다. 예를 들어 컴퓨팅 사용률이 35이면 함수가 최대 허용 시간의 35%에서 완료되었음을 의미합니다.

함수를 테스트하면 테스트 이벤트의 출력에서 컴퓨팅 사용률 값을 볼 수 있습니다. 프로덕션 함수의 경우, CloudFront 콘솔의 모니터링 페이지 또는 CloudWatch에서 컴퓨팅 사용률 지표를 볼 수 있습니다.

## Lambda@Edge에 대한 제한 사항

다음 제한 사항은 Lambda@Edge에만 적용됩니다.

목차


- DNS 확인
- HTTP 상태 코드
- Lambda 함수 버전
- Lambda 리전
- Lambda 역할 권한
- Lambda 기능
- 지원되는 런타임
- CloudFront 헤더
- 본문 포함 옵션이 적용된 요청 본문에 대한 제한 사항
- 응답 제한 시간 및 연결 유지 제한 시간(사용자 지정 오리진만 해당)


할당량에 대한 자세한 내용은 Lambda@Edge에 대한 할당량 단원을 참조하십시오.

### DNS 확인

CloudFront는 오리진 요청 Lambda@Edge 함수를 실행하기 전에 오리진 도메인 이름에 대해 DNS 확인을 수행합니다. 도메인의 DNS 서비스에 문제가 있고 CloudFront가 도메인 이름을 확인하여

IP 주소를 가져올 수 없는 경우 Lambda@Edge 함수가 간접 호출되지 않습니다. CloudFront는 클라이언트에 HTTP 502 상태 코드(잘못된 게이트웨이)를 반환합니다. 자세한 내용은 DNS 오류 (`NonS3OriginDnsError`) 섹션을 참조하세요.

함수 로직이 오리진 도메인 이름을 수정하는 경우 CloudFront는 함수 실행이 완료된 후 업데이트된 도메인 이름에 대해 또 다른 DNS 확인을 수행합니다.

관리에 대한 자세한 내용은 Amazon Route 53 개발자 안내서의 DNS 장애 조치 구성을 참조합니다.

## HTTP 상태 코드

뷰어 응답 이벤트에 대한 Lambda@Edge 함수는 응답이 오리진에서 왔는지, CloudFront 캐시에서 왔는지와 관계없이 응답의 HTTP 상태 코드를 수정할 수 없습니다.

## Lambda 함수 버전

`$LATEST` 또는 별칭이 아니라 번호가 매겨진 Lambda 함수 버전을 사용해야 합니다.

## Lambda 리전

Lambda 함수는 미국 동부(버지니아 북부) 리전에 있어야 합니다.

## Lambda 역할 권한

Lambda 함수와 연결된 IAM 실행 역할은 서비스 보안 주체 `lambda.amazonaws.com` 및 `edgelambda.amazonaws.com`에서 역할을 수임하도록 허용해야 합니다. 자세한 내용은 Lambda@Edge에 대한 IAM 권한 및 역할 설정 섹션을 참조하세요.

## Lambda 기능

다음 Lambda 함수는 Lambda@Edge에서 지원되지 않습니다.

- 기본값인 Auto(자동)가 아닌 Lambda 런타임 관리 구성
- VPC 내부 리소스에 액세스하도록 Lambda 함수 구성
- Lambda 함수 DLQ(Dead Letter Queue)
- Lambda 환경 변수(자동으로 지원되는 예약된 환경 변수 제외)
- Lambda 함수(계층을 통한 AWS Lambda 종속성 관리 포함)
- AWS X-Ray 사용
- Lambda 프로비저닝된 동시성

> **Note**
>
> Lambda@Edge 함수는 모든 Lambda 함수와 동일한 리전 동시성 기능을 공유합니다. 자세한 내용은 Lambda@Edge에 대한 할당량 섹션을 참조하세요.

- 컨테이너 이미지를 사용하여 Lambda 함수 생성
- arm64 아키텍처를 사용하는 Lambda 함수
- 512MB를 초과하는 임시 스토리지가 있는 Lambda 함수
- .zip 배포 패키지를 암호화하는 데 고객 관리형 키 사용

## 지원되는 런타임

Lambda@Edge는 최신 버전의 Node.js 및 Python 런타임을 지원합니다. 지원되는 버전 및 향후 사용 중단 날짜 목록은 AWS Lambda 개발자 안내서의 지원되는 런타임을 참조하세요.

> **Tip**
>
> - 성능 향상과 새로운 기능을 위해 제공된 런타임의 최신 버전을 사용하는 것이 가장 좋습니다.
> - 더 이상 사용되지 않는 버전의 Node.js로는 함수를 만들거나 업데이트할 수 없습니다. 기존 함수를 이러한 버전의 CloudFront 배포에만 연결할 수 있습니다. 배포와 관련된 이러한 버전의 함수는 계속 실행됩니다. 그러나 함수를 최신 버전의 Node.js로 옮기는 것이 좋습니다. 자세한 내용은 AWS Lambda 개발자 안내서의 런타임 사용 중단 정책과 GitHub의 Node.js 릴리스 일정을 참조하세요.

## CloudFront 헤더

Lambda@Edge 함수는 CloudFront 요청 헤더 추가에 나열된 CloudFront 헤더를 읽고, 편집하고, 제거하거나 추가할 수 있습니다.

> **Notes**
>
> - CloudFront에서 이러한 헤더를 추가하도록 하려면 캐시 정책 또는 오리진 요청 정책을 사용하여 추가하도록 CloudFront를 구성해야 합니다.

- CloudFront는 뷰어 요청 이벤트 이후에 헤더를 추가하므로 뷰어 요청의 Lambda@Edge 함수에서 헤더를 사용할 수 없습니다. 헤더는 오리진 요청 및 오리진 응답의 Lambda@Edge 함수에만 사용할 수 있습니다.
- 뷰어 요청에 이러한 이름을 가진 헤더가 포함되어 있으면 캐시 정책 또는 오리진 요청 정책을 사용하여 해당 헤더를 포함하도록 CloudFront를 구성해야 합니다. 그러면CloudFront는 뷰어 요청에 있었던 헤더 값을 덮어씁니다. 뷰어 방향 함수는 뷰어 요청의 헤더 값을 보는 반면, 오리진 방향 함수는 CloudFront에서 추가한 헤더 값을 확인합니다.
- 뷰어 요청 함수가 `CloudFront-Viewer-Country` 헤더를 추가하면 유효성 검사에 실패하고 CloudFront는 뷰어에 HTTP 상태 코드 502(잘못된 게이트웨이)를 반환합니다.

## 본문 포함 옵션이 적용된 요청 본문에 대한 제한 사항

본문 포함 옵션을 선택해 Lambda@Edge 함수에 요청 본문을 노출하면 노출 또는 교체된 본문 부분에 다음 정보 및 크기 제한이 적용됩니다.

- CloudFront는 항상 요청 본문을 base64로 인코딩한 후 Lambda@Edge에 노출합니다.
- 요청 본문이 크면 CloudFront는 요청 본문이 Lambda@Edge에 노출되기 전에 다음과 같이 자릅니다.
  - 뷰어 요청 이벤트의 경우 본문은 40KB에서 잘립니다.
  - 오리진 요청 이벤트의 경우 본문은 1MB에서 잘립니다.
- 요청 본문에 읽기 전용으로 액세스하는 경우 CloudFront는 전체 원본 요청 본문을 오리진으로 반환합니다.
- Lambda@Edge 함수가 요청 본문을 교체하면 함수가 반환하는 본문에 다음 크기 제한이 적용됩니다.
  - Lambda@Edge 함수가 본문을 일반 텍스트로 반환하는 경우:
    - 뷰어 요청 이벤트의 경우 본문 제한은 40KB입니다.
    - 오리진 요청 이벤트의 경우 본문 제한은 1MB입니다.
  - Lambda@Edge 함수가 본문을 base64로 인코딩된 텍스트로 반환하는 경우:
    - 뷰어 요청 이벤트의 경우 본문 제한은 53.2KB입니다.
    - 오리진 요청 이벤트의 경우 본문 제한은 1.33MB입니다.

> **Note**
>
> Lambda@Edge 함수가 이러한 제한을 초과하는 본문을 반환하는 경우 HTTP 502 상태 코드(Lambda 검증 오류)와 함께 요청이 실패합니다. 본문이 이러한 제한을 초과하지 않도록 Lambda@Edge 함수를 업데이트하는 것이 좋습니다.

## 응답 제한 시간 및 연결 유지 제한 시간(사용자 지정 오리진만 해당)

Lambda@Edge 함수를 사용하여 배포 오리진에 대한 응답 제한 시간 또는 연결 유지 제한 시간을 설정하는 경우 오리진이 지원할 수 있는 값을 지정해야 합니다. 자세한 내용은 응답 및 연결 유지 제한 시간 할당량 섹션을 참조하세요.

# 보고서, 지표 및 로그

CloudFront는 CloudFront 리소스에 대한 여러 가지 보고, 모니터링 및 로깅 옵션을 제공합니다. 다음 작업을 완료할 수 있습니다.

- 보고서를 보고 다운로드하여 CloudFront 배포에 대한 사용량과 활동을 확인합니다. 예를 들어, 결제 명세서, 캐시 통계, 인기 있는 콘텐츠, 인용을 가장 많이 한 사용자 등을 볼 수 있습니다.
- CloudFront 콘솔에서 직접 또는 Amazon CloudWatch를 사용하여 엣지 컴퓨팅 함수를 포함한 CloudFront를 모니터링하고 추적합니다. CloudFront는 배포 및 엣지 함수(Lambda@Edge 및 CloudFront Functions 모두)에 대한 지표를 CloudWatch로 전송합니다.
- CloudFront 배포가 표준 로그 또는 실시간 액세스 로그와 함께 수신하는 뷰어 요청에 대한 로그를 봅니다. 최종 사용자 요청 로그 외에도 CloudWatch Logs를 사용하여 엣지 함수(Lambda@Edge 및 CloudFront Functions 모두)에 대한 로그를 가져올 수 있습니다. AWS CloudTrail을 사용하여 AWS 계정의 CloudFront API 활동 로그를 가져올 수도 있습니다.
- AWS Config를 사용하여 CloudFront 리소스에 대한 구성 변경 사항을 추적합니다.

이러한 각 옵션에 대한 자세한 내용은 다음 주제를 참조하세요.

주제


- CloudFront에 대한 AWS 결제 및 사용 보고서
- 콘솔에서 CloudFront 보고서 보기
- Amazon CloudWatch를 사용한 CloudFront 지표 모니터링
- CloudFront 및 엣지 함수 로깅
- AWS Config를 사용하여 구성 변경 추적


## CloudFront에 대한 AWS 결제 및 사용 보고서

AWS는 CloudFront에 대해 다음 두 가지 사용 보고서를 제공합니다.

- AWS 결제 보고서는 CloudFront를 포함해 사용 중인 AWS 서비스에 대한 모든 활동을 개괄적으로 보여 줍니다.
- AWS 사용 보고서는 특정 서비스에 대한 활동을 시간, 일 또는 월 기준으로 요약합니다. 또한 CloudFront 사용을 그래픽으로 표시하는 사용 차트가 포함되어 있습니다.

> **Note**
>
> 다른 AWS 서비스와 마찬가지로 CloudFront는 사용한 만큼만 과금합니다. 자세한 내용은 CloudFront 요금을 참조하십시오.

주제


- CloudFront에 대한 AWS 결제 보고서
- CloudFront에 대한 AWS 사용 보고서 보기
- CloudFront에 대한 AWS 청구서 및 사용 보고서 해석


## CloudFront에 대한 AWS 결제 보고서

AWS 결제 및 비용 관리 콘솔의 청구서 페이지에서 서비스별로 나열된 AWS 사용 및 요금을 볼 수 있습니다.

AWS 결제 보고서를 보려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/costmanagement/에서 AWS 결제 및 비용 관리콘솔을 엽니다.
2. 탐색 창에서 청구서(Bills)를 선택합니다.
3. 결제 기간(예: 2023년 8월)을 선택합니다.
4. 서비스별 요금 탭에서 CloudFront를 선택한 다음 글로벌 또는 AWS 리전 이름을 확장합니다.
5. CSV 형식으로 세부 결제 보고서를 다운로드하려면 CSV로 모두 다운로드를 선택합니다.

AWS 청구서에 대한 자세한 내용은 AWS Billing 사용 설명서의 청구서 보기를 참조하세요.

결제 보고서에는 CloudFront에 적용되는 다음 값이 포함됩니다.

- ProductCode - `AmazonCloudFront`
- UsageType – 다음 값 중 하나입니다.
  - 데이터 전송의 유형을 식별하는 코드입니다.
  - `Invalidations`
  - `Executions-CloudFrontFunctions`
  - `KeyValueStore-APIOperations`

  - `KeyValueStore-EdgeReads`
  - `RealTimeLog-KinesisDataStream`
  - `SSL-Cert-Custom`
- ItemDescription - UsageType의 청구 요금에 대한 설명입니다.
- UsageStartDate 및 UsageEndDate - 사용이 적용되는 날짜로, UTC(협정 세계시)를 따릅니다.
- UsageQuantity - 다음 값 중 하나입니다.
  - 지정된 기간 동안 요청 수
  - 전송된 데이터의 양(기가바이트 단위)
  - 무효화된 객체 수
  - 활성화된 CloudFront 배포와 SSL 인증서가 연결된 개월 수를 비례 할당으로 계산한 합계입니다. 예를 들어 한 달 내내 활성화된 배포와 연결된 인증서가 하나 있고, 15일 동안 활성화된 배포와 연결된 또 다른 인증서가 있는 경우 이 값은 1.5가 됩니다.

## CloudFront에 대한 AWS 사용 보고서 보기

AWS는 결제 보고서보다 세부적이지만 CloudFront 액세스 로그보다 간략한 CloudFront 사용 보고서를 제공합니다. 사용 보고서는 시간, 일 또는 월 기준으로 누적 사용량 데이터를 제공하며, 오스트레일리아 지역 외부로 전송된 데이터 같은 지역 및 사용 유형에 따른 작업을 나열합니다.

AWS 사용 보고서를 보려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/costmanagement/에서 AWS 결제 및 비용 관리콘솔을 엽니다.
2. 탐색 창에서 Cost Explorer를 선택합니다.
3. 새 비용 및 사용량 보고서 페이지의 보고서 파라미터 창에서 보고서의 날짜 범위와 세부 수준을 선택합니다.
4. 필터, 서비스에서 CloudFront를 선택합니다.
5. 사용량 유형을 선택합니다.
6. 비용 및 사용량 분석에서 CSV로 다운로드를 선택합니다.

AWS 사용 보고서에 대한 자세한 내용은 AWS Data Exports 사용 설명서의 AWS 사용 보고서를 참조하세요.

CloudFront 사용 보고서에는 다음 값이 포함됩니다.

- 서비스 – `AmazonCloudFront`
- Operation - HTTP 메서드입니다. 값에는 `DELETE`, `GET`, `HEAD`, `OPTIONS`, `PATCH`, `POST` 및 `PUT`가 포함됩니다.
- UsageType – 다음 값 중 하나입니다.
  - 데이터 전송의 유형을 식별하는 코드입니다.
  - `Invalidations`
  - `Executions-CloudFrontFunctions`
  - `KeyValueStore-APIOperations`
  - `KeyValueStore-EdgeReads`
  - `RealTimeLog-KinesisDataStream`
  - `SSL-Cert-Custom`
- Resource - 사용과 연결된 CloudFront 배포의 ID 또는 CloudFront 배포와 연결한 SSL 인증서의 인증서 ID입니다.
- StartTime/EndTime - 사용이 적용되는 날짜로, UTC(협정 세계시)를 따릅니다.
- UsageValue - 1) 지정된 기간 동안 요청 수 또는 2) 바이트 단위로 전송된 데이터의 양입니다.

Amazon S3를 CloudFront의 오리진으로 사용하는 경우 Amazon S3에 대해서도 사용 보고서를 실행하는 것이 좋습니다. 그러나 Amazon S3를 CloudFront 배포를 위한 오리진이 아닌 다른 목적으로 사용하는 경우 CloudFront 사용에 어떤 부분이 적용되는지 명확하지 않을 수 있습니다.

> **Tip**
>
> CloudFront가 객체에 대해 수신하는 모든 요청에 대한 자세한 내용을 보려면 배포에 대한 CloudFront 액세스 로그를 활성화하세요. 자세한 내용은 the section called “액세스 로그(표준 로그)” 섹션을 참조하세요.

보고서의 CloudFront 요금 및 사용 유형을 이해하는 방법에 대한 자세한 내용은 the section called “CloudFront에 대한 AWS 청구서 및 사용 보고서 해석” 섹션을 참조하세요.

## CloudFront에 대한 AWS 청구서 및 사용 보고서 해석

결제 보고서와 사용 보고서가 준비되면 이 주제를 통해 청구서에 표시되는 각 CloudFront 요금과 각 요금에 해당하는 사용 유형을 해석하는 방법을 이해할 수 있습니다. 이 주제에는 두 보고서에 모두 표시될 수 있는 코드와 AWS 리전 약어가 포함되어 있습니다.

두 열 모두에 있는 대부분의 코드에는 활동의 위치를 나타내는 두 글자 약어가 포함됩니다. 다음 표에서 코드의 *region*은 AWS 청구서와 사용 보고서에 있는 다음 두 글자 약어 중 하나로 바뀝니다.

- AP: 홍콩, 필리핀, 한국, 대만 및 싱가포르(아시아 태평양)
- AU: 오스트레일리아
- CA: 캐나다
- EU: 유럽 및 이스라엘
- IN: 인도
- JP: 일본
- ME: 중동
- SA: 남미
- US: 미국
- ZA: 남아프리카

AWS 리전별 요금에 대한 자세한 내용은 Amazon CloudFront 요금을 참조하세요.

> **참고**
>
> - 이 표에는 Amazon S3 버킷에서 CloudFront 엣지 로케이션으로 객체를 전송하는 데 드는 요금이 포함되어 있지 않습니다. 이 요금은 존재하는 경우 AWS 청구서의 AWS 데이터 전송 부분에 나타납니다.
> - 첫째 열에는 AWS 결제 보고서에 나타나는 청구 항목이 나열되고 각 항목의 의미가 설명되어 있습니다.
> - 둘째 열에는 AWS 사용 보고서에 나타나는 항목이 나열되고, 청구서 요금과 사용 보고서 항목 간의 상관 관계가 표시됩니다.

| AWS 청구서에 포함된 CloudFront 요금 | AWS 사용 보고서의 UsageType 열에 있는 값 |
| --- | --- |
| *region*-DataTransfer-Out-Bytes<br><br>사용자 GET 및 HEAD 요청에 대한 응답으로 *region*의 CloudFront 엣지 로케이션에서 서비스되는 총 바이트. | *region*-Out-Bytes-HTTP-Static:<br><br>TTL이 3,600초보다 크거나 같은 객체에 대해 HTTP를 통해 서비스되는 바이트.<br><br>*region*-Out-Bytes-HTTPS-Static: |

| AWS 청구서에 포함된 CloudFront 요금 | AWS 사용 보고서의 UsageType 열에 있는 값 |
| --- | --- |
| | TTL이 3,600초보다 크거나 같은 객체에 대해 HTTPS를 통해 서비스되는 바이트.<br><br>*region*-Out-Bytes-HTTP-Dynamic:<br><br>TTL이 3,600초보다 작은 객체에 대해 HTTP를 통해 서비스되는 바이트.<br><br>*region*-Out-Bytes-HTTPS-Dynamic:<br><br>TTL이 3,600초보다 작은 객체에 대해 HTTPS를 통해 서비스되는 바이트.<br><br>*region*-Out-Bytes-HTTP-Proxy:<br><br>`DELETE`, `OPTIONS`, `PATCH`, `POST` 및 `PUT` 요청에 대한 응답으로 HTTP를 통해 CloudFront에서 최종 사용자에게 반환되는 바이트.<br><br>*region*-Out-Bytes-HTTPS-Proxy:<br><br>`DELETE`, `OPTIONS`, `PATCH`, `POST` 및 `PUT` 요청에 대한 응답으로 HTTPS를 통해 CloudFront에서 최종 사용자에게 반환되는 바이트.<br><br>여기에는 gRPC를 통해 CloudFront에서 최종 사용자에게 반환된 바이트가 포함됩니다. |

| AWS 청구서에 포함된 CloudFront 요금 | AWS 사용 보고서의 UsageType 열에 있는 값 |
| --- | --- |
| *region*-DataTransfer-Out-OBytes<br><br>DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 CloudFront 엣지 로케이션에서 오리진 또는 엣지 함수에 전송되는 총 바이트. 요금에는 WebSocket 데이터의 클라이언트에서 서버로의 전송에 따른 금액이 포함됩니다. | *region*-Out-OBytes-HTTP-Proxy<br><br>DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 HTTP를 통해 CloudFront 엣지 로케이션에서 오리진 또는 엣지 함수에 전송되는 총 바이트.<br><br>*region*-Out-OBytes-HTTPS-Proxy<br><br>DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 HTTPS를 통해 CloudFront 엣지 로케이션에서 오리진 또는 엣지 함수에 전송되는 총 바이트.<br><br>여기에는 gRPC를 통해 CloudFront 엣지 로케이션에서 오리진 또는 CloudFront Functions로 전송되는 바이트가 포함됩니다. |
| *region*-Requests-Tier1<br><br>HTTP GET 및 HEAD 요청 수. | *region*-Requests-HTTP-Static<br><br>TTL이 3,600초보다 크거나 같은 객체에 대해 서비스되는 HTTP GET 및 HEAD 요청의 수.<br><br>*region*-Requests-HTTP-Dynamic<br><br>TTL이 3,600초보다 작은 객체에 대해 서비스되는 HTTP GET 및 HEAD 요청의 수. |
| *region*-Requests-Tier2-HTTPS<br><br>HTTPS GET 및 HEAD 요청 수. | *region*-Requests-HTTPS-Static<br><br>TTL이 3,600초보다 크거나 같은 객체에 대해 서비스되는 HTTPS GET 및 HEAD 요청의 수.<br><br>*region*-Requests-HTTPS-Dynamic<br><br>TTL이 3,600초보다 작은 객체에 대해 서비스되는 HTTPS GET 및 HEAD 요청의 수. |

| AWS 청구서에 포함된 CloudFront 요금 | AWS 사용 보고서의 UsageType 열에 있는 값 |
| --- | --- |
| *region*-Requests-HTTP-Proxy<br><br>CloudFront가 오리진 또는 엣지 함수에 전달하는 HTTP DELETE, OPTIONS, PATCH, POST 및 PUT 요청의 수.<br><br>CloudFront가 오리진 또는 엣지 함수에 전달하는 HTTP WebSocket 요청(Upgrade: websocket 헤더가 있는 GET 요청)의 수도 포함됩니다. | *region*-Requests-HTTP-Proxy<br><br>CloudFront 청구서의 해당 항목과 같음. |
| *region*-Requests-HTTPS-Proxy<br><br>CloudFront가 오리진 또는 엣지 함수에 전달하는 HTTPS DELETE, OPTIONS, PATCH, POST 및 PUT 요청의 수.<br><br>다음 요청 유형도 포함됩니다.<br><br>• CloudFront가 오리진 또는 엣지 함수에 전달하는 HTTPS WebSocket 요청(Upgrade: websocket 헤더가 있는 GET 요청)의 수<br>• HTTPS gRPC 요청 수 | *region*-Requests-HTTPS-Proxy<br><br>CloudFront 청구서의 해당 항목과 같음. |
| *region*-Requests-HTTPS-Proxy-FLE<br><br>CloudFront가 오리진 또는 엣지 함수에 전달하는, 필드 레벨 암호화로 처리된 HTTPS DELETE, OPTIONS, PATCH 및 POST 요청의 수. | *region*-Requests-HTTPS-Proxy-FLE<br><br>CloudFront 청구서의 해당 항목과 같음. |
| *region*-Bytes-OriginShield<br><br>오리진에서 리전 엣지 캐시(Origin Shield로 사용하는 리전 엣지 캐시 포함)로 전송된 총 바이트 수입니다. | *region*-Bytes-OriginShield<br><br>CloudFront 청구서의 해당 항목과 같음. |

| AWS 청구서에 포함된 CloudFront 요금 | AWS 사용 보고서의 UsageType 열에 있는 값 |
| --- | --- |
| *region*-OBytes-OriginShield<br><br>리전 엣지 캐시(Origin Shield로 사용하는 리전 엣지 캐시 포함)에서 오리진으로 전송된 총 바이트 수입니다. | *region*-OBytes-OriginShield<br><br>CloudFront 청구서의 해당 항목과 같음. |
| *region*-Requests-OriginShield<br><br>증분 계층으로 사용되는 Origin Shield에 대한 요청의 수입니다. 오리진으로 프록시되는 동적(캐시할 수 없음) 요청의 경우 Origin Shield는 항상 증분 계층입니다. 캐시 가능한 요청의 경우 Origin Shield가 증분 계층인 경우가 있습니다.<br><br>자세한 내용은 the section called “Origin Shield 비용 추정” 섹션을 참조하세요. | *region*-Requests-OriginShield<br><br>CloudFront 청구서의 해당 항목과 같음. |
| 무효화<br><br>객체를 무효화(CloudFront 엣지 로케이션에서 객체 제거)하는 데 드는 요금입니다. 자세한 내용은 파일 무효화에 대한 요금 결제 섹션을 참조하세요. | 무효화<br><br>CloudFront 청구서의 해당 항목과 같음. |
| SSL-Cert-Custom<br><br>SSL 인증서를 기본 CloudFront SSL 인증서 대신 example.com 같은 CloudFront 대체 도메인 이름 및 CloudFront에서 배포에 할당한 도메인 이름과 함께 사용하는 데 드는 요금. | SSL-Cert-Custom<br><br>CloudFront 청구서의 해당 항목과 같음. |
| RealTimeLog-KinesisDataStream<br><br>실시간 액세스 로그에 생성된 라인 수에 대한 요금입니다. | RealTimeLog-KinesisDataStream<br><br>CloudFront 청구서의 해당 항목과 같음. |

| AWS 청구서에 포함된 CloudFront 요금 | AWS 사용 보고서의 UsageType 열에 있는 값 |
| --- | --- |
| Executions-CloudFrontFunctions<br><br>CloudFront Functions 간접 호출 횟수에 대한 요금입니다. | Executions-CloudFrontFunctions<br><br>CloudFront 청구서의 해당 항목과 같음. |
| ***region***-Lambda-Edge-Request<br><br>Lambda@Edge 함수 간접 호출 횟수에 대한 요금입니다. | ***region***-Lambda-Edge-Request<br><br>CloudFront 청구서의 해당 항목과 같음. |
| ***region***-Lambda-Edge-GB-Second<br><br>Lambda@Edge 함수가 간접적으로 호출된 시점부터 반환 또는 종료될 때까지의 기간에 대한 요금입니다. | ***region***-Lambda-Edge-GB-Second<br><br>CloudFront 청구서의 해당 항목과 같음. |
| KeyValueStore-EdgeReads<br><br>CloudFront KeyValueStore 메서드 `get()`, `exists()`, `meta()`의 읽기 직접 호출 횟수에 대한 요금입니다. 자세한 내용은 키 값 저장소를 위한 도우미 메서드 섹션을 참조하세요. | KeyValueStore-EdgeReads<br><br>CloudFront 청구서의 해당 항목과 같음. |
| KeyValueStore-APIOperations<br><br>CloudFront KeyValueStore API의 직접 호출 횟수에 대한 요금입니다. | KeyValueStore-APIOperations<br><br>CloudFront 청구서의 해당 항목과 같음. |

# 콘솔에서 CloudFront 보고서 보기

각 보고서는 상세한 정보와 시각화를 제공하므로 콘텐츠 전송을 최적화하고 성능 병목 현상을 식별하며 데이터에 기반한 결정을 내릴 수 있습니다. 캐시 효율성을 모니터링해야 하든, 트래픽 패턴을 분석해야 하든, 뷰어를 더 잘 이해해야 하든, 이 보고서를 사용하여 CloudFront 배포를 효과적으로 모니터링하고 분석할 수 있습니다.

콘솔에서 CloudFront 활동에 대한 다음 보고서를 볼 수 있습니다.

주제

- CloudFront 캐시 통계 보고서 보기
- CloudFront 인기 객체 보고서 보기
- CloudFront 상위 참조자 보고서
- CloudFront 사용 보고서 보기
- CloudFront 최종 사용자 보고서 보기

이러한 보고서 중 대부분은 CloudFront 액세스 로그의 데이터를 기반으로 합니다. 이 로그에는 CloudFront가 수신하는 모든 사용자 요청에 대한 세부 정보가 포함되어 있습니다. 보고서를 보기 위해 액세스 로그를 활성화할 필요가 없습니다. 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하십시오.

## CloudFront 캐시 통계 보고서 보기

Amazon CloudFront 캐시 통계 보고서에는 다음 정보가 표시됩니다.

- 요청 합계 - 해당 기간 동안의 전체 HTTP 상태 코드(예: 200 또는 404)와 전체 메서드(예: GET, HEAD, POST)의 총 요청 수
- 결과 유형별 뷰어 요청 비율 - 선택한 CloudFront 배포에 대한 총 뷰어 요청 중에서 적중, 누락 및 오류 비율
- 뷰어에 전송된 바이트 - 총 바이트 수와 누락 바이트 수
- HTTP 상태 코드 - HTTP 상태 코드별 뷰어 요청
- 다운로드가 완료되지 않은 GET 요청 비율 - 요청한 객체의 다운로드가 완료되지 않은 뷰어 GET 요청을 총 요청의 백분율로 표시한 값

이러한 통계에 대한 데이터는 CloudFront 액세스 로그와 동일한 소스에서 가져옵니다. 하지만 캐시 통계를 보기 위해 액세스 로깅을 활성화할 필요는 없습니다.

한 시간 또는 하루 기준의 데이터 지점을 포함해 지난 60일 동안 지정된 날짜 범위에 대한 차트를 표시할 수 있습니다. 일반적으로 한 시간 전까지 CloudFront에서 수신한 요청에 대한 데이터를 볼 수 있지만, 이따금씩 데이터가 24시간 가량 지연될 수 있습니다.

주제

- 콘솔에서 CloudFront 캐시 통계 보고서 보기

- 데이터를 CSV 형식으로 다운로드
- 캐시 통계 차트와 CloudFront 액세스 로그 데이터 간의 연결(액세스 로그)

## 콘솔에서 CloudFront 캐시 통계 보고서 보기

콘솔에서 CloudFront 캐시 통계 보고서를 볼 수 있습니다.

CloudFront 캐시 통계 보고서를 보려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 캐시 통계를 선택합니다.
3. CloudFront 캐시 통계 보고서(CloudFront Cache Statistics Reports) 창의 시작일(Start Date) 및 종료일(End Date) 필드에서 캐시 통계 차트를 표시할 날짜 범위를 선택합니다. 사용 가능한 범위는 세부 수준에 선택한 값에 따라 다릅니다.

   - 매일(Daily) - 하루에 데이터 포인트 하나씩 포함하는 차트를 표시하려면 이전 60일에 해당하는 날짜 범위를 선택합니다.
   - 매 시간(Hourly) - 한 시간에 데이터 포인트 하나씩 포함하는 차트를 표시하려면 이전 60일 내에서 날짜 범위를 최대 14일까지 선택합니다.

   날짜 및 시간이 협정 세계시(UTC)로 표시됩니다.
4. 세부 수준의 경우 차트에 하루에 데이터 지점 하나씩 표시할지, 한 시간에 데이터 지점 하나씩 표시할지 지정합니다. 날짜 범위를 14일보다 길게 지정할 경우 한 시간에 데이터 지점 하나를 지정하는 옵션을 사용할 수 없습니다.
5. 최종 사용자에서는 최종 사용자 요청이 시작된 대륙을 선택하거나 All Locations(모든 위치)를 선택합니다. 캐시 통계 보고서에는 CloudFront에서 지정된 위치로부터 수신된 요청에 대한 데이터가 포함됩니다.
6. 배포 목록에서 사용 차트에 데이터를 표시할 배포를 선택합니다.

   - 개별 배포(An individual distribution) - 이 차트에는 선택한 CloudFront 배포에 대한 데이터가 표시됩니다. 배포 목록은 배포에 대한 ID와 대체 도메인 이름(CNAME)이 있는 경우 이를 표시합니다. 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 오리진 도메인 이름이 포함됩니다.
   - 모든 배포 – 이 차트에는 사용자가 삭제한 배포를 제외한 현재 AWS 계정과 연결된 모든 배포에 대해 합산된 데이터가 표시됩니다.
7. 업데이트를 선택합니다.

> **Tip**
>
> - 차트 안에서 일별 또는 시간별 데이터를 보려면 마우스 포인터를 데이터 지점 위로 이동합니다.
> - 전송된 데이터를 보여 주는 차트의 경우 수직 눈금을 기가바이트, 메가바이트 또는 킬로바이트로 변경할 수 있습니다.

## 데이터를 CSV 형식으로 다운로드

캐시 통계 보고서를 CSV 형식으로 다운로드할 수 있습니다. 이 단원에서는 보고서를 다운로드하는 방법과 보고서의 값을 설명합니다.

캐시 통계 보고서를 CSV 형식으로 다운로드하려면

1. 캐시 통계 보고서를 표시한 상태에서 CSV를 선택합니다.
2. 파일 이름 열기 대화 상자에서 파일을 열지 또는 저장할지 여부를 선택합니다.

### 보고서에 대한 정보

보고서의 처음 몇 줄에는 다음 정보가 포함됩니다.

버전

이 CSV 파일 형식의 버전입니다.

보고서

보고서의 이름입니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

StartDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 시작 일시를 협정 세계시(UTC)로 표시합니다.

EndDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 종료 일시를 협정 세계시(UTC)로 표시합니다.

GeneratedTimeUTC

보고서를 실행 날짜 및 시간을 협정 세계시(UTC)로 표시합니다.

Granularity

보고서의 각 줄이 한 시간을 나타내는지 하루를 나타내는지 지정합니다.

ViewerLocation

최종 사용자 요청이 시작된 대륙, 도는 모든 위치에 대한 보고서를 다운로드하도록 선택한 경우 ALL입니다.

캐시 통계 보고서의 데이터

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 ALL입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

ViewerLocation

최종 사용자 요청이 시작된 대륙, 도는 모든 위치에 대한 보고서를 다운로드하도록 선택한 경우 ALL입니다.

TimeBucket

데이터가 적용되는 시간 또는 날짜를 협정 세계시(UTC)로 표시합니다.

RequestCount

해당 기간 동안의 전체 HTTP 상태 코드(예: 200 또는 404)와 전체 메서드(예: GET, HEAD, POST)의 총 요청 수입니다.

HitCount

CloudFront 엣지 캐시에서 객체가 제공되는 최종 사용자 요청 수입니다.

MissCount

현재 엣지 캐시에 객체가 없어서 CloudFront가 오리진에서 객체를 가져와야 하는 최종 사용자 요청 수입니다.

ErrorCount

오류가 발생하여 CloudFront에서 객체를 제공하지 못한 최종 사용자 요청 수입니다.

IncompleteDownloadCount

최종 사용자가 객체 다운로드를 시작했지만 완료하지 못한 최종 사용자 요청 수입니다.

HTTP2xx

HTTP 상태 코드가 2xx 값(성공)인 최종 사용자 요청 수입니다.

HTTP3xx

HTTP 상태 코드가 3xx 값(추가 작업 필요)인 최종 사용자 요청 수입니다.

HTTP4xx

HTTP 상태 코드가 4xx 값(클라이언트 오류)인 최종 사용자 요청 수입니다.

HTTP5xx

HTTP 상태 코드가 5xx 값(서버 오류)인 최종 사용자 요청 수입니다.

TotalBytes

모든 HTTP 메서드에 대한 모든 요청의 응답으로 CloudFront에서 최종 사용자에게 제공한 총 바이트 수입니다.

BytesFromMisses

요청 시 엣지 캐시에 없는 객체에 대해 최종 사용자에게 제공한 바이트 수입니다. 이 값은 오리진에서 CloudFront 엣지 캐시로 전송된 예상 정상 바이트 수입니다. 하지만 이미 엣지 캐시에 있지만 만료된 객체에 대한 요청은 제외됩니다.

## 캐시 통계 차트와 CloudFront 액세스 로그 데이터 간의 연결(액세스 로그)

다음 표에는 CloudFront 콘솔의 캐시 통계 차트가 CloudFront 액세스 로그의 값과 어떻게 연결되는지 보여 줍니다. CloudFront 액세스 로그에 대한 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하세요.

전체 요청

이 차트에는 해당 기간 동안의 전체 HTTP 상태 코드(예: 200 또는 404)와 전체 메서드(예: GET, HEAD 또는 POST)의 총 요청 수가 표시됩니다. 이 차트에 표시된 총 요청 수는 동일 기간 동안의 액세스 로그 파일의 총 요청 수와 동일합니다.

결과 유형별 뷰어 요청의 비율

이 차트에는 선택한 CloudFront 배포에 대한 적중, 누락 및 오류 수를 총 최종 사용자 요청 수의 백분율로 표시합니다.

- 적중(Hit) - 객체가 CloudFront 엣지 캐시에서 제공되는 최종 사용자 요청입니다. 액세스 로그에서 이러한 요청은 `x-edge-response-result-type`의 값이 `Hit`인 요청입니다.
- 누락(Miss) - 객체가 현재 엣지 캐시에 없어서 CloudFront가 객체를 오리진에서 가져와야 하는 최종 사용자 요청입니다. 액세스 로그에서 이러한 요청은 `x-edge-response-result-type`의 값이 `Miss`인 요청입니다.
- 오류(Error) - 오류가 발생하여 CloudFront에서 객체를 제공하지 못한 최종 사용자 요청 수입니다. 액세스 로그에서 이러한 요청은 `x-edge-response-result-type`의 값이 `Error`, `LimitExceeded` 또는 `CapacityExceeded`인 요청입니다.

이 차트에는 엣지 캐시에 있지만 만료된 객체에 대한 적중 새로 고침 요청이 포함되지 않습니다. 액세스 로그에서 적중 새로 고침은 `x-edge-response-result-type`의 값이 `RefreshHit`인 요청입니다.

뷰어로 전송되는 바이트 수

이 차트에는 두 가지 값이 표시됩니다.

- 총 바이트 수(Total bytes) - 모든 HTTP 메서드와 관련된 모든 요청에 대한 응답으로 CloudFront에서 최종 사용자에게 제공한 총 바이트 수입니다. CloudFront 액세스 로그에서 총 바이트 수(Total Bytes)는 동일 기간 동안 모든 요청에 대한 `sc-bytes` 열의 값 합계입니다.
- 누락 바이트 수(Bytes from misses) - 요청 시 엣지 캐시에 없는 객체에 대해 최종 사용자에게 제공한 바이트 수입니다. CloudFront 액세스 로그에서 누락 바이트 수는 `sc-bytes`의 값이 `x-edge-result-type`인 요청에 대한 `Miss` 열의 값 합계입니다. 이 값은 오리진에서 CloudFront 엣지 캐시로 전송된 예상 정상 바이트 수입니다. 하지만 이미 엣지 캐시에 있지만 만료된 객체에 대한 요청은 제외됩니다.

HTTP 상태 코드

이 차트에는 HTTP 상태 코드에 의한 최종 사용자 요청이 표시됩니다. CloudFront 액세스 로그에서 상태 코드는 `sc-status` 열에 나타납니다.

- 2xx - 요청이 성공했습니다.
- 3xx - 추가 작업이 필요합니다. 예를 들어 301(영구 이동됨)은 요청된 객체가 다른 위치로 이동되었음을 의미합니다.
- 4xx - 명백하게 클라이언트로 인해 오류가 발생했습니다. 예를 들어 404(찾을 수 없음)는 클라이언트가 찾을 수 없는 객체를 요청했음을 의미합니다.

- 5xx - 오리진 서버에서 요청을 이행하지 않았습니다. 예를 들어 503(서비스를 사용할 수 없음)은 오리진 서버를 현재 사용할 수 없음을 의미합니다.

다운로드를 마치지 않는 GET 요청의 비율

이 차트에는 요청한 객체의 다운로드를 완료하지 않은 최종 사용자 `GET` 요청이 총 요청 수의 백분율로 표시됩니다. 일반적으로 다른 링크를 클릭하거나 브라우저를 닫는 등의 동작으로 최종 사용자가 다운로드를 취소하여 객체 다운로드가 완료되지 않습니다. CloudFront 액세스 로그에서 이러한 요청은 `200` 열의 값이 `sc-status`이며 `Error` 열의 값은 `x-edge-result-type`입니다.

## CloudFront 인기 객체 보고서 보기

Amazon CloudFront 인기 객체 보고서를 통해 이전 60일 중 지정된 날짜 범위 동안 배포에서 가장 인기 있는 객체 50개를 확인할 수 있습니다. 또한 다음을 포함하여 해당 객체에 대한 통계를 볼 수 있습니다.

- 객체에 대한 요청 수
- 적중 및 누락 수
- 적중률
- 누락에 제공된 바이트 수
- 제공된 총 바이트
- 미완료 다운로드 수
- HTTP 상태 코드별 요청 수(2xx, 3xx, 4xx, 5xx)

이러한 통계에 대한 데이터는 CloudFront 액세스 로그와 동일한 소스에서 가져옵니다. 하지만 인기 객체를 보기 위해 액세스 로깅을 활성화할 필요는 없습니다.

주제

- 콘솔에서 CloudFront 인기 객체 보고서 보기
- CloudFront가 인기 객체 통계를 계산하는 방법
- 데이터를 CSV 형식으로 다운로드
- 인기 객체 보고서 데이터와 CloudFront 표준 로그(액세스 로그) 데이터 간의 연관성

### 콘솔에서 CloudFront 인기 객체 보고서 보기

콘솔에서 CloudFront 인기 객체 보고서를 볼 수 있습니다.

CloudFront 배포에서 인기 있는 객체를 보려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 인기 객체를 선택합니다.
3. CloudFront 인기 객체 보고서(CloudFront Popular Objects Report) 창의 시작일(Start Date) 및 종료일(End Date)에서 인기 객체 목록을 표시할 날짜 범위를 선택합니다. 이전 60일 내에서 원하는 날짜 범위를 선택할 수 있습니다.

   날짜 및 시간이 협정 세계시(UTC)로 표시됩니다.
4. 배포 목록에서 인기 객체 목록을 표시할 배포를 선택합니다.
5. 업데이트를 선택합니다.

## CloudFront가 인기 객체 통계를 계산하는 방법

배포에서 상위 50개 객체의 정확한 개수를 가져오기 위해 CloudFront는 자정부터 10분 간격으로 모든 객체에 대한 요청 수를 세고 다음 24시간 동안 상위 150개 객체의 누계를 기록합니다. 또한 CloudFront는 60일 동안의 상위 150개 객체에 대한 일일 합계도 유지합니다.

목록 아래쪽 부근에서는 객체가 계속해서 목록에 새로 올라오거나 목록에서 누락되므로 이러한 객체의 합계는 근사치입니다. 150개의 객체 목록에서 상위 50개 객체는 목록 내에서 순위가 올라가거나 내려갈 수 있지만, 목록에서 누락되는 일은 거의 드물며, 따라서 이러한 객체의 합계는 신뢰성이 더 높습니다.

어떤 객체가 상위 150개 객체 목록에서 누락된 후 하루가 지나서 다시 올라오는 경우 CloudFront는 해당 객체가 목록에서 누락되었던 기간 동안의 예상 요청 수를 추가합니다. 이 예상치는 객체가 해당 기간 동안 목록 아래쪽에 있을 때 받은 요청 수에 기초합니다.

객체가 당일 늦게 상위 50개 객체 범위 안으로 진입하는 경우 해당 객체가 상위 150개 객체에 포함되지 않았던 시간 동안 CloudFront에서 받은 요청 수의 예상치로 인해 일반적으로 인기 객체 보고서의 요청 수가 액세스 로그에서 해당 객체에 대해 표시되는 요청 수를 초과합니다.

## 데이터를 CSV 형식으로 다운로드

인기 객체 보고서를 CSV 형식으로 다운로드할 수 있습니다. 이 단원에서는 보고서를 다운로드하는 방법과 보고서의 값을 설명합니다.

인기 객체 보고서를 CSV 형식으로 다운로드하려면

1. 인기 객체 보고서를 표시한 상태에서 CSV를 선택합니다.
2. 파일 이름 열기 대화 상자에서 파일을 열지 또는 저장할지 여부를 선택합니다.

## 보고서에 대한 정보

보고서의 처음 몇 줄에는 다음 정보가 포함됩니다.

버전

이 CSV 파일 형식의 버전입니다.

보고서

보고서의 이름입니다.

DistributionID

보고서가 실행된 배포의 ID입니다.

StartDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 시작 일시를 협정 세계시(UTC)로 표시합니다.

EndDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 종료 일시를 협정 세계시(UTC)로 표시합니다.

GeneratedTimeUTC

보고서를 실행 날짜 및 시간을 협정 세계시(UTC)로 표시합니다.

## 인기 객체 보고서의 데이터

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

객체

객체 URL의 마지막 500자입니다.

RequestCount

이 객체에 대한 총 요청 수입니다.

HitCount

CloudFront 엣지 캐시에서 객체가 제공되는 최종 사용자 요청 수입니다.

MissCount

현재 엣지 캐시에 객체가 없어서 CloudFront가 오리진에서 객체를 가져와야 하는 최종 사용자 요청 수입니다.

HitCountPct

`HitCount`의 값을 `RequestCount` 값의 백분율로 표시합니다.

BytesFromMisses

요청 시 객체가 엣지 캐시에 없는 경우 이 객체에 대해 최종 사용자에게 제공된 바이트 수입니다.

TotalBytes

모든 HTTP 메서드와 관련된 모든 요청에 대한 응답으로 CloudFront에서 이 객체에 대해 최종 사용자에게 제공한 총 바이트 수입니다.

IncompleteDownloadCount

이 객체에 대해 최종 사용자가 객체 다운로드를 시작했지만 완료하지 못한 최종 사용자 요청 수입니다.

HTTP2xx

HTTP 상태 코드가 2xx 값(성공)인 최종 사용자 요청 수입니다.

HTTP3xx

HTTP 상태 코드가 3xx 값(추가 작업 필요)인 최종 사용자 요청 수입니다.

HTTP4xx

HTTP 상태 코드가 4xx 값(클라이언트 오류)인 최종 사용자 요청 수입니다.

HTTP5xx

HTTP 상태 코드가 5xx 값(서버 오류)인 최종 사용자 요청 수입니다.

## 인기 객체 보고서 데이터와 CloudFront 표준 로그(액세스 로그) 데이터 간의 연관성

다음 목록은 CloudFront 콘솔의 인기 객체 보고서 값이 CloudFront 액세스 로그의 값과 어떻게 연결되는지 보여 줍니다. CloudFront 액세스 로그에 대한 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하세요.

URL

최종 사용자가 객체에 액세스하는 데 사용하는 마지막 500자입니다.

요청

객체에 대한 총 요청 수입니다. 이 값은 일반적으로 CloudFront 액세스 로그의 객체에 대한 `GET` 요청 수와 거의 같습니다.

Hits

CloudFront 엣지 캐시에서 객체가 제공된 최종 사용자 요청 수입니다. 액세스 로그에서 이러한 요청은 `x-edge-response-result-type`의 값이 `Hit`인 요청입니다.

Misses

엣지 캐시에 객체가 없어서 CloudFront가 오리진에서 객체를 가져왔던 최종 사용자 요청 수입니다. 액세스 로그에서 이러한 요청은 `x-edge-response-result-type`의 값이 `Miss`인 요청입니다.

적중률

Hits(적중 수) 열의 값을 요청 열 값의 백분율로 표시합니다.

누락 바이트 수

요청 시 엣지 캐시에 없는 객체에 대해 최종 사용자에게 제공한 바이트 수입니다. CloudFront 액세스 로그에서 누락 바이트 수는 `sc-bytes`의 값이 `x-edge-result-type`인 요청에 대한 `Miss` 열의 값 합계입니다.

총 바이트 수

모든 HTTP 메서드에 대한 모든 요청의 응답으로 CloudFront에서 해당 객체에 대해 최종 사용자에게 제공한 총 바이트 수입니다. CloudFront 액세스 로그에서 총 바이트 수는 동일 기간 동안 모든 요청에 대한 `sc-bytes` 열의 값 합계입니다.

불완전 다운로드

요청한 객체에 대한 다운로드를 완료하지 못한 최종 사용자 요청 수입니다. 일반적으로 다운로드가 완료되지 않은 이유는 다른 링크를 클릭하거나 브라우저를 닫는 등의 동작으로 최종 사용자가

취소했기 때문입니다. CloudFront 액세스 로그에서 이러한 요청은 `200` 열의 값이 `sc-status`이며 `Error` 열의 값은 `x-edge-result-type`입니다.

2xx

HTTP 상태 코드가 `2xx`, `Successful`인 요청 수입니다. CloudFront 액세스 로그에서 상태 코드는 `sc-status` 열에 나타납니다.

3xx

HTTP 상태 코드가 `3xx`, `Redirection`인 요청 수입니다. `3xx` 상태 코드는 추가 조치가 필요함을 나타냅니다. 예를 들어 301(영구 이동됨)은 요청된 객체가 다른 위치로 이동되었음을 의미합니다.

4xx

HTTP 상태 코드가 `4xx`, `Client Error`인 요청 수입니다. `4xx` 상태 코드는 명백하게 클라이언트로 인해 오류가 발생했음을 나타냅니다. 예를 들어 404(찾을 수 없음)는 클라이언트가 찾을 수 없는 객체를 요청했음을 의미합니다.

5xx

HTTP 상태 코드가 `5xx`, `Server Error`인 요청 수입니다. `5xx` 상태 코드는 오리진 서버에서 요청을 이행하지 않았음을 나타냅니다. 예를 들어 503(서비스를 사용할 수 없음)은 오리진 서버를 현재 사용할 수 없음을 의미합니다.

## CloudFront 상위 참조자 보고서

CloudFront 상위 참조자 보고서는 이전 60일 중 원하는 날짜 범위에 대해 다음 정보를 포함합니다.

- 상위 25개 참조자(CloudFront에서 사용자의 배포에 대해 배포 중인 객체에 대한 대부분의 HTTP 및 HTTPS 요청이 시작된 웹 사이트 도메인)
- 참조자의 요청 수
- 지정된 기간 동안 참조에서 발생한 요청 수를 총 요청 수의 백분율로 표시합니다.

상위 참조자 보고서는 CloudFront 액세스 로그와 동일한 소스에서 가져옵니다. 하지만 상위 참조자를 보기 위해 액세스 로깅을 활성화할 필요는 없습니다.

상위 참조자는 검색 엔진, 객체에 직접 연결하는 기타 웹 사이트 또는 자체 웹 사이트일 수 있습니다. 예를 들어 `https://example.com/index.html`이 10개의 그래픽과 연결되는 경우 `example.com`은 10개 그래픽 모두의 참조자입니다.

> **Note**
> 사용자가 브라우저의 주소 표시줄에 URL을 직접 입력할 경우 다시 요청된 객체에 대한 참조자가 없습니다.

주제


- 콘솔에서 CloudFront 상위 참조자 보고서 보기
- CloudFront에서 상위 참조자 통계를 계산하는 방법
- 데이터를 CSV 형식으로 다운로드
- 상위 참조자 보고서 데이터와 CloudFront 표준 로그(액세스 로그) 데이터 간의 연관성


## 콘솔에서 CloudFront 상위 참조자 보고서 보기

콘솔에서 CloudFront 상위 참조자 보고서를 볼 수 있습니다.

CloudFront 배포에 대한 상위 참조자를 보려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 상위 참조자를 선택합니다.
3. CloudFront 상위 참조자 보고서(CloudFront Top Referrers Report) 창의 시작일(Start Date) 및 종료일(End Date)에서 상위 참조자 목록을 표시할 날짜 범위를 선택합니다.

   날짜 및 시간이 협정 세계시(UTC)로 표시됩니다.
4. 배포 목록에서 상위 참조자 목록을 표시할 배포를 선택합니다.
5. 업데이트를 선택합니다.

## CloudFront에서 상위 참조자 통계를 계산하는 방법

상위 25개 참조자의 정확한 개수를 가져오기 위해 CloudFront는 자정부터 10분 간격으로 모든 객체에 대한 요청 수를 세고 상위 75개 참조자의 누계를 기록합니다. 목록 아래쪽 부근에서는 참조자가 계속해서 목록에 새로 올라오거나 목록에서 누락되므로 이러한 참조자의 합계는 근사치입니다.

75개의 참조자 목록에서 상위 25개 객체는 목록 내에서 순위가 올라가거나 내려갈 수 있지만, 목록에서 누락되는 일은 거의 드물며, 따라서 이러한 참조자의 합계는 일반적으로 신뢰성이 더 높습니다.

## 데이터를 CSV 형식으로 다운로드

상위 참조자 보고서를 CSV 형식으로 다운로드할 수 있습니다. 이 단원에서는 보고서를 다운로드하는 방법과 보고서의 값을 설명합니다.

상위 참조자 보고서를 CSV 형식으로 다운로드하려면

1. 상위 참조자 보고서를 표시한 상태에서 CSV를 선택합니다.
2. 파일 이름 열기 대화 상자에서 파일을 열지 또는 저장할지 여부를 선택합니다.

### 보고서에 대한 정보

보고서의 처음 몇 줄에는 다음 정보가 포함됩니다.

버전

이 CSV 파일 형식의 버전입니다.

보고서

보고서의 이름입니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 ALL입니다.

StartDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 시작 일시를 협정 세계시(UTC)로 표시합니다.

EndDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 종료 일시를 협정 세계시(UTC)로 표시합니다.

GeneratedTimeUTC

보고서를 실행 날짜 및 시간을 협정 세계시(UTC)로 표시합니다.

### 상위 참조자 보고서의 데이터

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 ALL입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

Referrer

참조자의 도메인 이름입니다.

RequestCount

`Referrer` 열의 도메인 이름에서 받은 총 요청 수입니다.

RequestsPct

지정된 기간 동안 참조가 제출한 요청 수를 총 요청 수의 백분율로 표시합니다.

### 상위 참조자 보고서 데이터와 CloudFront 표준 로그(액세스 로그) 데이터 간의 연관성

다음 목록은 CloudFront 콘솔의 상위 참조자 보고서 값이 CloudFront 액세스 로그의 값과 어떻게 연결되는지 보여 줍니다. CloudFront 액세스 로그에 대한 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하세요.

Referrer

참조자의 도메인 이름입니다. 액세스 로그에서 참조자는 `cs(Referer)` 열에 나열됩니다.

요청 수

참조자 열의 도메인 이름에서 받은 총 요청 수입니다. 이 값은 일반적으로 CloudFront 액세스 로그의 참조자에서 받은 `GET` 요청 수와 거의 같습니다.

요청 %

지정된 기간 동안 참조가 제출한 요청 수를 총 요청 수의 백분율로 표시합니다. 참조자 수가 25개 이상이면 이 테이블의 데이터에 따라 요청 %(Request %)를 계산할 수 없습니다. 요청 수(request count) 열에 지정된 기간 동안의 모든 요청이 다 포함되지 않기 때문입니다.

## CloudFront 사용 보고서 보기

CloudFront 사용 보고서에는 다음 정보가 포함됩니다.

- 요청 수(Number of requests) - 지정된 CloudFront 배포에 대한 매 시간 간격 동안 선택한 리전의 엣지 로케이션에서 CloudFront가 응답하는 요청 수를 표시합니다.

- 프로토콜에서 전송한 데이터(Data transferred by protocol) 및 대상에서 전송한 데이터(data transferred by destination) - 둘 다 지정된 CloudFront 배포에 대한 매 시간 간격 동안 선택한 리전의 CloudFront 엣지 로케이션에서 전송한 총 데이터 양을 표시합니다. 다음과 같이 데이터를 구분합니다.
  - 프로토콜별(By protocol) – HTTP 또는 HTTPS 프로토콜로 데이터를 구분합니다
  - 대상별 – 대상(뷰어 또는 오리진)으로 데이터를 구분합니다

CloudFront 사용 보고서는 CloudFront에 대한 AWS 사용 보고서에 기초합니다. 이 보고서에는 추가 구성이 필요하지 않습니다. 자세한 내용은 CloudFront에 대한 AWS 사용 보고서 보기 단원을 참조하십시오.

한 시간 또는 하루 기준의 데이터 지점을 포함해 지난 60일 동안 지정된 날짜 범위에 대한 보고서를 볼 수 있습니다. 일반적으로 네 시간 전까지 CloudFront에서 수신한 요청에 대한 데이터를 볼 수 있지만, 이따금씩 데이터가 24시간 가량 지연될 수 있습니다.

자세한 내용은 사용 차트와 CloudFront 사용 보고서 데이터 간의 연결 단원을 참조하십시오.

주제


- 콘솔에서 CloudFront 보고서 보기
- 데이터를 CSV 형식으로 다운로드
- 사용 차트와 CloudFront 사용 보고서 데이터 간의 연결


## 콘솔에서 CloudFront 보고서 보기

콘솔에서 CloudFront 사용 보고서를 볼 수 있습니다.

CloudFront 사용 보고서를 보려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 Usage Reports(사용 보고서)를 선택합니다.
3. CloudFront 사용 보고서(CloudFront Usage Reports) 창의 시작일(Start Date) 및 종료일(End Date)에서 사용 차트를 표시할 날짜 범위를 선택합니다. 사용 가능한 범위는 세부 수준에 선택한 값에 따라 다릅니다.
   - 매일(Daily) - 하루에 데이터 포인트 하나씩 포함하는 차트를 표시하려면 이전 60일에 해당하는 날짜 범위를 선택합니다.

- 매 시간(Hourly) - 한 시간에 데이터 포인트 하나씩 포함하는 차트를 표시하려면 이전 60일 내에서 날짜 범위를 최대 14일까지 선택합니다.

날짜 및 시간이 협정 세계시(UTC)로 표시됩니다.

4. 세부 수준의 경우 차트에 하루에 데이터 지점 하나씩 표시할지, 한 시간에 데이터 지점 하나씩 표시할지 지정합니다. 날짜 범위를 14일보다 길게 지정할 경우 한 시간에 데이터 지점 하나를 지정하는 옵션을 사용할 수 없습니다.
5. 결제 리전(Billing Region)에서 표시할 데이터가 있는 CloudFront 결제 리전을 선택하거나 모든 리전(All Regions)를 선택합니다. 사용 차트에는 CloudFront가 지정된 리전의 엣지 로케이션에서 처리하는 요청에 대한 데이터가 포함됩니다. CloudFront가 요청을 처리하는 리전은 뷰어의 위치에 해당되거나 해당되지 않을 수 있습니다.

   배포에 대한 가격 등급에 포함된 리전만 선택합니다. 그러지 않으면 사용 차트에 데이터가 포함되지 않을 가능성이 큽니다. 예를 들어 배포에 대해 가격 등급 200을 선택한 경우 남미 및 오스트레일리아 결제 리전이 포함되지 않으므로, CloudFront에서는 일반적으로 이 리전의 요청을 처리하지 않습니다. 요금 등급에 대한 자세한 내용은 CloudFront 요금을 참조합니다.
6. 배포 목록에서 사용 차트에 데이터를 표시할 배포를 선택합니다.
   - 개별 배포(An individual distribution) - 이 차트에는 선택한 CloudFront 배포에 대한 데이터가 표시됩니다. 배포 목록은 배포에 대한 ID와 대체 도메인 이름(CNAME)이 있는 경우 이를 표시합니다. 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.
   - 모든 배포(삭제된 배포 제외)(All distributions (excludes deleted)) — 이 차트에는 사용자가 삭제한 배포를 제외한 현재 AWS 계정과 연결된 모든 배포에 대해 합산된 데이터가 표시됩니다.
   - 삭제된 모든 배포(All Deleted Distributions) — 차트에 현재 AWS 계정과 연결되어 있고 지난 60일 동안 삭제된 모든 배포에 대해 합산된 데이터가 표시됩니다.
7. 그래프 업데이트를 선택합니다.

> **Tip**
>
> - 차트 안에서 일별 또는 시간별 데이터를 보려면 마우스 포인터를 데이터 지점 위로 이동합니다.
> - 전송된 데이터를 보여주는 차트의 경우 수직 눈금을 기가바이트, 메가바이트 또는 킬로바이트로 변경할 수 있습니다.

## 데이터를 CSV 형식으로 다운로드

사용 보고서를 CSV 형식으로 다운로드할 수 있습니다. 이 단원에서는 보고서를 다운로드하는 방법과 보고서의 값을 설명합니다.

사용 보고서를 CSV 형식으로 다운로드하려면

1. 사용 보고서를 표시한 상태에서 CSV를 선택합니다.
2. 파일 이름 열기 대화 상자에서 파일을 열지 또는 저장할지 여부를 선택합니다.

### 보고서에 대한 정보

보고서의 처음 몇 줄에는 다음 정보가 포함됩니다.

버전

이 CSV 파일 형식의 버전입니다.

보고서

보고서의 이름입니다.

DistributionID

보고서를 실행한 배포의 ID로, 모든 배포에 대한 보고서를 실행한 경우 `ALL`, 또는 삭제된 모든 배포에 대한 보고서를 실행한 경우 `ALL_DELETED`입니다.

StartDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 시작 일시를 협정 세계시(UTC)로 표시합니다.

EndDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 종료 일시를 협정 세계시(UTC)로 표시합니다.

GeneratedTimeUTC

보고서를 실행 날짜 및 시간을 협정 세계시(UTC)로 표시합니다.

Granularity

보고서의 각 줄이 한 시간을 나타내는지 하루를 나타내는지 지정합니다.

BillingRegion

최종 사용자 요청이 시작된 대륙, 또는 모든 결제 리전에 대한 보고서를 다운로드하도록 선택한 경우 `ALL`입니다.

### 사용 보고서의 데이터

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서를 실행한 배포의 ID로, 모든 배포에 대한 보고서를 실행한 경우 `ALL`, 또는 삭제된 모든 배포에 대한 보고서를 실행한 경우 `ALL_DELETED`입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

BillingRegion

보고서가 실행된 CloudFront 결제 리전, 또는 `ALL`입니다.

TimeBucket

데이터가 적용되는 시간 또는 날짜를 협정 세계시(UTC)로 표시합니다.

HTTP

지정된 CloudFront 배포에 대한 매 시간 간격 동안 선택한 리전의 엣지 로케이션에서 CloudFront가 응답한 HTTP 요청 수입니다. 값은 다음과 같습니다.

- CloudFront에서 데이터를 뷰어에게 전송하게 하는 `GET` 및 `HEAD` 요청 수
- CloudFront에서 데이터를 오리진으로 전송하게 하는 `DELETE`, `OPTIONS`, `PATCH`, `POST` 및 `PUT` 요청 수

HTTPS

지정된 CloudFront 배포에 대한 매 시간 간격 동안 선택한 리전의 엣지 로케이션에서 CloudFront가 응답한 HTTPS 요청 수입니다. 값은 다음과 같습니다.

- CloudFront에서 데이터를 뷰어에게 전송하게 하는 `GET` 및 `HEAD` 요청 수
- CloudFront에서 데이터를 오리진으로 전송하게 하는 `DELETE`, `OPTIONS`, `PATCH`, `POST` 및 `PUT` 요청 수

HTTPBytes

지정된 CloudFront 배포에 대해 일정 기간 동안 선택된 결제 리전의 CloudFront 엣지 로케이션에서 HTTP를 통해 전송된 총 데이터 양입니다. 값은 다음과 같습니다.

- GET 및 HEAD 요청에 대한 응답으로 CloudFront에서 뷰어에게 전송된 데이터
- DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대해 뷰어에서 CloudFront로 전송된 데이터
- DELETE, OPTIONS, PATCH, POST, PUT 요청에 대한 응답으로 CloudFront에서 뷰어에게 전송된 데이터

HTTPSBytes

지정된 CloudFront 배포에 대해 일정 기간 동안 선택된 결제 리전의 CloudFront 엣지 로케이션에서 HTTPS를 통해 전송된 총 데이터 양입니다. 값은 다음과 같습니다.

- GET 및 HEAD 요청에 대한 응답으로 CloudFront에서 뷰어에게 전송된 데이터
- DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대해 뷰어에서 CloudFront로 전송된 데이터
- DELETE, OPTIONS, PATCH, POST, PUT 요청에 대한 응답으로 CloudFront에서 뷰어에게 전송된 데이터

BytesIn

지정된 CloudFront 배포에 대해 각 기간 동안 선택한 리전의 DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대해 CloudFront에서 오리진으로 전송된 총 데이터 양입니다.

BytesOut

지정된 CloudFront 배포에 대해 일정 기간 동안 HTTP 및 HTTPS를 통해 CloudFront에서 선택된 결제 리전의 뷰어에게 전송된 총 데이터 양입니다. 값은 다음과 같습니다.

- GET 및 HEAD 요청에 대한 응답으로 CloudFront에서 뷰어에게 전송된 데이터
- DELETE, OPTIONS, PATCH, POST, PUT 요청에 대한 응답으로 CloudFront에서 뷰어에게 전송된 데이터

## 사용 차트와 CloudFront 사용 보고서 데이터 간의 연결

다음 목록은 CloudFront 콘솔의 사용 차트가 CloudFront 사용 보고서의 사용 유형(Usage Type) 열의 값과 어떻게 연결되는지 보여 줍니다.

주제


- 요청 수
- 프로토콜별 전송 데이터
- 대상별 전송 데이터

### 요청 수

이 차트는 지정된 CloudFront 배포의 각 시간 간격에서 선택한 리전에 있는 엣지 로케이션에서 CloudFront가 응답하는 요청의 총 수를 프로토콜(HTTP 또는 HTTPS)과 유형(정적, 동적 또는 프록시)으로 구분하여 보여줍니다.

HTTP 요청 수

- *region*-Requests-HTTP-Static: TTL이 3600초보다 크거나 같은 객체에 서비스되는 HTTP `GET` 및 `HEAD` 요청의 수
- *region*-Requests-HTTP-Dynamic: TTL이 3600초보다 작은 객체에 서비스되는 HTTP `GET` 및 `HEAD` 요청의 수
- *region*-Requests-HTTP-Proxy: CloudFront가 오리진으로 전달하는 HTTP `DELETE`, `OPTIONS`, `PATCH`, `POST` 및 `PUT` 요청의 수

HTTPS 요청 수

- *region*-Requests-HTTPS-Static: TTL이 3600초보다 크거나 같은 객체에 서비스되는 HTTPS `GET` 및 `HEAD` 요청의 수
- *region*-Requests-HTTPS-Dynamic: TTL이 3600초보다 작은 객체에 서비스되는 HTTPS `GET` 및 `HEAD` 요청의 수
- *region*-Requests-HTTPS-Proxy: CloudFront가 오리진으로 전달하는 HTTPS `DELETE`, `OPTIONS`, `PATCH`, `POST` 및 `PUT` 요청의 수

### 프로토콜별 전송 데이터

이 차트는 프로토콜(HTTP 또는 HTTPS), 유형(정적, 동적 또는 프록시) 및 대상(뷰어 또는 오리진)으로 구분하여 지정된 CloudFront 배포의 각 시간 간격 동안 선택한 리전의 CloudFront 엣지 로케이션에서 전송된 데이터의 총 양을 보여줍니다.

HTTP를 통해 전송된 데이터

- *region*-Out-Bytes-HTTP-Static: TTL이 3600초보다 크거나 같은 객체에 대해 HTTP를 통해 서비스되는 바이트
- *region*-Out-Bytes-HTTP-Dynamic: TTL이 3600초보다 작은 객체에 대해 HTTP를 통해 서비스되는 바이트
- *region*-Out-Bytes-HTTP-Proxy: `DELETE`, `OPTIONS`, `PATCH`, `POST` 및 `PUT` 요청에 대한 응답으로 HTTP를 통해 CloudFront에서 최종 사용자에게 반환되는 바이트

- *region*-Out-OBytes-HTTP-Proxy: DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 HTTP를 통해 CloudFront 엣지 로케이션에서 오리진으로 전송되는 총 바이트

HTTPS를 통해 전송된 데이터

- *region*-Out-Bytes-HTTPS-Static: TTL이 3600초보다 크거나 같은 객체에 대해 HTTPS를 통해 서비스되는 바이트
- *region*-Out-Bytes-HTTPS-Dynamic: TTL이 3600초보다 작은 객체에 대해 HTTPS를 통해 서비스되는 바이트
- *region*-Out-Bytes-HTTPS-Proxy: DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 HTTPS를 통해 CloudFront에서 최종 사용자에게 반환되는 바이트
- *region*-Out-OBytes-HTTPS-Proxy: DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 HTTPS를 통해 CloudFront 엣지 로케이션에서 오리진으로 전송되는 총 바이트

대상별 전송 데이터

이 차트는 대상(뷰어 또는 오리진), 프로토콜(HTTP 또는 HTTPS), 유형(정적, 동적 또는 프록시)으로 구분하여 지정된 CloudFront 배포의 각 시간 간격 동안 선택한 리전의 CloudFront 엣지 로케이션에서 전송된 데이터의 총 양을 보여줍니다.

CloudFront에서 뷰어에게 전송된 데이터

- *region*-Out-Bytes-HTTP-Static: TTL이 3600초보다 크거나 같은 객체에 대해 HTTP를 통해 서비스되는 바이트
- *region*-Out-Bytes-HTTPS-Static: TTL이 3600초보다 크거나 같은 객체에 대해 HTTPS를 통해 서비스되는 바이트
- *region*-Out-Bytes-HTTP-Dynamic: TTL이 3600초보다 작은 객체에 대해 HTTP를 통해 서비스되는 바이트
- *region*-Out-Bytes-HTTPS-Dynamic: TTL이 3600초보다 작은 객체에 대해 HTTPS를 통해 서비스되는 바이트
- *region*-Out-Bytes-HTTP-Proxy: DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 HTTP를 통해 CloudFront에서 최종 사용자에게 반환되는 바이트
- *region*-Out-Bytes-HTTPS-Proxy: DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 HTTPS를 통해 CloudFront에서 최종 사용자에게 반환되는 바이트

CloudFront에서 오리진으로 전송된 데이터

- *region*-Out-OBytes-HTTP-Proxy: DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 HTTP를 통해 CloudFront 엣지 로케이션에서 오리진으로 전송되는 총 바이트

- *region*-Out-OBytes-HTTPS-Proxy: DELETE, OPTIONS, PATCH, POST 및 PUT 요청에 대한 응답으로 HTTPS를 통해 CloudFront 엣지 로케이션에서 오리진으로 전송되는 총 바이트

## CloudFront 최종 사용자 보고서 보기

CloudFront 상위 참조자 보고서는 이전 60일 중 원하는 날짜 범위에 대해 다음 정보를 포함합니다.

- 디바이스 - 콘텐츠에 액세스하는 데 가장 자주 사용되는 디바이스 유형(예: 데스크톱 또는 모바일)
- 브라우저 - 콘텐츠에 액세스하는 데 가장 자주 사용되는 상위 10개 브라우저(예: Chrome 또는 Firefox)
- 운영 체제 - 콘텐츠에 액세스할 때 가장 자주 사용되는 상위 10개 운영 체제(예: Linux, macOS 또는 Windows)
- 위치 - 콘텐츠에 가장 자주 액세스하는 최종 사용자의 상위 50개 위치(국가 또는 미국 주/지역)
  - 이전 60일 내에서 최대 14일까지 원하는 날짜 범위에 대해 시간별 데이터 지점이 포함된 위치를 볼 수도 있습니다.

> **Note**
>
> 뷰어 차트 및 보고서를 보기 위해 액세스 로깅을 활성화할 필요는 없습니다.

주제


- 콘솔에서 최종 사용자 차트 및 보고서 보기
- 데이터를 CSV 형식으로 다운로드
- 최종 사용자 보고서에 포함되는 데이터
- 위치 보고서 데이터와 CloudFront 표준 로그(액세스 로그) 데이터 간의 연관성


### 콘솔에서 최종 사용자 차트 및 보고서 보기

콘솔에서 CloudFront 최종 사용자 차트 및 보고서를 볼 수 있습니다.

CloudFront 최종 사용자 차트 및 보고서를 보려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.

2. 탐색 창에서 최종 사용자를 선택합니다.
3. CloudFront 최종 사용자(CloudFront Viewers) 창의 시작일(Start Date) 및 종료일(End Date)에서 최종 사용자 차트 및 보고서를 표시할 날짜 범위를 선택합니다.

   Locations 차트의 경우 사용 가능한 범위는 세부 수준에 선택한 값에 따라 다릅니다.

   - 매일(Daily) - 하루에 데이터 포인트 하나씩 포함하는 차트를 표시하려면 이전 60일에 해당하는 날짜 범위를 선택합니다.
   - 매 시간(Hourly) - 한 시간에 데이터 포인트 하나씩 포함하는 차트를 표시하려면 이전 60일 내에서 날짜 범위를 최대 14일까지 선택합니다.

   날짜 및 시간이 협정 세계시(UTC)로 표시됩니다.
4. (Browsers 및 Operating Systems 차트에만 해당) 그룹화에는 이름별로(Chrome, Firefox) 또는 이름 및 버전별로(Chrome 40.0, Firefox 35.0) 브라우저와 운영 체제를 그룹화할지 여부를 지정합니다.
5. (Locations 차트에만 해당) 세부 수준에는 차트에 하루에 데이터 지점 하나씩 표시할지, 한 시간에 데이터 지점 하나씩 표시할지 지정합니다. 날짜 범위를 14일보다 길게 지정할 경우 한 시간에 데이터 지점 하나를 지정하는 옵션을 사용할 수 없습니다.
6. (Locations에만 해당) 세부 정보에는 국가 또는 미국 주별 상위 위치를 표시할지 여부를 지정합니다.
7. 배포 목록에서 사용 차트에 데이터를 표시할 배포를 선택합니다.

   - 개별 배포(An individual distribution) - 이 차트에는 선택한 CloudFront 배포에 대한 데이터가 표시됩니다. 배포 목록에는 배포에 대한 ID와 대체 도메인 이름(CNAME)이 표시됩니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.
   - 모든 배포(삭제된 배포 제외)(All distributions (excludes deleted)) — 이 차트에는 사용자가 삭제한 배포를 제외한 현재 AWS 계정과 연결된 모든 배포에 대해 합산된 데이터가 표시됩니다.
8. 업데이트를 선택합니다.

차트 안에서 일별 또는 시간별 데이터를 보려면 마우스 포인터를 데이터 지점 위로 이동합니다.

## 데이터를 CSV 형식으로 다운로드

각각의 뷰어 보고서를 CSV 형식으로 다운로드할 수 있습니다. 이 단원에서는 보고서를 다운로드하는 방법과 보고서의 값을 설명합니다.

뷰어 보고서를 CSV 형식으로 다운로드하려면

1. 최종 사용자 보고서를 표시한 상태에서 CSV를 선택합니다.
2. 디바이스 또는 Devices Trends(디바이스 추세) 같은, 다운로드할 데이터를 선택합니다.
3. 파일 이름 열기 대화 상자에서 파일을 열지 또는 저장할지 여부를 선택합니다.

## 최종 사용자 보고서에 포함되는 데이터

각 보고서의 처음 몇 줄에는 다음 정보가 포함됩니다.

버전

이 CSV 파일 형식의 버전입니다.

보고서

보고서의 이름입니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

StartDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 시작 일시를 협정 세계시(UTC)로 표시합니다.

EndDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 종료 일시를 협정 세계시(UTC)로 표시합니다.

GeneratedTimeUTC

보고서를 실행 날짜 및 시간을 협정 세계시(UTC)로 표시합니다.

그룹화(브라우저 및 운영 체제 보고서에만 해당)

데이터를 이름별로 그룹화할지 브라우저나 운영 체제의 이름 및 버전별로 그룹화할지 여부를 지정합니다.

Granularity

보고서의 각 줄이 한 시간을 나타내는지 하루를 나타내는지 지정합니다.

세부 정보(위치 보고서에만 해당)

요청을 국가별로 나열할지 또는 미국 주별로 나열할지 여부를 지정합니다.

다음 주제에서는 다양한 최종 사용자 보고서 정보를 설명합니다.

주제


- 디바이스 보고서
- 디바이스 추세 보고서
- 브라우저 보고서
- 브라우저 추세 보고서
- 운영 체제 보고서
- 운영 체제 추세 보고서
- 위치 보고서
- 위치 추세 보고서


## 디바이스 보고서

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

요청

CloudFront가 각 디바이스 유형에서 받은 요청 수입니다.

RequestsPct

CloudFront가 각 디바이스 유형에서 받은 요청 수를 CloudFront가 모든 디바이스에서 받은 총 요청 수의 백분율로 표시합니다.

사용자 지정(Custom)

`User-Agent` HTTP 헤더 값이 표준 디바이스 유형(예: `Desktop` 또는 `Mobile`) 중 하나와 연결되지 않은 요청입니다.

디바이스 추세 보고서

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

TimeBucket

데이터가 적용되는 시간 또는 날짜를 협정 세계시(UTC)로 표시합니다.

Desktop

해당 기간 동안 CloudFront가 데스크톱 컴퓨터에서 받은 요청 수입니다.

모바일

해당 기간 동안 CloudFront가 모바일 디바이스에서 받은 요청 수입니다. 모바일 디바이스에는 태블릿과 휴대폰이 포함될 수 있습니다. CloudFront가 요청이 모바일 디바이스나 태블릿에서 시작되었는지 확인할 수 없는 경우 해당 요청이 `Mobile` 열의 개수에 포함됩니다.

Smart-TV

해당 기간 동안 CloudFront가 스마트 TV에서 받은 요청 수입니다.

Tablet

해당 기간 동안 CloudFront가 태블릿에서 받은 요청 수입니다. CloudFront가 요청이 모바일 디바이스나 태블릿에서 시작되었는지 확인할 수 없는 경우 해당 요청이 `Mobile` 열의 개수에 포함됩니다.

알 수 없음

`User-Agent` HTTP 헤더 값이 표준 디바이스 유형(예: `Desktop` 또는 `Mobile`) 중 하나와 연결되지 않은 요청입니다.

Empty

해당 기간 동안 CloudFront가 받은 요청 중에서 HTTP `User-Agent` 헤더에 값이 포함되지 않은 요청의 수입니다.

### 브라우저 보고서

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

그룹

`Grouping` 값에 따라, CloudFront에서 요청을 받은 브라우저 또는 브라우저 및 버전입니다. 브라우저 이름 외에도 다음과 같은 값을 사용할 수 있습니다.

- 봇/크롤러(Bot/Crawler) - 기본적으로 콘텐츠를 인덱싱하는 검색 엔진의 요청입니다.
- 비어 있음(Empty) - `User-Agent` HTTP 헤더의 값이 비어 있는 요청입니다.
- 기타(Other) - CloudFront에서 식별되었지만 잘 사용되지 않는 브라우저입니다. 처음 9개 값 중에 `Bot/Crawler`, `Empty` 및/또는 `Unknown`이 표시되지 않으면 이러한 값이 `Other`에도 포함됩니다.
- 알 수 없음(Unknown) - `User-Agent` HTTP 헤더의 값이 표준 브라우저와 연결되지 않은 요청입니다. 이 범주에 속하는 대부분의 요청은 사용자 지정 애플리케이션이나 스크립트에서 들어옵니다.

요청

CloudFront가 각 브라우저 유형에서 받은 요청 수입니다.

RequestsPct

CloudFront가 각 브라우저 유형에서 받은 요청 수를 해당 기간 동안 CloudFront가 받은 총 요청 수의 백분율로 표시합니다.

### 브라우저 추세 보고서

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

TimeBucket

데이터가 적용되는 시간 또는 날짜를 협정 세계시(UTC)로 표시합니다.

(브라우저)

보고서의 나머지 열에는 `Grouping` 값에 따라 브라우저 또는 브라우저 및 버전이 나열됩니다. 브라우저 이름 외에도 다음과 같은 값을 사용할 수 있습니다.

- 봇/크롤러(Bot/Crawler) - 기본적으로 콘텐츠를 인덱싱하는 검색 엔진의 요청입니다.
- 비어 있음(Empty) - `User-Agent` HTTP 헤더의 값이 비어 있는 요청입니다.
- 기타(Other) - CloudFront에서 식별되었지만 잘 사용되지 않는 브라우저입니다. 처음 9개 값 중에 `Bot/Crawler`, `Empty` 및/또는 `Unknown`이 표시되지 않으면 이러한 값이 `Other`에도 포함됩니다.
- 알 수 없음(Unknown) - `User-Agent` HTTP 헤더의 값이 표준 브라우저와 연결되지 않은 요청입니다. 이 범주에 속하는 대부분의 요청은 사용자 지정 애플리케이션이나 스크립트에서 들어옵니다.

운영 체제 보고서

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

그룹

`Grouping` 값에 따라, CloudFront에서 요청을 받은 운영 체제 또는 운영 체제 및 버전입니다. 운영 체제 이름 외에도 다음과 같은 값을 사용할 수 있습니다.

- 봇/크롤러(Bot/Crawler) - 기본적으로 콘텐츠를 인덱싱하는 검색 엔진의 요청입니다.
- 비어 있음(Empty) - `User-Agent` HTTP 헤더의 값이 비어 있는 요청입니다.

- 기타(Other) - CloudFront에서 식별되었지만 잘 사용되지 않는 운영 체제입니다. 처음 9개 값 중에 `Bot/Crawler`, `Empty` 및/또는 `Unknown`이 표시되지 않으면 이러한 값이 `Other`에도 포함됩니다.
- 알 수 없음(Unknown) - `User-Agent` HTTP 헤더의 값이 표준 브라우저와 연결되지 않은 요청입니다. 이 범주에 속하는 대부분의 요청은 사용자 지정 애플리케이션이나 스크립트에서 들어옵니다.

요청

CloudFront가 각 운영 체제 유형에서 받은 요청 수입니다.

RequestsPct

CloudFront가 각 운영 체제 유형에서 받은 요청 수를 해당 기간 동안 CloudFront가 받은 총 요청 수의 백분율로 표시합니다.

운영 체제 추세 보고서

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

TimeBucket

데이터가 적용되는 시간 또는 날짜를 협정 세계시(UTC)로 표시합니다.

(운영 체제)

보고서의 나머지 열에는 `Grouping` 값에 따라 운영 체제 또는 운영 체제 및 버전이 나열됩니다. 운영 체제 이름 외에도 다음과 같은 값을 사용할 수 있습니다.

- 봇/크롤러(Bot/Crawler) - 기본적으로 콘텐츠를 인덱싱하는 검색 엔진의 요청입니다.
- 비어 있음(Empty) - `User-Agent` HTTP 헤더의 값이 비어 있는 요청입니다.
- 기타(Other) - CloudFront에서 식별되었지만 잘 사용되지 않는 운영 체제입니다. 처음 9개 값 중에 `Bot/Crawler`, `Empty` 및/또는 `Unknown`이 표시되지 않으면 이러한 값이 `Other`에도 포함됩니다.

- 알 수 없음(Unknown) - `User-Agent` HTTP 헤더에 운영 체제가 지정되지 않은 요청입니다.

### 위치 보고서

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

LocationCode

CloudFront에서 요청을 받은 위치의 약어입니다. 사용 가능한 값에 대한 자세한 내용은 위치 보고서 데이터와 CloudFront 표준 로그(액세스 로그) 데이터 간의 연관성의 Location 설명을 참조하십시오.

LocationName

CloudFront에서 요청을 받은 위치의 이름입니다.

요청

CloudFront가 각 위치에서 받은 요청 수입니다.

RequestsPct

CloudFront가 각 위치에서 받은 요청 수를 해당 기간 동안 CloudFront가 모든 위치에서 받은 총 요청 수의 백분율로 표시합니다.

TotalBytes

CloudFront가 지정된 배포 및 기간 동안 이 국가나 주의 최종 사용자에게 제공한 바이트 수입니다.

### 위치 추세 보고서

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서가 실행된 배포의 ID 또는, 모든 배포에 대한 보고서를 실행한 경우 `ALL`입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

TimeBucket

데이터가 적용되는 시간 또는 날짜를 협정 세계시(UTC)로 표시합니다.

(위치)

보고서의 나머지 열에는 CloudFront가 요청을 받은 위치가 나열됩니다. 사용 가능한 값에 대한 자세한 내용은 위치 보고서 데이터와 CloudFront 표준 로그(액세스 로그) 데이터 간의 연관성의 Location 설명을 참조하십시오.

## 위치 보고서 데이터와 CloudFront 표준 로그(액세스 로그) 데이터 간의 연관성

다음 목록은 CloudFront 콘솔의 Locations 보고서 데이터가 CloudFront 액세스 로그의 값과 어떻게 연결되는지 보여 줍니다. CloudFront 액세스 로그에 대한 자세한 내용은 액세스 로그(표준 로그) 단원을 참조하세요.

위치

최종 사용자가 위치한 국가 또는 미국 주입니다. 액세스 로그에서 `c-ip` 열에는 최종 사용자가 실행 중인 디바이스의 IP 주소가 포함됩니다. 지리적 위치 데이터를 사용하여 IP 주소에 따라 디바이스의 지리적 위치를 식별합니다.

국가별로 위치 보고서를 표시하는 경우 국가 목록은 ISO 3166-2, Codes for the representation of names of countries and their subdivisions – Part 2: Country subdivision code에 기초합니다. 국가 목록에는 다음 추가 값이 포함됩니다.

- 익명 프록시(Anonymous Proxy) - 익명 프록시에서 시작된 요청입니다.
- 위성 공급자(Satellite Provider) - 여러 국가에 인터넷 서비스를 제공하는 위성 공급자로부터 시작된 요청입니다. 뷰어가 사기 위험이 높은 국가에 있을 수 있습니다.
- 유럽(알 수 없음(Europe (Unknown)) - 여러 유럽 국가에서 사용되는 블록의 IP에서 시작된 요청입니다. 요청이 시작된 국가를 확인할 수 없습니다. CloudFront는 유럽(알 수 없음)(Europe (Unknown))을 기본값으로 사용합니다.
- 아시아/태평양(알 수 없음)(Asia/Pacific (Unknown)) - 여러 아시아/태평양 리전 국가에서 사용되는 블록의 IP에서 시작된 요청입니다. 요청이 시작된 국가를 확인할 수 없습니다. CloudFront는 아시아/태평양(알 수 없음)(Asia/Pacific (Unknown))을 기본값으로 사용합니다.

미국 주별로 위치 보고서를 표시하는 경우 보고서에 미국 지역과 미군 주둔 리전이 포함될 수 있습니다.

> **Note**
>
> CloudFront에서 사용자 위치를 확인할 수 없는 경우에는 최종 사용자 보고서에 위치가 "알 수 없음"으로 표시됩니다.

Request Count

지정된 배포 및 기간 동안 최종 사용자가 위치한 국가나 미국 주에서 받은 총 요청 수입니다. 일반적으로 이 값은 CloudFront 액세스 로그의 해당 국가 또는 주의 IP 주소에서 받은 GET 요청 수와 거의 같습니다.

요청 %

세부 정보에 선택한 값에 따라 다음 중 하나입니다.

- 국가(Countries) - 이 국가의 요청을 총 요청 수의 백분율로 표시합니다.
- 주(U.S. States) - 이 주의 요청을 미국 내 총 요청 수의 백분율로 표시합니다.

50개 이상의 국가에서 요청을 받은 경우 이 테이블의 데이터에 따라 Request %(요청 %)를 계산할 수 없습니다. Request Count(요청 수) 열에 지정된 기간 동안의 모든 요청이 다 포함되지 않기 때문입니다.

바이트

CloudFront가 지정된 배포 및 기간 동안 이 국가나 주의 최종 사용자에게 제공한 바이트 수입니다. 이 열의 데이터 표시를 KB, MB 또는 GB로 변경하려면 열 머리글의 링크를 선택합니다.

# Amazon CloudWatch를 사용한 CloudFront 지표 모니터링

Amazon CloudFront는 Amazon CloudWatch와 통합되어 배포 및 엣지 함수(Lambda@Edge 및 CloudFront Functions 모두)에 대한 운영 지표를 자동으로 게시합니다. 이러한 지표를 사용하여 문제를 해결, 추적 및 디버깅할 수 있습니다. 이러한 지표 중 다수는 CloudFront 콘솔에서 일련의 그래프로 표시되며 CloudFront API 또는 CLI를 사용하여 액세스할 수도 있습니다. 이러한 모든 지표는 CloudWatch 콘솔에서 또는 CloudWatch API 또는 CLI를 통해 사용할 수 있습니다. CloudFront 지표는 CloudWatch 할당량(이전에는 한도라고 함)이 적용되지 않아 추가 비용이 발생하지 않습니다.

CloudFront 배포에 대한 기본 지표 외에도 추가 비용이 드는 추가 지표를 활성화할 수 있습니다. 추가 지표는 CloudFront 배포에 적용되며, 각 배포에 대해 개별적으로 활성화해야 합니다. 비용에 관한 자세한 내용은 the section called "추가 CloudFront 지표에 대한 비용 추정" 단원을 참조하십시오.

CloudFront 콘솔이나 CloudWatch 콘솔, API 또는 CLI에서 이러한 지표에 따라 경보를 설정할 수도 있습니다. 예를 들어, 응답의 HTTP 상태 코드가 `5xxErrorRate~500`(경계값 포함) 범위에 있는 모든 최종 사용자 요청의 백분율을 나타내는 `599` 지표에 따라 경보를 설정할 수 있습니다. 오류 발생률이 일정 시간 동안 일정 값에 도달하면(예: 연속 5분 동안 요청의 5%), 경보가 트리거됩니다. 경보를 만들 때 경보의 값과 시간 단위를 지정합니다.

> **Notes**
>
> - CloudFront 콘솔에서 CloudWatch 경보를 생성하면 미국 동부(버지니아 북부) 리전(`us-east-1`)에 경보가 하나 생성됩니다. CloudWatch 콘솔에서 경보를 생성하는 경우 동일한 리전을 사용해야 합니다. CloudFront는 글로벌 서비스이므로 이 서비스에 대한 지표는 미국 동부(버지니아 북부)로 전송됩니다.
> - 경보를 생성하는 데는 표준 CloudWatch 요금이 적용됩니다.

주제


- CloudFront 및 엣지 함수 지표 보기
- 지표에 대한 경보 생성
- 지표 데이터를 CSV 형식으로 다운로드
- CloudFront의 지표 유형


## CloudFront 및 엣지 함수 지표 보기

CloudFront 콘솔에서 CloudFront 배포 및 엣지 함수에 대한 운영 지표를 볼 수 있습니다.

CloudFront에서 CloudFront 및 엣지 함수 지표를 보려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 모니터링을 선택합니다.

3. 특정 CloudFront 배포 또는 엣지 함수의 활동에 대한 그래프를 보려면 그래프를 선택한 후 View distribution metrics(배포 지표 보기) 또는 View metrics(지표 보기)를 선택합니다.
4. 다음을 수행하여 그래프를 사용자 지정할 수 있습니다.
   a. 그래프에 표시된 정보의 시간 범위를 변경하려면 1h(1시간), 3h(3시간) 또는 다른 범위를 선택하거나 사용자 지정 범위를 지정합니다.
   b. CloudFront가 그래프의 정보를 업데이트하는 빈도를 변경하려면 새로 고침 아이콘 옆에 있는 아래쪽 화살표를 선택하고 새로 고침 속도를 선택합니다. 기본 새로 고침 빈도는 1분이지만 다른 옵션을 선택할 수 있습니다.
5. CloudFront 콘솔에서 CloudFront 그래프를 보려면 대시보드에 추가를 선택합니다. CloudWatch 콘솔에서 그래프를 보려면 미국 동부(버지니아 북부) 리전을 사용해야 합니다.

주제


- 기본 CloudFront 배포 지표
- 추가 CloudFront 배포 지표 활성화
- 기본 Lambda @Edge 함수 지표
- 기본 CloudFront Functions 지표


## 기본 CloudFront 배포 지표

다음 기본 지표는 모든 CloudFront 배포에 포함되며, 추가 비용은 들지 않습니다.

요청

모든 HTTP 메서드와 HTTP 및 HTTPS 요청에 대해 CloudFront에서 수신한 최종 사용자 요청의 총 수입니다.

다운로드된 바이트

`GET` 및 `HEAD` 요청에 대해 뷰어가 다운로드한 총 바이트 수입니다.

업로드된 바이트

뷰어가 `OPTIONS`, `POST` 및 `PUT` 요청을 사용하여 CloudFront에 업로드한 총 바이트 수입니다.

4xx 오류 발생률

응답의 HTTP 상태 코드가 `4xx`인 모든 최종 사용자 요청의 백분율입니다.

5xx 오류 발생률

응답의 HTTP 상태 코드가 5xx인 모든 최종 사용자 요청의 백분율입니다.

총 오류 발생률

응답의 HTTP 상태 코드가 4xx 또는 5xx인 모든 최종 사용자 요청의 백분율입니다.

이러한 지표는 CloudFront 콘솔의 모니터링 페이지에서 각 CloudFront 배포에 대해 그래프로 표시됩니다. 각 그래프에서 합계는 1분 간격으로 표시됩니다. 그래프를 보는 것 외에도 지표 보고서를 CSV 파일로 다운로드할 수도 있습니다.

## 추가 CloudFront 배포 지표 활성화

기본 지표 외에도 추가 비용이 드는 추가 지표를 활성화할 수 있습니다. 비용에 관한 자세한 내용은 the section called “추가 CloudFront 지표에 대한 비용 추정” 단원을 참조하십시오.

각 배포에 대해 다음 추가 지표를 개별적으로 활성화해야 합니다.

캐시 적중률

CloudFront가 캐시에서 콘텐츠를 제공하는 캐시 가능한 모든 요청의 비율입니다. HTTP POST 및 PUT 요청, 오류는 캐시 가능한 요청으로 간주되지 않습니다.

오리진 지연 시간

CloudFront에서 요청을 수신한 시점부터 CloudFront 캐시가 아닌 오리진에서 제공되는 요청에 대해 네트워크(최종 사용자 아님)에 대한 응답을 제공하기 시작한 시점까지 소요되는 총 시간입니다. 이것을 첫 번째 바이트 대기 시간 또는 time-to-first-byte라고도 합니다.

상태 코드별 오류 발생률

응답의 HTTP 상태 코드가 4xx 또는 5xx 범위의 특정 코드인 모든 최종 사용자 요청의 백분율입니다. 이 지표는 오류 코드 401, 403, 404, 502, 503, 504에 모두 사용할 수 있습니다.

CloudFormation, AWS Command Line Interface(AWS CLI) 또는 CloudFront API를 사용하여 CloudFront 콘솔에서 추가 지표를 활성화할 수 있습니다.

### Console

추가 지표를 활성화하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 모니터링을 선택합니다.
3. 추가 지표를 활성화할 배포를 선택한 다음 배포 지표 보기를 선택합니다.
4. Manage additional metrics(추가 지표 관리)를 선택합니다.
5. Manage additional metrics(추가 지표 관리) 창에서 Enabled(활성화됨)를 켭니다. 추가 지표를 활성화한 후에는 Manage additional metrics(추가 지표 관리) 창을 닫을 수 있습니다.

   추가 지표를 활성화하면 그래프로 표시됩니다. 각 그래프에서 합계는 1분 간격으로 표시됩니다. 그래프를 보는 것 외에도 지표 보고서를 CSV 파일로 다운로드할 수도 있습니다.

### CloudFormation

CloudFormation에서 추가 지표를 활성화하려는 경우 `AWS::CloudFront::MonitoringSubscription` 리소스 유형을 사용합니다. 다음 예는 추가 지표를 활성화하는 CloudFormation 템플릿 구문을 YAML 형식으로 보여줍니다.

```
Type: AWS::CloudFront::MonitoringSubscription
Properties:
  DistributionId: EDFDVBD6EXAMPLE
  MonitoringSubscription:
    RealtimeMetricsSubscriptionConfig:
      RealtimeMetricsSubscriptionStatus: Enabled
```

### CLI

AWS Command Line Interface(AWS CLI)를 사용하여 추가 지표를 관리하려면 다음 명령 중 하나를 사용합니다.

배포에 대한 추가 지표를 활성화하려면

- 다음 예제에서처럼 create-monitoring-subscription 명령을 사용합니다. *EDFDVBD6EXAMPLE*을 추가 지표를 활성화할 배포의 ID로 바꿉니다.

```
aws cloudfront create-monitoring-subscription --
distribution-id EDFDVBD6EXAMPLE --monitoring-subscription
 RealtimeMetricsSubscriptionConfig={RealtimeMetricsSubscriptionStatus=Enabled}
```

배포에 대해 추가 지표가 활성화되어 있는지 여부를 확인하려면

- 다음 예제에서처럼 get-monitoring-subscription 명령을 사용합니다. *EDFDVBD6EXAMPLE*을 확인할 배포의 ID로 바꿉니다.

```
aws cloudfront get-monitoring-subscription --distribution-id EDFDVBD6EXAMPLE
```

배포에 대한 추가 지표를 비활성화하려면

- 다음 예제에서처럼 delete-monitoring-subscription 명령을 사용합니다. *EDFDVBD6EXAMPLE*을 추가 지표를 비활성화할 배포의 ID로 바꿉니다.

```
aws cloudfront delete-monitoring-subscription --distribution-id EDFDVBD6EXAMPLE
```

API

CloudFront API를 사용하여 추가 지표를 관리하려면 다음 API 연산 중 하나를 사용합니다.

- 배포에 대한 추가 지표를 활성화하려면 CreateMonitoringSubscription을 사용합니다.
- 배포에 대해 추가 지표가 활성화되어 있는지 여부를 확인하려면 GetMonitoringSubscription을 사용합니다.
- 배포에 대한 추가 지표를 비활성화하려면 DeleteMonitoringSubscription을 사용합니다.

이러한 API 작업에 대한 자세한 내용은 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

## 추가 CloudFront 지표에 대한 비용 추정

배포에 대한 추가 지표를 활성화하면 CloudFront는 미국 동부(버지니아 북부) 리전의 CloudWatch에 최대 8개의 지표를 전송합니다. CloudWatch는 각 지표에 대해 낮은 정액제 요금을 부과합니다. 이 요

금은 지표당 한 달에 한 번만 청구됩니다(배포당 최대 8개 지표). 이는 정액제 요금이므로 CloudFront 배포가 보내거나 받는 요청 또는 응답 수에 관계없이 비용이 동일하게 유지됩니다. 지표당 청구되는 요금은 Amazon CloudWatch 요금 페이지 및 CloudWatch 요금 계산기를 참조하십시오. CloudWatch API를 사용하여 지표를 검색할 때 추가 API 요금이 적용됩니다.

## 기본 Lambda @Edge 함수 지표

CloudWatch 지표를 사용하여 Lambda@Edge 함수의 문제를 실시간으로 모니터링할 수 있습니다. 이러한 지표에 대한 추가 요금은 없습니다.

Lambda@Edge 함수를 CloudFront 배포의 캐시 동작에 연결하면 Lambda가 CloudWatch에 지표를 자동으로 전송하기 시작합니다. 지표는 모든 Lambda 리전에서 사용할 수 있지만 CloudWatch 콘솔에서 지표를 보거나 CloudWatch API에서 지표 데이터를 가져오려면 미국 동부(버지니아 북부) 리전(`us-east-1`)을 사용해야 합니다. 지표 그룹 이름은 `AWS/CloudFront/`*`distribution-ID`* 형식으로 지정됩니다. 여기서 *`distribution-ID`*는 Lambda@Edge 함수에 연결된 CloudFront 배포의 ID입니다. CloudWatch 지표에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하세요.

다음 기본 지표는 CloudFront 콘솔의 모니터링 페이지에서 각 Lambda@Edge 함수에 대해 그래프로 표시됩니다.

- `5xx`Lambda@Edge의 오류 발생률
- Lambda 실행 오류
- 잘못된 Lambda 응답
- Lambda 제한

그래프에는 호출 수, 오류 수, 제한 수 등이 포함되어 있습니다. 각 그래프에서 합계는 1분 간격으로 표시되며 AWS 리전별로 그룹화됩니다.

조사할 오류가 급증할 경우 함수를 선택한 다음 AWS 리전별로 로그 파일을 확인하여, 어떤 함수가 문제를 야기했는지 및 어느 AWS 리전에서 발생했는지 파악할 수 있습니다. Lambda@Edge 오류 문제 해결에 대한 자세한 내용은 다음 단원을 참조하십시오.

- the section called “실패 유형을 결정하는 방법”
- AWS에서 콘텐츠 전송 디버깅을 위한 4단계

## 기본 CloudFront Functions 지표

CloudFront 함수는 사용자가 기능을 모니터링할 수 있도록 운영 지표를 Amazon CloudWatch로 전송합니다. 이러한 지표를 보면 문제 해결, 추적 및 디버깅에 도움이 될 수 있습니다. CloudFront 함수는 다음과 같은 지표를 CloudWatch에 게시합니다.

- 간접 호출 (`FunctionInvocations`) — 지정된 기간 동안 함수가 시작된(호출된) 횟수입니다.
- 유효성 검사 오류 (`FunctionValidationErrors`) — 지정된 기간 동안 함수에서 생성된 유효성 검사 오류의 수입니다. 유효성 검사 오류는 함수가 성공적으로 실행되지만 잘못된 데이터(잘못된 이벤트 객체)를 반환할 때 발생합니다.
- 실행 오류 (`FunctionExecutionErrors`) — 지정된 기간 동안 발생한 실행 오류의 수입니다. 함수가 성공적으로 완료되지 않을 때 실행 오류가 발생합니다.
- 컴퓨팅 활용률 (`FunctionComputeUtilization`) — 함수가 실행되는 데 걸린 시간의 최대 허용 시간의 백분율입니다. 예를 들어, 값이 35이면 함수가 최대 허용 시간의 35%에서 완료되었음을 의미합니다. 이 측정 단위는 0에서 100 사이의 숫자입니다.

  이 값이 100에 도달하거나 100에 가까워지면 함수가 허용된 실행 시간을 사용했거나 거의 사용한 것이므로 후속 요청의 제한 현상이 발생할 수 있습니다. 함수가 80% 이상의 사용률로 실행되는 경우 함수를 검토하여 실행 시간을 줄이고 사용률을 높이는 것이 좋습니다. 예를 들어 오류만 기록하고, 복잡한 정규식 식을 단순화하거나, 복잡한 JSON 객체의 불필요한 구문 분석을 제거하고 싶을 수 있습니다.
- 제한(`FunctionThrottles`) - 지정된 기간 동안 함수가 제한된 횟수입니다. 함수는 다음과 같은 이유로 제한될 수 있습니다.
  - 함수가 실행에 허용된 최대 시간을 지속적으로 초과함
  - 함수에서 컴파일 오류가 발생함
  - 초당 요청 수가 비정상적으로 많음

또한 CloudFront KeyValueStore는 다음과 같은 운영 지표를 Amazon CloudWatch에 전송합니다.

- 읽기 요청(`KvsReadRequests`) - 지정된 기간 동안 함수가 키 값 저장소에서 성공적으로 읽은 횟수입니다.
- 읽기 오류(`KvsReadErrors`) - 지정된 기간 동안 함수가 키 값 저장소에서 읽지 못한 횟수입니다.

이러한 모든 지표는 미국 동부(버지니아 북부) 리전(us-east-1), CloudFront 네임스페이스의 CloudWatch에 게시됩니다. CloudWatch 콘솔에서 이러한 지표를 볼 수도 있습니다. CloudWatch 콘솔에서 함수별 또는 배포당 함수별 지표를 볼 수 있습니다.

CloudWatch를 사용하여 이러한 지표를 기반으로 경보를 설정할 수도 있습니다. 예를 들어 함수를 실행하는 데 걸린 사용 가능한 시간의 백분율을 나타내는 실행 시간(FunctionComputeUtilization) 지표를 기반으로 경보를 설정할 수 있습니다. 실행 시간이 특정 시간 동안 특정 값에 도달할 때. 예를 들어, 15분 연속 사용 가능한 시간의 70% 이상을 선택하면 경보가 트리거됩니다. 경보를 만들 때 경보의 값과 시간 단위를 지정합니다.

> **Note**
>
> CloudFront Functions는 프로덕션 요청 및 응답에 대한 응답으로 실행되는 LIVE 스테이지의 함수에 대해서만 지표를 CloudWatch로 전송합니다. 함수를 테스트할 때 CloudFront는 지표를 CloudWatch로 보내지 않습니다. 테스트 출력에는 오류, 컴퓨팅 사용률 및 함수 로그(console.log() 명령문)에 대한 정보가 포함되지만 이 정보는 CloudWatch로 전송되지 않습니다.

CloudWatch API를 사용하여 이러한 지표를 얻는 방법에 대한 자세한 내용은 the section called "CloudFront 지표" 섹션을 참조하세요.

## 지표에 대한 경보 생성

CloudFront 콘솔에서 특정 CloudFront 지표에 따라 Amazon Simple Notification Service(SNS)에서 사용자에게 알리도록 경보를 설정할 수 있습니다.

지표에 대한 경보를 생성하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 경보(Alarms)를 선택하세요.
3. 경보 생성을 선택하세요.
4. 세부 정보에서 다음을 지정합니다.
   a. 경보 이름 - 경보의 이름입니다.
   b. 배포 - 경보를 생성하는 CloudFront 배포입니다.
5. 조건에서 다음을 지정합니다.

a. 지표 - 경보를 생성하는 데 사용할 지표입니다.

b. "IF" <condition> – CloudWatch에서 경보를 트리거하고 Amazon SNS 주제로 알림을 전송하는 시점의 임계값입니다. 예를 들어, 5xx 오류 발생률이 1%를 넘는 경우 알림을 받으려면 다음과 같이 지정할 수 있습니다.

   5xx 오류 발생률 > 1

c. "FOR" 연속 기간 – 경보가 트리거되기 전에 조건이 충족되어야 하는 기간입니다. 값을 선택할 때 일시적 문제에 대해서는 경보가 발생하지 않지만 지속되거나 실제 문제에 대해서는 경보가 발생하도록 값을 적절하게 조정해야 합니다.

d. (선택 사항) 알림 - 이 지표가 경보를 트리거할 경우 알림을 보낼 Amazon SNS 주제입니다.

6. 경보 생성(Create alarm)을 선택하세요.

> **Notes**
>
> - 조건 값을 입력할 때는 문장 부호 없이 정수를 사용합니다. 예를 들어, 천을 지정하려면 **1000**을 입력합니다.
> - 4xx, 5xx 및 총 오류 발생률은 지정하는 값이 백분율입니다.
> - 요청, 다운로드한 바이트 및 업로드한 바이트는 지정하는 값이 단위입니다. 예를 들면 1073742000 바이트입니다.

Amazon SNS 주제 생성 방법에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 안내서의 Amazon SNS 주제 생성을 참조하세요.

## 지표 데이터를 CSV 형식으로 다운로드

CloudFront 배포에 대한 CloudWatch 지표 데이터를 CSV 형식으로 다운로드할 수 있습니다.

지표 데이터를 CSV 형식으로 다운로드하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 탐색 창에서 모니터링을 선택합니다.
3. 배포를 선택한 다음 배포 지표 보기를 선택합니다.
4. CSV 다운로드를 선택한 다음 기간(예: 지난 1일(1시간))을 선택합니.

5. 파일을 다운로드한 후 파일을 열면 다음 정보를 볼 수 있습니다.

주제


- 보고서에 대한 정보
- 지표 보고서의 데이터


## 보고서에 대한 정보

보고서의 처음 몇 줄에는 다음 정보가 포함됩니다.

버전

CloudFront 보고 버전입니다.

보고서

보고서의 이름입니다.

DistributionID

보고서를 실행한 배포의 ID입니다.

StartDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 시작 일시를 협정 세계시(UTC)로 표시합니다.

EndDateUTC

보고서 실행 기간에 해당하는 날짜 범위의 종료 일시를 협정 세계시(UTC)로 표시합니다.

GeneratedTimeUTC

보고서를 실행 날짜 및 시간을 협정 세계시(UTC)로 표시합니다.

Granularity

보고서 각 행의 기간입니다(예: `ONE_MINUTE`).

## 지표 보고서의 데이터

이 보고서에는 다음 값이 포함됩니다.

DistributionID

보고서를 실행한 배포의 ID입니다.

FriendlyName

배포의 대체 도메인 이름(CNAME)입니다(있는 경우). 배포에 대체 도메인 이름이 없으면 목록에 배포에 대한 원래 도메인 이름이 포함됩니다.

TimeBucket

데이터가 적용되는 시간 또는 날짜를 협정 세계시(UTC)로 표시합니다.

요청

해당 기간 동안의 전체 HTTP 상태 코드(예: `200`, `404` 등)와 전체 메소드(예: `GET`, `HEAD`, `POST` 등)의 총 요청 수입니다.

BytesDownloaded

해당 기간 동안 지정한 배포에 대해 최종 사용자가 다운로드한 바이트 수입니다.

BytesUploaded

해당 기간 동안 지정한 배포에 대해 최종 사용자가 업로드한 바이트 수입니다.

TotalErrorRatePct

해당 기간 동안 지정한 배포에 대해 HTTP 상태 코드가 `4xx` 또는 `5xx` 오류였던 요청의 백분율입니다.

4xxErrorRatePct

해당 기간 동안 지정한 배포에 대해 HTTP 상태 코드가 `4xx` 오류였던 요청의 백분율입니다.

5xxErrorRatePct

해당 기간 동안 지정한 배포에 대해 HTTP 상태 코드가 `5xx` 오류였던 요청의 백분율입니다.

배포에 대해 추가 지표를 활성화한 경우에는 보고서에 다음과 같은 추가 값도 포함됩니다.

401ErrorRatePct

해당 기간 동안 지정한 배포에 대해 HTTP 상태 코드가 `401` 오류였던 요청의 백분율입니다.

403ErrorRatePct

해당 기간 동안 지정한 배포에 대해 HTTP 상태 코드가 `403` 오류였던 요청의 백분율입니다.

404ErrorRatePct

해당 기간 동안 지정한 배포에 대해 HTTP 상태 코드가 `404` 오류였던 요청의 백분율입니다.

502ErrorRatePct

해당 기간 동안 지정한 배포에 대해 HTTP 상태 코드가 502 오류였던 요청의 백분율입니다.

503ErrorRatePct

해당 기간 동안 지정한 배포에 대해 HTTP 상태 코드가 503 오류였던 요청의 백분율입니다.

504ErrorRatePct

해당 기간 동안 지정한 배포에 대해 HTTP 상태 코드가 504 오류였던 요청의 백분율입니다.

OriginLatency

CloudFront에서 요청을 수신한 시점부터 CloudFront 캐시가 아닌 오리진에서 제공되는 요청에 대해 네트워크(최종 사용자 아님)에 대한 응답을 제공하기 시작한 시점까지 소요되는 총 시간(밀리초)입니다. 이것을 첫 번째 바이트 대기 시간 또는 time-to-first-byte라고도 합니다.

CacheHitRate

CloudFront가 캐시에서 콘텐츠를 제공하는 캐시 가능한 모든 요청의 비율입니다. HTTP POST 및 PUT 요청, 오류는 캐시 가능한 요청으로 간주되지 않습니다.

## CloudFront의 지표 유형

CloudWatch API 또는 AWS Command Line Interface(AWS CLI)를 사용하여, 빌드한 프로그램이나 애플리케이션의 CloudFront 지표를 가져올 수 있습니다. 원시 데이터를 사용하여 사용자 지정 대시보드, 사용자 고유의 경고 도구 등을 만들 수 있습니다.

자세한 내용은 AWS CLI 명령 참조의 get-metric-data 또는 Amazon CloudWatch API 참조의 GetMetricData API 작업을참조하세요.

주제

- 모든 CloudFront 지표에 대한 값
- CloudFront 배포 지표에 대한 값
- CloudFront 함수 지표에 대한 값

> **Note**
>
> CloudWatch API에서 CloudFront 지표를 가져오려면 미국 동부(버지니아 북부) 리전(us-east-1)을 사용해야 합니다. 또한 각 지표의 특정 값과 유형을 알아야 합니다.

## 모든 CloudFront 지표에 대한 값

다음 값은 모든 CloudFront 지표에 적용됩니다.

**Namespace**

`Namespace`의 값은 항상 `AWS/CloudFront`입니다.

Dimensions

각 CloudFront 지표에는 다음의 차원이 있습니다.

**DistributionId**

지표를 가져오려는 CloudFront 배포의 ID입니다.

**FunctionName**

(CloudFront 함수에서) 지표를 가져오려는 함수 이름입니다.

이 차원은 함수에만 적용됩니다.

**Region**

`Region`의 값은 항상 `Global`입니다. CloudFront는 글로벌 서비스이기 때문입니다.

## CloudFront 배포 지표에 대한 값

다음 목록의 정보를 사용하여 CloudWatch API에서 특정 CloudFront 배포 지표에 대한 세부 정보를 얻을 수 있습니다. 이러한 지표 중 일부는 배포에 대한 추가 지표를 활성화한 경우에만 사용할 수 있습니다.

> **Note**
>
> 각 지표에 대해 하나의 통계, `Average` 또는 `Sum`만 적용할 수 있습니다. 다음 목록은 해당 지표에 적용할 수 있는 통계를 지정합니다.

4xx 오류 발생률

응답의 HTTP 상태 코드가 `4xx`인 모든 최종 사용자 요청의 백분율입니다.

- 지표 이름: `4xxErrorRate`
- 유효한 통계: `Average`

- 단위: Percent

401 오류 발생률

응답의 HTTP 상태 코드가 401인 모든 최종 사용자 요청의 백분율입니다. 이 지표를 가져오려면 먼저 추가 지표를 활성화해야 합니다.

- 지표 이름: 401ErrorRate
- 유효한 통계: Average
- 단위: Percent

403 오류 발생률

응답의 HTTP 상태 코드가 403인 모든 최종 사용자 요청의 백분율입니다. 이 지표를 가져오려면 먼저 추가 지표를 활성화해야 합니다.

- 지표 이름: 403ErrorRate
- 유효한 통계: Average
- 단위: Percent

404 오류 발생률

응답의 HTTP 상태 코드가 404인 모든 최종 사용자 요청의 백분율입니다. 이 지표를 가져오려면 먼저 추가 지표를 활성화해야 합니다.

- 지표 이름: 404ErrorRate
- 유효한 통계: Average
- 단위: Percent

5xx 오류 발생률

응답의 HTTP 상태 코드가 5xx인 모든 최종 사용자 요청의 백분율입니다.

- 지표 이름: 5xxErrorRate
- 유효한 통계: Average
- 단위: Percent

502 오류 발생률

응답의 HTTP 상태 코드가 502인 모든 최종 사용자 요청의 백분율입니다. 이 지표를 가져오려면 먼저 추가 지표를 활성화해야 합니다.

- 지표 이름: 502ErrorRate
- 유효한 통계: Average

- 단위: Percent

503 오류 발생률

응답의 HTTP 상태 코드가 503인 모든 최종 사용자 요청의 백분율입니다. 이 지표를 가져오려면 먼저 추가 지표를 활성화해야 합니다.

- 지표 이름: 503ErrorRate
- 유효한 통계: Average
- 단위: Percent

504 오류 발생률

응답의 HTTP 상태 코드가 504인 모든 최종 사용자 요청의 백분율입니다. 이 지표를 가져오려면 먼저 추가 지표를 활성화해야 합니다.

- 지표 이름: 504ErrorRate
- 유효한 통계: Average
- 단위: Percent

다운로드된 바이트

GET 및 HEAD 요청에 대해 뷰어가 다운로드한 총 바이트 수입니다.

- 지표 이름: BytesDownloaded
- 유효한 통계: Sum
- 단위: None

업로드된 바이트

뷰어가 OPTIONS, POST 및 PUT 요청을 사용하여 CloudFront에 업로드한 총 바이트 수입니다.

- 지표 이름: BytesUploaded
- 유효한 통계: Sum
- 단위: None

캐시 적중률

CloudFront가 캐시에서 콘텐츠를 제공하는 캐시 가능한 모든 요청의 비율입니다. HTTP POST 및 PUT 요청, 오류는 캐시 가능한 요청으로 간주되지 않습니다. 이 지표를 가져오려면 먼저 추가 지표를 활성화해야 합니다.

- 지표 이름: CacheHitRate
- 유효한 통계: Average

- 단위: `Percent`

오리진 지연 시간

CloudFront에서 요청을 수신한 시점부터 CloudFront 캐시가 아닌 오리진에서 제공되는 요청에 대해 네트워크(최종 사용자 아님)에 대한 응답을 제공하기 시작하는 시점까지 소요되는 총 시간(밀리초)입니다. 이것을 첫 번째 바이트 대기 시간 또는 time-to-first-byte라고도 합니다. 이 지표를 가져오려면 먼저 추가 지표를 활성화해야 합니다.

- 지표 이름: `OriginLatency`
- 유효한 통계: `Percentile`
- 단위: `Milliseconds`

> **Note**
>
> CloudWatch API에서 `Percentile` 통계를 확인하려면 `ExtendedStatistics`가 아닌 `Statistics` 파라미터를 사용합니다. 자세한 내용은 Amazon CloudWatch API 참조의 GetMetricStatistics 또는 AWS SDK 관련 참조 설명서를 참조하세요.

요청

모든 HTTP 메서드와 HTTP 및 HTTPS 요청에 대해 CloudFront에서 수신한 최종 사용자 요청의 총 수입니다.

- 지표 이름: `Requests`
- 유효한 통계: `Sum`
- 단위: `None`

총 오류 발생률

응답의 HTTP 상태 코드가 `4xx` 또는 `5xx`인 모든 최종 사용자 요청의 백분율입니다.

- 지표 이름: `TotalErrorRate`
- 유효한 통계: `Average`
- 단위: `Percent`

## CloudFront 함수 지표에 대한 값

다음 목록의 정보를 사용하여 CloudWatch API에서 특정 CloudFront 함수 지표에 대한 세부 정보를 얻을 수 있습니다.

> **Note**
>
> 각 지표에 대해 하나의 통계, Average 또는 Sum만 적용할 수 있습니다. 다음 목록은 해당 지표에 적용할 수 있는 통계를 지정합니다.

호출

지정된 기간 동안 함수가 시작된(호출된) 횟수입니다.

- 지표 이름: FunctionInvocations
- 유효한 통계: Sum
- 단위: None

유효성 검사 오류

지정된 기간 동안 함수에서 생성된 유효성 검사 오류의 수입니다. 유효성 검사 오류는 함수가 성공적으로 실행되지만 잘못된 데이터(잘못된 이벤트 객체)를 반환할 때 발생합니다.

- 지표 이름: FunctionValidationErrors
- 유효한 통계: Sum
- 단위: None

실행 오류

지정된 기간 동안 발생한 실행 오류의 수입니다. 함수가 성공적으로 완료되지 않을 때 실행 오류가 발생합니다.

- 지표 이름: FunctionExecutionErrors
- 유효한 통계: Sum
- 단위: None

컴퓨팅 사용률

함수가 실행되는 데 걸린 시간(0~100)의 최대 허용 시간의 백분율입니다. 예를 들어, 값이 35이면 함수가 최대 허용 시간의 35%에서 완료되었음을 의미합니다.

- 지표 이름: FunctionComputeUtilization
- 유효한 통계: Average
- 단위: Percent

제한

지정된 기간 동안 함수가 제한된 횟수입니다.

- 지표 이름: `FunctionThrottles`
- 유효한 통계: `Sum`
- 단위: `None`

# CloudFront 및 엣지 함수 로깅

Amazon CloudFront는 다양한 종류의 로깅을 제공합니다. CloudFront 배포에 전달되는 최종 사용자 요청을 로깅하거나 AWS 계정에 CloudFront 서비스 활동(API 활동)을 로깅할 수 있습니다. CloudFront Functions 및 Lambda@Edge 함수의 로그를 가져올 수도 있습니다.

## 요청 로깅

CloudFront는 다음과 같이 배포에 전달되는 요청을 로깅하는 방법을 제공합니다.

액세스 로그(표준 로그)

CloudFront 액세스 로그는 배포에 대해 이루어진 모든 요청에 대한 상세 기록을 제공합니다. 보안 및 액세스 감사와 같은 시나리오에 로그를 사용할 수 있습니다.

CloudFront 액세스 로그는 지정한 전송 대상으로 전송됩니다.

필요한 경우 액세스 로그를 사용합니다.

- 기록 분석 및 보고
- 보안 감사 및 규정 준수 요구 사항
- 비용 효율적인 장기 로그 보존

자세한 내용은 액세스 로그(표준 로그) 섹션을 참조하세요.

실시간 액세스 로그

CloudFront 실시간 액세스 로그는 요청 수신 후 몇 초 이내에 제공되며, 배포에 대해 이루어진 요청에 대한 정보를 실시간으로 제공합니다. 실시간 액세스 로그에 대한 샘플링 비율(즉, 실시간 액세스 로그 레코드를 수신하려는 요청의 백분율)을 선택할 수 있습니다. 또한 로그 레코드에서 수신하려는 특정 필드도 선택할 수 있습니다. 실시간 액세스 로그는 콘텐츠 전송 성능을 실시간으로 모니터링하는 데 적합합니다.

CloudFront 실시간 액세스 로그는 Amazon Kinesis Data Streams에서 선택한 데이터 스트림으로 전송됩니다. CloudFront는 Kinesis Data Streams 사용 요금 외에도, 실시간 액세스 로그에 대한 요금을 청구합니다.

필요한 경우 실시간 액세스 로그를 사용합니다.

- 실시간 모니터링 및 알림
- 라이브 대시보드 및 운영 인사이트

자세한 내용은 실시간 액세스 로그 사용 섹션을 참조하세요.

연결 로그

연결 로그는 mTLS 지원 배포를 위한 서버와 클라이언트 간의 연결에 대한 자세한 정보를 제공합니다. 연결 로그는 클라이언트 인증서 정보, mTLS 인증 실패 이유, 연결 허용 또는 거부 여부에 대한 가시성을 제공합니다.

액세스 로그(표준 로그)와 마찬가지로 연결 로그는 지정한 전송 대상으로 전송됩니다.

> **Note**
>
> 연결 로그를 활성화하려면 먼저 배포에 대해 mTLS를 활성화해야 합니다.

필요한 경우 연결 로그를 사용합니다.

- TLS 핸드셰이크 중 연결 성공 또는 실패 이유
- 클라이언트 인증서 정보에 대한 가시성

자세한 내용은 연결 로그를 사용한 관찰성 섹션을 참조하세요.

## 엣지 함수 로깅

Amazon CloudWatch Logs를 사용하여 엣지 함수(Lambda@Edge 및 CloudFront Functions 모두)에 대한 로그를 가져올 수 있습니다. CloudWatch 콘솔 또는 CloudWatch Logs API를 사용하여 로그에 액세스할 수 있습니다. 자세한 내용은 the section called “엣지 함수 로그” 섹션을 참조하세요.

## 서비스 활동 로깅

AWS CloudTrail을 사용하여 AWS 계정에 CloudFront 서비스 활동(API 활동)을 로깅할 수 있습니다. CloudTrail은 CloudFront에서 사용자, 역할 또는 AWS 서비스가 수행한 API 작업의 기록을 제공합니다. CloudTrail에서 수집한 정보를 사용하여 CloudFront에 수행된 API 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

자세한 내용은 AWS CloudTrail을 사용하여 Amazon CloudFront API 직접 호출 로깅 섹션을 참조하세요.

로깅에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

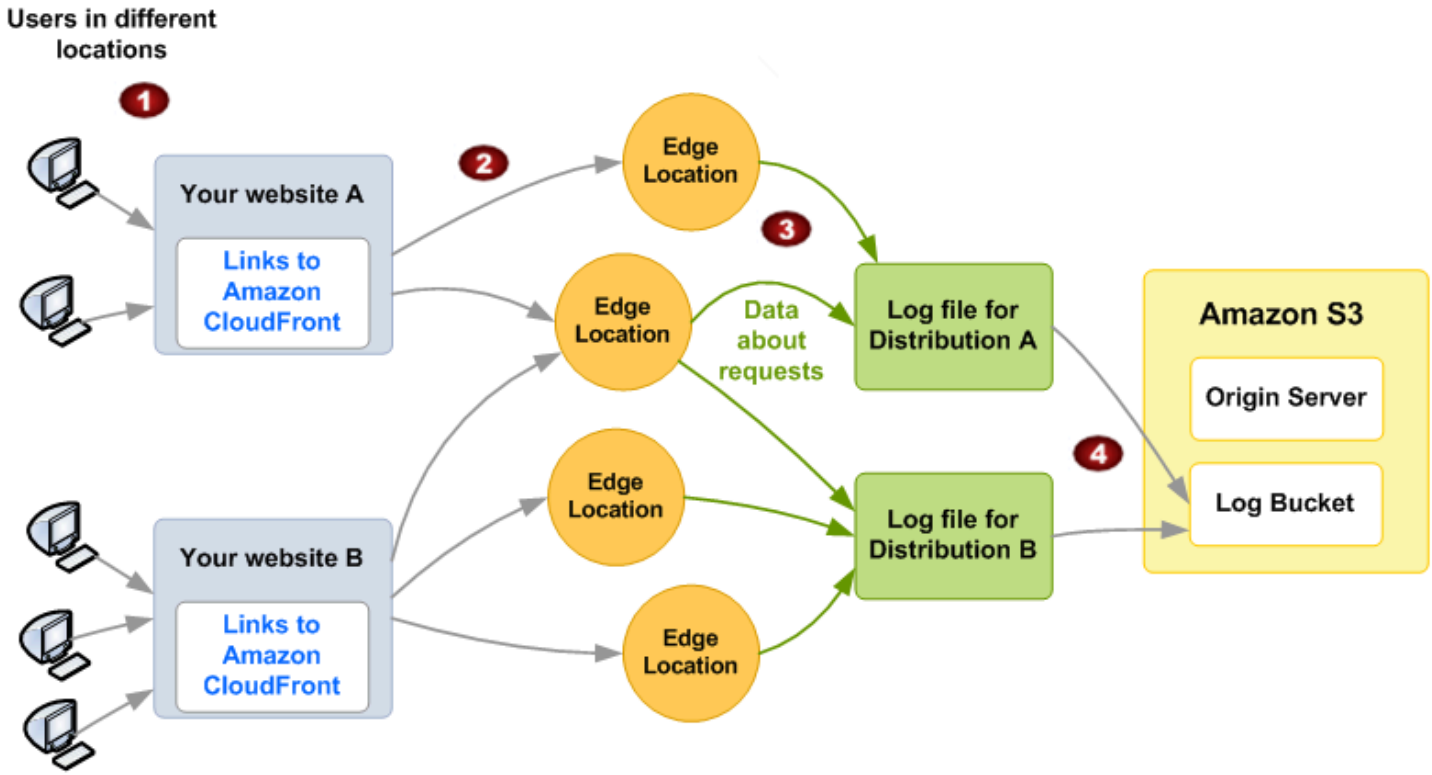
- 액세스 로그(표준 로그)
- 실시간 액세스 로그 사용
- 엣지 함수 로그
- AWS CloudTrail을 사용하여 Amazon CloudFront API 직접 호출 로깅

## 액세스 로그(표준 로그)

로그 파일을 만들도록 CloudFront를 구성할 수 있습니다. 이러한 로그 파일에는 CloudFront에서 수신하는 모든 사용자(뷰어) 요청에 대한 세부 정보가 포함됩니다. 이러한 로그를 액세스 로그라고 부르며 표준 로그라고도 합니다.

각 로그에는 요청을 받은 시간, 처리 시간, 요청 경로, 서버 응답과 같은 정보가 포함되어 있습니다. 이러한 액세스 로그를 사용하여 응답 시간을 분석하고 문제를 해결할 수 있습니다.

다음 다이어그램은 CloudFront에서 객체 요청에 대한 정보를 어떻게 기록하는지 보여 줍니다. 이 예제에서는 배포가 Amazon S3 버킷으로 액세스 로그를 전송하도록 구성됩니다.



1. 이 예제에서는 A와 B라는 두 웹 사이트가 있고 그에 해당하는 CloudFront 배포 2개가 있습니다. 사용자가 배포와 연결된 URL을 사용하여 객체를 요청합니다.
2. CloudFront에서 각 요청을 해당 엣지 로케이션으로 라우팅합니다.
3. CloudFront에서 각 요청에 대한 데이터를 해당 배포 관련 로그 파일에 기록합니다. 이 예제에서는 배포 A와 관련된 요청에 대한 정보가 배포 A 로그 파일에 기록되고, 배포 B와 관련된 요청에 대한 정보가 배포 B 로그 파일에 기록됩니다.
4. CloudFront는 로깅을 활성화할 때 지정한 Amazon S3 버킷에 배포 관련 로그 파일을 정기적으로 저장합니다. 그런 다음, CloudFront는 새 로그 파일에 배포에 대한 후속 요청에 대한 정보를 저장하기 시작합니다.

   지정된 시간 동안 콘텐츠에 액세스하는 뷰어가 없는 경우 해당 시간 동안 로그 파일이 수신되지 않습니다.

> **Note**
>
> 모든 요청을 완전히 살펴보기 보다는 콘텐츠에 대한 요청 특성을 이해하는 데 로그를 사용하는 것이 좋습니다. CloudFront는 최대 효과에 기초하여 액세스 로그를 전송합니다. 요청에 따라서는 실제로 요청이 처리된 지 한참 후에 로그 레코드가 전송되거나 아예 전송되지 않을 수도 있습니다. 로그 항목이 액세스 로그에서 생략되는 경우, 액세스 로그의 항목 수가 AWS 결제 및 사용 보고서에 표시되는 사용량과 일치하지 않습니다.

CloudFront는 2가지 버전의 표준 로깅을 지원합니다. 표준 로깅(레거시)은 Amazon S3로만 액세스 로그 전송을 지원합니다. 표준 로깅(v2)은 추가 전송 대상을 지원합니다. 배포에 대해 하나 또는 두 로깅 옵션을 모두 구성할 수 있습니다. 자세한 내용은 다음 항목을 참조하세요.

주제

- 표준 로깅(v2) 구성
- 표준 로깅(레거시) 구성
- 표준 로깅 참조

> **Tip**
>
> 또한 CloudFront의 실시간 액세스 로그는 배포에 대해 이루어진 요청에 대한 정보를 실시간으로 제공합니다(로그는 요청을 받은 후 몇 초 내에 전달됩니다). 실시간 액세스 로그를 사용하여

콘텐츠 전송 성능을 모니터링 및 분석하고 이에 기초해 조치를 취할 수 있습니다. 자세한 내용은 실시간 액세스 로그 사용 섹션을 참조하세요.

## 표준 로깅(v2) 구성

배포를 만들거나 업데이트할 때 액세스 로그(표준 로그)를 활성화할 수 있습니다. 표준 로깅(v2)에는 다음 기능이 포함됩니다.

- 액세스 로그를 Amazon CloudWatch Logs, Amazon Data Firehose, Amazon Simple Storage Service(Amazon S3)로 전달합니다.
- 원하는 로그 필드를 선택합니다. 실시간 액세스 로그 필드의 하위 집합을 선택할 수도 있습니다.
- 추가 출력 로그 파일 형식을 선택합니다.

Amazon S3를 사용하는 경우 다음의 선택 사항 기능이 제공됩니다.

- 옵트인 AWS 리전에 로그를 전송합니다.
- 파티셔닝을 사용하여 로그를 구성합니다.
- Hive 호환 파일 이름을 사용합니다.

자세한 내용은 Amazon S3로 로그 전송 섹션을 참조하세요.

표준 로깅을 시작하려면 다음 작업을 완료합니다.

1. 로그를 수신할 지정된 AWS 서비스에 필수 권한을 설정합니다.
2. CloudFront 콘솔 또는 CloudWatch API에서 표준 로깅을 구성합니다.
3. 액세스 로그를 확인합니다.

> **Note**
>
> - 표준 로깅(v2)을 사용 설정해도 표준 로깅(레거시)에 영향을 미치거나 해당 로그가 변경되지 않습니다. 표준 로깅(v2)을 사용하는 것 외에도 배포 시 표준 로깅(레거시)을 계속 사용할 수 있습니다. 자세한 내용은 표준 로깅(레거시) 구성 섹션을 참조하세요.
> - 표준 로깅(레거시)을 이미 사용 설정했고 Amazon S3에 대한 표준 로깅(v2)을 사용하려는 경우 다른 Amazon S3 버킷을 지정하거나 동일한 버킷에서 별도의 경로를 사용하는 것이 좋습

니다(예: 로그 접두사 또는 파티셔닝 사용). 이렇게 하면 어떤 로그 파일이 어떤 배포와 연결되어 있는지 추적하고 로그 파일이 서로 덮어쓰는 것을 방지할 수 있습니다.

### 권한

CloudFront는 CloudWatch 판매된 로그를 사용하여 액세스 로그를 전달합니다. 이렇게 하려면 로그 전송을 사용 설정할 수 있도록 지정된 AWS 서비스에 대한 권한이 필요합니다.

각 로그 대상에 필요한 권한을 보려면 Amazon CloudWatch Logs 사용 설명서의 다음 주제 중에서 선택합니다.

- CloudWatch 로그
- Firehose –
- Amazon S3:

로그 대상에 대한 권한을 설정한 후 배포에 대한 표준 로깅을 사용 설정할 수 있습니다.

> **Note**
>
> CloudFront는 액세스 로그를 다른 AWS 계정(크로스 계정)로 전송하는 것을 지원합니다. 크로스 계정 전송을 사용하려면 두 계정(계정 및 수신 계정)에 필수 권한이 부여되어 있어야 합니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서에 나와 있는 크로스 계정 전송을 위한 표준 로깅 사용 섹션 또는 Cross-account delivery example을 참조하시기 바랍니다.

### 표준 로깅 사용

표준 로깅을 사용하기 위해 CloudFront 콘솔 또는 CloudWatch API를 사용할 수 있습니다.

목차

- 표준 로깅 사용(CloudFront 콘솔)
- 표준 로깅 사용(CloudWatch API)

### 표준 로깅 사용(CloudFront 콘솔)

CloudFront 배포 시 표준 로깅을 사용하려면 다음과 같이 하세요(콘솔의 경우).

1. CloudFront 콘솔을 사용하여 기존 배포를 업데이트합니다.
2. Logging(로깅) 탭을 선택합니다.
3. 추가를 선택한 다음 로그를 수신할 서비스를 선택합니다.
   - CloudWatch Logs
   - Firehose
   - Amazon S3
4. 대상에서 서비스의 리소스를 선택합니다. 리소스를 아직 만들지 않은 경우 생성을 선택하거나 다음 설명서를 통해 확인할 수 있습니다.
   - CloudWatch Logs에서 로그 그룹 이름을 입력합니다.
   - Firehose에 Firehose 전송 스트림을 입력합니다.
   - Amazon S3에서 버킷 이름을 입력합니다.

   > **Tip**
   > 접두사를 지정하려면 버킷 이름 뒤에 접두사를 입력합니다(예: `amzn-s3-demo-bucket.s3.amazonaws.com/MyLogPrefix`). 접두사를 지정하지 않으면 CloudFront에서 자동으로 접두사를 추가합니다. 자세한 내용은 Amazon S3로 로그 전송 섹션을 참조하세요.

5. 추가 설정 - 선택 사항에서 다음 옵션을 지정할 수 있습니다.
   a. 필드 선택에서 대상으로 전송할 로그 필드 이름을 선택합니다. 액세스 로그 필드 및 실시간 액세스 로그 필드의 하위 집합을 선택할 수 있습니다.
   b. (Amazon S3만 해당) 파티셔닝에서 로그 파일 데이터를 파티셔닝할 경로를 지정합니다.
   c. (Amazon S3만 해당) Hive 호환 파일 형식에서 확인란을 선택하여 Hive 호환 S3 경로를 사용할 수 있습니다. 이렇게 하면 Hive 호환 도구에 새 데이터를 쉽게 로드할 수 있습니다.
   d. 출력 형식에서 원하는 형식을 지정합니다.

> **Note**
>
> Parquet를 선택하면 액세스 로그를 Apache Parquet로 변환하는 데 CloudWatch 요금이 발생합니다. 자세한 내용은 Vended Logs section for CloudWatch pricing을 참조하시기 바랍니다.

e. 필드 구분 기호에서 로그 필드를 구분하는 방법을 지정합니다.

6. 배포를 업데이트하거나 만드는 단계를 완료합니다.
7. 다른 대상을 추가하려면 3~6단계를 반복합니다.
8. 로그 페이지에서 배포 옆의 표준 로그 상태가 활성화됨인지 확인합니다.
9. (선택 사항) 쿠키 로깅을 사용 설정하려면 관리, 설정을 선택하고 쿠키 로깅을 켠 다음 변경 사항 저장을 선택합니다.

> **Tip**
>
> 쿠키 로깅은 배포의 모든 표준 로깅에 적용되는 전역 설정입니다. 별도의 전송 대상에서 이 설정을 재정의할 수 없습니다.

표준 로깅 전송 및 로그 필드에 대한 자세한 내용은 표준 로깅 참조 섹션을 참조하시기 바랍니다.

표준 로깅 사용(CloudWatch API)

CloudWatch API를 사용하여 배포에 대한 표준 로깅을 사용 설정할 수도 있습니다.

> **참고**
>
> - 표준 로깅을 사용하기 위해 CloudWatch API를 호출할 때 다른 대상으로 리전 간 전송을 사용하려는 경우에도 미국 동부(버지니아 북부) 리전(`us-east-1`)을 지정해야 합니다. 예를 들어 유럽(아일랜드) 리전(`eu-west-1`)의 S3 버킷에 액세스 로그를 보내려면 `us-east-1` 리전에서 CloudWatch API를 사용합니다.
> - 표준 로깅에 쿠키를 포함하는 추가 옵션이 있습니다. CloudFront API에서 이는 `IncludeCookies` 파라미터입니다. CloudWatch API를 사용하여 액세스 로깅을 구성하고 쿠키를 포함하도록 지정하는 경우 CloudFront 콘솔 또는 CloudFront API를 사용하여 쿠키를 포함하도록 배포를 업데이트해야 합니다. 그렇지 않으면 CloudFront는 로그 대상으로 쿠키를 전송할 수 없습니다. 자세한 내용은 쿠키 로깅 섹션을 참조하세요.

배포에 대한 표준 로깅을 사용하려면 다음과 같이 합니다(CloudWatch API의 경우).

1. 배포를 만든 후 Amazon 리소스 이름(ARN)을 가져옵니다.

   CloudFront 콘솔의 배포 페이지에서 ARN을 찾거나 GetDistribution API 작업을 사용할 수 있습니다. 배포 ARN은 `arn:aws:cloudfront::123456789012:distribution/d111111abcdef8` 형식을 따릅니다.

2. 다음으로 CloudWatch PutDeliverySource API 작업을 사용하여 배포에 대한 전송 소스를 만듭니다.

   a. 전송 소스 이름을 입력합니다.

   b. 배포의 `resourceArn`을 전달합니다.

   c. `logType`에서 수집되는 로그 유형으로 `ACCESS_LOGS`를 지정합니다.

   d. Example예: AWS CLI put-delivery-source 명령

      다음은 배포에 대한 전송 소스를 구성하는 예제입니다.

```
aws logs put-delivery-source --name S3-delivery --resource-arn
 arn:aws:cloudfront::123456789012:distribution/d111111abcdef8 --log-type
 ACCESS_LOGS
```

      출력

```
{
 "deliverySource": {
 "name": "S3-delivery",
 "arn": "arn:aws:logs:us-east-1:123456789012:delivery-source:S3-delivery",
 "resourceArns": [
 "arn:aws:cloudfront::123456789012:distribution/d111111abcdef8"
 ],
 "service": "cloudfront",
 "logType": "ACCESS_LOGS"
 }
}
```

3. PutDeliveryDestination API 작업을 사용하여 로그를 저장할 위치를 구성합니다.

   a. `destinationResourceArn`에서 대상의 ARN을 지정합니다. CloudWatch Logs 로그 그룹, Firehose 전송 스트림, Amazon S3 버킷이 될 수 있습니다.

   b. `outputFormat`에서 로그의 출력 형식을 지정합니다.

c. Example예: AWS CLI put-delivery-destination 명령

다음은 Amazon S3 버킷에 대한 전송 대상을 구성하는 예제입니다.

```
aws logs put-delivery-destination --name S3-destination --delivery-destination-
configuration destinationResourceArn=arn:aws:s3:::amzn-s3-demo-bucket
```

출력

```
{
    "name": "S3-destination",
    "arn": "arn:aws:logs:us-east-1:123456789012:delivery-destination:S3-
destination",
    "deliveryDestinationType": "S3",
    "deliveryDestinationConfiguration": {
        "destinationResourceArn": "arn:aws:s3:::amzn-s3-demo-bucket"
    }
}
```

> **Note**
>
> 크로스 계정에 로그를 전달하는 경우 PutDeliveryDestinationPolicy API 작업을 사용하여 대상 계정 AWS Identity and Access Management(IAM) 정책을 할당해야 합니다. IAM 정책은 한 계정에서 다른 계정으로의 전송을 허용합니다.

4. CreateDelivery API 작업을 사용하여 이전 단계에서 만든 대상에 전송 소스를 연결합니다. 이 API 작업은 전송 소스를 최종 대상과 연결합니다.

   a. `deliverySourceName`에서 소스의 이름을 지정합니다.

   b. `deliveryDestinationArn`에서 전송 대상의 ARN을 지정합니다.

   c. `fieldDelimiter`에서 각 로그 필드를 구분할 문자열을 지정합니다.

   d. `recordFields`에서 원하는 로그 필드를 지정합니다.

   e. S3를 사용하는 경우 `enableHiveCompatiblePath` 및 `suffixPath`를 사용할지 여부를 지정합니다.

Example예: AWS CLI create-delivery 명령

다음은 전송을 만드는 예제입니다.

```
aws logs create-delivery --delivery-source-name cf-delivery --delivery-destination-
arn arn:aws:logs:us-east-1:123456789012:delivery-destination:S3-destination
```

출력

```
{
    "id": "abcNegnBoTR123",
    "arn": "arn:aws:logs:us-east-1:123456789012:delivery:abcNegnBoTR123",
    "deliverySourceName": "cf-delivery",
    "deliveryDestinationArn": "arn:aws:logs:us-east-1:123456789012:delivery-
destination:S3-destination",
    "deliveryDestinationType": "S3",
    "recordFields": [
        "date",
        "time",
        "x-edge-location",
        "sc-bytes",
        "c-ip",
        "cs-method",
        "cs(Host)",
        "cs-uri-stem",
        "sc-status",
        "cs(Referer)",
        "cs(User-Agent)",
        "cs-uri-query",
        "cs(Cookie)",
        "x-edge-result-type",
        "x-edge-request-id",
        "x-host-header",
        "cs-protocol",
        "cs-bytes",
        "time-taken",
        "x-forwarded-for",
        "ssl-protocol",
        "ssl-cipher",
        "x-edge-response-result-type",
        "cs-protocol-version",
```

```
        "fle-status",
        "fle-encrypted-fields",
        "c-port",
        "time-to-first-byte",
        "x-edge-detailed-result-type",
        "sc-content-type",
        "sc-content-len",
        "sc-range-start",
        "sc-range-end",
        "c-country",
        "cache-behavior-path-pattern"
    ],
     "fieldDelimiter": ""
}
```

5. CloudFront 콘솔의 로그 페이지에서 배포 옆의 표준 로그 상태가 활성화됨인지 확인합니다.

   표준 로깅 전송 및 로그 필드에 대한 자세한 내용은 표준 로깅 참조 섹션을 참조하시기 바랍니다.

> **Note**
>
> AWS CloudFormation을 사용하여 CloudFront에 대한 표준 로깅(v2)을 사용하려면 다음 CloudWatch Logs 속성을 사용할 수 있습니다.
>
> - Delivery
> - DeliveryDestination
> - DeliverySource
>
> `ResourceArn`은 CloudFront 배포이며, `LogType`은 지원되는 로그 유형의 `ACCESS_LOGS`여야 합니다.

## 크로스 계정 전송을 위한 표준 로깅 사용

AWS 계정에 대해 표준 로깅을 활성화하고 액세스 로그를 다른 계정으로 전송하려면 소스 계정과 대상 계정을 올바르게 구성해야 합니다. CloudFront 배포가 있는 소스 계정은 액세스 로그를 대상 계정으로 전송합니다.

이 예제 절차에서 소스 계정(*111111111111*)은 대상 계정(*222222222222*)의 Amazon S3 버킷으로 액세스 로그를 전송합니다. 대상 계정의 Amazon S3 버킷으로 액세스 로그를 보내려면 AWS CLI를 사용합니다.

대상 계정 구성

대상 계정의 경우 다음 절차를 완료합니다.

대상 계정을 구성하려면 다음과 같이 합니다.

1. 로그 전송 대상을 만들려면 다음 AWS CLI 명령을 입력합니다. 이 예제에서는 *MyLogPrefix* 문자열을 사용하여 액세스 로그의 접두사를 만듭니다.

```
aws logs put-delivery-destination --name cloudfront-delivery-destination --
delivery-destination-configuration "destinationResourceArn=arn:aws:s3:::amzn-s3-
demo-bucket-cloudfront-logs/MyLogPrefix"
```

출력

```
{
    "deliveryDestination": {
        "name": "cloudfront-delivery-destination",
        "arn": "arn:aws:logs:us-east-1:222222222222:delivery-
destination:cloudfront-delivery-destination",
        "deliveryDestinationType": "S3",
        "deliveryDestinationConfiguration": {"destinationResourceArn":
 "arn:aws:s3:::amzn-s3-demo-bucket-cloudfront-logs/MyLogPrefix"}
    }
}
```

> **Note**
>
> 접두사 없이 S3 버킷을 지정하면 CloudFront는 `AWSLogs/<account-ID>/CloudFront`를 S3 전송 대상의 `suffixPath`에 표시되는 접두사로 자동으로 추가합니다. 자세한 내용은 S3DeliveryConfiguration을 참조하시기 바랍니다.

2. 소스 계정이 로그 전송을 만들 수 있도록 로그 전송 대상에 대한 리소스 정책을 추가합니다.

   다음 정책에서 *111111111111*을 소스 계정 ID로 교체하고 1단계의 출력에서 전송 대상 ARN을 지정합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCreateDelivery",
            "Effect": "Allow",
            "Principal": {"AWS": "111111111111"},
            "Action": ["logs:CreateDelivery"],
            "Resource": "arn:aws:logs:us-east-1:222222222222:delivery-
destination:cloudfront-delivery-destination"
        }
    ]
}
```

3. `deliverypolicy.json`과 같은 파일을 저장합니다.
4. 이전 정책을 전송 대상에 연결하려면 다음 AWS CLI 명령을 입력합니다.

```
aws logs put-delivery-destination-policy --delivery-destination-name cloudfront-
delivery-destination --delivery-destination-policy file://deliverypolicy.json
```

5. 아래 스테이트먼트를 대상 Amazon S3 버킷 정책에 추가하여 리소스 ARN과 소스 계정 ID를 교체합니다. 이 정책을 사용하면 `delivery.logs.amazonaws.com` 서비스 위탁자가 `s3:PutObject` 작업을 수행할 수 있습니다.

```
{
    "Sid": "AWSLogsDeliveryWrite",
    "Effect": "Allow",
    "Principal": {"Service": "delivery.logs.amazonaws.com"},
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-cloudfront-logs/*",
    "Condition": {
        "StringEquals": {
            "s3:x-amz-acl": "bucket-owner-full-control",
            "aws:SourceAccount": "111111111111"
        },
        "ArnLike": {"aws:SourceArn": "arn:aws:logs:us-east-1:111111111111:delivery-
source:*"}
    }
```

```
}
```

6. 버킷에 AWS KMS를 사용하는 경우 KMS 키 정책에 다음 스테이트먼트를 추가하여 `delivery.logs.amazonaws.com` 서비스 위탁자에 대한 권한을 부여합니다.

```
{
    "Sid": "Allow Logs Delivery to use the key",
    "Effect": "Allow",
    "Principal": {"Service": "delivery.logs.amazonaws.com"},
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {"aws:SourceAccount": "111111111111"},
        "ArnLike": {"aws:SourceArn": "arn:aws:logs:us-east-1:111111111111:delivery-
source:*"}
    }
}
```

소스 계정 구성

대상 계정을 구성한 후 다음 절차에 따라 전송 소스를 만들고 소스 계정의 배포에 대한 로깅을 사용 설정합니다.

소스 계정을 구성하려면 다음과 같이 합니다.

1. CloudFront 표준 로깅을 위한 전송 소스를 만들어 CloudWatch Logs로 로그 파일을 보낼 수 있습니다.

   다음 AWS CLI 명령을 입력하여 이름과 배포 ARN을 교체할 수 있습니다.

```
aws logs put-delivery-source --name s3-cf-delivery --resource-arn
 arn:aws:cloudfront::111111111111:distribution/E1TR1RHV123ABC --log-type
 ACCESS_LOGS
```

   출력

```
{
    "deliverySource": {
        "name": "s3-cf-delivery",
        "arn": "arn:aws:logs:us-east-1:111111111111:delivery-source:s3-cf-
delivery",
        "resourceArns":
 ["arn:aws:cloudfront::111111111111:distribution/E1TR1RHV123ABC"],
        "service": "cloudfront",
        "logType": "ACCESS_LOGS"
    }
}
```

2. 소스 계정의 로그 전송 소스와 대상 계정의 로그 전송 대상을 매핑하는 전송을 만듭니다.

   다음 AWS CLI 명령에서 1단계: 대상 계정 구성의 출력에서 전송 대상 ARN을 지정합니다.

```
aws logs create-delivery --delivery-source-name s3-cf-delivery --
delivery-destination-arn arn:aws:logs:us-east-1:222222222222:delivery-
destination:cloudfront-delivery-destination
```

   출력

```
{
    "delivery": {
        "id": "OPmOpLahVzhx1234",
        "arn": "arn:aws:logs:us-east-1:111111111111:delivery:OPmOpLahVzhx1234",
        "deliverySourceName": "s3-cf-delivery",
        "deliveryDestinationArn": "arn:aws:logs:us-east-1:222222222222:delivery-
destination:cloudfront-delivery-destination",
        "deliveryDestinationType": "S3",
        "recordFields": [
            "date",
            "time",
            "x-edge-location",
            "sc-bytes",
            "c-ip",
            "cs-method",
            "cs(Host)",
            "cs-uri-stem",
            "sc-status",
            "cs(Referer)",
            "cs(User-Agent)",
```

```
            "cs-uri-query",
            "cs(Cookie)",
            "x-edge-result-type",
            "x-edge-request-id",
            "x-host-header",
            "cs-protocol",
            "cs-bytes",
            "time-taken",
            "x-forwarded-for",
            "ssl-protocol",
            "ssl-cipher",
            "x-edge-response-result-type",
            "cs-protocol-version",
            "fle-status",
            "fle-encrypted-fields",
            "c-port",
            "time-to-first-byte",
            "x-edge-detailed-result-type",
            "sc-content-type",
            "sc-content-len",
            "sc-range-start",
            "sc-range-end",
            "c-country",
            "cache-behavior-path-pattern"
        ],
        "fieldDelimiter": "\t"
    }
}
```

3. 크로스 계정 전송이 성공했는지 확인합니다.

   a. *##* 계정에서 CloudFront 콘솔에 로그인하고 배포를 선택합니다. 로깅 탭의 유형에서 S3 크로스 계정 로그 전송을 위해 만든 항목을 볼 수 있습니다.

   b. *##* 계정에서 Amazon S3 콘솔에 로그인하고 Amazon S3 버킷을 선택합니다. 버킷 이름에 *MyLogPrefix* 접두사와 해당 폴더에 전달된 액세스 로그가 표시됩니다.

### 출력 파일 형식

선택한 전송 대상에 따라 로그 파일에 대해 다음 형식 중 하나를 지정할 수 있습니다.

- JSON
- Plain

- w3c
- 원시
- Parquet(Amazon S3만 해당)

> **Note**
>
> 전송 대상을 처음 만들 때만 출력 형식을 설정할 수 있습니다. 이는 나중에 업데이트할 수 없습니다. 출력 형식을 변경하려면 전송을 삭제하고 다른 전송을 만듭니다.

자세한 내용은 Amazon CloudWatch Logs API 참조에 나와 있는 PutDeliveryDestination을 참조하시기 바랍니다.

표준 로깅 설정 편집

CloudFront 콘솔 또는 CloudWatch API를 사용하여 로깅을 사용 또는 사용 해제하고 다른 로그 설정을 업데이트할 수 있습니다. 로깅 설정에 대한 변경 내용은 12시간 내에 적용됩니다.

자세한 내용은 다음 항목을 참조하세요.

- CloudFront 콘솔을 사용하여 배포를 업데이트하려면 배포 업데이트 섹션을 참조하시기 바랍니다.
- CloudFront API를 사용하여 배포를 업데이트하려면 Amazon CloudFront API 참조에 나와 있는 UpdateDistribution을 참조하시기 바랍니다.
- CloudWatch Logs API 작업에 대한 자세한 내용은 Amazon CloudWatch Logs API 참조를 참조하시기 바랍니다.

액세스 로그 필드

표준 로깅(레거시)에서 지원하는 로그 필드와 동일한 로그 필드를 선택할 수 있습니다. 자세한 내용은 로그 파일 필드를 참조하시기 바랍니다.

또한 다음의 실시간 액세스 로그 필드를 선택할 수 있습니다.

1. **`timestamp(ms)`** - 밀리초 단위의 타임스탬프
2. **`origin-fbl`** - CloudFront와 오리진 간의 첫 바이트 지연 시간(초)
3. **`origin-lbl`** - CloudFront와 오리진 간의 마지막 바이트 지연 시간(초)
4. **`asn`** - 뷰어의 Autonomous System Number(ASN)

5. **`c-country`** - 뷰어의 지리적 위치를 나타내는 국가 코드이며, 뷰어의 IP 주소로 결정됨 국가 코드 목록은 ISO 3166-1 alpha-2 단원을 참조하세요.
6. **`cache-behavior-path-pattern`** - 뷰어 요청과 일치하는 캐시 동작을 식별하는 경로 패턴

### CloudWatch Logs에 로그 전송

CloudWatch Logs로 로그를 전송하려면 CloudWatch Logs 로그 그룹을 만들거나 기존 버전을 사용합니다. CloudWatch Logs 로그 그룹의 구성에 대한 자세한 내용은 Working with Log Groups and Log Streams을 참조하시기 바랍니다.

로그 그룹을 만든 후에는 표준 로깅을 허용하는 데 필요한 권한이 있어야 합니다. 필수 권한에 대한 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 Logs sent to CloudWatch Logs를 참조하시기 바랍니다.

> **참고**
>
> - CloudWatch Logs 로그 그룹의 이름을 지정할 경우 `[\w-]` 정규식 패턴만 사용합니다. 자세한 내용은 Amazon CloudWatch Logs API 참조에 나와 있는 PutDeliveryDestination API 작업을 참조하시기 바랍니다.
> - 로그 그룹 리소스 정책이 크기 제한을 초과하지 않는지 확인합니다. CloudWatch Logs 주제의 로그 그룹 리소스 정책 크기 제한 고려 사항 섹션을 참조하시기 바랍니다.

### CloudWatch Logs로 전송된 액세스 로그 예제

```
{
"date": "2024-11-14",
"time": "21:34:06",
"x-edge-location": "SOF50-P2",
"asn": "16509",
"timestamp(ms)": "1731620046814",
"origin-fbl": "0.251",
"origin-lbl": "0.251",
"x-host-header": "d111111abcdef8.cloudfront.net",
"cs(Cookie)": "examplecookie=value"
}
```

### Firehose로 로그 전송

Firehose로 로그를 보내려면 기존 Firehose 전송 스트림을 만들거나 사용합니다. 그런 다음 Firehose 전송 스트림을 로그 전송 대상으로 지정합니다. 미국 동부(버지니아 북부) us-east-1 리전에서 Firehose 전송 스트림을 지정해야 합니다.

전송 스트림 생성에 관한 자세한 내용은 Amazon Data Firehose 전송 스트림 생성을 참조하세요.

전송 스트림을 만든 후 표준 로깅을 허용하는 데 필요한 권한이 있어야 합니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 Logs sent to Firehose를 참조하시기 바랍니다.

> **Note**
>
> Firehose 스트림의 이름을 지정할 경우 `[\w-]` 정규식 패턴만 사용합니다. 자세한 내용은 Amazon CloudWatch Logs API 참조에 나와 있는 PutDeliveryDestination API 작업을 참조하시기 바랍니다.

### Firehose로 전송된 액세스 로그 예제

```
{"date":"2024-11-15","time":"19:45:51","x-edge-location":"SOF50-
P2","asn":"16509","timestamp(ms)":"1731699951183","origin-fbl":"0.254","origin-
lbl":"0.254","x-host-
header":"d111111abcdef8.cloudfront.net","cs(Cookie)":"examplecookie=value"}
{"date":"2024-11-15","time":"19:45:52","x-edge-location":"SOF50-
P2","asn":"16509","timestamp(ms)":"1731699952950","origin-fbl":"0.125","origin-
lbl":"0.125","x-host-
header":"d111111abcdef8.cloudfront.net","cs(Cookie)":"examplecookie=value"}
```

### Amazon S3로 로그 전송

Amazon S3로 액세스 로그를 전송하려면 S3 버킷을 만들거나 기존 버킷을 사용합니다. CloudFront에서 로깅을 사용할 때 버킷 이름을 지정합니다. 버킷 생성에 대해 자세히 알아보려면 Amazon Simple Storage Service 사용 설명서의 버킷 생성을 참조하시기 바랍니다.

버킷을 만든 후 표준 로깅을 허용하는 데 필요한 권한이 있어야 합니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 Amazon S3로 전송된 로그를 참조하세요.

- 로깅을 사용 설정하면 AWS가 자동으로 필요한 버킷 정책을 추가합니다.
- 옵트인 AWS 리전에서 S3 버킷을 사용할 수도 있습니다.

> **Note**
>
> 표준 로깅(레거시)을 이미 사용 설정했고 Amazon S3에 대한 표준 로깅(v2)을 사용하려는 경우 다른 Amazon S3 버킷을 지정하거나 동일한 버킷에서 별도의 경로를 사용하는 것이 좋습니다(예: 로그 접두사 또는 파티셔닝 사용). 이렇게 하면 어떤 로그 파일이 어떤 배포와 연결되어 있는지 추적하고 로그 파일이 서로 덮어쓰는 것을 방지할 수 있습니다.

주제


- S3 버킷 지정
- 분할
- Hive 호환 파일 이름 형식
- 로그에 액세스하는 경로 예제
- Amazon S3로 전송된 액세스 로그의 예제


## S3 버킷 지정

S3 버킷을 전송 대상으로 지정할 때 다음 사항에 유의하시기 바랍니다.

S3 버킷 이름은 `[\w-]` 정규식 패턴만 사용할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs API 참조에 나와 있는 PutDeliveryDestination API 작업을 참조하시기 바랍니다.

S3 버킷에 접두사를 지정한 경우 해당 경로 아래에 로그가 나타납니다. 접두사를 지정하지 않으면 CloudFront에서 자동으로 `AWSLogs/{account-id}/CloudFront` 접두사를 추가합니다.

자세한 내용은 로그에 액세스하는 경로 예제 섹션을 참조하세요.

## 분할

CloudFront가 S3 버킷으로 액세스 로그를 전송할 때 파티셔닝을 사용하여 액세스 로그를 구성할 수 있습니다. 이렇게 하면 원하는 경로를 기반으로 액세스 로그를 구성하고 찾을 수 있습니다.

다음 변수를 사용하여 폴더 경로를 만들 수 있습니다.

- `{DistributionId}` 또는 `{distributionid}`
- `{yyyy}`
- `{MM}`
- `{dd}`

- {HH}
- {accountid}

원하는 수의 변수를 사용하고 경로에 폴더 이름을 지정할 수 있습니다. 그런 다음 CloudFront는 이 경로를 사용하여 S3 버킷에 폴더 구조를 만듭니다.

예제

- *my_distribution_log_data*/{DistributionId}/*logs*
- /*cloudfront*/{DistributionId}/*my_distribution_log_data*/{yyyy}/{MM}/{dd}/{HH}/*logs*

> **Note**
>
> 접미사 경로의 배포 ID에 두 변수 중 하나를 사용할 수 있습니다. 그러나 AWS Glue에 액세스 로그를 보내는 경우 AWS Glue는 파티션 이름을 소문자로 예상하므로 {distributionid} 변수를 사용해야 합니다. CloudFront에서 기존 로그 구성을 업데이트하여 {DistributionId}를 {distributionid}로 바꿉니다.

Hive 호환 파일 이름 형식

전송된 액세스 로그가 포함된 S3 객체가 Apache Hive와의 통합을 허용하는 접두사 구조를 사용하도록 이 옵션을 사용할 수 있습니다. 자세한 내용은 CreateDelivery API 작업을 참조하시기 바랍니다.

Example예제

```
/cloudfront/DistributionId={DistributionId}/my_distribution_log_data/year={yyyy}/
month={MM}/day={dd}/hour={HH}/logs
```

파티셔닝 및 Hive 호환 옵션에 대한 자세한 내용은 Amazon CloudWatch Logs API 참조의 S3DeliveryConfiguration 요소를 참조하시기 바랍니다.

로그에 액세스하는 경로 예제

S3 버킷을 대상으로 지정할 경우 다음 옵션을 사용하여 액세스 로그의 경로를 만들 수 있습니다.

- Amazon S3 버킷(접두사가 있거나 없는 경우)

- CloudFront 제공 변수를 사용하거나 직접 입력하여 파티셔닝
- Hive 호환 옵션 사용

다음 표는 선택한 옵션에 따라 버킷에 액세스 로그가 표시되는 방법을 보여줍니다.

접두사가 있는 Amazon S3 버킷

| Amazon S3 버킷 이름 | 접미사 경로에서 지정하는 파티션 | 접미사 경로 업데이트 | Hive 호환 사용 설정 여부 | 액세스 로그 전송 대상 |
| --- | --- | --- | --- | --- |
| amzn-s3-demo-bucket/MyLogPrefix | 없음 | 없음 | 아니요 | amzn-s3-demo-bucket/MyLogPrefix/ |
| amzn-s3-demo-bucket/MyLogPrefix | myFolderA/ | myFolderA/ | 아니요 | amzn-s3-demo-bucket/MyLogPrefix/myFolderA/ |
| amzn-s3-demo-bucket/MyLogPrefix | myFolderA/{yyyy} | myFolderA/{yyyy} | 예 | amzn-s3-demo-bucket/MyLogPrefix/myFolderA/year=2025 |

접두사가 없는 Amazon S3 버킷

| Amazon S3 버킷 이름 | 접미사 경로에서 지정하는 파티션 | 접미사 경로 업데이트 | Hive 호환 사용 설정 여부 | 액세스 로그 전송 대상 |
| --- | --- | --- | --- | --- |
| amzn-s3-demo-bucket | 없음 | AWSLogs/{account-i | 아니요 | amzn-s3-demo-bucket/AWSLogs |

| Amazon S3 버킷 이름 | 접미사 경로에서 지정하는 파티션 | 접미사 경로 업데이트 | Hive 호환 사용 설정 여부 | 액세스 로그 전송 대상 |
|---|---|---|---|---|
| | | d}/CloudFront/ | | / <your-account-ID> / CloudFront/ |
| amzn-s3-demo-bucket | myFolderA/ | AWSLogs/{account-id}/CloudFront/myFolderA/ | 아니요 | amzn-s3-demo-bucket/AWSLogs/ <your-account-ID> / CloudFront/myFolderA/ |
| amzn-s3-demo-bucket | myFolderA/ | AWSLogs/{account-id}/CloudFront/myFolderA/ | 예 | amzn-s3-demo-bucket/AWSLogs/aws-account-id=<your-account-ID> / CloudFront/myFolderA/ |
| amzn-s3-demo-bucket | myFolderA/{yyyy} | AWSLogs/{account-id}/CloudFront/myFolderA/{yyyy} | 예 | amzn-s3-demo-bucket/AWSLogs/aws-account-id=<your-account-ID> / CloudFront/myFolderA/year=2025 |

파티션 역할 AWS 계정 ID

| Amazon S3 버킷 이름 | 접미사 경로에서 지정하는 파티션 | 접미사 경로 업데이트 | Hive 호환 사용 설정 여부 | 액세스 로그 전송 대상 |
|---|---|---|---|---|
| amzn-s3-demo-bucket | 없음 | AWSLogs/{account-id}/CloudFront/ | 예 | amzn-s3-demo-bucket/AWSLogs/aws-account-id=*<your-account-ID>* /CloudFront/ |
| amzn-s3-demo-bucket | myFolderA/{accountid} | AWSLogs/{account-id}/CloudFront/myFolderA/{accountid} | 예 | amzn-s3-demo-bucket/AWSLogs/aws-account-id=*<your-account-ID>*/CloudFront/myFolderA/accountid= *<your-account-ID>* |

참고

- {account-id} 변수는 CloudFront용으로 예약되어 있습니다. 접두사 없이 Amazon S3 버킷을 지정하면 CloudFront가 자동으로 접미사 경로에 이 변수를 추가합니다. 로그가 Hive와 호환되는 경우 이 변수는 aws-account-id로 표시됩니다.
- CloudFront가 접미사 경로에 계정 ID를 추가하도록 {accountid} 변수를 사용할 수 있습니다. 로그가 Hive와 호환되는 경우 이 변수는 accountid로 표시됩니다.
- 접미사 경로에 대한 자세한 내용은 S3DeliveryConfiguration을 참조하시기 바랍니다.

### Amazon S3로 전송된 액세스 로그의 예제

```
#Fields: date time x-edge-location asn timestamp(ms) x-host-header cs(Cookie)
2024-11-14    22:30:25    SOF50-P2    16509    1731623425421
d111111abcdef8.cloudfront.net    examplecookie=value2
```

### 표준 로깅 사용 해제

배포가 더 이상 필요하지 않은 경우 배포에 대한 표준 로깅을 사용 해제할 수 있습니다.

표준 로깅을 사용 해제하려면 다음과 같이 합니다.

1. CloudFront 콘솔에 로그인합니다.
2. 배포를 선택한 다음 배포 ID를 선택합니다.
3. 로깅을 선택한 다음 액세스 로그 대상에서 대상을 선택합니다.
4. 관리를 선택한 다음 삭제를 선택합니다.
5. 표준 로깅이 2개 이상인 경우 이전 단계를 반복합니다.

> **Note**
>
> CloudFront 콘솔에서 표준 로깅을 삭제하면 이 작업은 전송 및 전송 대상만 삭제합니다. AWS 계정에서 전송 소스를 삭제하지 않습니다. 전송 소스를 삭제하려면 `aws logs delete-delivery-source --name DeliverySourceName` 명령에서 전송 소스 이름을 지정합니다. 자세한 내용은 Amazon CloudWatch Logs API 참조에 나와 있는 DeleteDeliverySource를 참조하시기 바랍니다.

### 문제 해결

CloudFront 표준 로깅(v2)을 사용할 때 발생하는 일반적인 문제를 해결하려면 다음 정보를 사용합니다.

#### 전송 소스가 이미 존재함

배포에 대한 표준 로깅을 사용하는 경우 전송 소스를 만듭니다. 그런 다음 해당 전송 소스를 사용하여 원하는 대상 유형인 CloudWatch Logs, Firehose, Amazon S3에 대한 전송을 만듭니다. 현재 배포당 하나의 전송 소스만 보유할 수 있습니다. 동일한 배포에 대해 다른 전송 소스를 만들려고 하면 다음 오류 메시지가 나타납니다.

```
This ResourceId has already been used in another Delivery Source in this account
```

다른 전송 소스를 만들려면 먼저 기존 전송 소스를 삭제합니다. 자세한 내용은 Amazon CloudWatch Logs API 참조에 나와 있는 DeleteDeliverySource를 참조하시기 바랍니다.

접미사 경로를 변경했는데 Amazon S3 버킷이 내 로그를 수신할 수 없음

표준 로깅(v2)을 사용 설정하고 접두사 없이 버킷 ARN을 지정하는 경우 CloudFront는 접미사 경로에 다음 기본값(`AWSLogs/{account-id}/CloudFront`)을 추가합니다. CloudFront 콘솔 또는 UpdateDeliveryConfiguration API 작업을 사용하여 다른 접미사 경로를 지정하는 경우 동일한 경로를 사용하도록 Amazon S3 버킷 정책을 업데이트해야 합니다.

Example예: 접미사 경로 업데이트

1. 기본 접미사 경로는 `AWSLogs/{account-id}/CloudFront`이며 `myFolderA`로 바꿉니다.
2. 새 접미사 경로가 Amazon S3 버킷 정책에 지정된 경로와 다르기 때문에 액세스 로그가 전달되지 않습니다.
3. 다음 단계 중 하나를 수행할 수 있습니다.
   - Amazon S3 버킷 권한을 `amzn-s3-demo-bucket/AWSLogs/<your-account-ID>/CloudFront/*`에서 `amzn-s3-demo-bucket/myFolderA/*`로 업데이트합니다.
   - 기본 접미사를 다시 사용하도록 로깅 구성을 업데이트합니다(`AWSLogs/{account-id}/CloudFront`).

자세한 내용은 권한 섹션을 참조하세요.

로그 파일 삭제

CloudFront는 대상에서 로그 파일을 자동으로 삭제하지 않습니다. 로그 파일 삭제에 대한 자세한 내용은 다음 주제를 참조하시기 바랍니다.

Amazon S3

- Amazon Simple Storage Service 콘솔 사용 설명서의 객체 삭제

CloudWatch Logs

- Amazon CloudWatch Logs 사용 설명서의 Working with log groups and log streams
- Amazon CloudWatch Logs API 참조의 DeleteLogGroup

Firehose

- Amazon Data Firehose API 참조의 DeleteDeliveryStream

가격 책정

CloudFront는 표준 로그 사용에 대한 요금을 부과하지 않습니다. 그러나 선택한 로그 전송 대상에 따라 전송, 수집, 스토리지, 액세스에 대한 요금이 발생할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs 요금을 참조하시기 바랍니다. 유료 티어에서 로그 탭을 선택한 다음 판매된 로그에서 각 전송 대상의 정보를 참조하시기 바랍니다.

AWS 서비스의 요금에 대한 자세한 내용은 다음 주제를 참조하시기 바랍니다.

- Amazon CloudWatch Logs 요금
- Amazon Data Firehose 요금
- Amazon S3 요금

> **Note**
>
> 로그 파일을 저장하고 액세스하는 데 Amazon S3 요금이 발생하지만 Amazon S3로 로그를 전송하는 것에 대한 추가 요금은 없습니다. Parquet 옵션을 사용하여 액세스 로그를 Apache Parquet로 변환하면 이 옵션에 대한 CloudWatch 요금이 발생합니다. 자세한 내용은 Vended Logs section for CloudWatch pricing을 참조하시기 바랍니다.

## 표준 로깅(레거시) 구성

> **Notes**
>
> - 이 주제는 이전 버전의 표준 로깅에 관한 내용입니다. 최신 버전은 표준 로깅(v2) 구성를 참조하세요.
> - 표준 로깅(레거시)을 이미 사용 설정했고 Amazon S3에 대한 표준 로깅(v2)을 사용하려는 경우 다른 Amazon S3 버킷을 지정하거나 동일한 버킷에서 별도의 경로를 사용하는 것이 좋습니다(예: 로그 접두사 또는 파티셔닝 사용). 이렇게 하면 어떤 로그 파일이 어떤 배포와 연결되어 있는지 추적하고 로그 파일이 서로 덮어쓰는 것을 방지할 수 있습니다.

표준 로깅(레거시)을 시작하려면 다음 단계를 완료합니다.

1. 로그를 수신할 Amazon S3 버킷을 선택하고 필수 권한을 추가합니다.
2. CloudFront 콘솔 또는 CloudFront API에서 표준 로깅(레거시)을 구성합니다. 로그를 수신할 Amazon S3 버킷만 선택할 수 있습니다.
3. 액세스 로그를 확인합니다.

### 표준 로그용 Amazon S3 버킷 선택

배포에 대해 로깅을 활성화하는 경우 CloudFront에서 로그 파일을 저장할 Amazon S3 버킷을 지정할 수 있습니다. Amazon S3를 오리진으로 사용할 경우에는 로그 파일에 별도의 버킷을 사용하는 것이 좋습니다.

CloudFront에서 액세스 로그를 저장하려는 Amazon S3 버킷을 지정합니다(예: `amzn-s3-demo-bucket.s3.amazonaws.com`).

동일 버킷에 여러 배포에 대한 로그 파일을 저장할 수 있습니다. 로깅을 활성화할 경우 파일 이름에 접두사(선택 사항)를 지정하여 배포와 연결되는 로그 파일을 추적할 수 있습니다.

> **S3 버킷 선택 관련**
>
> - 버킷에 액세스 제어 목록(ACL)이 활성화되어 있어야 합니다. CloudFront 콘솔에서 ACL이 활성화되지 않은 버킷을 선택하면 오류 메시지가 표시됩니다. 권한 섹션을 참조하세요.
> - 버킷 소유자로 설정된 S3 객체 소유권이 적용된 Amazon S3 버킷을 선택하지 마세요. 이 설정은 버킷과 그 안의 객체에 대한 ACL을 비활성화하여 CloudFront에서 버킷으로 로그 파일을 전송하지 못하도록 합니다.
> - 다음 AWS 리전에서는 Amazon S3 버킷을 선택하지 마세요. CloudFront는 이러한 리전의 버킷에 표준 로그를 전달하지 않습니다.
>  - 아프리카(케이프타운)
>  - 아시아 태평양(홍콩)
>  - 아시아 태평양(하이데라바드)
>  - 아시아 태평양(자카르타)
>  - 아시아 태평양(멜버른)
>  - 캐나다 서부(캘거리)
>  - 유럽(밀라노)

- 유럽(스페인)
- 유럽(취리히)
- 이스라엘(텔아비브)
- 중동(바레인)
- 중동(UAE)

권한

> **Important**
>
> 2023년 4월부터 CloudFront 표준 로그에 사용되는 새 S3 버킷에 대해 S3 ACL을 활성화해야 합니다. 버킷을 생성할 때 ACL을 활성화하거나 기존 버킷에 ACL을 활성화할 수 있습니다. 변경 사항에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 새 S3 버킷에 대한 기본 설정 FAQ와 AWS 뉴스 블로그의 2023년 4월에 예정된 Amazon S3 보안 변경 사항을 참조하세요.

로그 파일용으로 지정하는 버킷에 대해 AWS 계정에 다음 권한이 있어야 합니다.

- 버킷의 ACL에서 `FULL_CONTROL` 권한을 부여받아야 합니다. 버킷 소유자인 경우 기본적으로 계정에 이 권한이 있습니다. 이 권한이 없을 경우 버킷 소유자가 버킷에 대한 ACL을 업데이트해야 합니다.
- `s3:GetBucketAcl`
- `s3:PutBucketAcl`

버킷에 대한 ACL

배포를 생성하거나 업데이트하고 로깅을 활성화할 경우 CloudFront는 이러한 권한을 사용하여 버킷에 대한 ACL을 업데이트함으로써 `awslogsdelivery` 계정에 `FULL_CONTROL` 권한을 부여합니다. `awslogsdelivery` 계정이 로그 파일을 버킷에 기록합니다. ACL을 업데이트하는 데 필요한 권한이 계정에 없을 경우 배포를 생성하거나 업데이트할 수 없습니다.

프로그래밍 방식으로 버킷 생성 요청을 제출했는데 지정한 이름의 버킷이 이미 존재하는 경우, S3는 해당 버킷에 대한 권한을 기본값으로 재설정합니다. 액세스 로그를 S3 버킷에 저장하도록 CloudFront를 구성했는데 더 이상 해당 버킷에서 로그를 가져오지 않는 경우, 버킷에 대한 권한을 점검하여 CloudFront가 필요한 권한을 가지고 있는지 확인하세요.

버킷에 대한 ACL 복원

`awslogsdelivery` 계정에 대한 권한을 제거할 경우 CloudFront는 S3 버킷에 로그를 저장할 수 없습니다. CloudFront에서 배포에 대한 로그를 다시 저장할 수 있게 하려면 다음 중 하나를 수행하여 ACL 권한을 복원합니다.

- CloudFront의 배포에 대한 로깅을 비활성화한 다음 이를 다시 활성화합니다. 자세한 내용은 표준 로깅 단원을 참조하십시오.
- Amazon S3 콘솔의 S3 버킷으로 이동하고 권한을 추가하여 `awslogsdelivery`에 대한 ACL 권한을 수동으로 추가합니다. `awslogsdelivery`에 대한 ACL을 추가하려면 계정에 대해 다음과 같이 정식 ID를 입력해야 합니다.

  ```
  c4c1ede66af53448b93c283ce9448c4ba468c9432aa01d700d3878632f77d2d0
  ```

  S3 버킷에 ACL을 추가하는 방법에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 ACL 구성을 참조하세요.

각 로그 파일의 ACL

버킷의 ACL 외에도 각 로그 파일마다 ACL이 있습니다. 버킷 소유자에게는 각 로그 파일에 대한 `FULL_CONTROL` 권한이 있으며, 배포 소유자(버킷 소유자와 다른 경우)에게는 아무런 권한이 없습니다. `awslogsdelivery` 계정에는 읽기 및 쓰기 권한이 있습니다.

로깅 비활성화

로깅을 비활성화하더라도 CloudFront가 버킷이나 로그 파일에 대한 ACL이 자동으로 삭제되지 않습니다. 필요한 경우 ACL을 삭제할 수 있습니다.

SSE-KMS 버킷에 필요한 키 정책

표준 로그의 S3 버킷이 고객 관리형 키를 사용하여 AWS KMS keys(SSE-KMS)로 서버 측 암호화를 사용하는 경우, 고객 관리형 키의 키 정책에 다음 설명을 추가해야 합니다. 이렇게 하면 CloudFront가 버킷에 로그 파일을 쓸 수 있습니다. CloudFront는 로그 파일을 버킷에 쓸 수 없기 때문에 SSE-KMS와 함께 AWS 관리형 키를 사용할 수 없습니다.

```
{
    "Sid": "Allow CloudFront to use the key to deliver logs",
    "Effect": "Allow",
    "Principal": {
        "Service": "delivery.logs.amazonaws.com"
    },
```

```
    "Action": "kms:GenerateDataKey*",
    "Resource": "*"
}
```

표준 로그의 S3 버킷이 S3 버킷 키와 함께 SSE-KMS를 사용하는 경우 정책 설명에 `kms:Decrypt` 권한을 추가해야 합니다. 이 경우 전체 정책 설명은 다음과 같습니다.

```
{
    "Sid": "Allow CloudFront to use the key to deliver logs",
    "Effect": "Allow",
    "Principal": {
        "Service": "delivery.logs.amazonaws.com"
    },
    "Action": [
        "kms:GenerateDataKey*",
        "kms:Decrypt"
    ],
    "Resource": "*"
}
```

> **Note**
>
> S3 버킷에 대해 SSE-KMS를 사용 설정할 때 고객 관리형 키에 대한 전체 ARN을 지정합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 AWS KMS keys(SSE-KMS)로 서버 측 암호화 지정을 참조하세요.

## 표준 로깅(레거시) 사용

표준 로그를 사용하려면 CloudFront 콘솔 또는 CloudFront API를 사용합니다.

목차


- 표준 로깅(레거시) 사용(CloudFront 콘솔)
- 표준 로깅(레거시) 사용(CloudFront API)


### 표준 로깅(레거시) 사용(CloudFront 콘솔)

CloudFront 배포에 대한 표준 로그를 사용하려면 다음과 같이 합니다(콘솔의 경우).

1. CloudFront 콘솔을 사용하여 새 배포를 만들거나 기존 배포를 업데이트합니다.

2. 표준 로깅 섹션의 로그 전송에서 켜기를 선택합니다.
3. (선택 사항) 쿠키 로깅에서 로그에 쿠키를 포함하려면 켜기를 선택합니다. 자세한 내용은 쿠키 로깅 단원을 참조하십시오.

> **Tip**
> 쿠키 로깅은 배포의 모든 표준 로그에 적용되는 전역 설정입니다. 별도의 전송 대상에서 이 설정을 재정의할 수 없습니다.

4. 전송 대상 섹션에서 Amazon S3(레거시)를 지정합니다.
5. Amazon S3 버킷을 지정합니다. 버킷이 아직 없는 경우 생성을 선택하거나 설명서를 보고 버킷을 만들 수 있습니다.
6. (선택 사항) 로그 접두사에서, CloudFront에서 이 배포에 대한 액세스 로그 파일 이름에 접두사로 지정하려는 문자열(해당하는 경우)을 지정합니다(예: `exampleprefix/`). 후행 슬래시(/)는 선택 사항이지만 로그 파일을 쉽게 검색할 수 있도록 넣는 것이 좋습니다. 자세한 내용은 로그 접두사 단원을 참조하십시오.
7. 배포를 업데이트하거나 만드는 단계를 완료합니다.
8. 로그 페이지에서 배포 옆의 표준 로그 상태가 활성화됨인지 확인합니다.

   표준 로깅 전송 및 로그 필드에 대한 자세한 내용은 표준 로깅 참조 섹션을 참조하시기 바랍니다.

### 표준 로깅(레거시) 사용(CloudFront API)

CloudFront API를 사용하여 배포에 대한 표준 로그를 사용 설정할 수도 있습니다.

배포에 대한 표준 로그를 사용하려면 다음과 같이 합니다(CloudFront API의 경우).

- CreateDistribution 또는 UpdateDistribution API 작업을 사용하고 LoggingConfig 객체를 구성합니다.

### 표준 로깅 설정 편집

CloudFront 콘솔 또는 CloudFront API를 사용하여 로깅을 활성화/비활성화하거나, 로그가 저장되는 Amazon S3 버킷을 변경하고 로그 파일의 접두사를 변경할 수 있습니다. 로깅 설정에 대한 변경 내용은 12시간 내에 적용됩니다.

자세한 정보는 다음 주제를 참조하십시오.

- CloudFront 콘솔을 사용하여 배포를 업데이트하려면 배포 업데이트 섹션을 참조하세요.
- CloudFront API를 사용하여 배포를 업데이트하려면 Amazon CloudFront API 참조의 UpdateDistribution을 참조하세요.

## Amazon S3로 로그 전송

Amazon S3로 로그를 전송하면 로그가 다음 형식으로 표시됩니다.

### 파일 이름 형식

CloudFront에서 Amazon S3 버킷에 저장하는 각 로그 파일의 이름에는 다음 파일 이름 형식이 사용됩니다.

```
<optional prefix>/<distribution ID>.YYYY-MM-DD-HH.unique-ID.gz
```

날짜 및 시간이 협정 세계시(UTC)로 표시됩니다.

예를 들어 접두사로 `example-prefix`를 사용하고 배포 ID가 `EMLARXS9EXAMPLE`이면 파일 이름은 다음과 같습니다.

```
example-prefix/EMLARXS9EXAMPLE.2019-11-14-20.RT4KCN4SGK9.gz
```

배포에 대한 로깅을 활성화할 경우 파일 이름에 접두사(선택 사항)를 지정하여 배포와 연결되는 로그 파일을 추적할 수 있습니다. 로그 파일 접두사의 값을 포함하고 접두사가 슬래시(/)로 끝나지 않으면 CloudFront에서 슬래시를 자동으로 추가합니다. 접두사가 슬래시로 끝나면 CloudFront에서 슬래시를 추가하지 않습니다.

파일 이름의 끝에 붙은 `.gz`는 CloudFront에서 로그 파일이 gzip으로 압축되었음을 나타냅니다.

### 표준 로그 파일 형식

로그 파일의 각 항목은 단일 최종 사용자 요청에 대한 세부 정보를 제공합니다. 이러한 로그 파일은 다음과 같은 특성을 갖습니다.

- W3C 확장 로그 파일 형식을 사용합니다.
- 탭으로 구분된 값을 포함합니다.
- 레코드를 포함합니다. 이 때 레코드가 시간 순서대로 나열되어 있지 않을 수도 있습니다.
- 두 헤더 행(파일 형식 버전이 표시된 행과 각 레코드에 포함된 W3C 필드가 나열된 행)을 포함합니다.

- 필드 값에서 공백 및 다른 특정 문자에 상응하는 URL 인코딩 값을 포함합니다.

  상응하는 URL 인코딩 값은 다음과 같은 문자에 사용됩니다.

  - ASCII 문자 코드 0~32
  - ASCII 문자 코드 127 이상
  - 다음 표의 모든 문자

  URL 인코딩 표준은 RFC 1738에 정의되어 있습니다.

| URL 인코딩 값 | 문자 |
|---|---|
| %3C | < |
| %3E | > |
| %22 | " |
| %23 | # |
| %25 | % |
| %7B | { |
| %7D | } |
| %7C | \| |
| %5C | \ |
| %5E | ^ |
| %7E | ~ |
| %5B | [ |
| %5D | ] |
| %60 | ` |
| %27 | ' |

| URL 인코딩 값 | 문자 |
| --- | --- |
| %20 | 공백 |

### 로그 파일 삭제

CloudFront는 Amazon S3 버킷에서 로그 파일을 자동으로 삭제하지 않습니다. Amazon S3 버킷에서 로그 파일을 삭제하는 방법에 대한 자세한 내용은 Amazon Simple Storage Service 콘솔 사용 설명서의 객체 삭제를 참조하시기 바랍니다.

### 요금

표준 로깅은 CloudFront의 선택적 기능입니다. CloudFront는 표준 로그 사용에 대한 요금을 부과하지 않습니다. 하지만 Amazon S3에 파일을 저장하고 액세스하는 데 일반적인 Amazon S3 요금이 부과됩니다. 사용자는 언제든지 이를 삭제할 수 있습니다.

Amazon S3 요금에 대한 자세한 내용은 Amazon S3 요금을 참조하세요.

CloudFront 요금에 대한 자세한 내용은 CloudFront 요금을 참조하세요.

## 표준 로깅 참조

다음 섹션은 표준 로깅(v2)과 표준 로깅(레거시) 모두에 적용됩니다.

주제

- 로그 파일 전송 타이밍
- 요청 URL 또는 헤더가 최대 크기를 초과했을 때 요청이 기록되는 방법
- 로그 파일 필드
- 로그 분석

### 로그 파일 전송 타이밍

CloudFront는 1시간 동안 여러 번 배포에 대한 표준 로그를 전송합니다. 일반적으로 로그 파일에는 지정된 시간 동안 CloudFront에서 수신한 요청에 대한 정보가 들어 있습니다. CloudFront는 일반적으로 로그에 표시된 이벤트가 발생하고 한 시간 이내에 해당 기간 동안의 로그 파일을 대상으로 전송합니다. 하지만 한 기간에 대한 일부 또는 전체 로그 파일 항목이 때때로 최대 24시간까지 지연되기도 합니다. 로그 항목이 지연되는 경우 CloudFront는 해당 항목을 로그 파일에 저장하며 파일 이름에는 파일이 전송된 날짜 및 시간이 아니라 요청이 발생한 기간의 날짜 및 시간이 포함됩니다.

CloudFront는 로그 파일을 생성할 때 로그 파일에 포함되는 기간 중에 객체에 대한 요청을 받은 모든 엣지 로케이션의 배포 관련 정보를 통합합니다.

CloudFront는 CloudFront에서 배포와 연결된 객체에 대해 받는 요청 수에 따라 한 기간 동안 두 개 이상의 파일을 저장할 수 있습니다.

CloudFront는 로깅을 활성화하고 약 4시간이 지나면 액세스 로그들을 안정적으로 전송하기 시작합니다. 그 전에는 일부 액세스 로그들을 가져올 수 있습니다.

> **Note**
>
> 해당 기간 동안 객체를 요청하는 사용자가 없으면 해당 기간 동안 아무런 로그 파일도 수신되지 않습니다.

요청 URL 또는 헤더가 최대 크기를 초과했을 때 요청이 기록되는 방법

쿠키를 포함하여 모든 요청 헤더의 총 크기가 20KB를 초과하거나 URL이 8192바이트를 초과하면 CloudFront가 요청을 완전히 구문 분석할 수 없고 요청을 기록할 수 없습니다. 요청이 기록되지 않으므로 로그 파일에 반환된 HTTP 오류 상태 코드가 표시되지 않습니다.

요청 본문이 최대 크기를 초과하면 HTTP 오류 상태 코드를 포함하여 요청이 기록됩니다.

로그 파일 필드

배포에 대한 로그 파일에는 33개의 필드가 포함되어 있습니다. 다음 목록에는 각 필드 이름이 해당 필드의 정보에 대한 설명과 함께 순서대로 포함되어 있습니다.

1. **`date`**

   이벤트가 발생한 날짜가 `YYYY-MM-DD` 형식으로 표시됩니다. 예를 들어 `2019-06-30`입니다. 날짜 및 시간이 협정 세계시(UTC)로 표시됩니다. WebSocket 연결의 경우, 연결이 종료된 날짜가 됩니다.

2. **`time`**

   CloudFront 서버에서 요청 응답을 완료한 시간(UTC)입니다(예: `01:42:39`). WebSocket 연결의 경우, 연결이 종료된 시간이 됩니다.

3. **`x-edge-location`**

   요청을 처리한 엣지 로케이션입니다. 각 엣지 로케이션은 3자 코드와 임의로 배정된 번호로 식별됩니다(예: DFW3). 3자 코드는 일반적으로 엣지 로케이션의 지리적 위치 부근 공항을 나타내는 국제 항공 수송 협회(IATA) 공항 코드에 상응합니다. 이러한 약어는 향후에 변경될 수 있습니다.

4. **sc-bytes**

   요청에 대한 응답으로 서버가 최종 사용자에게 보낸 총 바이트 수(헤더 포함)입니다. WebSocket 및 gRPC 연결의 경우, 연결을 통해 서버에서 클라이언트로 전송된 총 바이트 수입니다.

5. **c-ip**

   요청을 한 최종 사용자의 IP 주소(예: `192.0.2.183` 또는 `2001:0db8:85a3::8a2e:0370:7334`)입니다. 최종 사용자가 HTTP 프록시 또는 로드 밸런서를 사용하여 요청을 전송하는 경우, 해당 필드 값은 프록시 또는 로드 밸런서의 IP 주소입니다. `x-forwarded-for` 필드도 참조하십시오.

6. **cs-method**

   최종 사용자로부터 수신된 HTTP 요청 메서드입니다.

7. **cs(Host)**

   CloudFront 배포의 도메인 이름(예: d111111abcdef8.cloudfront.net)입니다.

8. **cs-uri-stem**

   경로 및 객체를 식별하는 요청 URL의 일부(예: `/images/cat.jpg`)입니다. URL과 쿼리 문자열의 물음표(?)는 로그에 포함되지 않습니다.

9. **sc-status**

   다음 값 중 하나를 포함합니다.

   - 서버 응답의 HTTP 상태 코드(예: `200`)입니다.
   - `000`(서버가 요청에 응답하기 전에 최종 사용자가 연결을 종료했음을 나타냄) 서버가 응답을 전송하기 시작한 후 최종 사용자가 연결을 종료하는 경우, 이 필드에는 서버가 전송하기 시작하는 응답의 HTTP 상태 코드가 포함됩니다.

10. **cs(Referer)**

    요청의 `Referer` 헤더 값입니다. 다음은 요청을 시작한 도메인 이름입니다. 일반적인 참조자에는 검색 엔진, 객체에 직접 연결하는 기타 웹 사이트 및 자체 웹 사이트가 포함됩니다.

11. **cs(User-Agent)**

    요청의 `User-Agent` 헤더 값입니다. `User-Agent` 헤더는 요청을 제출한 디바이스 유형 및 브라우저 등의 요청 소스를 식별하고 검색 엔진에서 요청이 온 경우에는 어느 검색 엔진인지 식별합니다.

12. **cs-uri-query**

    요청 URL의 쿼리 문자열 부문(있는 경우)입니다.

URL에 쿼리 문자열이 포함되지 않은 경우 이 필드의 값은 하이픈(-)입니다. 자세한 내용은 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱 섹션을 참조하세요.

13. **`cs(Cookie)`**

이름-값 페어 및 관련 속성을 포함한 요청의 `Cookie` 헤더입니다.

쿠키 로깅을 활성화하는 경우 CloudFront는 오리진에 전달하도록 선택한 쿠키와 관계없이 모든 요청의 쿠키를 기록합니다. 요청에 쿠키 헤더가 포함되지 않은 경우 이 필드의 값은 하이픈(-)입니다. 쿠키에 대한 자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하십시오.

14. **`x-edge-result-type`**

마지막 바이트가 서버를 떠난 후 서버가 응답을 분류한 방식입니다. 경우에 따라 서버가 응답을 전송할 준비가 된 시간과 가 응답 전송을 완료한 시간 사이에 결과 유형이 변경될 수 있습니다. `x-edge-response-result-type` 필드도 참조하십시오.

예를 들어 HTTP 스트리밍에서 서버가 캐시 스트림의 한 세그먼트를 찾는다고 가정해 보십시오. 이 시나리오에서 이 필드의 값은 보통 `Hit`입니다. 그러나 서버가 전체 세그먼트를 전달하기 전에 최종 사용자가 연결을 종료하면 최종 결과 유형(및 이 필드 값)은 `Error`가 됩니다.

콘텐츠가 캐시 가능하지 않고 오리진으로 직접 프록시 처리되므로 WebSocket 및 gRPC 연결에서 이 필드의 값이 `Miss`가 됩니다.

가능한 값은 다음과 같습니다.

- `Hit` - 서버가 캐시에서 최종 사용자에게 객체를 제공했습니다.
- `RefreshHit` - 서버가 캐시에서 객체를 찾았지만 객체가 만료되었기 때문에 서버가 오리진에 접속하여 캐시에 최신 버전의 객체가 있는지 확인했습니다.
- `Miss` - 캐시의 객체로는 요청을 충족할 수 없어서 서버가 요청을 오리진으로 전달했으며 결과가 최종 사용자에게 반환되었습니다.
- `LimitExceeded` - CloudFront 할당량(이전에는 제한이라고 함)이 초과되어 요청이 거부되었습니다.
- `CapacityExceeded` - 객체 제공을 요청하는 시점에 용량이 충분하지 않아 서버에서 HTTP 503 상태 코드가 반환되었습니다.
- `Error` - 일반적으로 이는 요청으로 인해 클라이언트 오류(`sc-status` 필드 값이 `4xx` 범위에 있음) 또는 서버 오류(`sc-status` 필드 값이 `5xx` 범위에 있음)가 발생했음을 의미합니다. `sc-status` 필드의 값이 `200`인 경우 또는 이 필드의 값이 `Error`이고 `x-edge-response-`

`result-type` 필드의 값이 `Error`가 아닌 경우, HTTP 요청이 성공했지만 모든 바이트를 수신하기 전에 클라이언트의 연결이 끊어졌음을 의미합니다.

- `Redirect` - 서버는 배포 설정에 따라 최종 사용자를 HTTP에서 HTTPS로 리디렉션했습니다.
- `LambdaExecutionError` - 잘못된 연결, 함수 시간 초과, AWS 종속성 문제 또는 기타 일반 가용성 문제로 인해 배포와 연결된 Lambda@Edge 함수가 완료되지 않았습니다.

15. **`x-edge-request-id`**

요청을 고유하게 식별하는 불투명 문자열입니다. CloudFront는 `x-amz-cf-id` 응답 헤더에도 이 문자열을 전송합니다.

16. **`x-host-header`**

최종 사용자가 이 요청에 대한 `Host` 헤더에 포함한 값입니다. 객체 URL에 CloudFront 도메인 이름(예: d111111abcdef8.cloudfront.net)을 사용하는 경우, 이 필드에는 도메인 이름이 포함됩니다. 객체 URL(예: www.example.com)에 대체 도메인 이름(CNAME)을 사용하는 경우, 이 필드에 대체 도메인 이름이 포함됩니다.

대체 도메인 이름을 사용하는 경우 배포와 연결된 도메인 이름은 필드7에 있는 `cs(Host)` 섹션을 참조하세요.

17. **`cs-protocol`**

최종 사용자 요청 프로토콜(`http`, `https`, `grpcs`, `ws` 또는 `wss`)입니다.

18. **`cs-bytes`**

최종 사용자가 요청에 포함한 데이터의 바이트 수(헤더 포함)입니다. WebSocket 및 gRPC 연결의 경우, 연결에서 클라이언트로부터 서버로 전송된 총 바이트 수입니다.

19. **`time-taken`**

서버가 최종 사용자 요청을 수신한 시점부터 서버에서 출력 대기열에 대한 응답의 최종 바이트를 쓰는 시점 간의 시간 차이(1/1,000초 단위, 예: 0.082)로, 서버에서 측정됩니다. 최종 사용자 관점에서 보면 전체 응답을 가져오는 데 걸리는 총 시간은 네트워크 지연 시간과 TCP 버퍼링으로 인해 이 값보다 큽니다.

20. **`x-forwarded-for`**

최종 사용자가 HTTP 프록시 또는 로드 밸런서를 사용하여 요청을 전송하는 경우, `c-ip` 필드의 값은 프록시 또는 로드 밸런서의 IP 주소입니다. 이 경우 이 필드는 요청을 시작한 최종 사용자의 IP 주소입니다. 이 필드에는 여러 개의 쉼표로 구분된 IP 주소가 포함될 수 있습니다. 각 IP 주소는 IPv4

주소(예:192.0.2.183) 또는 IPv6 주소(예:2001:0db8:85a3::8a2e:0370:7334)일 수 있습니다.

최종 사용자가 HTTP 프록시 또는 로드 밸런서를 사용하지 않은 경우 이 필드의 값은 하이픈(-)입니다.

21 **ssl-protocol**

요청에 HTTPS가 사용된 경우, 이 필드에는 최종 사용자 및 서버가 요청 및 응답 전송을 위해 협상한 SSL/TLS 프로토콜이 포함됩니다. 가능한 값 목록은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호에서 지원되는 SSL/TLS 프로토콜을 참조하세요.

필드 17의 `cs-protocol`이 `http`인 경우 이 필드의 값은 하이픈(-)입니다.

22 **ssl-cipher**

요청에 HTTPS가 사용되는 경우, 이 필드에는 최종 사용자 및 서버가 요청 및 응답 암호화를 위해 협상한 SSL/TLS 암호가 포함됩니다. 가능한 값 목록은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호에서 지원되는 SSL/TLS 암호를 참조하세요.

필드 17의 `cs-protocol`이 `http`인 경우 이 필드의 값은 하이픈(-)입니다.

23 **x-edge-response-result-type**

최종 사용자에게 응답을 반환하기 직전에 서버에서 응답을 분류한 방식입니다. `x-edge-result-type` 필드도 참조하십시오. 가능한 값은 다음과 같습니다.

- `Hit` - 서버가 캐시에서 최종 사용자에게 객체를 제공했습니다.
- `RefreshHit` - 서버가 캐시에서 객체를 찾았지만 객체가 만료되었기 때문에 서버가 오리진에 접속하여 캐시에 최신 버전의 객체가 있는지 확인했습니다.
- `Miss` - 캐시의 객체로는 요청을 충족할 수 없어서 서버가 요청을 오리진 서버로 전달했으며 결과가 최종 사용자에게 반환되었습니다.
- `LimitExceeded` - CloudFront 할당량(이전에는 제한이라고 함)이 초과되어 요청이 거부되었습니다.
- `CapacityExceeded` - 객체 제공을 요청하는 시점에 엣지 로케이션의 용량이 충분하지 않아 서버에서 503 오류가 반환되었습니다.
- `Error` - 일반적으로 이는 요청으로 인해 클라이언트 오류(`sc-status` 필드 값이 `4xx` 범위에 있음) 또는 서버 오류(`sc-status` 필드 값이 `5xx` 범위에 있음)가 발생했음을 의미합니다.

  `x-edge-result-type` 필드의 값이 `Error`이고 이 필드의 값이 `Error`가 아닌 경우 다운로드를 완료하기 전에 클라이언트 연결이 끊어졌습니다.

- `Redirect` - 서버는 배포 설정에 따라 최종 사용자를 HTTP에서 HTTPS로 리디렉션했습니다.
- `LambdaExecutionError` - 잘못된 연결, 함수 시간 초과, AWS 종속성 문제 또는 기타 일반 가용성 문제로 인해 배포와 연결된 Lambda@Edge 함수가 완료되지 않았습니다.

24. **`cs-protocol-version`**

최종 사용자가 요청에서 지정한 HTTP 버전입니다. 가능한 값은 `HTTP/0.9`, `HTTP/1.0`, `HTTP/1.1`, `HTTP/2.0` 및 `HTTP/3.0`입니다.

25. **`fle-status`**

배포에 대해 필드 레벨 암호화가 구성될 때, 요청 본문이 처리되었는지를 나타내는 코드가 이 필드에 포함됩니다. 서버가 성공적으로 요청 본문을 처리하고, 지정된 필드 값을 암호화하고, 요청을 오리진으로 전달할 때 이 필드 값은 `Processed`입니다. 이 경우 `x-edge-result-type` 값은 계속해서 클라이언트 측 또는 서버 측 오류를 나타낼 수 있습니다.

이 필드에 가능한 값은 다음과 같습니다.

- `ForwardedByContentType` - 어떠한 콘텐츠 유형도 구성되지 않았기 때문에 서버가 요청을 구문 분석 또는 암호화 없이 오리진에 전달했습니다.
- `ForwardedByQueryArgs` - 요청에 필드 레벨 암호화 구성에 없는 쿼리 인수가 포함되기 때문에 서버가 요청을 구문 분석 또는 암호화 없이 오리진에 전달했습니다.
- `ForwardedDueToNoProfile` - 필드 레벨 암호화 구성에 어떠한 프로필도 지정되지 않았기 때문에 서버가 요청을 구문 분석 또는 암호화 없이 오리진에 전달했습니다.
- `MalformedContentTypeClientError` - `Content-Type` 헤더 값이 유효하지 않은 형식이기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.
- `MalformedInputClientError` - 요청 본문이 유효하지 않은 형식이기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.
- `MalformedQueryArgsClientError` - 쿼리 인수가 비어 있거나 유효하지 않은 형식이기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.
- `RejectedByContentType` - 필드 레벨 암호화 구성에 어떠한 콘텐츠 유형도 지정되지 않았기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.
- `RejectedByQueryArgs` - 필드 레벨 암호화 구성에 어떠한 쿼리 인수도 지정되지 않았기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.
- `ServerError` - 오리진 서버가 오류를 반환했습니다.

요청이 필드 레벨 암호화 할당량(이전에는 제한이라고 함) 을 초과하면 이 필드에 다음 오류 코드 중 하나가 포함되고 서버는 HTTP 상태 코드 400을 최종 사용자에게 반환합니다. 필드 레벨 암호화의 현재 할당량에 대한 목록은 필드 레벨 암호화에 대한 할당량 단원을 참조하세요.

- `FieldLengthLimitClientError` - 암호화되도록 구성된 필드가 허용되는 최대 길이를 초과했습니다.
- `FieldNumberLimitClientError` - 배포가 암호화하도록 구성된 요청에 허용된 필드 수보다 많은 필드가 있습니다.
- `RequestLengthLimitClientError` - 필드 레벨 암호화 구성 시 요청 본문의 길이가 허용되는 최대 길이를 초과했습니다.

배포에 대해 필드 레벨 암호화가 구성되지 않은 경우 이 필드의 값은 하이픈(-)입니다.

26. **`fle-encrypted-fields`**

서버가 암호화하여 오리진에 전달한 필드 레벨 암호화의 수입니다. CloudFront 서버는 데이터를 암호화하면서 처리된 요청을 오리진으로 스트리밍합니다. 따라서 `fle-status` 값이 오류인 경우에도 이 필드에 값이 있을 수 있습니다.

배포에 대해 필드 레벨 암호화가 구성되지 않은 경우 이 필드의 값은 하이픈(-)입니다.

27. **`c-port`**

최종 사용자가 보낸 요청의 포트 번호입니다.

28. **`time-to-first-byte`**

요청을 수신한 후 응답의 첫 바이트를 쓸 때까지의 시간(초)이며 서버에서 측정합니다.

29. **`x-edge-detailed-result-type`**

이 필드에는 `x-edge-result-type` 필드와 동일한 값이 포함됩니다. 단, 다음과 같은 경우는 예외입니다.

- 객체가 Origin Shield 계층에서 최종 사용자에게 제공된 경우 이 필드에 `OriginShieldHit`가 포함합니다.
- 객체가 CloudFront 캐시에 없고 오리진 요청 Lambda @Edge 함수에 의해 응답이 생성된 경우 이 필드에는 `MissGeneratedResponse`가 포함됩니다.
- `x-edge-result-type` 필드의 값이 `Error`인 경우 이 필드에는 오류에 대한 추가 정보를 제공하는 다음 값 중 하나가 포함됩니다.
  - `AbortedOrigin` - 서버에 오리진 관련된 문제가 발생했습니다.

- `ClientCommError` - 서버와 최종 사용자 간의 통신 문제로 인해 최종 사용자에게 대한 응답이 중단되었습니다.
- `ClientGeoBlocked` - 최종 사용자의 지리적 위치에서 요청을 거부하도록 배포가 구성됩니다.
- `ClientHungUpRequest` - 요청을 보내는 동안 최종 사용자가 중간에 중지했습니다.
- `Error` - 오류 유형이 기타 모든 카테고리에 적합하지 않아 오류가 발생했습니다. 이러한 오류 유형은 서버가 캐시에서 오류 응답을 제공할 때 발생할 수 있습니다.
- `InvalidRequest` - 서버가 최종 사용자로부터 잘못된 요청을 받았습니다.
- `InvalidRequestBlocked` - 요청한 리소스에 대한 액세스가 차단됩니다.
- `InvalidRequestCertificate` - 배포가 HTTPS 연결이 설정된 SSL/TLS 인증서와 일치하지 않습니다.
- `InvalidRequestHeader` - 요청에 잘못된 헤더가 포함되어 있습니다.
- `InvalidRequestMethod` - 사용되었던 HTTP 요청 메서드를 처리하도록 배포가 구성되지 않았습니다. 이러한 문제는 배포에서 캐시 가능한 요청만 지원하는 경우 발생할 수 있습니다.
- `OriginCommError` - 오리진에 연결하거나 오리진에서 데이터를 읽는 동안 요청 시간이 초과되었습니다.
- `OriginConnectError` - 서버를 오리진에 연결할 수 없습니다.
- `OriginContentRangeLengthError` - 오리진 응답의 `Content-Length` 헤더가 `Content-Range` 헤더의 길이와 일치하지 않습니다.
- `OriginDnsError` - 서버에서 오리진의 도메인 이름을 확인할 수 없습니다.
- `OriginError` - 오리진에서 잘못된 응답을 반환했습니다.
- `OriginHeaderTooBigError` - 오리진에서 반환한 헤더가 너무 커서 엣지 서버가 처리할 수 없습니다.
- `OriginInvalidResponseError` - 오리진에서 잘못된 응답을 반환했습니다.
- `OriginReadError` - 오리진에서 서버를 읽을 수 없습니다.
- `OriginWriteError` - 오리진에 서버를 쓸 수 없습니다.
- `OriginZeroSizeObjectError` - 오리진에서 전송된 크기가 0인 객체로 인해 오류가 발생했습니다.
- `SlowReaderOriginError` - 최종 사용자가 오리진 오류를 일으킨 메시지를 느리게 읽었습니다.

30 **`sc-content-type`**

응답의 HTTP `Content-Type` 헤더 값입니다.

31 **sc-content-len**

응답의 HTTP `Content-Length` 헤더 값입니다.

32 **sc-range-start**

응답에 HTTP `Content-Range` 헤더가 포함되어 있으면 이 필드에 범위 시작 값이 포함됩니다.

33 **sc-range-end**

응답에 HTTP `Content-Range` 헤더가 포함되어 있으면 이 필드에 범위 끝 값이 포함됩니다.

34 **distribution-tenant-id**

배포 테넌트의 ID입니다.

35 **connection-id**

TLS 연결의 고유 식별자입니다.

이 필드에 대한 정보를 가져오려면 먼저 배포에 대해 mTLS를 활성화해야 합니다. 자세한 내용은 CloudFront에 대한 상호 TLS 인증(뷰어 mTLS) 섹션을 참조하세요.

다음은 배포 시 로그 파일 예제입니다.

```
#Version: 1.0
#Fields: date time x-edge-location sc-bytes c-ip cs-method cs(Host) cs-uri-stem sc-
status cs(Referer) cs(User-Agent) cs-uri-query cs(Cookie) x-edge-result-type x-edge-
request-id x-host-header cs-protocol cs-bytes time-taken x-forwarded-for ssl-protocol
 ssl-cipher x-edge-response-result-type cs-protocol-version fle-status fle-encrypted-
fields c-port time-to-first-byte x-edge-detailed-result-type sc-content-type sc-
content-len sc-range-start sc-range-end
2019-12-04 21:02:31 LAX1 392 192.0.2.100 GET d111111abcdef8.cloudfront.net /
index.html 200 - Mozilla/5.0%20(Windows%20NT%2010.0;%20Win64;
%20x64)%20AppleWebKit/537.36%20(KHTML,%20like
%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Hit
 SOX4xwn4XV6Q4rgb7XiVGOHms_BGlTAC4KyHmureZmBNrjGdRLiNIQ== d111111abcdef8.cloudfront.net
 https 23 0.001 - TLSv1.2 ECDHE-RSA-AES128-GCM-SHA256 Hit HTTP/2.0 - - 11040 0.001 Hit
 text/html 78 - -
2019-12-04 21:02:31 LAX1 392 192.0.2.100 GET d111111abcdef8.cloudfront.net /
index.html 200 - Mozilla/5.0%20(Windows%20NT%2010.0;%20Win64;
%20x64)%20AppleWebKit/537.36%20(KHTML,%20like
%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Hit
 k6WGMNkEzR5BEM_SaF47gjtX9zBDO2m349OY2an0QPEaUum1ZOLrow== d111111abcdef8.cloudfront.net
```

```
 https 23 0.000 - TLSv1.2 ECDHE-RSA-AES128-GCM-SHA256 Hit HTTP/2.0 - - 11040 0.000 Hit
 text/html 78 - -
2019-12-04 21:02:31 LAX1 392 192.0.2.100 GET d111111abcdef8.cloudfront.net /
index.html 200 - Mozilla/5.0%20(Windows%20NT%2010.0;%20Win64;
%20x64)%20AppleWebKit/537.36%20(KHTML,%20like
%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Hit
 f37nTMVvnKvV2ZSvEsivup_c2kZ7VXzYdjC-GUQZ5qNs-89BlWazbw== d111111abcdef8.cloudfront.net
 https 23 0.001 - TLSv1.2 ECDHE-RSA-AES128-GCM-SHA256 Hit HTTP/2.0 - - 11040 0.001 Hit
 text/html 78 - -
2019-12-13 22:36:27 SEA19-C1 900 192.0.2.200 GET d111111abcdef8.cloudfront.net /
favicon.ico 502 http://www.example.com/ Mozilla/5.0%20(Windows
%20NT%2010.0;%20Win64;%20x64)%20AppleWebKit/537.36%20(KHTML,
%20like%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Error
 1pkpNfBQ39sYMnjjUQjmH2w1wdJnbHYTbag21o_3OfcQgPzdL2RSSQ== www.example.com http 675
 0.102 - - - Error HTTP/1.1 - - 25260 0.102 OriginDnsError text/html 507 - -
2019-12-13 22:36:26 SEA19-C1 900 192.0.2.200 GET d111111abcdef8.cloudfront.net / 502
 - Mozilla/5.0%20(Windows%20NT%2010.0;%20Win64;%20x64)%20AppleWebKit/537.36%20(KHTML,
%20like%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Error
 3AqrZGCnF_g0-5KOvfA7c9XLcf4YGvMFSeFdIetR1N_2y8jSis8Zxg== www.example.com http 735
 0.107 - - - Error HTTP/1.1 - - 3802 0.107 OriginDnsError text/html 507 - -
2019-12-13 22:37:02 SEA19-C2 900 192.0.2.200 GET d111111abcdef8.cloudfront.net / 502
 - curl/7.55.1 - - Error kBkDzGnceVtWHqSCqBUqtA_cEs2T3tFUBbnBNkB9El_uVRhHgcZfcw==
 www.example.com http 387 0.103 - - - Error HTTP/1.1 - - 12644 0.103 OriginDnsError
 text/html 507 - -
```

## 로그 분석

시간당 액세스 로그를 여러 개 받을 수 있기 때문에, 지정된 기간에 받는 모든 로그 파일을 한 파일로 결합하는 것이 좋습니다. 그러고 나면 이 기간의 데이터를 보다 정확하고 완전하게 분석할 수 있습니다.

액세스 로그를 분석하는 한 가지 방법은 Amazon Athena를 사용하는 것입니다. Athena는 CloudFront를 비롯하여 AWS 서비스에 대한 데이터를 분석할 수 있는 대화형 쿼리 서비스입니다. 자세한 내용은 Amazon Athena 사용 설명서의 Amazon CloudFront 로그 쿼리를 참조하세요.

또한 다음 AWS 블로그 게시물에서는 액세스 로그를 분석하는 몇 가지 방법을 설명합니다.

- Amazon CloudFront Request Logging(HTTP를 통해 전송되는 콘텐츠의 경우)
- Enhanced CloudFront Logs, Now With Query Strings

# 실시간 액세스 로그 사용

CloudFront 실시간 액세스 로그를 사용하여 배포에 대해 이루어진 요청에 대한 정보를 실시간으로 제공합니다(로그는 요청을 받은 후 몇 초 내에 전달됨). 실시간 액세스 로그를 사용하여 콘텐츠 전송 성능을 모니터링 및 분석하고 이에 기초해 조치를 취할 수 있습니다.

CloudFront 실시간 액세스 로그는 구성 가능합니다. 선택 항목:

- 실시간 로그에 대한 샘플링 비율(즉, 실시간 액세스 로그 레코드를 수신하려는 요청의 백분율).
- 로그 레코드에서 수신하려는 특정 필드
- 실시간 로그를 수신하려는 특정 캐시 동작(경로 패턴)

CloudFront 실시간 액세스 로그는 Amazon Kinesis Data Streams에서 선택한 데이터 스트림으로 전송됩니다. 자체 Kinesis 데이터 스트림 소비자를 생성하거나 Amazon Data Firehose를 사용하여 로그 데이터를 Amazon Simple Storage Service(S3), Amazon Redshift, Amazon OpenSearch Service(OpenSearch Service) 또는 서드 파티 로그 처리 서비스로 전송할 수 있습니다.

CloudFront는 Kinesis Data Streams 사용 요금 외에도, 실시간 액세스 로그에 대한 요금을 청구합니다. 비용에 대한 자세한 내용은 Amazon CloudFront 요금 및 Amazon Kinesis Data Streams 요금 단원을 참조하십시오.

> **Important**
>
> 모든 요청을 완전히 살펴보기 보다는 콘텐츠에 대한 요청 특성을 이해하는 데 로그를 사용하는 것이 좋습니다. CloudFront는 최대 효과에 기초하여 실시간 액세스 로그를 전송합니다. 요청에 따라서는 실제로 요청이 처리된 지 한참 후에 로그 레코드가 전송되거나 아예 전송되지 않을 수도 있습니다. 로그 항목이 실시간 액세스 로그에서 생략되는 경우 실시간 액세스 로그의 항목 수가 AWS 결제 및 사용 보고서에 표시되는 사용량과 일치하지 않습니다.

주제


- 실시간 액세스 로그 구성 생성 및 사용
- 실시간 액세스 로그 구성 이해
- Kinesis Data Streams 소비자 생성
- 실시간 액세스 로그 문제 해결

## 실시간 액세스 로그 구성 생성 및 사용

배포에 대한 요청 관련 정보를 실시간으로 확인하도록 실시간 액세스 로그 구성을 사용할 수 있습니다. 로그는 요청 수신 후 몇 초 이내에 전달됩니다. AWS Command Line Interface(AWS CLI) 또는 CloudFront API를 사용하여 CloudFront 콘솔에서 실시간 액세스 로그 구성을 생성할 수 있습니다.

실시간 액세스 로그 구성을 사용하려면 CloudFront 배포의 하나 이상의 캐시 동작에 해당 로그 구성을 연결합니다.

Console

실시간 액세스 로그 구성을 생성하려는 경우

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudfront/v4/home?#/logs의 CloudFront 콘솔에서 로그 페이지를 엽니다.
2. 실시간 로그 구성 탭을 선택합니다.
3. 구성 생성을 선택합니다.
4. 이름에 구성용 이름을 입력합니다.
5. 샘플링 비율은 로그 기록을 수신할 요청의 백분율을 입력합니다.
6. 필드의 경우 실시간 액세스 로그에서 수신할 필드를 선택합니다.
   - 로그에 모든 CMCD 필드를 포함하려면 CMCD all key를 선택합니다.
7. 엔드포인트의 경우 실시간 액세스 로그를 수신할 Kinesis 데이터 스트림을 하나 이상 선택합니다.

   > **Note**
   >
   > CloudFront 실시간 액세스 로그는 Kinesis 데이터 스트림에서 지정한 데이터 스트림으로 전달됩니다. 실시간 액세스 로그를 읽고 분석하기 위해 자신만의 Kinesis 데이터 스트림 소비자를 구축할 수 있습니다. Firehose를 사용하여 로그 데이터를 Amazon S3, Amazon Redshift, Amazon OpenSearch Service 또는 서드 파티 로그 처리 서비스에 전송할 수 있습니다.

8. IAM 역할의 경우, 새 서비스 역할 만들기를 선택하거나 기존 역할을 선택합니다. IAM 역할을 생성할 권한이 있어야 합니다.
9. (선택 사항) 배포의 경우, 실시간 액세스 로그 구성에 첨부할 CloudFront 배포 및 캐시 동작을 선택합니다.
10. 구성 생성을 선택합니다.

성공하면 콘솔에 방금 만든 실시간 액세스 로그 구성의 세부 정보가 표시됩니다.

자세한 내용은 실시간 액세스 로그 구성 이해 섹션을 참조하세요.

AWS CLI

AWS CLI를 사용하여 실시간 액세스 로그 구성을 생성하려면 aws cloudfront create-realtime-log-config 명령을 사용합니다. 각 개별 파라미터를 명령줄 입력으로 지정하는 대신 입력 파일을 사용하여 명령의 입력 파라미터를 제공할 수 있습니다.

실시간 액세스 로그 구성을 만드는 방법(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 `rtl-config.yaml` 명령에 대한 모든 입력 파라미터가 포함된 create-realtime-log-config이라는 파일을 만듭니다.

```
aws cloudfront create-realtime-log-config --generate-cli-skeleton yaml-input >
 rtl-config.yaml
```

2. 방금 생성한 `rtl-config.yaml`이라는 파일을 엽니다. 파일을 편집하여 원하는 실시간 액세스 로그 구성 설정을 지정한 다음 파일을 저장합니다. 다음 사항에 유의하세요.

   - `StreamType`의 유일한 유효 값은 `Kinesis`입니다.

   실시간 긴 구성 설정에 대한 자세한 내용은 실시간 액세스 로그 구성 이해 단원을 참조하십시오.

3. 다음 명령을 사용하여 `rtl-config.yaml` 파일의 입력 파라미터로 실시간 액세스 로그 구성을 만듭니다.

```
aws cloudfront create-realtime-log-config --cli-input-yaml file://rtl-
config.yaml
```

성공하면 명령 출력에 방금 만든 실시간 액세스 로그 구성의 세부 정보가 표시됩니다.

기존 배포에 실시간 액세스 로그 구성을 연결하는 방법(입력 파일이 있는 CLI)

1. 다음 명령을 사용하여 업데이트할 CloudFront 배포에 대한 배포 구성을 저장합니다. *`distribution_ID`*를 배포 ID로 바꿉니다.

```
aws cloudfront get-distribution-config --id distribution_ID --output yaml >
 dist-config.yaml
```

2. 방금 생성한 `dist-config.yaml`이라는 파일을 엽니다. 실시간 액세스 로그 구성을 사용하도록 업데이트하려는 각 캐시 동작을 다음과 같이 변경하여 파일을 편집합니다.

   - 캐시 동작에서 `RealtimeLogConfigArn`이라는 필드를 추가합니다. 필드 값에는 이 캐시 동작에 연결할 실시간 액세스 로그 구성의 ARN을 사용합니다.
   - `ETag` 필드의 이름을 `IfMatch`로 바꾸지만 필드 값은 변경하지 마세요.

   완료되면 파일을 저장합니다.

3. 실시간 액세스 로그 구성을 사용하도록 배포를 업데이트하려면 다음 명령을 사용합니다. *distribution_ID*를 배포 ID로 바꿉니다.

```
aws cloudfront update-distribution --id distribution_ID --cli-input-yaml file://
dist-config.yaml
```

성공하면 명령 출력에 방금 만든 배포의 세부 정보가 표시됩니다.

API

CloudFront API를 사용하여 실시간 액세스 로그 구성을 생성하려면 CreateRealTimeLogConfig API 작업을 사용합니다. 이 API 호출에서 지정하는 파라미터에 대한 자세한 내용은 실시간 액세스 로그 구성 이해 및 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

실시간 액세스 로그 구성을 생성한 후 다음 API 작업 중 하나를 사용하여 캐시 동작에 연결할 수 있습니다.

- 기존 배포의 캐시 동작에 연결하려면 UpdateDistribution을 사용합니다.
- 새 배포의 캐시 동작에 연결하려면 CreateDistribution을 사용합니다.

이 두 API 작업에 대해 캐시 동작 내의 `RealtimeLogConfigArn` 필드에 실시간 액세스 로그 구성의 ARN을 입력합니다. 이러한 API 호출에서 지정하는 다른 필드에 대한 자세한 내용은 모든 배포 설정 참조와 AWS SDK 또는 기타 API 클라이언트에 대한 API 참조 설명서를 참조하세요.

## 실시간 액세스 로그 구성 이해

CloudFront 실시간 액세스 로그를 사용하려면 먼저 실시간 액세스 로그 구성을 만듭니다. 실시간 액세스 로그 구성에는 수신할 로그 필드, 로그 레코드의 샘플링 비율 및 로그를 전달할 Kinesis 데이터 스트림에 대한 정보가 포함됩니다.

특히 실시간 액세스 로그 구성에는 다음 설정이 포함되어 있습니다.

목차


- 이름
- 샘플링 비율
- Fields
- 엔드포인트(Kinesis Data Streams)
- IAM 역할


### 이름

실시간 액세스 로그 구성을 식별하는 이름입니다.

### 샘플링 비율

샘플링 비율은 실시간 액세스 로그 레코드인 Kinesis Data Streams로 전송되는 뷰어 요청 비율을 결정하는 1~100의 정수입니다. 실시간 액세스 로그에 모든 뷰어 요청을 포함하려면 샘플링 비율을 100으로 지정합니다. 낮은 샘플링 비율을 선택하면 실시간 액세스 로그에서 요청 데이터의 대표 샘플을 수신하면서 비용을 줄일 수 있습니다.

### Fields

각 실시간 액세스 로그 레코드에 포함된 필드 목록입니다. 각 로그 레코드는 최대 40개의 필드를 포함할 수 있으며 사용 가능한 모든 필드를 수신하도록 선택하거나 성능 모니터링 및 분석에 필요한 필드만 수신하도록 선택할 수 있습니다.

다음 목록에는 각 필드 이름과 해당 필드의 정보에 대한 설명이 포함되어 있습니다. 필드는 Kinesis Data Streams에 전달된 로그 레코드에 나타나는 순서대로 나열됩니다.

필드 46-63은 미디어 플레이어 클라이언트가 각 요청과 함께 CDN에 보낼 수 있는 일반 미디어 클라이언트 데이터(CMCD)입니다. 이 데이터를 사용하여 미디어 유형(오디오, 비디오), 재생 속도, 스트리밍

길이와 같은 각 요청을 이해할 수 있습니다. 이러한 필드는 CloudFront로 전송된 경우에만 실시간 액세스 로그에 표시됩니다.

1. **`timestamp`**

   엣지 서버에서 요청에 대한 응답을 완료한 날짜 및 시간입니다.

2. **`c-ip`**

   요청을 한 최종 사용자의 IP 주소(예: `192.0.2.183` 또는 `2001:0db8:85a3::8a2e:0370:7334`)입니다. 최종 사용자가 HTTP 프록시 또는 로드 밸런서를 사용하여 요청을 전송하는 경우, 해당 필드 값은 프록시 또는 로드 밸런서의 IP 주소입니다. `x-forwarded-for` 필드도 참조하십시오.

3. **`s-ip`**

   요청을 처리한 CloudFront 서버의 IP 주소(예: `192.0.2.183` 또는 `2001:0db8:85a3::8a2e:0370:7334`).

4. **`time-to-first-byte`**

   요청을 수신한 후 응답의 첫 바이트를 쓸 때까지의 시간(초)이며 서버에서 측정합니다.

5. **`sc-status`**

   서버 응답의 HTTP 상태 코드(예: `200`)입니다.

6. **`sc-bytes`**

   요청에 대한 응답으로 서버가 최종 사용자에게 보낸 총 바이트 수(헤더 포함)입니다. WebSocket 및 gRPC 연결의 경우, 연결을 통해 서버에서 클라이언트로 전송된 총 바이트 수입니다.

7. **`cs-method`**

   최종 사용자로부터 수신된 HTTP 요청 메서드입니다.

8. **`cs-protocol`**

   최종 사용자 요청 프로토콜(`http`, `https`, `grpcs`, `ws` 또는 `wss`)입니다.

9. **`cs-host`**

   최종 사용자가 이 요청에 대한 `Host` 헤더에 포함한 값입니다. 객체 URL에 CloudFront 도메인 이름(예: d111111abcdef8.cloudfront.net)을 사용하는 경우, 이 필드에는 도메인 이름이 포함됩니다. 객체 URL(예: www.example.com)에 대체 도메인 이름(CNAME)을 사용하는 경우, 이 필드에 대체 도메인 이름이 포함됩니다.

10. **`cs-uri-stem`**

쿼리 문자열(있는 경우)을 포함하지만 도메인 이름은 포함하지 않는 전체 요청 URL입니다. 예를 들어 `/images/cat.jpg?mobile=true`입니다.

> **Note**
>
> 표준 로그에서 `cs-uri-stem` 값에 쿼리 문자열이 포함되지 않습니다.

11. **`cs-bytes`**

    최종 사용자가 요청에 포함한 데이터의 바이트 수(헤더 포함)입니다. WebSocket 및 gRPC 연결의 경우, 연결에서 클라이언트로부터 서버로 전송된 총 바이트 수입니다.

12. **`x-edge-location`**

    요청을 처리한 엣지 로케이션입니다. 각 엣지 로케이션은 3자 코드와 임의로 배정된 번호로 식별됩니다(예: DFW3). 3자 코드는 일반적으로 엣지 로케이션의 지리적 위치 부근 공항을 나타내는 국제 항공 수송 협회(IATA) 공항 코드에 상응합니다. (이러한 약어는 향후에 변경될 수 있습니다.)

13. **`x-edge-request-id`**

    요청을 고유하게 식별하는 불투명 문자열입니다. CloudFront는 `x-amz-cf-id` 응답 헤더에도 이 문자열을 전송합니다.

14. **`x-host-header`**

    CloudFront 배포의 도메인 이름(예: d111111abcdef8.cloudfront.net)입니다.

15. **`time-taken`**

    서버가 최종 사용자 요청을 수신한 시점부터 서버에서 출력 대기열에 대한 응답의 최종 바이트를 쓰는 시점 간의 시간 차이(1/1,000초 단위, 예: 0.082)로, 서버에서 측정됩니다. 최종 사용자 관점에서 보면 전체 응답을 가져오는 데 걸리는 총 시간은 네트워크 지연 시간과 TCP 버퍼링으로 인해 이 값보다 큽니다.

16. **`cs-protocol-version`**

    최종 사용자가 요청에서 지정한 HTTP 버전입니다. 가능한 값은 `HTTP/0.9`, `HTTP/1.0`, `HTTP/1.1`, `HTTP/2.0` 및 `HTTP/3.0`입니다.

17. **`c-ip-version`**

    요청의 IP 버전(IPv4 또는 IPv6)입니다.

18. **`cs-user-agent`**

요청의 User-Agent 헤더 값입니다. User-Agent 헤더는 요청을 제출한 디바이스 유형 및 브라우저 등의 요청 소스를 식별하고 검색 엔진에서 요청이 온 경우에는 어느 검색 엔진인지 식별합니다.

19. **cs-referer**

요청의 Referer 헤더 값입니다. 다음은 요청을 시작한 도메인 이름입니다. 일반적인 참조자에는 검색 엔진, 객체에 직접 연결하는 기타 웹 사이트 및 자체 웹 사이트가 포함됩니다.

20. **cs-cookie**

이름-값 페어 및 관련 속성을 포함한 요청의 Cookie 헤더입니다.

> **Note**
>
> 이 필드는 800바이트로 잘립니다.

21. **cs-uri-query**

요청 URL의 쿼리 문자열 부문(있는 경우)입니다.

22. **x-edge-response-result-type**

최종 사용자에게 응답을 반환하기 직전에 서버에서 응답을 분류한 방식입니다. x-edge-result-type 필드도 참조하십시오. 가능한 값은 다음과 같습니다.

- Hit - 서버가 캐시에서 최종 사용자에게 객체를 제공했습니다.
- RefreshHit - 서버가 캐시에서 객체를 찾았지만 객체가 만료되었기 때문에 서버가 오리진에 접속하여 캐시에 최신 버전의 객체가 있는지 확인했습니다.
- Miss - 캐시의 객체로는 요청을 충족할 수 없어서 서버가 요청을 오리진 서버로 전달했으며 결과가 최종 사용자에게 반환되었습니다.
- LimitExceeded - CloudFront 할당량(이전에는 제한이라고 함)이 초과되어 요청이 거부되었습니다.
- CapacityExceeded - 객체 제공을 요청하는 시점에 엣지 로케이션의 용량이 충분하지 않아 서버에서 503 오류가 반환되었습니다.
- Error - 일반적으로 이는 요청으로 인해 클라이언트 오류(sc-status 필드 값이 4xx 범위에 있음) 또는 서버 오류(sc-status 필드 값이 5xx 범위에 있음)가 발생했음을 의미합니다.

  x-edge-result-type 필드의 값이 Error이고 이 필드의 값이 Error가 아닌 경우 다운로드를 완료하기 전에 클라이언트 연결이 끊어졌습니다.
- Redirect - 서버는 배포 설정에 따라 최종 사용자를 HTTP에서 HTTPS로 리디렉션했습니다.

- `LambdaExecutionError` - 잘못된 연결, 함수 시간 초과, AWS 종속성 문제 또는 기타 일반 가용성 문제로 인해 배포와 연결된 Lambda@Edge 함수가 완료되지 않았습니다.

23 **`x-forwarded-for`**

최종 사용자가 HTTP 프록시 또는 로드 밸런서를 사용하여 요청을 전송하는 경우, `c-ip` 필드의 값은 프록시 또는 로드 밸런서의 IP 주소입니다. 이 경우 이 필드는 요청을 시작한 최종 사용자의 IP 주소입니다. 이 필드에는 여러 개의 쉼표로 구분된 IP 주소가 포함될 수 있습니다. 각 IP 주소는 IPv4 주소(예:`192.0.2.183`) 또는 IPv6 주소(예: `2001:0db8:85a3::8a2e:0370:7334`)일 수 있습니다.

24 **`ssl-protocol`**

요청에 HTTPS가 사용된 경우, 이 필드에는 최종 사용자 및 서버가 요청 및 응답 전송을 위해 협상한 SSL/TLS 프로토콜이 포함됩니다. 가능한 값 목록은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호에서 지원되는 SSL/TLS 프로토콜을 참조하세요.

25 **`ssl-cipher`**

요청에 HTTPS가 사용되는 경우, 이 필드에는 최종 사용자 및 서버가 요청 및 응답 암호화를 위해 협상한 SSL/TLS 암호가 포함됩니다. 가능한 값 목록은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호에서 지원되는 SSL/TLS 암호를 참조하세요.

26 **`x-edge-result-type`**

마지막 바이트가 서버를 떠난 후 서버가 응답을 분류한 방식입니다. 경우에 따라 서버가 응답을 전송할 준비가 된 시간과 가 응답 전송을 완료한 시간 사이에 결과 유형이 변경될 수 있습니다. `x-edge-response-result-type` 필드도 참조하십시오.

예를 들어 HTTP 스트리밍에서 서버가 캐시 스트림의 한 세그먼트를 찾는다고 가정해 보십시오. 이 시나리오에서 이 필드의 값은 보통 `Hit`입니다. 그러나 서버가 전체 세그먼트를 전달하기 전에 최종 사용자가 연결을 종료하면 최종 결과 유형(및 이 필드 값)은 `Error`가 됩니다.

콘텐츠가 캐시 가능하지 않고 오리진으로 직접 프록시 처리되므로 WebSocket 및 gRPC 연결에서 이 필드의 값이 `Miss`가 됩니다.

가능한 값은 다음과 같습니다.

- `Hit` - 서버가 캐시에서 최종 사용자에게 객체를 제공했습니다.
- `RefreshHit` - 서버가 캐시에서 객체를 찾았지만 객체가 만료되었기 때문에 서버가 오리진에 접속하여 캐시에 최신 버전의 객체가 있는지 확인했습니다.

- `Miss` - 캐시의 객체로는 요청을 충족할 수 없어서 서버가 요청을 오리진으로 전달했으며 결과가 최종 사용자에게 반환되었습니다.
- `LimitExceeded` - CloudFront 할당량(이전에는 제한이라고 함)이 초과되어 요청이 거부되었습니다.
- `CapacityExceeded` - 객체 제공을 요청하는 시점에 용량이 충분하지 않아 서버에서 HTTP 503 상태 코드가 반환되었습니다.
- `Error` - 일반적으로 이는 요청으로 인해 클라이언트 오류(`sc-status` 필드 값이 `4xx` 범위에 있음) 또는 서버 오류(`sc-status` 필드 값이 `5xx` 범위에 있음)가 발생했음을 의미합니다. `sc-status` 필드의 값이 `200`인 경우 또는 이 필드의 값이 `Error`이고 `x-edge-response-result-type` 필드의 값이 `Error`가 아닌 경우, HTTP 요청이 성공했지만 모든 바이트를 수신하기 전에 클라이언트의 연결이 끊어졌음을 의미합니다.
- `Redirect` - 서버는 배포 설정에 따라 최종 사용자를 HTTP에서 HTTPS로 리디렉션했습니다.
- `LambdaExecutionError` - 잘못된 연결, 함수 시간 초과, AWS 종속성 문제 또는 기타 일반 가용성 문제로 인해 배포와 연결된 Lambda@Edge 함수가 완료되지 않았습니다.

27. **`fle-encrypted-fields`**

서버가 암호화하여 오리진에 전달한 필드 레벨 암호화의 수입니다. CloudFront 서버는 데이터를 암호화하면서 처리된 요청을 오리진으로 스트리밍합니다. 따라서 `fle-status` 값이 오류인 경우에도 이 필드에 값이 있을 수 있습니다.

28. **`fle-status`**

배포에 대해 필드 레벨 암호화가 구성될 때, 요청 본문이 처리되었는지를 나타내는 코드가 이 필드에 포함됩니다. 서버가 성공적으로 요청 본문을 처리하고, 지정된 필드 값을 암호화하고, 요청을 오리진으로 전달할 때 이 필드 값은 `Processed`입니다. 이 경우 `x-edge-result-type` 값은 계속해서 클라이언트 측 또는 서버 측 오류를 나타낼 수 있습니다.

이 필드에 가능한 값은 다음과 같습니다.

- `ForwardedByContentType` - 어떠한 콘텐츠 유형도 구성되지 않았기 때문에 서버가 요청을 구문 분석 또는 암호화 없이 오리진에 전달했습니다.
- `ForwardedByQueryArgs` - 요청에 필드 레벨 암호화 구성에 없는 쿼리 인수가 포함되기 때문에 서버가 요청을 구문 분석 또는 암호화 없이 오리진에 전달했습니다.
- `ForwardedDueToNoProfile` - 필드 레벨 암호화 구성에 어떠한 프로필도 지정되지 않았기 때문에 서버가 요청을 구문 분석 또는 암호화 없이 오리진에 전달했습니다.
- `MalformedContentTypeClientError` - `Content-Type` 헤더 값이 유효하지 않은 형식이기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.

- `MalformedInputClientError` - 요청 본문이 유효하지 않은 형식이기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.
- `MalformedQueryArgsClientError` - 쿼리 인수가 비어 있거나 유효하지 않은 형식이기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.
- `RejectedByContentType` - 필드 레벨 암호화 구성에 어떠한 콘텐츠 유형도 지정되지 않았기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.
- `RejectedByQueryArgs` - 필드 레벨 암호화 구성에 어떠한 쿼리 인수도 지정되지 않았기 때문에 서버가 요청을 거부하고 최종 사용자에게 HTTP 400 상태 코드를 반환했습니다.
- `ServerError` - 오리진 서버가 오류를 반환했습니다.

요청이 필드 레벨 암호화 할당량(이전에는 제한이라고 함) 을 초과하면 이 필드에 다음 오류 코드 중 하나가 포함되고 서버는 HTTP 상태 코드 400을 최종 사용자에게 반환합니다. 필드 레벨 암호화의 현재 할당량에 대한 목록은 필드 레벨 암호화에 대한 할당량 단원을 참조하세요.

- `FieldLengthLimitClientError` - 암호화되도록 구성된 필드가 허용되는 최대 길이를 초과했습니다.
- `FieldNumberLimitClientError` - 배포가 암호화하도록 구성된 요청에 허용된 필드 수보다 많은 필드가 있습니다.
- `RequestLengthLimitClientError` - 필드 레벨 암호화 구성 시 요청 본문의 길이가 허용되는 최대 길이를 초과했습니다.

29 **`sc-content-type`**

응답의 HTTP `Content-Type` 헤더 값입니다.

30 **`sc-content-len`**

응답의 HTTP `Content-Length` 헤더 값입니다.

31 **`sc-range-start`**

응답에 HTTP `Content-Range` 헤더가 포함되어 있으면 이 필드에 범위 시작 값이 포함됩니다.

32 **`sc-range-end`**

응답에 HTTP `Content-Range` 헤더가 포함되어 있으면 이 필드에 범위 끝 값이 포함됩니다.

33 **`c-port`**

최종 사용자가 보낸 요청의 포트 번호입니다.

34 **`x-edge-detailed-result-type`**

이 필드에는 `x-edge-result-type` 필드와 동일한 값이 포함됩니다. 단, 다음과 같은 경우는 예외입니다.

- 객체가 Origin Shield 계층에서 최종 사용자에게 제공된 경우 이 필드에 `OriginShieldHit`가 포함합니다.
- 객체가 CloudFront 캐시에 없고 오리진 요청 Lambda @Edge 함수에 의해 응답이 생성된 경우 이 필드에는 `MissGeneratedResponse`가 포함됩니다.
- `x-edge-result-type` 필드의 값이 `Error`인 경우 이 필드에는 오류에 대한 추가 정보를 제공하는 다음 값 중 하나가 포함됩니다.
  - `AbortedOrigin` - 서버에 오리진 관련된 문제가 발생했습니다.
  - `ClientCommError` - 서버와 최종 사용자 간의 통신 문제로 인해 최종 사용자에게 대한 응답이 중단되었습니다.
  - `ClientGeoBlocked` - 최종 사용자의 지리적 위치에서 요청을 거부하도록 배포가 구성됩니다.
  - `ClientHungUpRequest` - 요청을 보내는 동안 최종 사용자가 중간에 중지했습니다.
  - `Error` - 오류 유형이 기타 모든 카테고리에 적합하지 않아 오류가 발생했습니다. 이러한 오류 유형은 서버가 캐시에서 오류 응답을 제공할 때 발생할 수 있습니다.
  - `InvalidRequest` - 서버가 최종 사용자로부터 잘못된 요청을 받았습니다.
  - `InvalidRequestBlocked` - 요청한 리소스에 대한 액세스가 차단됩니다.
  - `InvalidRequestCertificate` - 배포가 HTTPS 연결이 설정된 SSL/TLS 인증서와 일치하지 않습니다.
  - `InvalidRequestHeader` - 요청에 잘못된 헤더가 포함되어 있습니다.
  - `InvalidRequestMethod` - 사용되었던 HTTP 요청 메서드를 처리하도록 배포가 구성되지 않았습니다. 이러한 문제는 배포에서 캐시 가능한 요청만 지원하는 경우 발생할 수 있습니다.
  - `OriginCommError` - 오리진에 연결하거나 오리진에서 데이터를 읽는 동안 요청 시간이 초과되었습니다.
  - `OriginConnectError` - 서버를 오리진에 연결할 수 없습니다.
  - `OriginContentRangeLengthError` - 오리진 응답의 `Content-Length` 헤더가 `Content-Range` 헤더의 길이와 일치하지 않습니다.
  - `OriginDnsError` - 서버에서 오리진의 도메인 이름을 확인할 수 없습니다.
  - `OriginError` - 오리진에서 잘못된 응답을 반환했습니다.
  - `OriginHeaderTooBigError` - 오리진에서 반환한 헤더가 너무 커서 엣지 서버가 처리할 수 없습니다.
  - `OriginInvalidResponseError` - 오리진에서 잘못된 응답을 반환했습니다.

- `OriginReadError` - 오리진에서 서버를 읽을 수 없습니다.
- `OriginWriteError` - 오리진에 서버를 쓸 수 없습니다.
- `OriginZeroSizeObjectError` - 오리진에서 전송된 크기가 0인 객체로 인해 오류가 발생했습니다.
- `SlowReaderOriginError` - 최종 사용자가 오리진 오류를 일으킨 메시지를 느리게 읽었습니다.

35. **`c-country`**

최종 사용자의 IP 주소로 결정되는 최종 사용자의 지리적 위치를 나타내는 국가 코드입니다. 국가 코드 목록은 ISO 3166-1 alpha-2 단원을 참조하세요.

36. **`cs-accept-encoding`**

최종 사용자 요청의 `Accept-Encoding` 헤더 값입니다.

37. **`cs-accept`**

최종 사용자 요청의 `Accept` 헤더 값입니다.

38. **`cache-behavior-path-pattern`**

최종 사용자 요청과 일치하는 캐시 동작을 식별하는 경로 패턴입니다.

39. **`cs-headers`**

최종 사용자 요청의 HTTP 헤더(이름 및 값)입니다.

> **Note**
> 이 필드는 800바이트로 잘립니다.

40. **`cs-header-names`**

최종 사용자 요청의 HTTP 헤더 이름(값 아님)입니다.

> **Note**
> 이 필드는 800바이트로 잘립니다.

41. **`cs-headers-count`**

최종 사용자 요청의 HTTP 헤더 수입니다.

42. **primary-distribution-id**

    연속 배포가 활성화된 경우 이 ID는 현재 배포의 기본 배포를 식별하는 데 사용됩니다.

43. **primary-distribution-dns-name**

    연속 배포가 활성화된 경우 이 값에는 현재 CloudFront 배포와 연결된 기본 도메인 이름(예: d111111abcdef8.cloudfront.net)이 표시됩니다.

44. **origin-fbl**

    CloudFront와 오리진 간의 첫 바이트 지연 시간(초)입니다.

45. **origin-lbl**

    CloudFront와 오리진 간의 마지막 바이트 지연 시간(초)입니다.

46. **asn**

    최종 사용자의 Autonomous System Number(ASN)입니다.

47.

    > **실시간 액세스 로그의 CMCD 필드**
    >
    > 이러한 필드에 대한 자세한 내용은 CTA 사양 웹 애플리케이션 비디오 에코시스템 - 일반 미디어 클라이언트 데이터 CTA-5004 문서를 참조합니다.

48. **cmcd-encoded-bitrate**

    요청된 오디오 또는 비디오 객체의 인코딩된 비트레이트입니다.

49. **cmcd-buffer-length**

    요청된 미디어 객체의 버퍼 길이입니다.

50. **cmcd-buffer-starvation**

    이전 요청과 객체 요청 사이의 특정 시점에서 버퍼가 부족했는지 여부. 이로 인해 플레이어가 리버퍼링 상태에 빠져 비디오 또는 오디오 재생이 중단될 수 있습니다.

51. **cmcd-content-id**

    현재 콘텐츠를 식별하는 고유한 문자열입니다.

52. **cmcd-object-duration**

    요청된 객체의 재생 시간(밀리초).

53. **cmcd-deadline**

    버퍼 언더런 상태나 기타 재생 문제를 방지하기 위해 이 객체의 첫 번째 샘플을 사용할 수 있어야 하는 요청 시점으로부터의 기한.

54. **cmcd-measured-throughput**

    클라이언트가 측정한 클라이언트와 서버 간 처리량.

55. **cmcd-next-object-request**

    다음 요청된 객체의 상대 경로.

56. **cmcd-next-range-request**

    다음 요청이 부분 객체 요청인 경우 이 문자열은 요청될 바이트 범위를 나타냅니다.

57. **cmcd-object-type**

    요청 중인 현재 객체의 미디어 유형.

58. **cmcd-playback-rate**

    실시간이면 1, 배속이면 2, 재생되지 않는 경우 0.

59. **cmcd-requested-maximum-throughput**

    요청된 최대 처리량 중 클라이언트가 자산 전달에 충분하다고 간주하는 최대 처리량.

60. **cmcd-streaming-format**

    현재 요청을 정의하는 스트리밍 형식.

61. **cmcd-session-id**

    현재 재생 세션을 식별하는 GUID.

62. **cmcd-stream-type**

    세그먼트 가용성을 식별하는 토큰. v= 모든 세그먼트를 사용할 수 있습니다. l= 시간이 지나면 세그먼트를 사용할 수 있게 됩니다.

63. **cmcd-startup**

    버퍼가 비어있는 이벤트 이후 스타트 업, 검색 또는 복구 중에 객체가 긴급하게 필요한 경우 키는 값 없이 포함됩니다.

64. **cmcd-top-bitrate**

클라이언트가 재생할 수 있는 최고 비트레이트 렌디션입니다.

65 **`cmcd-version`**

정의된 키 이름 및 값을 해석하는 데 사용되는 이 사양의 버전입니다. 이 키를 생략하면 클라이언트와 서버는 값을 버전 1에 정의된 것으로 해석해야 합니다.

66 **`r-host`**

이 필드는 오리진 요청에 대해 전송되며 객체를 제공하는 데 사용되는 오리진 서버의 도메인을 나타냅니다. 오류가 발생하는 경우 이 필드를 사용하여 마지막으로 시도한 오리진을 찾을 수 있습니다 (예: `cd8jhdejh6a.mediapackagev2.us-east-1.amazonaws.com`).

67 **`sr-reason`**

이 필드를 통해 오리진을 선택한 이유를 알 수 있습니다. 기본 오리진에 대한 요청이 성공하면 비어 있습니다.

오리진 장애 조치가 발생하면 필드에 `Failover:403` 또는 `Failover:502`와 같이 장애 조치로 이어진 HTTP 오류 코드가 포함됩니다. 오리진 장애 조치의 경우 재시도된 요청도 실패하고 사용자 지정 오류 페이지를 구성하지 않으면 `r-status`에서 두 번째 오리진의 응답을 나타냅니다. 그러나 오리진 장애 조치와 함께 사용자 지정 오류 페이지를 구성하면 요청이 실패하고 사용자 지정 오류 페이지가 대신 반환된 경우 두 번째 오리진의 응답이 포함됩니다.

오리진 장애 조치가 발생하지 않지만 미디어 품질 인식 복원력(MQAR) 오리진 옵션이 발생하는 경우 `MediaQuality`로 로그됩니다. 자세한 내용은 미디어 품질 인식 복원력 섹션을 참조하세요.

68 **`x-edge-mqcs`**

이 필드는 CloudFront가 MediaPackage v2의 CMSD 응답 헤더에서 검색한 미디어 세그먼트의 미디어 품질 신뢰도 점수(MQCS)(범위: 0~100)를 나타냅니다. 이 필드는 MQAR 지원 오리진 그룹이 있는 캐시 동작과 일치하는 요청에 사용할 수 있습니다. CloudFront는 오리진 요청 외에도 캐시에서 제공되는 미디어 세그먼트에 대해 이 필드를 로그합니다. 자세한 내용은 미디어 품질 인식 복원력 섹션을 참조하세요.

69 **`distribution-tenant-id`**

배포 테넌트의 ID입니다.

70 **`connection-id`**

TLS 연결의 고유 식별자입니다.

이 필드에 대한 정보를 가져오려면 먼저 배포에 대해 mTLS를 활성화해야 합니다. 자세한 내용은 CloudFront에 대한 상호 TLS 인증(뷰어 mTLS) 섹션을 참조하세요.

엔드포인트(Kinesis Data Streams)

엔드포인트에는 실시간 로그를 전송하려는 Kinesis Data Streams에 대한 정보가 포함되어 있습니다. 데이터 스트림의 Amazon 리소스 이름(ARN)을 제공합니다.

Kinesis Data Streams 생성에 대한 자세한 내용은 Amazon Kinesis Data Streams 개발자 안내서의 다음 주제를 참조하세요.

- 스트림 생성 및 관리
- 를 사용하여 기본 Kinesis Data Streams 작업 수행AWS CLI
- 스트림 생성(AWS SDK for Java 사용)

데이터 스트림을 생성할 때 샤드 수를 지정해야 합니다. 다음 정보를 사용하여 필요한 샤드 수를 예측할 수 있습니다.

Kinesis 데이터 스트림의 샤드 수를 추정하는 방법

1. CloudFront 배포에서 수신하는 초당 요청 수를 계산(또는 예측)합니다.

   CloudFront 사용 보고서(CloudFront 콘솔)와 CloudFront 지표(CloudFront 및 Amazon CloudWatch 콘솔)를 사용하여 초당 요청 수를 계산할 수 있습니다.
2. 단일 실시간 액세스 로그 레코드의 일반적인 크기를 확인합니다.

   일반적으로 단일 로그 레코드는 약 500바이트입니다. 사용 가능한 모든 필드를 포함하는 큰 레코드 하나는 보통 약 1KB입니다.

   로그 레코드 크기가 확실하지 않으면 샘플링 비율을 낮게 설정하여(예: 1%) 실시간 로그를 활성화한 다음 Kinesis Data Streams의 모니터링 데이터를 사용해 레코드의 평균 크기를 계산할 수 있습니다(수신되는 총 바이트 수를 총 레코드 수로 나눔).
3. Amazon Kinesis Data Streams 요금 페이지의 AWS Pricing Calculator 아래에서 지금 사용자 지정 예상 비용 생성을 선택합니다.

   - 계산기에 초당 요청(레코드) 수를 입력합니다.
   - 단일 로그 레코드의 평균 레코드 크기를 입력합니다.

- 그런 다음, 계산 표시를 선택합니다.

요금 계산기는 필요한 샤드 수와 예상 비용을 보여줍니다.

IAM 역할

Kinesis 데이터 스트림에 실시간 액세스 로그를 전달하는 CloudFront 권한을 부여하는 AWS Identity and Access Management(IAM) 역할입니다.

CloudFront 콘솔을 사용하여 실시간 액세스 로그 구성을 만들 때 새 서비스 역할 만들기를 선택하여 콘솔에서 IAM 역할을 만들 수 있습니다.

AWS CloudFormation 또는 CloudFront API(AWS CLI 또는 SDK)를 사용하여 실시간 액세스 로그 구성을 생성하는 경우 직접 IAM 역할을 생성하고 역할 ARN을 입력해야 합니다. IAM 역할을 직접 만들려면 다음 정책을 사용하세요.

IAM 역할 신뢰 정책

다음 IAM 역할 신뢰 정책을 사용하려면 *111122223333*을 해당 AWS 계정 번호로 바꿉니다. CloudFront가 AWS 계정의 배포를 대신하여 이 역할만 맡을 수 있기 때문에 이 정책의 `Condition` 요소는 혼동된 대리자 문제를 방지하는 데 도움이 됩니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudfront.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                }
            }
        }
    ]
```

```
}
```

암호화되지 않은 데이터 스트림에 대한 IAM 역할 권한 정책

다음 정책을 사용하려면 *arn:aws:kinesis:us-east-2:123456789012:stream/StreamName*을 Kinesis 데이터 스트림의 ARN으로 바꿉니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStreamSummary",
                "kinesis:DescribeStream",
                "kinesis:PutRecord",
                "kinesis:PutRecords"
            ],
            "Resource": [
                "arn:aws:kinesis:us-east-2:123456789012:stream/StreamName"
            ]
        }
    ]
}
```

암호화된 데이터 스트림에 대한 IAM 역할 권한 정책

다음 정책을 사용하려면 *arn:aws:kinesis:us-east-2:123456789012:stream/StreamName*을 Kinesis 데이터 스트림의 ARN으로 교체하고 *arn:aws:kms:us-east-2:123456789012:key/e58a3d0b-fe4f-4047-a495-ae03cc73d486*을 본인 AWS KMS key의 ARN으로 교체합니다.

JSON

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

```
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStreamSummary",
                "kinesis:DescribeStream",
                "kinesis:PutRecord",
                "kinesis:PutRecords"
            ],
            "Resource": [
                "arn:aws:kinesis:us-east-2:123456789012:stream/StreamName"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": [
                "arn:aws:kms:us-east-2:123456789012:key/e58a3d0b-fe4f-4047-a495-
ae03cc73d486"
            ]
        }
    ]
}
```

## Kinesis Data Streams 소비자 생성

실시간 액세스 로그를 읽고 분석하려면 Kinesis Data Streams 소비자를 생성하거나 사용합니다. CloudFront 실시간 로그의 소비자를 생성할 때는 모든 실시간 액세스 로그 레코드의 필드가 항상 Fields 섹션에 나열된 것과 동일한 순서로 전달된다는 것을 알고 있어야 합니다. 이 고정된 순서를 적용하도록 소비자를 빌드해야 합니다.

예를 들어 세 개의 필드(`time-to-first-byte`, `sc-status` 및 `c-country`)만 포함하는 실시간 액세스 로그 구성을 가정해 보겠습니다. 이 시나리오에서 마지막 필드인 `c-country`는 항상 모든 로그 레코드의 필드 번호 3입니다. 그러나 나중에 실시간 액세스 로그 구성에 필드를 추가하면 각 레코드 필드의 배치가 변경될 수 있습니다.

예를 들어 `sc-bytes` 및 `time-taken` 필드를 실시간 액세스 로그 구성에 추가하는 경우, 이 필드는 Fields 단원에 표시된 순서에 따라 각 로그 레코드에 삽입됩니다. 다섯 필드의 결과 순서는 `time-to-`

`first-byte`, `sc-status`, `sc-bytes`, `time-taken` 및 `c-country`입니다. 원래 `c-country` 필드는 필드 번호 3이었지만 이제는 필드 번호 5입니다. 실시간 액세스 로그 구성에 필드를 추가하는 경우, 소비자 애플리케이션이 로그 레코드에서 위치를 변경하는 필드를 처리할 수 있는지 확인합니다.

### 실시간 액세스 로그 문제 해결

실시간 액세스 로그 구성을 생성한 후에는 모든 레코드(또는 일부 레코드)가 Kinesis Data Streams에 전달되지 않을 수 있습니다. 이 경우 먼저 CloudFront 배포가 최종 사용자 요청을 수신하고 있는지 확인해야 합니다. 이 경우 다음 설정을 확인하여 문제 해결을 계속할 수 있습니다.

IAM 역할 권한

Kinesis 데이터 스트림에 실시간 액세스 로그 레코드를 전달하기 위해 CloudFront에서는 실시간 액세스 로그 구성에 IAM 역할을 사용합니다. 역할 신뢰 정책과 역할 권한 정책이 IAM 역할에 표시된 정책과 일치하는지 확인합니다.

Kinesis Data Streams 조절

CloudFront가 스트림이 처리할 수 있는 것보다 더 빠르게 Kinesis 데이터 스트림에 실시간 액세스 로그 레코드를 쓰는 경우, Kinesis Data Streams에서 CloudFront의 요청을 조절할 수 있습니다. 이 경우 Kinesis 데이터 스트림의 샤드 수를 늘릴 수 있습니다. 각 샤드는 초당 최대 1,000개의 레코드 쓰기를 지원할 수 있으며 최대 데이터 쓰기는 초당 1MB입니다.

## 엣지 함수 로그

Amazon CloudWatch Logs를 사용하여 엣지 함수(Lambda@Edge 및 CloudFront Functions 모두)에 대한 로그를 가져올 수 있습니다. CloudWatch 콘솔 또는 CloudWatch Logs API를 사용하여 로그에 액세스할 수 있습니다.

> **Important**
>
> 모든 요청을 완전히 살펴보기 보다는 콘텐츠에 대한 요청 특성을 이해하는 데 로그를 사용하는 것이 좋습니다. CloudFront는 최대 효과에 기초하여 엣지 함수 로그를 전송합니다. 요청에 따라서는 실제로 요청이 처리된 지 한참 후에 로그 레코드가 전송되거나 아예 전송되지 않을 수도 있습니다. 로그 항목이 엣지 함수 로그에서 생략되는 경우 엣지 함수 로그의 항목 수가 AWS 결제 및 사용 보고서에 표시되는 사용량과 일치하지 않습니다.

주제

- Lambda@Edge 로그
- CloudFront Functions 로그

## Lambda@Edge 로그

Lambda@Edge는 함수 로그를 CloudWatch Logs에 자동으로 전송하여 함수가 취소되는 AWS 리전에서 로그 스트림을 만듭니다. AWS Lambda에서 함수를 생성하거나 수정할 때 기본 CloudWatch 로그 그룹 이름을 사용하거나 이름을 사용자 지정할 수 있습니다.

- 기본 로그 그룹 이름은 `/aws/lambda/<FunctionName>`입니다. 여기서 `<FunctionName>`은 함수를 생성할 때 지정한 이름입니다. CloudWatch로 로그를 전송할 때 Lambda@Edge는 함수 이름에 `us-east-1` 접두사를 자동으로 추가하여 로그 그룹 이름이 `/aws/lambda/us-east-1.<FunctionName>`이 되도록 합니다. 이 접두사는 함수가 생성된 AWS 리전에 해당합니다. 이 접두사는 함수가 간접적으로 호출되는 다른 리전에서도 로그 그룹 이름의 일부로 유지됩니다.
- `/MyLogGroup`과 같은 사용자 지정 로그 그룹 이름을 지정하면 Lambda@Edge가 리전 접두사를 추가하지 않습니다. 로그 그룹 이름은 함수가 간접적으로 호출되는 다른 모든 리전에서 동일하게 유지됩니다.

> **Note**
>
> 사용자 지정 로그 그룹을 생성하고 기본 `/aws/lambda/<FunctionName>`과 동일한 이름을 지정하면Lambda@Edge가 함수 이름에 `us-east-1` 접두사를 추가합니다.

Lambda@Edge 함수는 로그 그룹 이름 사용자 지정 외에도 JSON 및 일반 텍스트 로그 형식과 로그 수준 필터링을 지원합니다. 자세한 내용은 AWS Lambda 개발자 안내서의 Lambda 함수에 대한 고급 로깅 제어 구성을 참조하세요.

> **Note**
>
> Lambda@Edge는 요청 볼륨과 로그 크기에 따라 로그를 제한합니다.

Lambda@Edge 함수 로그 파일을 확인하려면 올바른 리전의 CloudWatch Logs 파일을 검토해야 합니다. Lambda@Edge 함수가 실행되는 리전을 보려면 CloudFront 콘솔의 함수에 대한 지표 그래프를 확인하세요. 지표는 각 리전별로 표시됩니다. 동일한 페이지에서 리전을 선택하고 해당 리전의 로그 파일을 확인하여 문제를 조사할 수 있습니다.

Lambda@Edge 함수와 함께 CloudWatch Logs를 사용하는 방법을 자세히 알아보려면 다음 주제를 참조하시기 바랍니다.

- CloudFront 콘솔의 모니터링(Monitoring) 섹션에서 그래프를 보는 방법에 대한 자세한 내용은 the section called "Amazon CloudWatch를 사용한 CloudFront 지표 모니터링" 단원을 참조하세요.
- CloudWatch Logs로 데이터를 전송하는 데 필요한 권한에 대한 자세한 내용은 the section called "IAM 권한 및 역할 설정" 섹션을 참조하세요.
- 로깅을 Lambda@Edge 함수에 추가하는 방법에 대한 자세한 내용은 AWS Lambda 개발자 안내서의 Node.js에 AWS Lambda 함수 로깅 또는 Python에서 AWS Lambda 함수 로깅 섹션을 참조하세요.
- CloudWatch Logs 할당량(이전에는 제한이라고 함)에 대한 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 CloudWatch Logs 할당량 단원을 참조하세요.

## CloudFront Functions 로그

CloudFront 함수의 코드에 `console.log()` 명령문이 포함되어 있는 경우 CloudFront Functions는 자동으로 이러한 로그 라인을 CloudWatch Logs로 전송합니다. `console.log()` 명령문이 없으면 아무것도 CloudWatch Logs로 전송되지 않습니다.

CloudFront 함수는 함수를 실행한 엣지 로케이션에 관계없이 항상 미국 동부(버지니아 북부) 리전(`us-east-1`)에서 로그 스트림을 생성합니다. 로그 스트림 이름은 `YYYY/M/D/`*`UUID`* 형식입니다.

로그 그룹 이름은 다음 형식을 사용합니다.

- 캐시 동작 수준의 CloudFront Functions경우 형식은 `/aws/cloudfront/function/`*`<FunctionName>`*입니다.
- 배포 수준의 CloudFront Functions(연결 함수)의 경우 형식은 `/aws/cloudfront/connection-function/`*`<FunctionName>`*입니다.

*`<FunctionName>`*는 함수를 생성할 때 함수에 부여한 이름입니다.

### Example뷰어 요청

다음은 CloudWatch Logs로 전송된 로그 메시지 예제입니다. 각 줄은 CloudFront 요청을 고유하게 식별하는 ID로 시작됩니다. 메시지는 CloudFront 배포 ID가 포함된 `START` 행으로 시작하고 하나의 `END` 행으로 끝납니다. `START` 및 `END` 줄 사이에는 함수의 `console.log()` 명령문에 의해 생성된 로그 라인이 있습니다.

```
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV5Oyq-vmAtH8HADpjhw== START DistributionID:
 E3E5D42GADAXZZ
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV5Oyq-vmAtH8HADpjhw== Example function log output
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV5Oyq-vmAtH8HADpjhw== END
```

Example연결 요청

다음은 CloudWatch Logs로 전송된 로그 메시지 예제입니다. 각 줄은 CloudFront 요청을 고유하게 식별하는 ID로 시작됩니다. 메시지는 CloudFront 배포 ID가 포함된 `START` 행으로 시작하고 하나의 `END` 행으로 끝납니다. `START` 및 `END` 줄 사이에는 함수의 `console.log()` 명령문에 의해 생성된 로그 라인이 있습니다.

```
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV5Oyq-vmAtH8HADpjhw== START DistributionID:
 E3E5D42GADA123
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV5Oyq-vmAtH8HADpjhw== 1.2.3.4
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV5Oyq-vmAtH8HADpjhw== END
```

> **Note**
>
> CloudFront Functions는 프로덕션 요청 및 응답에 대한 응답으로 실행되는 `LIVE` 스테이지의 함수에 대해서만 로그를 CloudWatch로 전송합니다. 함수를 테스트할 때 CloudFront는 로그를 CloudWatch로 보내지 않습니다. 테스트 출력에는 오류, 컴퓨팅 사용률 및 함수 로그(`console.log()` 명령문)에 대한 정보가 포함되지만 이 정보는 CloudWatch로 전송되지 않습니다.

CloudFront 함수는 AWS Identity and Access Management(IAM) 서비스 연결 역할을 사용하여 계정의 CloudWatch Logs로 로그를 전송합니다. 서비스 연결 역할은 AWS 서비스에 직접 연결된 IAM 역할입니다. 서비스 연결 역할은 해당 서비스에서 사전 정의하며 서비스에서 다른 AWS 서비스를 호출하기 위해 필요한 모든 권한을 포함합니다. CloudFront Functions에서는 AWSServiceRoleForCloudFrontLogger 서비스 연결 역할을 사용합니다. 이 역할에 대한 자세한 내용은 the section called “Lambda@Edge의 서비스 연결 역할” (Lambda@Edge에서 동일한 서비스 연결 역할 사용) 단원을 참조하세요.

유효성 검사 오류 또는 실행 오류로 인해 함수가 실패하면 정보가 표준 로그 및 실시간 액세스 로그에 기록됩니다. 오류에 대한 구체적인 내용은 `x-edge-result-type`, `x-edge-response-result-type` 및 `x-edge-detailed-result-type` 필드를 참조하시기 바랍니다.

# AWS CloudTrail을 사용하여 Amazon CloudFront API 직접 호출 로깅

CloudFront는 사용자, 역할 또는 AWS 서비스가 수행한 작업의 기록을 제공하는 서비스인 AWS CloudTrail와 통합되어 있습니다. CloudTrail은 Client VPN에 대한 모든 API 호출을CloudFront 이벤트로 캡처합니다. 캡처되는 호출에는 CloudFront 콘솔로부터의 호출과 CloudFront API 작업에 대한 코드 호출이 포함됩니다. CloudTrail에서 수집한 정보를 사용하여 CloudFront에 수행된 요청, 요청이 수행된 IP 주소, 요청이 수행된 시간, 추가 세부 정보를 확인할 수 있습니다.

모든 이벤트 또는 로그 항목에는 요청을 생성했던 사용자에 대한 정보가 포함됩니다. 자격 증명을 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트 사용자로 했는지 사용자 보안 인증으로 했는지 여부.
- IAM Identity Center 사용자를 대신하여 요청이 이루어졌는지 여부입니다.
- 역할 또는 페더레이션 사용자에 대한 임시 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 다른 AWS 서비스에서 요청했는지 여부.

계정을 생성할 때 AWS 계정에서 CloudTrail이 활성 상태이며, CloudTrail 이벤트 기록에 자동으로 액세스할 수 있습니다. CloudTrail 이벤트 기록은 지난 90일 간 AWS 리전의 관리 이벤트에 대해 보기, 검색 및 다운로드가 가능하고, 수정이 불가능한 레코드를 제공합니다. 자세한 설명은 AWS CloudTrail 사용 설명서의 CloudTrail 이벤트 기록 작업을 참조하세요. Event history(이벤트 기록) 보기는 CloudTrail 요금이 부과되지 않습니다.

지난 90일 동안 AWS 계정에서 진행 중인 이벤트 기록을 보려면 추적 또는 CloudTrail Lake 이벤트 데이터 스토어를 생성합니다.

CloudTrail 추적

CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. AWS Management Console을 사용하여 만든 추적은 모두 다중 리전입니다. AWS CLI를 사용하여 단일 리전 또는 다중 리전 추적을 생성할 수 있습니다. 계정의 모든 AWS 리전에서 활동을 캡처하므로, 다중 리전 추적 생성이 권장됩니다. 단일 리전 추적을 생성하는 경우 추적의 AWS 리전에 로깅된 이벤트만 볼 수 있습니다. 추적에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 Creating a trail for your AWS 계정 및 Creating a trail for an organization을 참조하세요.

CloudTrail에서 추적을 생성하여 진행 중인 관리 이벤트의 사본 하나를 Amazon S3 버킷으로 무료로 전송할 수는 있지만, Amazon S3 스토리지 요금이 부과됩니다. CloudTrail 요금에 대한 자세한 내용은 AWS CloudTrail 요금을 참조하세요. Amazon S3 요금에 대한 자세한 내용은 Amazon S3 요금을 참조하세요.

### CloudTrail Lake 이벤트 데이터 스토어

CloudTrail Lake를 사용하면 이벤트에 대해 SQL 기반 쿼리를 실행할 수 있습니다. CloudTrail Lake는 행 기반 JSON 형식의 기존 이벤트를 Apache ORC 형식으로 변환합니다. ORC는 빠른 데이터 검색에 최적화된 열 기반 스토리지 형식입니다. 이벤트는 이벤트 데이터 스토어로 집계되며, 이벤트 데이터 스토어는 고급 이벤트 선택기를 적용하여 선택한 기준을 기반으로 하는 변경 불가능한 이벤트 컬렉션입니다. 이벤트 데이터 스토어에 적용하는 선택기는 어떤 이벤트가 지속되고 쿼리할 수 있는지 제어합니다. CloudTrail Lake에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 Working with AWS CloudTrail Lake를 참조하세요.

CloudTrail Lake 이벤트 데이터 스토어 및 쿼리에는 비용이 발생합니다. 이벤트 데이터 스토어를 생성할 때 이벤트 데이터 스토어에 사용할 요금 옵션을 선택합니다. 요금 옵션에 따라 이벤트 모으기 및 저장 비용과 이벤트 데이터 스토어의 기본 및 최대 보존 기간이 결정됩니다. CloudTrail 요금에 대한 자세한 내용은 AWS CloudTrail 요금을 참조하세요.

> **Note**
>
> CloudFront는 글로벌 서비스입니다. CloudTrail은 미국 동부(버지니아 북부) 리전에서 CloudFront에 대한 이벤트를 녹합니다. 자세한 내용을 알아보려면 AWS CloudTrail 사용 설명서의 글로벌 서비스 이벤트를 참조하세요.
> AWS Security Token Service을 사용하여 임시 보안 자격 증명을 사용하는 경우 `us-west-2`와 같은 리전 및 엔드포인트에 대한 호출은 CloudTrail에서 해당 지역으로 기록됩니다.
> CloudFront 엔드포인트에 대한 자세한 내용은 AWS 일반 참조에서 CloudFront 엔드포인트 및 할당량을 참조합니다.

## CloudTrail의 CloudFront 데이터 이벤트

데이터 이벤트는 리소스 기반 또는 리소스에서 수행된 리소스 작업에 대한 정보를 제공합니다(예: CloudFront 배포판에서 읽기 또는 쓰기). 이를 데이터 영역 작업이라고도 합니다. 데이터 이벤트가 대량 활동인 경우도 있습니다. 기본적으로 CloudTrail은 데이터 이벤트를 로깅하지 않습니다. CloudTrail 이벤트 기록은 데이터 이벤트를 기록하지 않습니다.

데이터 이벤트에는 추가 요금이 적용됩니다. CloudTrail 요금에 대한 자세한 내용은 AWS CloudTrail 요금을 참조하세요.

CloudTrail 콘솔, AWS CLI 또는 CloudTrail API 작업을 사용하여 CloudFront 리소스 유형에 대한 데이터 이벤트에 로깅할 수 있습니다. 데이터 이벤트를 로깅하는 방법에 대한 자세한 내용은 AWS

CloudTrail 사용 설명서의 Logging data events with the AWS Management Console 및 Logging data events with the AWS Command Line Interface를 참조하세요.

다음 표에는 데이터 이벤트를 로깅할 수 있는 CloudFront 리소스 유형이 나열되어 있습니다. 데이터 이벤트 유형(콘솔) 열에는 CloudTrail 콘솔의 데이터 이벤트 유형 목록에서 선택할 값이 표시됩니다. resources.type 값 열에는 AWS CLI 또는 CloudTrail API를 사용하여 고급 이벤트 선택기를 구성할 때 지정하는 `resources.type` 값이 표시됩니다. CloudTrail에 로깅되는 데이터 API 열에는 리소스 유형에 대해 CloudTrail에 로깅된 API 호출이 표시됩니다.

| 데이터 이벤트 유형(콘솔) | resources.type 값 | CloudTrail에 로깅되는 데이터 API |
| --- | --- | --- |
| CloudFront KeyValueStore | `AWS::CloudFront::KeyValueStore` | • DeleteKeys<br>• DescribeKeyValueStore<br>• GetKey<br>• ListKeys<br>• PutKeys<br>• UpdateKeys |

`eventName`, `readOnly` 및 `resources.ARN` 필드를 필터링하여 중요한 이벤트만 로깅하도록 고급 이벤트 선택기를 구성할 수 있습니다. 이러한 필드에 대한 자세한 내용은 AWS CloudTrail API 참조의 AdvancedFieldSelector 섹션을 참조하세요.

## CloudTrail의 CloudFront 관리 이벤트

관리 이벤트는 AWS 계정의 리소스에 대해 수행되는 관리 작업에 대한 정보를 제공합니다. 이를 컨트롤 플레인 작업이라고도 합니다. 기본적으로 CloudTrail은 관리 이벤트를 로깅합니다.

Amazon CloudFront는 CloudFront 컨트롤 플레인 작업을 관리 이벤트로 로깅합니다. CloudFront가 CloudTrail에 로그하는 Amazon CloudFront 컨트롤 플레인 작업의 목록은 Amazon CloudFront API 참조를 살펴봅니다.

## CloudFront 이벤트 예제

이벤트는 모든 소스로부터의 단일 요청을 나타내며 요청된 API 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 들어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 직접 호출의 주문 스택 추적이 아니므로 이벤트가 특정 순서로 표시되지 않습니다.

목차


- 예시: UpdateDistribution
- 예시: UpdateKeys


## 예시: UpdateDistribution

다음 예는 UpdateDistribution 작업을 보여주는 CloudTrail 이벤트를 보여줍니다.

CloudFront API에 대한 호출의 경우 `eventSource`가 `cloudfront.amazonaws.com`입니다.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:role-session-name",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/role-session-name",
        "accountId": "111122223333",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-02-02T19:23:50Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-02-02T19:26:01Z",
    "eventSource": "cloudfront.amazonaws.com",
    "eventName": "UpdateDistribution",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "52.94.133.137",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/121.0.0.0 Safari/537.36",
    "requestParameters": {
        "distributionConfig": {
```

```
            "defaultRootObject": "",
            "aliases": {
                "quantity": 3,
                "items": [
                    "alejandro_rosalez.awsps.myinstance.com",
                    "cross-testing.alejandro_rosalez.awsps.myinstance.com",
                    "*.alejandro_rosalez.awsps.myinstance.com"
                ]
            },
            "cacheBehaviors": {
                "quantity": 0,
                "items": []
            },
            "httpVersion": "http2and3",
            "originGroups": {
                "quantity": 0,
                "items": []
            },
            "viewerCertificate": {
                "minimumProtocolVersion": "TLSv1.2_2021",
                "cloudFrontDefaultCertificate": false,
                "aCMCertificateArn": "arn:aws:acm:us-east-1:111122223333:certificate/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                "sSLSupportMethod": "sni-only"
            },
            "webACLId": "arn:aws:wafv2:us-east-1:111122223333:global/webacl/testing-
acl/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "customErrorResponses": {
                "quantity": 0,
                "items": []
            },
            "logging": {
                "includeCookies": false,
                "prefix": "",
                "enabled": false,
                "bucket": ""
            },
            "priceClass": "PriceClass_All",
            "restrictions": {
                "geoRestriction": {
                    "restrictionType": "none",
                    "quantity": 0,
                    "items": []
                }
```

```
            },
            "isIPV6Enabled": true,
            "callerReference": "1578329170895",
            "continuousDeploymentPolicyId": "",
            "enabled": true,
            "defaultCacheBehavior": {
                "targetOriginId": "d111111abcdef8",
                "minTTL": 0,
                "compress": false,
                "maxTTL": 31536000,
                "functionAssociations": {
                    "quantity": 0,
                    "items": []
                },
                "trustedKeyGroups": {
                    "quantity": 0,
                    "items": [],
                    "enabled": false
                },
                "smoothStreaming": false,
                "fieldLevelEncryptionId": "",
                "defaultTTL": 86400,
                "lambdaFunctionAssociations": {
                    "quantity": 0,
                    "items": []
                },
                "viewerProtocolPolicy": "redirect-to-https",
                "forwardedValues": {
                    "cookies": {"forward": "none"},
                    "queryStringCacheKeys": {
                        "quantity": 0,
                        "items": []
                    },
                    "queryString": false,
                    "headers": {
                        "quantity": 1,
                        "items": ["*"]
                    }
                },
                "trustedSigners": {
                    "items": [],
                    "enabled": false,
                    "quantity": 0
                },
```

```
                "allowedMethods": {
                    "quantity": 2,
                    "items": [
                        "HEAD",
                        "GET"
                    ],
                    "cachedMethods": {
                        "quantity": 2,
                        "items": [
                            "HEAD",
                            "GET"
                        ]
                    }
                }
            },
            "staging": false,
            "origins": {
                "quantity": 1,
                "items": [
                    {
                        "originPath": "",
                        "connectionTimeout": 10,
                        "customOriginConfig": {
                            "originReadTimeout": 30,
                            "hTTPSPort": 443,
                            "originProtocolPolicy": "https-only",
                            "originKeepaliveTimeout": 5,
                            "hTTPPort": 80,
                            "originSslProtocols": {
                                "quantity": 3,
                                "items": [
                                    "TLSv1",
                                    "TLSv1.1",
                                    "TLSv1.2"
                                ]
                            }
                        },
                        "id": "d111111abcdef8",
                        "domainName": "d111111abcdef8.cloudfront.net",
                        "connectionAttempts": 3,
                        "customHeaders": {
                            "quantity": 0,
                            "items": []
                        },
```

```
                        "originShield": {"enabled": false},
                        "originAccessControlId": ""
                    }
                ]
            },
            "comment": "HIDDEN_DUE_TO_SECURITY_REASONS"
        },
        "id": "EDFDVBD6EXAMPLE",
        "ifMatch": "E1RTLUR9YES76O"
    },
    "responseElements": {
        "distribution": {
            "activeTrustedSigners": {
                "quantity": 0,
                "enabled": false
            },
            "id": "EDFDVBD6EXAMPLE",
            "domainName": "d111111abcdef8.cloudfront.net",
            "distributionConfig": {
                "defaultRootObject": "",
                "aliases": {
                    "quantity": 3,
                    "items": [
                        "alejandro_rosalez.awsps.myinstance.com",
                        "cross-testing.alejandro_rosalez.awsps.myinstance.com",
                        "*.alejandro_rosalez.awsps.myinstance.com"
                    ]
                },
                "cacheBehaviors": {"quantity": 0},
                "httpVersion": "http2and3",
                "originGroups": {"quantity": 0},
                "viewerCertificate": {
                    "minimumProtocolVersion": "TLSv1.2_2021",
                    "cloudFrontDefaultCertificate": false,
                    "aCMCertificateArn": "arn:aws:acm:us-
east-1:111122223333:certificate/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                    "sSLSupportMethod": "sni-only",
                    "certificateSource": "acm",
                    "certificate": "arn:aws:acm:us-east-1:111122223333:certificate/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
                },
                "webACLId": "arn:aws:wafv2:us-east-1:111122223333:global/webacl/
testing-acl/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
                "customErrorResponses": {"quantity": 0},
```

```
                "logging": {
                    "includeCookies": false,
                    "prefix": "",
                    "enabled": false,
                    "bucket": ""
                },
                "priceClass": "PriceClass_All",
                "restrictions": {
                    "geoRestriction": {
                        "restrictionType": "none",
                        "quantity": 0
                    }
                },
                "isIPV6Enabled": true,
                "callerReference": "1578329170895",
                "continuousDeploymentPolicyId": "",
                "enabled": true,
                "defaultCacheBehavior": {
                    "targetOriginId": "d111111abcdef8",
                    "minTTL": 0,
                    "compress": false,
                    "maxTTL": 31536000,
                    "functionAssociations": {"quantity": 0},
                    "trustedKeyGroups": {
                        "quantity": 0,
                        "enabled": false
                    },
                    "smoothStreaming": false,
                    "fieldLevelEncryptionId": "",
                    "defaultTTL": 86400,
                    "lambdaFunctionAssociations": {"quantity": 0},
                    "viewerProtocolPolicy": "redirect-to-https",
                    "forwardedValues": {
                        "cookies": {"forward": "none"},
                        "queryStringCacheKeys": {"quantity": 0},
                        "queryString": false,
                        "headers": {
                            "quantity": 1,
                            "items": ["*"]
                        }
                    },
                    "trustedSigners": {
                        "enabled": false,
                        "quantity": 0
```

```
                    },
                    "allowedMethods": {
                        "quantity": 2,
                        "items": [
                            "HEAD",
                            "GET"
                        ],
                        "cachedMethods": {
                            "quantity": 2,
                            "items": [
                                "HEAD",
                                "GET"
                            ]
                        }
                    }
                },
                "staging": false,
                "origins": {
                    "quantity": 1,
                    "items": [
                        {
                            "originPath": "",
                            "connectionTimeout": 10,
                            "customOriginConfig": {
                                "originReadTimeout": 30,
                                "hTTPSPort": 443,
                                "originProtocolPolicy": "https-only",
                                "originKeepaliveTimeout": 5,
                                "hTTPPort": 80,
                                "originSslProtocols": {
                                    "quantity": 3,
                                    "items": [
                                        "TLSv1",
                                        "TLSv1.1",
                                        "TLSv1.2"
                                    ]
                                }
                            },
                            "id": "d111111abcdef8",
                            "domainName": "d111111abcdef8.cloudfront.net",
                            "connectionAttempts": 3,
                            "customHeaders": {"quantity": 0},
                            "originShield": {"enabled": false},
                            "originAccessControlId": ""
```

```
                        }
                    ]
                },
                "comment": "HIDDEN_DUE_TO_SECURITY_REASONS"
            },
            "aliasICPRecordals": [
                {
                    "cNAME": "alejandro_rosalez.awsps.myinstance.com",
                    "iCPRecordalStatus": "APPROVED"
                },
                {
                    "cNAME": "cross-testing.alejandro_rosalez.awsps.myinstance.com",
                    "iCPRecordalStatus": "APPROVED"
                },
                {
                    "cNAME": "*.alejandro_rosalez.awsps.myinstance.com",
                    "iCPRecordalStatus": "APPROVED"
                }
            ],
            "aRN": "arn:aws:cloudfront::111122223333:distribution/EDFDVBD6EXAMPLE",
            "status": "InProgress",
            "lastModifiedTime": "Feb 2, 2024 7:26:01 PM",
            "activeTrustedKeyGroups": {
                "enabled": false,
                "quantity": 0
            },
            "inProgressInvalidationBatches": 0
        },
        "eTag": "E1YHBLAB2BJY1G"
    },
    "requestID": "4e6b66f9-d548-11e3-a8a9-73e33example",
    "eventID": "5ab02562-0fc5-43d0-b7b6-90293example",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "apiVersion": "2020_05_31",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "cloudfront.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
```

```
}
```

예시: UpdateKeys

다음 예제에서는 UpdateKeys 작업을 보여주는 CloudTrail 이벤트를 보여줍니다.

CloudFront KeyValueStore API 호출의 경우, `eventSource`는 `cloudfront.amazonaws.com`이 아닌 `edgekeyvaluestore.amazonaws.com`입니다.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:role-session-name",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/role-session-name",
        "accountId": "111122223333",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2023-11-01T23:41:14Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-11-01T23:41:28Z",
    "eventSource": "edgekeyvaluestore.amazonaws.com",
    "eventName": "UpdateKeys",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "3.235.183.252",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/121.0.0.0 Safari/537.36,
    "requestParameters": {
        "kvsARN": "arn:aws:cloudfront::111122223333:key-value-store/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
        "ifMatch": "KV3O6B1CX531EBP",
        "deletes": [
```

```
            {"key": "key1"}
        ]
    },
    "responseElements": {
        "itemCount": 0,
        "totalSizeInBytes": 0,
        "eTag": "KVDC9VEVZ71ZGO"
    },
    "requestID": "5ccf104c-acce-4ea1-b7fc-73e33example",
    "eventID": "a0b1b5c7-906c-439d-9925-90293example",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::CloudFront::KeyValueStore",
            "ARN": "arn:aws:cloudfront::111122223333:key-value-store/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "111122223333.cloudfront-kvs.global.api.aws"
    }
}
```

CloudTrail 레코드 콘텐츠에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail record contents를 참조하세요.

# AWS Config를 사용하여 구성 변경 추적

AWS 리소스 구성을 기록하고 평가하려면 배포의 구성을 자세히 볼 수 있는 AWS Config를 사용할 수 있습니다. 여기에는 리소스가 서로 어떻게 관련되는지, 과거에 어떻게 구성되었는지가 포함되어 시간에 따른 변화를 검토할 수 있습니다.

또한 AWS Config를 사용하여 CloudFront 배포 설정 변경에 대한 구성 변경 사항을 기록할 수 있습니다. 배포 상태, 가격 등급, 오리진, 지리적 제한 설정, Lambda@Edge 구성에 대한 변경 사항을 캡처할 수 있습니다.

> **Note**
>
> AWS Config에서는 CloudFront 스트리밍 배포에 대한 키-값 태그를 기록하지 않습니다.

목차


- CloudFront로 AWS Config 설정
- CloudFront 구성 기록 보기
- AWS Config 규칙을 사용하여 CloudFront 구성 평가


## CloudFront로 AWS Config 설정

AWS Config를 설정할 때는 지원되는 모든 AWS 리소스를 기록하는 방법을 선택하거나, CloudFront의 변경 사항만 기록하는 등 지정된 리소스만 기록하도록 지정할 수 있습니다. 지원되는 CloudFront 리소스의 목록을 보려면 AWS Config 개발자 안내서에서 지원되는 리소스 유형 주제의 Amazon CloudFront 섹션을 참조하세요.

> **참고**
>
> - CloudFront 배포에 대한 구성 변경 사항을 추적하려면 미국 동부(버지니아 북부) AWS 리전의 CloudFront 콘솔에 로그인해야 합니다.
> - AWS Config에서 리소스 레코딩이 지연될 수 있습니다. AWS Config는 리소스를 검색한 후에만 리소스를 기록합니다.

Console

CloudFront로 AWS Config를 설정하려면

1. AWS Management Console 에 로그인한 다음 AWS Config 콘솔을 엽니다.
2. [Get Started Now]를 선택합니다.
3. 설정 페이지의 기록할 리소스 유형에서 AWS에 기록할 AWS Config 리소스 유형을 지정합니다. CloudFront 변경 사항만 기록하고 싶은 경우에는 특정 유형을 선택한 다음, CloudFront에서 변경 사항을 추적하려는 배포 또는 스트리밍 배포를 선택합니다.

   추적할 배포를 추가 또는 변경하려면 초기 설정을 완료한 후에 왼쪽의 설정을 선택합니다.

4. AWS Config에 필요한 추가 옵션(예: 알림 설정, 구성 정보의 위치 지정, 리소스 유형 평가를 위한 규칙 추가 등)을 지정합니다.

자세한 내용은 AWS Config 개발자 안내서의 콘솔을 통해 AWS Config 설정을 참조하세요.

AWS CLI

AWS CLI에 AWS Config를 설정하려면 AWS Config 개발자 안내서의 AWS CLI를 사용하여 AWS Config 설정을 참조하세요.

AWS Config API

AWS Config API를 사용하여 CloudFront에 AWS Config를 설정하려면 AWS Config API 참조의 StartConfigurationRecorder API 작업을 참조하세요.

## CloudFront 구성 기록 보기

AWS Config가 배포에 구성 변경 사항을 기록하기 시작한 후에는 CloudFront에 대해 구성한 모든 배포의 구성 기록을 가져올 수 있습니다.

다음 방법을 사용하여 구성 내역을 볼 수 있습니다.

Console

기록된 리소스 각각에 대해 구성 세부 정보의 내역을 제공하는 타임라인 페이지를 볼 수 있습니다. 이 페이지를 보려면 전용 호스트 페이지의 Config 타임라인 열에서 회색 아이콘을 선택합니다.

보다 자세한 내용은 AWS Config 개발자 안내서의 AWS Config 콘솔에서 구성 세부 사항 보기를 참조하세요.

AWS CLI

모든 배포 목록을 얻으려면 다음 예시에서와 같이 list-discovered-resources 명령을 사용합니다.

```
aws configservice list-discovered-resources --resource-type
 AWS::CloudFront::Distribution
```

특정 기간 동안의 배포에 대한 구성 세부 정보를 가져오려면 get-resource-config-history 명령을 실행합니다.

보다 자세한 내용은 AWS Config 개발자 안내서의 CLI를 사용하여 구성 세부 정보 보기 섹션을 참조하세요.

AWS Config API

모든 배포 목록을 얻으려면 ListDiscoveredResources API 작업을 사용합니다.

특정 기간 동안의 배포에 대한 구성 세부 정보를 가져오려면 GetResourceConfigHistory API 작업을 사용합니다. 자세한 내용은 AWS Config API 참조를 참조하십시오.

## AWS Config 규칙을 사용하여 CloudFront 구성 평가

AWS Config 규칙을 사용하여 원하는 구성을 기준으로 구성을 평가할 수 있습니다. 예를 들어 AWS Config 규칙을 사용하면 CloudFront 리소스가 일반적인 보안 모범 사례를 준수하는지 여부를 평가할 수 있습니다. 최종 사용자 정책 HTTPS, SNI 활성화, OAC 활성화, 오리진 장애 조치 활성화, AWS WAF WebACL과 같은 관리형 규칙 또는 구성 변경 시 트리거할 AWS Shield Advanced 리소스 정책을 선택할 수 있습니다.

관리형 규칙은 선택한 빈도로 정기적으로 평가를 실행할 수 있습니다. AWS Firewall Manager는 자동 알림 및 문제 해결을 위해 AWS Config에 의존합니다. 자세한 내용은 AWS Config 개발자 안내서의 AWS Config 규칙으로 리소스 평가 및 AWS Config 관리형 규칙 목록을 참조하세요.

# Amazon CloudFront의 보안

AWS은 클라우드 보안을 가장 중요하게 생각합니다. AWS 고객으로서 여러분은 가장 높은 보안 요구 사항을 충족하기 위해 설계된 데이터 센터 및 네트워크 아키텍처의 혜택을 받게 됩니다.

보안은 AWS과 사용자의 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드 내 보안 및 클라우드의 보안으로 설명합니다.

- 클라우드의 보안 - AWS는 AWS Cloud에서 AWS 서비스를 실행하는 인프라를 보호합니다. AWS는 또한 안전하게 사용할 수 있는 서비스를 제공합니다. 서드 파티 감사원은 정기적으로 AWS규정 준수 프로그램의 일환으로 보안 효과를 테스트하고 검증합니다. Amazon CloudFront에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 프로그램 제공 범위 내 AWS 서비스를 참조하세요.
- 클라우드 내 보안 – 사용자의 책임은 사용자가 사용하는 AWS 서비스에 의해 결정됩니다. 또한 데이터의 민감도, 조직의 요건 및 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 CloudFront를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 주제에서는 보안 및 규정 준수 목적에 맞게 CloudFront를 구성하는 방법을 보여줍니다. 또한 CloudFront 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법도 알아봅니다.

주제


- Amazon CloudFront의 데이터 보호
- Amazon CloudFront용 Identity and Access Management
- Amazon CloudFront의 로깅 및 모니터링
- Amazon CloudFront에 대한 규정 준수 확인
- Amazon CloudFront의 복원성
- Amazon CloudFront의 인프라 보안


## Amazon CloudFront의 데이터 보호

AWS 공동 책임 모델은 Amazon CloudFront의 데이터 보호에 적용됩니다. 이 모델에서 설명하는 것처럼 AWS는 모든 AWS 클라우드를 실행하는 글로벌 인프라를 보호할 책임이 있습니다. 사용자는 이 인프라에 호스팅되는 콘텐츠에 대한 통제 권한을 유지할 책임이 있습니다. 사용하는 AWS 서비스의 보안 구성과 관리 태스크에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 관한 자세한 내용은

데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS보안 블로그의 AWS공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터를 보호하려면 AWS 계정자격 증명을 보호하고 AWS IAM Identity Center또는 AWS Identity and Access Management(IAM)를 통해 개별 사용자 계정을 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용합니다.
- SSL/TLS를 사용하여 AWS리소스와 통신하세요. TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- AWS CloudTrail으로 API 및 사용자 활동 로깅을 설정하세요. AWS 활동 캡처에 CloudTrail 추적을 사용하는 방법에 대한 자세한 내용은 AWS CloudTrail사용 설명서의 CloudTrail 추적 작업을 참조하세요.
- AWS 암호화 솔루션을 AWS 서비스내의 모든 기본 보안 컨트롤과 함께 사용하세요.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용합니다.
- 명령줄 인터페이스 또는 API를 통해 AWS에 액세스할 때 FIPS 140-3 검증된 암호화 모듈이 필요한 경우, FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 연방 정보 처리 표준(FIPS) 140-3을 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 형식 텍스트 필드에 입력하지 않는 것이 좋습니다. 여기에는 CloudFront 또는 기타 AWS 서비스 서비스에서 콘솔, API, AWS CLI 또는 AWS SDK를 사용하여 작업하는 경우가 포함됩니다. 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 자격 증명을 URL에 포함해서는 안 됩니다.

Amazon CloudFront는 제공하는 콘텐츠의 보안을 유지하는 데 사용할 수 있는 몇 가지 옵션을 제공합니다.

- HTTPS 연결 구성
- 전송 중 특정 데이터에 대한 추가 보안을 제공하도록 필드 수준 암호화를 구성합니다.
- 특정 사람이나 특정 영역의 사람만 볼 수 있도록 콘텐츠에 대한 액세스 제한

다음 주제에서는 이러한 옵션에 대해 자세히 설명합니다.

주제


- 전송 중 데이터 암호화
- 저장된 암호화
- 콘텐츠에 대한 액세스 제한


## 전송 중 데이터 암호화

전송 중 데이터를 암호화하려면 최종 사용자가 HTTPS를 사용하여 파일을 요청하도록 Amazon CloudFront를 구성합니다. 이렇게 하면 CloudFront가 최종 사용자와 통신할 때 연결이 암호화됩니다. 또한 CloudFront가 오리진의 파일을 받을 때 HTTPS를 사용하도록 구성할 수 있습니다. 이렇게 하면 CloudFront가 오리진과 통신할 때 연결이 암호화됩니다.

자세한 내용은 CloudFront에서 HTTPS 사용 단원을 참조하세요.

필드 레벨 암호화는 HTTPS와 함께 추가 보안 레이어를 추가하여 시스템 처리 전체에서 특정 데이터를 보호하고 특정 애플리케이션만 이를 볼 수 있도록 합니다. CloudFront에서 필드 수준 암호화를 구성하여 사용자가 제출한 중요한 정보를 웹 서버에 안전하게 업로드할 수 있습니다. 클라이언트가 제공하는 중요한 정보는 사용자에게 더 가까운 엣지에서 암호화됩니다. 전체 애플리케이션 스택에서 암호화된 상태로 유지되므로 데이터가 필요하고 이를 해독할 자격 증명이 있는 애플리케이션만 이 작업을 수행할 수 있습니다.

자세한 내용은 필드 수준 암호화를 사용하여 민감한 데이터 보호 단원을 참조하세요.

CloudFront API 엔드포인트, `cloudfront.amazonaws.com` 및 `cloudfront-fips.amazonaws.com`는 HTTPS 트래픽만 허용합니다. 즉, CloudFront API를 사용하여 정보를 주고받을 때 배포 구성, 캐시 정책 및 오리진 요청 정책, 키 그룹 및 퍼블릭 키, CloudFront 함수의 함수 코드 등의 데이터가 전송 중에 항상 암호화됩니다. 또한 CloudFront API 엔드포인트로 전송된 모든 요청은 AWS 자격 증명으로 서명되고 AWS CloudTrail에 로그인됩니다.

CloudFront 함수의 함수 코드 및 구성은 엣지 로케이션 포인트(POP)로 복사되거나 CloudFront에서 사용하는 다른 스토리지 위치 간에 복사될 때 전송 중에 항상 암호화됩니다.

## 저장된 암호화

CloudFront Functions의 함수 코드 및 구성은 항상 엣지 로케이션 POP와 CloudFront에서 사용하는 다른 스토리지 위치에 암호화된 형식으로 저장됩니다.

# 콘텐츠에 대한 액세스 제한

인터넷을 통해 콘텐츠를 배포하는 많은 회사에서는 일부 사용자를 대상으로 하는 콘텐츠, 문서, 비즈니스 데이터 또는 미디어 스트림에 대한 액세스를 제한하고자 합니다. Amazon CloudFront를 사용하여 이 콘텐츠를 안전하게 제공하려면 다음 중 하나 이상을 수행하면 됩니다.

서명된 URL 또는 쿠키 사용

서명된 URL 또는 서명된 쿠키를 사용하여 CloudFront를 통해 이 프라이빗 콘텐츠를 제공함으로써 선택된 사용자(예: 유료 사용자)를 대상으로 하는 콘텐츠에 대한 액세스를 제한할 수 있습니다. 자세한 내용은 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공 단원을 참조하세요.

Amazon S3 버킷의 콘텐츠에 대한 액세스 제한

예를 들어 CloudFront 서명된 URL 또는 서명된 쿠키를 사용해 콘텐츠에 대한 액세스를 제한하는 경우 사용자가 파일에 대한 직접 URL을 사용하여 파일을 보는 것도 원치 않을 것입니다. 대신, 보호가 작동하도록 CloudFront URL을 사용해야만 파일에 액세스하도록 허용할 수 있습니다.

Amazon S3 버킷을 CloudFront 배포의 오리진으로 사용하는 경우 오리진 액세스 제어(OAC)를 설정하여 S3 버킷에 대한 액세스를 제한할 수 있습니다. 자세한 내용은 the section called “Amazon S3 오리진에 대한 액세스 제한” 섹션을 참조하세요.

Application Load Balancer가 제공하는 콘텐츠에 대한 액세스 제한

Elastic Load Balancing에서 Application Load Balancer와 함께 CloudFront를 오리진으로 사용하면 사용자가 Application Load Balancer에 직접 액세스할 수 없도록 CloudFront를 구성할 수 있습니다. 이렇게 하면 사용자가 CloudFront를 통해서만 Application Load Balancer에 액세스할 수 있으므로 CloudFront를 사용할 때의 이점을 실현할 수 있습니다. 자세한 내용은 Application Load Balancer에 대한 액세스 제한 단원을 참조하세요.

AWS WAF 웹 ACL 사용

웹 애플리케이션 방화벽 서비스인 AWS WAF를 사용하여 웹 액세스 제어 목록(웹 ACL)을 만들어 콘텐츠에 대한 액세스를 제한할 수 있습니다. 요청이 시작되는 IP 주소 또는 쿼리 문자열의 값과 같이 사용자가 지정하는 조건에 따라 CloudFront는 요청된 콘텐츠 또는 HTTP 403 상태 코드(사용 권한 없음)로 요청에 응답합니다. 자세한 내용은 AWS WAF 보호 사용 단원을 참조하세요.

지리적 제한 사용

지리적 차단이라고도 하는 지리적 제한을 사용하여 특정 지리적 위치에 있는 사용자가 CloudFront 배포를 통해 제공한 콘텐츠에 액세스하는 것을 차단할 수 있습니다. 지리적 제한을 구성할 때 몇 가지 옵션을 선택할 수 있습니다. 자세한 내용은 콘텐츠의 지리적 배포 제한 단원을 참조하세요.

# Amazon CloudFront용 Identity and Access Management

AWS Identity and Access Management(IAM)는 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어할 수 있도록 지원하는 AWS 서비스입니다. IAM 관리자는 어떤 사용자가 CloudFront 리소스를 사용할 수 있는 '인증'(로그인) 및 '권한'(권한 있음)을 받을 수 있는지 제어합니다. IAM은 추가 비용 없이 사용할 수 있는 AWS 서비스입니다.

주제


- 고객
- ID를 통한 인증
- 정책을 사용하여 액세스 관리
- Amazon CloudFront와 함께 IAM을 사용하는 방법
- Amazon CloudFront에 대한 자격 증명 기반 정책 예시
- AWSAmazon CloudFront용 관리형 정책
- CloudFront의 서비스 연결 역할 사용
- Amazon CloudFront 자격 증명 및 액세스 문제 해결


## 고객

AWS Identity and Access Management(IAM)를 사용하는 방법은 역할에 따라 다릅니다.

- 서비스 사용자 - 기능에 액세스할 수 없는 경우 관리자에게 권한 요청(참조Amazon CloudFront 자격 증명 및 액세스 문제 해결)
- 서비스 관리자 - 사용자 액세스 결정 및 권한 요청 제출(Amazon CloudFront와 함께 IAM을 사용하는 방법 참조)
- IAM 관리자 - 액세스를 관리하기 위한 정책 작성(Amazon CloudFront에 대한 자격 증명 기반 정책 예시 참조)

## ID를 통한 인증

인증은 ID 자격 증명을 사용하여 AWS에 로그인하는 방식입니다. AWS 계정 루트 사용자이나 IAM 사용자로, 또는 IAM 역할을 수임하여 인증(에 로그인)받아야 합니다.

AWS IAM Identity Center(IAM Identity Center), Single Sign-On 인증 또는 Google/Facebook 자격 증명과 같은 자격 증명 소스의 자격 증명을 사용하여 페더레이션 ID로 로그인할 수 있습니다. 로그인하는 방법에 대한 자세한 내용은 AWS Sign-In사용 설명서의 AWS 계정에 로그인하는 방법 섹션을 참조하세요.

프로그래밍 방식 액세스를 위해 AWS는 요청에 암호화 방식으로 서명할 수 있는 SDK 및 CLI를 제공합니다. 자세한 내용은 IAM 사용자 설명서의 API 요청용 AWS Signature Version 4 섹션을 참조하세요.

## AWS 계정 루트 사용자

AWS 계정을 생성하는 경우에는 모든 AWS 서비스 서비스와 리소스에 대한 완전한 액세스 권한이 있는 AWS 계정 루트 사용자라는 단일 로그인 ID로 시작합니다. 일상적인 태스크에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 자격 증명이 필요한 작업은 IAM 사용자 설명서의 루트 사용자 자격 증명이 필요한 작업 섹션을 참조하세요.

## 페더레이션 ID

가장 좋은 방법은 인간 사용자가 ID 공급자와의 페더레이션을 사용하여 임시 자격 증명으로 AWS 서비스에 액세스하도록 하는 것입니다.

페더레이션 ID는 엔터프라이즈 사용자 디렉터리, 웹 ID 제공업체 또는 Directory Service의 사용자로, ID의 자격 증명을 사용하여 AWS 서비스에 액세스합니다. 페더레이션 ID는 임시 자격 증명을 제공하는 역할을 수임합니다.

중앙 집중식 액세스 관리를 위해 AWS IAM Identity Center를 추천합니다. 자세한 정보는 AWS IAM Identity Center사용 설명서의 What is IAM Identity Center?를 참조하세요.

## IAM 사용자 및 그룹

IAM 사용자는 단일 개인 또는 애플리케이션에 대한 특정 권한을 가진 ID입니다. 장기 자격 증명이 있는 IAM 사용자 대신 임시 자격 증명을 사용하는 것이 좋습니다. 자세한 내용은 IAM 사용자 설명서에서 임시 자격 증명을 사용하여 AWS에 액세스하려면 인간 사용자가 ID 제공업체와의 페더레이션을 사용하도록 요구를 참조하세요.

IAM 그룹은 IAM 사용자 모음을 지정하고 대규모 사용자 집합에 대한 관리 권한을 더 쉽게 만듭니다. 자세한 내용은 IAM 사용자 설명서의 IAM 사용자 사용 사례 섹션을 참조하세요.

### IAM 역할

IAM 역할은 임시 자격 증명을 제공하는 특정 권한이 있는 자격 증명입니다. 사용자에서 IAM 역할(콘솔)로 전환하거나 AWS CLI또는 AWS API 작업을 직접 호출하여 역할을 수임할 수 있습니다. 자세한 내용은 IAM 사용자 설명서의 역할 수임 방법을 참조하세요.

IAM 역할은 페더레이션 사용자 액세스, 임시 IAM 사용자 권한, 교차 계정 액세스, 교차 서비스 액세스 및 Amazon EC2에서 실행되는 애플리케이션에 유용합니다. 자세한 내용은 IAM 사용자 설명서의 교차 계정 리소스 액세스를 참조하세요.

## 정책을 사용하여 액세스 관리

정책을 생성하고 AWS ID 또는 리소스에 연결하여 AWS에서 내 액세스를 제어합니다. 정책은 자격 증명이나 리소스와 연결될 때 해당 권한을 정의합니다. AWS는 보안 주체가 요청을 보낼 때 이러한 정책을 평가합니다. 대부분의 정책은 AWS에 JSON 문서로 저장됩니다. JSON 정책 문서에 대한 자세한 내용은 IAM 사용자 설명서의 JSON 정책 개요 섹션을 참조하세요.

정책을 사용하여 관리자는 어떤 보안 주체가 어떤 리소스에 대해 어떤 조건에서 작업을 수행할 수 있는지 정의하여 누가 무엇을 액세스할 수 있는지 지정합니다.

기본적으로 사용자 및 역할에는 어떠한 권한도 없습니다. IAM 관리자는 IAM 정책을 생성하고 사용자가 수임할 수 있는 역할에 추가합니다. IAM 정책은 작업을 수행하기 위해 사용하는 방법과 관계없이 작업에 대한 권한을 정의합니다.

### 자격 증명 기반 정책

ID 기반 정책은 ID(사용자, 사용자 그룹 또는 역할)에 연결하는 JSON 권한 정책 문서입니다. 이러한 정책은 자격 증명이 수행할 수 있는 작업, 대상 리소스 및 이에 관한 조건을 제어합니다. ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용자 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

ID 기반 정책은 인라인 정책(단일 ID에 직접 포함) 또는 관리형 정책(여러 ID에 연결된 독립 실행형 정책)일 수 있습니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용자 설명서의 관리형 정책 및 인라인 정책 중에서 선택 섹션을 참조하세요.

### 리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 예를 들어 IAM 역할 신뢰 정책 및 Amazon S3 버킷 정책이 있습니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다.

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. 리소스 기반 정책에서는 IAM의 AWS관리형 정책을 사용할 수 없습니다.

### 기타 정책 유형

AWS는 이러한 정책 타입이 부여하는 최대 권한을 설정할 수 있는 추가 정책 타입을 지지합니다.

- 권한 경계 - ID 기반 정책에서 IAM 엔터티에 부여할 수 있는 최대 권한을 설정합니다. 자세한 정보는 IAM 사용자 설명서의 IAM 엔터티의 권한 범위를 참조하세요.
- 서비스 제어 정책(SCP) - AWS Organizations내 조직 또는 조직 단위에 대한 최대 권한을 지정합니다. 자세한 내용은 AWS Organizations사용 설명서의 서비스 제어 정책을 참조하세요.
- 리소스 제어 정책(RCP) – 계정의 리소스에 사용할 수 있는 최대 권한을 설정합니다. 자세한 내용은 AWS Organizations사용 설명서의 리소스 제어 정책(RCP)을 참조하세요.
- 세션 정책 – 역할 또는 페더레이션 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 자세한 내용은 IAM 사용자 설명서의 세션 정책을 참조하세요.

### 여러 정책 유형

여러 정책 유형이 요청에 적용되는 경우, 결과 권한은 이해하기가 더 복잡합니다. 여러 정책 유형이 관련될 때 AWS가 요청을 허용할지 여부를 결정하는 방법을 알아보려면 IAM 사용자 설명서의 정책 평가 로직을 참조하세요.

## Amazon CloudFront와 함께 IAM을 사용하는 방법

IAM을 사용하여 CloudFront에 대한 액세스를 관리하기 전에 CloudFront와 함께 사용할 수 있는 IAM 기능을 알아보세요.

Amazon CloudFront에서 사용할 수 있는 IAM 기능

| IAM 특성 | CloudFront 지원 |
| --- | --- |
| 자격 증명 기반 정책 | 예 |
| 리소스 기반 정책 | 아니요 |
| 정책 작업 | 예 |
| 정책 리소스 | 예 |

| IAM 특성 | CloudFront 지원 |
| --- | --- |
| 정책 조건 키(서비스별) | 예 |
| ACL | 아니요 |
| ABAC(정책 내 태그) | 부분 |
| 임시 자격 증명 | 예 |
| 전달 액세스 세션(FAS) | 아니요 |
| 서비스 역할 | 아니요 |
| 서비스 연결 역할 | 예 |

CloudFront 및 기타 AWS 서비스에서 대부분의 IAM 기능을 사용하는 방법을 전체적으로 알아보려면 IAM 사용 설명서에서 IAM으로 작업하는 AWS 서비스를 참조하세요.

## CloudFront에 대한 자격 증명 기반 정책

자격 증명 기반 정책 지원: 예

자격 증명 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 ID에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자 및 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업과 리소스뿐 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. JSON 정책에서 사용할 수 있는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

CloudFront에 대한 자격 증명 기반 정책 예시

CloudFront 자격 증명 기반 정책 예시를 보려면 Amazon CloudFront에 대한 자격 증명 기반 정책 예시 섹션을 참조하세요.

## CloudFront 내 리소스 기반 정책

리소스 기반 정책 지원: 아니요

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예제는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 위탁자에는 계정, 사용자, 역할, 페더레이션 사용자 또는 AWS 서비스가 포함될 수 있습니다.

교차 계정 액세스를 활성화하려는 경우, 전체 계정이나 다른 계정의 IAM 개체를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM에서 교차 계정 리소스 액세스를 참조하세요.

## CloudFront 정책 작업

정책 작업 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지를 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 작업을 설명합니다. 연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하세요.

CloudFront 작업 목록을 보려면 서비스 권한 부여 참조에서 Amazon CloudFront에서 정의한 작업을 참조하세요.

CloudFront의 정책 작업은 작업 앞에 다음 접두사를 사용합니다.

```
cloudfront
```

단일 문에서 여러 작업을 지정하려면 쉼표로 구분합니다.

```
"Action": [
      "cloudfront:action1",
      "cloudfront:action2"
         ]
```

CloudFront 자격 증명 기반 정책 예시를 보려면 Amazon CloudFront에 대한 자격 증명 기반 정책 예시 섹션을 참조하세요.

## CloudFront 정책 리소스

정책 리소스 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지를 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Resource` JSON 정책 요소는 작업이 적용되는 하나 이상의 객체를 지정합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

```
"Resource": "*"
```

CloudFront 리소스 유형 및 해당 ARN 목록을 보려면 서비스 승인 참조에서 Amazon RDS에서 정의한 리소스를 참조하세요. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 Amazon CloudFront에서 정의한 작업을 참조하세요.

CloudFront 자격 증명 기반 정책 예시를 보려면 Amazon CloudFront에 대한 자격 증명 기반 정책 예시 섹션을 참조하세요.

## CloudFront 정책 조건 키

서비스별 정책 조건 키 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지를 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Condition` 요소는 정의된 기준에 따라 문이 실행되는 시기를 지정합니다. 같음(equals) 또는 미만(less than)과 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다. 모든 AWS 전역 조건 키를 보려면 IAM 사용자 설명서의 AWS 전역 조건 컨텍스트 키를 참조하세요.

CloudFront 조건 키 목록을 보려면 서비스 승인 참조의 Amazon CloudFront에 사용되는 조건 키를 참조하세요. 조건 키를 사용할 수 있는 작업과 리소스를 알아보려면 Amazon CloudFront에서 정의한 작업을 참조하세요.

CloudFront 자격 증명 기반 정책 예시를 보려면 Amazon CloudFront에 대한 자격 증명 기반 정책 예시 섹션을 참조하세요.

## CloudFront 내의 ACL

ACL 지원: 아니요

액세스 제어 목록(ACL)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

## CloudFront와 ABAC

ABAC 지원(정책의 태그): 부분적

속성 기반 액세스 제어(ABAC)는 태그라고 불리는 속성을 기반으로 권한을 정의하는 권한 부여 전략입니다. IAM 엔터티 및 AWS 리소스에 태그를 연결하면 보안 주체의 태그가 리소스 태그와 일치할 때 작업을 허용하도록 ABAC 정책을 설계할 수 있습니다.

태그에 근거하여 액세스를 제어하려면 `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`또는 `aws:TagKeys`조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다.

서비스가 모든 리소스 유형에 대해 세 가지 조건 키를 모두 지원하는 경우, 값은 서비스에 대해 예입니다. 서비스가 일부 리소스 유형에 대해서만 세 가지 조건 키를 모두 지원하는 경우, 값은 부분적입니다.

ABAC에 대한 자세한 내용은 IAM 사용자 설명서의 ABAC 권한 부여를 통한 권한 정의를 참조하세요. ABAC 설정 단계가 포함된 자습서를 보려면 IAM 사용자 설명서의 속성 기반 액세스 제어(ABAC) 사용을 참조하세요.

CloudFront는 배포, 테넌트 또는 트러스트 스토어와 같은 태그를 지원하는 CloudFront 리소스에 대해 ABAC를 지원합니다.

## CloudFront에서 임시 보안 인증 정보 사용

임시 자격 증명 지원: 예

임시 자격 증명은 AWS리소스에 대한 단기 액세스를 제공하며 페더레이션 또는 전환 역할을 사용할 때 자동으로 생성됩니다. AWS에서는 장기 액세스 키를 사용하는 대신 임시 자격 증명을 동적으로 생성할 것을 권장합니다. 자세한 내용은 IAM 사용 설명서의 IAM 임시 보안 자격 증명 및 IAM으로 작업하는 AWS 서비스 섹션을 참조하세요.

## CloudFront에 대한 전달 액세스 세션

전달 액세스 세션(FAS) 지원: 아니요

전달 액세스 세션(FAS)은 AWS 서비스를 직접 호출하는 위탁자의 권한과 요청하는 AWS 서비스를 함께 사용하여 다운스트림 서비스에 대한 요청을 수행합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.

### CloudFront에 대한 서비스 역할

서비스 역할 지원: 아니요

서비스 역할은 서비스가 사용자를 대신하여 작업을 수행하는 것으로 가정하는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 AWS 서비스 AWS에 권한을 위임할 역할 생성을 참조하세요.

> **Warning**
> 서비스 역할에 대한 권한을 변경하면 CloudFront 기능이 중단될 수 있습니다. CloudFront에서 관련 지침을 제공하는 경우에만 서비스 역할을 편집하세요.

### CloudFront 서비스 연결 역할

서비스 연결 역할 지원: 예

서비스 연결 역할은 AWS 서비스에 연결된 서비스 역할의 한 유형입니다. 서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은 AWS 계정에 나타나고, 서비스가 소유합니다. IAM 관리자는 서비스 연결 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.

CloudFront는 서비스 연결 역할을 사용하여 사용자를 대신하여 작업을 수행합니다. CloudFront 서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 CloudFront의 서비스 연결 역할 사용 섹션을 참조하세요. Lambda@Edge 서비스 연결 역할을 만들거나 관리하는 방법에 대한 자세한 내용은 Lambda@Edge의 서비스 연결 역할 섹션을 참조하시기 바랍니다.

서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 IAM으로 작업하는 AWS서비스를 참조하세요. 서비스 연결 역할 열에서 Yes가 포함된 서비스를 테이블에서 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예(Yes) 링크를 선택합니다.

## Amazon CloudFront에 대한 자격 증명 기반 정책 예시

기본적으로 사용자 및 역할에는 CloudFront 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성(콘솔)을 참조하세요.

각 리소스 유형에 대한 ARN 형식을 포함하여 CloudFront에서 정의한 작업 및 리소스 유형에 대한 자세한 내용은 서비스 승인 참조의 Amazon CloudFront에 사용되는 작업, 리소스 및 조건 키를 참조하세요.

주제

- 정책 모범 사례
- 사용자가 자신의 고유한 권한을 볼 수 있도록 허용
- 프로그래밍 방식으로 CloudFront에 액세스할 수 있는 권한
- CloudFront 콘솔을 사용하는 데 필요한 권한
- 고객 관리형 정책 예

## 정책 모범 사례

자격 증명 기반 정책에 따라 계정에서 사용자가 CloudFront 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부가 결정됩니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. 자격 증명 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책으로 시작하고 최소 권한을 향해 나아가기 - 사용자 및 워크로드에 권한 부여를 시작하려면 많은 일반 사용 사례에 대한 권한을 부여하는 AWS관리형 정책을 사용합니다. AWS 계정에서 사용할 수 있습니다. 사용 사례에 고유한 AWS고객 관리형 정책을 정의하여 권한을 줄이는 것이 좋습니다. 자세한 내용은 IAM 사용자 설명서의 AWS 관리형 정책 또는 AWS직무에 대한 관리형 정책을 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우, 작업을 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용자 설명서에 있는 IAM의 정책 및 권한을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어, SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. CloudFormation와 같이, 특정 AWS 서비스를 통해 사용되는 경우에만 서비스 작업에 대한 액세스 권한을 부여할 수도 있습니다. 자세한 내용은 IAM 사용자 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.
- IAM Access Analyzer를 통해 IAM 정책을 확인하여 안전하고 기능적인 권한 보장 - IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 새로운 및 기존 정책을 확인합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하여

안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 내용은 IAM 사용자 설명서의 IAM Access Analyzer에서 정책 검증을 참조하세요.

- 다중 인증(MFA) 필요 – AWS 계정에 IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우, 추가 보안을 위해 MFA를 설정합니다. API 작업을 직접적으로 호출할 때 MFA가 필요하면 정책에 MFA 조건을 추가합니다. 자세한 내용은 IAM 사용자 설명서의 MFA를 통한 보안 API 액세스를 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용자 설명서의 IAM의 보안 모범 사례를 참조하세요.

## 사용자가 자신의 고유한 권한을 볼 수 있도록 허용

이 예제는 IAM 사용자가 자신의 사용자 ID에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 AWS CLI나 AWS API를 사용하여 프로그래밍 방식으로 이 작업을 완료할 수 있는 권한이 포함됩니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
```

```
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

## 프로그래밍 방식으로 CloudFront에 액세스할 수 있는 권한

다음에서는 권한 정책을 보여 줍니다. `Sid`(명령문 ID)는 선택 사항입니다.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "AllowAllCloudFrontPermissions",
         "Effect": "Allow",
         "Action": ["cloudfront:*"],
         "Resource": "*"
      }
   ]
}
```

정책은 모든 CloudFront 작업을 수행할 수 있는 권한을 부여하며, 이러한 권한은 프로그래밍 방식으로 CloudFront에 액세스하는 데 충분합니다. 콘솔을 사용하여 CloudFront에 액세스하는 경우 CloudFront 콘솔을 사용하는 데 필요한 권한 단원을 참조하십시오.

작업과 각 작업을 사용할 권한을 부여하거나 거부하기 위해 지정하는 ARN의 목록은 서비스 승인 참조의 Amazon CloudFront에 사용되는 작업, 리소스 및 조건 키를 참조하세요.

## CloudFront 콘솔을 사용하는 데 필요한 권한

CloudFront 콘솔에 대한 전체 액세스 권한을 부여하려면 다음 권한 정책에서 권한을 부여합니다.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement":[
       {
          "Effect":"Allow",
          "Action":[
             "acm:ListCertificates",
             "cloudfront:*",
             "cloudwatch:DescribeAlarms",
             "cloudwatch:PutMetricAlarm",
             "cloudwatch:GetMetricStatistics",
             "elasticloadbalancing:DescribeLoadBalancers",
             "iam:ListServerCertificates",
             "sns:ListSubscriptionsByTopic",
             "sns:ListTopics",
             "waf:GetWebACL",
             "waf:ListWebACLs"
          ],
          "Resource":"*"
       },
       {
          "Effect":"Allow",
          "Action":[
             "s3:ListAllMyBuckets",
             "s3:PutBucketPolicy"
          ],
          "Resource":"arn:aws:s3:::*"
       }
    ]
}
```

다음은 권한이 필요한 이유입니다.

**`acm:ListCertificates`**

CloudFront 콘솔을 사용하여 배포를 생성하고 업데이트할 때 최종 사용자와 CloudFront 간에 또는 CloudFront와 원본 간에 HTTPS를 요구하도록 CloudFront를 구성하려는 경우, ACM 인증서 목록을 볼 수 있도록 허용합니다.

CloudFront 콘솔을 사용하지 않는 경우에는 이 권한이 필요하지 않습니다.

**`cloudfront:*`**

모든 CloudFront 작업을 수행할 수 있도록 허용합니다.

**`cloudwatch:DescribeAlarms` 및 `cloudwatch:PutMetricAlarm`**

CloudFront 콘솔에서 CloudWatch 경보를 생성하고 볼 수 있습니다. `sns:ListSubscriptionsByTopic` 및 `sns:ListTopics`도 참조하십시오.

CloudFront 콘솔을 사용하지 않는 경우에는 이러한 권한이 필요하지 않습니다.

**`cloudwatch:GetMetricStatistics`**

CloudFront가 CloudFront 콘솔에서 CloudWatch 지표를 렌더링할 수 있도록 허용합니다.

CloudFront 콘솔을 사용하지 않는 경우에는 이 권한이 필요하지 않습니다.

**`elasticloadbalancing:DescribeLoadBalancers`**

배포를 생성하고 업데이트할 때, 사용 가능한 원본 목록에서 Elastic Load Balancing 로드 밸런서 목록을 볼 수 있도록 허용합니다.

CloudFront 콘솔을 사용하지 않는 경우에는 이 권한이 필요하지 않습니다.

**`iam:ListServerCertificates`**

CloudFront 콘솔을 사용하여 배포를 생성하고 업데이트할 때 최종 사용자와 CloudFront 간에 또는 CloudFront와 원본 간에 HTTPS를 요구하도록 CloudFront를 구성하려는 경우, IAM 인증서 저장소에서 인증서 목록을 볼 수 있도록 허용합니다.

CloudFront 콘솔을 사용하지 않는 경우에는 이 권한이 필요하지 않습니다.

**`s3:ListAllMyBuckets`**

배포를 생성하고 업데이트할 때, 다음 작업을 수행할 수 있도록 허용합니다.

- 사용 가능한 원본 목록에서 S3 버킷 목록 보기
- 액세스 로그를 저장할 수 있는 S3 버킷 목록 보기

CloudFront 콘솔을 사용하지 않는 경우에는 이 권한이 필요하지 않습니다.

**`S3:PutBucketPolicy`**

액세스를 S3 버킷으로 제한하는 배포를 생성하거나 업데이트할 때, 사용자가 버킷 정책을 업데이트하여 CloudFront 원본 액세스 ID에 대한 액세스 권한을 부여할 수 있도록 허용합니다. 자세한 내용은 the section called “오리진 액세스 ID 사용(레거시, 권장하지 않음)” 섹션을 참조하세요.

CloudFront 콘솔을 사용하지 않는 경우에는 이 권한이 필요하지 않습니다.

**`sns:ListSubscriptionsByTopic` 및 `sns:ListTopics`**

CloudFront 콘솔에서 CloudWatch 경보를 생성할 때 알림에 대한 SNS 주제를 선택할 수 있도록 허용합니다.

CloudFront 콘솔을 사용하지 않는 경우에는 이러한 권한이 필요하지 않습니다.

**`waf:GetWebACL` 및 `waf:ListWebACLs`**

CloudFront 콘솔에서 AWS WAF 웹 ACL 목록을 볼 수 있도록 허용합니다.

CloudFront 콘솔을 사용하지 않는 경우에는 이러한 권한이 필요하지 않습니다.

CloudFront 콘솔에 대한 권한 전용 작업

CloudFront 보안 절감형 번들 페이지에서 다음 CloudFront 작업을 수행할 수 있습니다. API 작업은 코드에서 호출할 수 없으며 AWS CLI 및 AWS SDK에 포함되지 않습니다.

| 작업 | 설명 |
| --- | --- |
| `CreateSavingsPlan` | 새로운 절감형 플랜을 만들 수 있는 권한을 부여합니다. |
| `GetSavingsPlan` | 절감형 플랜을 가져올 수 있는 권한을 부여합니다. |
| `ListRateCards` | 계정에 대한 CloudFront 요금 카드를 나열할 수 있는 권한을 부여합니다. |
| `ListSavingsPlans` | 계정의 절감형 플랜을 나열할 수 있는 권한을 부여합니다. |
| `ListUsages` | CloudFront 사용을 나열할 수 있는 권한을 부여합니다. |
| `UpdateSavingsPlan` | 절감형 플랜을 업데이트할 수 있는 권한을 부여합니다. |

> **참고**
>
> - CloudFront 절감형 플랜에 대한 자세한 내용은 Amazon CloudFront FAQ의 CloudFront 보안 절감형 번들 섹션을 참조하시기 바랍니다.
> - CloudFront에 대한 절감형 플랜을 만든 다음, 나중에 삭제하려면 AWS Support에 문의하시기 바랍니다.

## 고객 관리형 정책 예

CloudFront API 작업에 대한 권한을 허용하는 고유의 사용자 지정 IAM 정책을 생성할 수 있습니다. 지정된 권한이 필요한 IAM 사용자 또는 그룹에 이러한 사용자 지정 정책을 연결할 수 있습니다. 이러한 정책은 CloudFront API, AWS SDK 또는 AWS CLI를 사용하는 경우에 유효합니다. 다음 예제에서는 몇 가지 일반 사용 사례에 대한 권한을 보여 줍니다. 사용자에게 CloudFront에 대한 전체 액세스 권한을 부여하는 정책에 대해서는 CloudFront 콘솔을 사용하는 데 필요한 권한 단원을 참조하십시오.

예제:

- 예제 1: 모든 배포에 대한 읽기 액세스 허용
- 예 2: 배포 생성, 업데이트, 삭제 허용
- 예제 3: 무효화 생성 및 나열 허용
- 예 4: 배포물 생성 허용

### 예제 1: 모든 배포에 대한 읽기 액세스 허용

다음 권한 정책은 사용자에게 CloudFront 콘솔에서 모든 배포를 볼 수 있는 사용자 권한을 부여합니다.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "acm:ListCertificates",
            "cloudfront:GetDistribution",
            "cloudfront:GetDistributionConfig",
```

```
            "cloudfront:ListDistributions",
            "cloudfront:ListCloudFrontOriginAccessIdentities",
            "elasticloadbalancing:DescribeLoadBalancers",
            "iam:ListServerCertificates",
            "sns:ListSubscriptionsByTopic",
            "sns:ListTopics",
            "waf:GetWebACL",
            "waf:ListWebACLs"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListAllMyBuckets"
         ],
         "Resource":"arn:aws:s3:::*"
      }
   ]
}
```

예 2: 배포 생성, 업데이트, 삭제 허용

다음 권한 정책은 사용자가 CloudFront 콘솔을 사용하여 배포를 생성, 업데이트 및 삭제할 수 있도록 허용합니다.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "acm:ListCertificates",
            "cloudfront:CreateDistribution",
            "cloudfront:DeleteDistribution",
            "cloudfront:GetDistribution",
            "cloudfront:GetDistributionConfig",
            "cloudfront:ListDistributions",
            "cloudfront:UpdateDistribution",
            "cloudfront:ListCloudFrontOriginAccessIdentities",
```

```
            "elasticloadbalancing:DescribeLoadBalancers",
            "iam:ListServerCertificates",
            "sns:ListSubscriptionsByTopic",
            "sns:ListTopics",
            "waf:GetWebACL",
            "waf:ListWebACLs"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListAllMyBuckets",
            "s3:PutBucketPolicy"
         ],
         "Resource":"arn:aws:s3:::*"
      }
   ]
}
```

`cloudfront:ListCloudFrontOriginAccessIdentities` 권한이 있으면 사용자가 Amazon S3 버킷의 객체에 액세스할 권한을 기존 원본 액세스 ID에 자동으로 부여할 수 있습니다. 또한 사용자가 원본 액세스 ID를 생성할 수 있도록 하려면 `cloudfront:CreateCloudFrontOriginAccessIdentity` 권한도 허용해야 합니다.

예제 3: 무효화 생성 및 나열 허용

다음 권한 정책은 사용자가 무효화를 생성하고 나열할 수 있도록 허용합니다. 먼저 배포에 대한 설정을 표시하여 무효화를 생성하고 보기 때문에 CloudFront 배포에 대한 읽기 액세스 권한도 여기에 포함됩니다.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "acm:ListCertificates",
            "cloudfront:GetDistribution",
```

```
            "cloudfront:GetStreamingDistribution",
            "cloudfront:GetDistributionConfig",
            "cloudfront:ListDistributions",
            "cloudfront:ListCloudFrontOriginAccessIdentities",
            "cloudfront:CreateInvalidation",
            "cloudfront:GetInvalidation",
            "cloudfront:ListInvalidations",
            "elasticloadbalancing:DescribeLoadBalancers",
            "iam:ListServerCertificates",
            "sns:ListSubscriptionsByTopic",
            "sns:ListTopics",
            "waf:GetWebACL",
            "waf:ListWebACLs"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListAllMyBuckets"
         ],
         "Resource":"arn:aws:s3:::*"
      }
   ]
}
```

예 4: 배포물 생성 허용

다음 권한 정책은 사용자에게 CloudFront 콘솔에서 배포를 생성하고 나열할 수 있는 권한을 부여합니다. `CreateDistribution` 작업의 경우 배포 ARN(`arn:aws:cloudfront::123456789012:distribution/*`)에 와일드카드 대신 `Resource`에 와일드카드(*) 문자를 지정합니다. `Resource` 요소에 대한 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 요소: 리소스를 참조합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
```

```
            "Effect": "Allow",
            "Action": "cloudfront:CreateDistribution",
            "Resource": "*"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": "cloudfront:ListDistributions",
            "Resource": "*"
        }
    ]
}
```

## AWSAmazon CloudFront용 관리형 정책

사용자, 그룹 또는 역할에 권한을 추가하려면 정책을 직접 작성하는 것보다 AWS 관리형 정책을 사용하는 것이 편리합니다. 사용자가 필요한 권한만 제공하는 IAM 고객 관리형 정책을 생성하는 데 시간과 전문 지식이 필요합니다. 빠르게 시작하려면 AWS 관리형 정책을 사용하면 됩니다. 이 정책은 일반적인 사용 사례를 다루며 사용자의 AWS 계정에서 사용할 수 있습니다. AWS 관리형 정책에 대한 자세한 내용은 IAM 사용 설명서에서 AWS 관리형 정책을 참조하세요.

AWS 서비스 유지 관리 및 AWS 관리형 정책 업데이트입니다. AWS 관리형 정책에서 권한을 변경할 수 없습니다. 서비스에서 때때로 추가 권한을 AWS 관리형 정책에 추가하여 새로운 기능을 지원합니다. 이 유형의 업데이트는 정책이 연결된 모든 ID(사용자, 그룹 및 역할)에 적용됩니다. 새 기능이 출시되거나 새 권한이 사용 가능해지면 서비스는 AWS 관리형 정책을 업데이트합니다. 서비스는 AWS 관리형 정책에서 권한을 제거하지 않기 때문에 정책 업데이트로 인해 기존 권한이 손상되지 않습니다.

또한 AWS는 여러 서비스에 걸쳐 있는 직무에 관한 관리형 정책을 지원합니다. 예를 들어 ReadOnlyAccess라는 이름의 AWS 관리형 정책은 모든 AWS 서비스 및 리소스에 대한 읽기 전용 액세스 권한을 제공합니다. 서비스에서 새 기능을 시작하면 AWS가 새 작업 및 리소스에 대한 읽기 전용 권한을 추가합니다. 직무 정책의 목록과 설명은 IAM 사용 설명서의 직무에 관한 AWS 관리형 정책을 참조하세요.

주제

- AWS 관리형 정책: CloudFrontReadOnlyAccess
- AWS 관리형 정책: CloudFrontFullAccess
- AWS 관리형 정책: AWSCloudFrontLogger

- AWS 관리형 정책: AWSLambdaReplicator
- AWS 관리형 정책: AWSCloudFrontVPCOriginServiceRolePolicy
- AWS 관리형 정책에 대해 CloudFront 업데이트

## AWS 관리형 정책: CloudFrontReadOnlyAccess

CloudFrontReadOnlyAccess 정책을 IAM 자격 증명에 연결할 수 있습니다. 이 정책은 CloudFront 리소스에 읽기 전용 권한을 허용합니다. 또한 CloudFront와 관련되어 있고 CloudFront 콘솔에 표시되는 AWS 서비스 리소스에 읽기 전용 권한을 허용합니다.

### 권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `cloudfront:Describe*` - 보안 주체가 CloudFront 리소스에 대한 메타데이터 정보를 가져올 수 있습니다.
- `cloudfront:Get*` - 보안 주체가 CloudFront 리소스에 대한 자세한 정보 및 구성을 가져올 수 있습니다.
- `cloudfront:List*` - 보안 주체가 CloudFront 리소스 목록을 가져올 수 있습니다.
- `cloudfront-keyvaluestore:Describe*` - 보안 주체가 키 값 저장소에 대한 정보를 얻을 수 있습니다.
- `cloudfront-keyvaluestore:Get*` - 보안 주체가 키 값 저장소에 대한 자세한 정보 및 구성을 가져올 수 있습니다.
- `cloudfront-keyvaluestore:List*` - 보안 주체가 키 값 저장소 목록을 얻을 수 있습니다.
- `acm:DescribeCertificate` - 위탁자가 ACM 인증서에 대한 세부 정보를 가져올 있습니다.
- `acm:ListCertificates` - 보안 주체가 ACM 인증서 목록을 가져오도록 허용합니다.
- `iam:ListServerCertificates` - 보안 주체가 IAM에 저장된 서버 인증서 목록을 가져올 수 있습니다.
- `route53:List*` - 보안 주체가 Route 53 리소스 목록을 가져올 수 있습니다.
- `waf:ListWebACLs` - 보안 주체가 에서 웹 ACL 목록을 가져올 수 있도록 허용합니다.AWS WAF
- `waf:GetWebACL` - 보안 주체가 에서 웹 ACL에 대한 세부 정보를 가져올 수 있도록 허용합니다.AWS WAF

- `wafv2:ListWebACLs` - 보안 주체가 에서 웹 ACL 목록을 가져올 수 있도록 허용합니다.AWS WAF
- `wafv2:GetWebACL` - 보안 주체가 에서 웹 ACL에 대한 세부 정보를 가져올 수 있도록 허용합니다.AWS WAF
- `pricingplanmanager:GetSubscription` - 보안 주체가 요금제 구독에 대한 세부 정보를 가져올 수 있는 읽기 전용 액세스를 허용합니다.
- `pricingplanmanager:ListSubscriptions` - 보안 주체가 요금제 구독을 나열할 수 있는 읽기 전용 액세스를 허용합니다.
- `ec2:DescribeIpamPools` - 보안 주체가 IPAM 풀에 대한 세부 정보를 가져올 수 있도록 허용합니다.
- `ec2:GetIpamPoolCidrs` - 보안 주체가 IPAM 풀에 프로비저닝된 CIDR을 가져올 수 있도록 허용합니다.

이 정책의 권한을 보려면 AWS 관리형 정책 참조의 CloudFrontReadOnlyAccess를 확인하시기 바랍니다.

## AWS 관리형 정책: CloudFrontFullAccess

CloudFrontFullAccess 정책을 IAM 자격 증명에 연결할 수 있습니다. 이 정책은 CloudFront 리소스에 대한 관리 권한을 허용합니다. 또한 CloudFront와 관련되어 있고 CloudFront 콘솔에 표시되는 AWS 서비스 리소스에 읽기 전용 권한을 허용합니다.

### 권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `s3:ListAllMyBuckets` - 보안 주체가 모든 Amazon S3 버킷 목록을 가져오도록 허용합니다.
- `acm:DescribeCertificate` - 위탁자가 ACM 인증서에 대한 세부 정보를 가져올 있습니다.
- `acm:ListCertificates` - 보안 주체가 ACM 인증서 목록을 가져오도록 허용합니다.
- `acm:RequestCertificate` - 위탁자가 ACM에서 관리형 인증서를 요청할 수 있도록 허용합니다.
- `cloudfront:*` - 보안 주체가 모든 CloudFront 리소스에서 모든 작업을 수행하도록 허용합니다.
- `cloudfront-keyvaluestore:*` - 보안 주체가 키 값 저장소에 대한 모든 작업을 수행할 수 있습니다.
- `iam:ListServerCertificates` - 보안 주체가 IAM에 저장된 서버 인증서 목록을 가져올 수 있습니다.

- `waf:ListWebACLs` - 보안 주체가 에서 웹 ACL 목록을 가져올 수 있도록 허용합니다.AWS WAF
- `waf:GetWebACL` - 보안 주체가 에서 웹 ACL에 대한 세부 정보를 가져올 수 있도록 허용합니다.AWS WAF
- `waf:CreateWebACLs` - 보안 주체가 AWS WAF에서 웹 ACL을 생성할 수 있도록 허용합니다.
- `wafv2:ListWebACLs` - 보안 주체가 에서 웹 ACL 목록을 가져올 수 있도록 허용합니다.AWS WAF
- `wafv2:GetWebACL` - 보안 주체가 에서 웹 ACL에 대한 세부 정보를 가져올 수 있도록 허용합니다.AWS WAF
- `kinesis:ListStreams` - 보안 주체가 Amazon Kinesis Streams 목록을 가져올 수 있도록 허용합니다.
- `elasticloadbalancing:DescribeLoadBalancers` - 위탁자가 Elastic Load Balancing의 로드 밸런서에 대한 자세한 정보를 얻을 수 있습니다.
- `kinesis:DescribeStream` - 보안 주체가 Kinesis 스트림에 대한 세부 정보를 가져올 수 있도록 허용합니다.
- `iam:ListRoles` - 보안 주체가 IAM에서 역할 목록을 가져오도록 허용합니다.
- `pricingplanmanager:AssociateResourcesToSubscription` - 보안 주체가 리소스를 구독에 연결할 수 있도록 허용합니다. 이렇게 하면 리소스에 구독의 요금제를 적용할 수 있습니다.
- `pricingplanmanager:CancelSubscription` - 보안 주체가 기존 구독을 취소할 수 있도록 허용합니다.
- `pricingplanmanager:CancelSubscriptionChange` - 변경 사항이 적용되기 전에 보안 주체가 기존 구독에 대해 보류 중인 변경(예: 플랜 업그레이드)을 취소할 수 있습니다.
- `pricingplanmanager:CreateSubscription` - 보안 주체가 요금제 구독을 생성할 수 있도록 허용합니다.
- `pricingplanmanager:DisassociateResourcesFromSubscription` - 보안 주체가 리소스와 기존 구독 간의 연결을 제거할 수 있도록 허용합니다.
- `pricingplanmanager:UpdateSubscription` - 보안 주체가 요금제 변경과 같은 기존 구독을 수정할 수 있도록 허용합니다.
- `pricingplanmanager:GetSubscription` - 보안 주체가 요금제 구독에 대한 세부 정보를 가져올 수 있는 읽기 전용 액세스를 허용합니다.
- `pricingplanmanager:ListSubscriptions` - 보안 주체가 요금제 구독을 나열할 수 있는 읽기 전용 액세스를 허용합니다.
- `ec2:DescribeInstances` - 위탁자가 Amazon EC2의 인스턴스에 대한 자세한 정보를 얻을 수 있습니다.

- `ec2:DescribeInternetGateways` - 위탁자가 Amazon EC2의 인터넷 게이트웨이에 대한 자세한 정보를 얻을 수 있습니다.
- `ec2:DescribeIpamPools` - 보안 주체가 IPAM 풀에 대한 세부 정보를 가져올 수 있도록 허용합니다.
- `ec2:GetIpamPoolCidrs` - 보안 주체가 IPAM 풀에 프로비저닝된 CIDR을 가져올 수 있도록 허용합니다.

이 정책의 권한을 보려면 AWS 관리형 정책 참조의 CloudFrontFullAccess를 확인하시기 바랍니다.

> **Important**
>
> CloudFront에서 액세스 로그를 생성하여 저장하도록 하려면 추가 권한을 부여해야 합니다. 자세한 내용은 권한 섹션을 참조하세요.

## AWS 관리형 정책: AWSCloudFrontLogger

AWSCloudFrontLogger 정책을 IAM 자격 증명에 연결할 수 없습니다. 이 정책은 CloudFront에서 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 the section called “Lambda@Edge의 서비스 연결 역할” 섹션을 참조하세요.

이 정책은 CloudFront에서 로그 파일을 Amazon CloudFront로 푸시할 수 있도록 허용합니다. 이 정책에 포함된 권한에 대한 자세한 내용은 the section called “CloudFront Logger에 대한 서비스 연결 역할 권한” 섹션을 참조하세요.

이 정책의 권한을 보려면 AWS 관리형 정책 참조에서 AWSCloudFrontLogger를 참조하시기 바랍니다.

## AWS 관리형 정책: AWSLambdaReplicator

AWSLambdaReplicator 정책을 IAM 자격 증명에 연결할 수 없습니다. 이 정책은 CloudFront에서 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 the section called “Lambda@Edge의 서비스 연결 역할” 섹션을 참조하세요.

이 정책을 통해 CloudFront는 AWS Lambda에서 함수를 생성, 삭제 및 비활성화하여 Lambda @Edge 함수를 AWS 리전에 복제할 수 있습니다. 이 정책에 포함된 권한에 대한 자세한 내용은 the section called “Lambda Replicator의 서비스 연결 역할 권한” 섹션을 참조하세요.

이 정책의 권한을 보려면 AWS 관리형 정책 참조에서 AWSLambdaReplicator를 참조하시기 바랍니다.

## AWS 관리형 정책: AWSCloudFrontVPCOriginServiceRolePolicy

AWSCloudFrontVPCOriginServiceRolePolicy를 IAM 엔터티에 연결할 수 없습니다. 이 정책은 CloudFront에서 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 CloudFront의 서비스 연결 역할 사용 섹션을 참조하세요.

CloudFront가 사용자를 대신하여 EC2 탄력적 네트워크 인터페이스 및 보안 그룹을 관리할 수 있도록 허용합니다. 이 정책에 포함된 권한에 대한 자세한 내용은 the section called “CloudFront VPC 오리진에 대한 서비스 연결 역할 권한” 섹션을 참조하세요.

이 정책의 권한을 보려면 AWS 관리형 정책 참조의 AWSCloudFrontVPCOriginServiceRolePolicy를 참조하시기 바랍니다.

## AWS 관리형 정책에 대해 CloudFront 업데이트

이 서비스가 이러한 변경 내용을 추적하기 시작한 이후부터 CloudFront의 AWS 관리형 정책 업데이트에 대한 세부 정보를 봅니다. 이 페이지의 변경 사항에 대한 자동 알림을 받아보려면 CloudFront 문서 기록 페이지에서 RSS 피드를 구독하세요.

| 변경 | 설명 | 날짜 |
| --- | --- | --- |
| CloudFrontReadOnlyAccess - 기존 정책에 대한 업데이트 | CloudFront에 Amazon EC2에 대한 새 권한을 추가했습니다.<br><br>새 권한을 통해 보안 주체는 `ec2:DescribeIpamPools` 및 `ec2:GetIpamPoolCidrs` 작업을 사용할 수 있습니다. | 2025년 11월 24일 |
| CloudFrontFullAccess - 기존 정책에 대한 업데이트 | CloudFront에 Amazon EC2에 대한 새 권한을 추가했습니다.<br><br>새 권한을 통해 보안 주체는 `ec2:DescribeIpamPools` 및 `ec2:GetIpamPoolCidrs` 작업을 사용할 수 있습니다. | 2025년 11월 24일 |

| 변경 | 설명 | 날짜 |
| --- | --- | --- |
| CloudFrontFullAccess - 기존 정책에 대한 업데이트 | CloudFront에 AWS WAF ACL 리소스를 생성할 수 있는 새 권한을 추가하고 AWS Pricing Plan Manager에 생성, 업데이트, 삭제 및 읽기 권한을 추가했습니다. | 2025년 11월 18일 |
| CloudFrontFullAccess - 기존 정책에 대한 업데이트 | CloudFront에 AWS WAF ACL 리소스를 생성할 수 있는 새 권한을 추가하고 AWS Pricing Plan Manager에 생성, 업데이트, 삭제 및 읽기 권한을 추가했습니다. | 2025년 11월 18일 |
| CloudFrontReadOnlyAccess - 기존 정책에 대한 업데이트 | CloudFront에 AWS Pricing Plan Manager에 대한 읽기 전용 액세스 권한을 새로 추가했습니다. | 2025년 11월 18일 |
| CloudFrontReadOnlyAccess - 기존 정책에 대한 업데이트 | CloudFront에 AWS Pricing Plan Manager에 대한 읽기 전용 액세스 권한을 새로 추가했습니다. | 2025년 11월 18일 |
| CloudFrontReadOnlyAccess - 기존 정책에 대한 업데이트 | CloudFront에 ACM에 대한 새 권한을 추가했습니다.<br><br>새 권한을 통해 위탁자는 ACM 인증서에 대한 세부 정보를 가져올 수 있습니다. | 2025년 4월 28일 |

| 변경 | 설명 | 날짜 |
| --- | --- | --- |
| CloudFrontFullAccess - 기존 정책에 대한 업데이트 | CloudFront에 ACM에 대한 새 권한을 추가했습니다.<br><br>새 권한을 통해 위탁자는 ACM 인증서에 대한 세부 정보를 가져오고 ACM에서 관리형 인증서를 요청할 수 있습니다. | 2025년 4월 28일 |
| CloudFrontFullAccess - 기존 정책에 대한 업데이트 | CloudFront에서 Amazon EC2 및 Elastic Load Balancing에 대한 새 권한을 추가했습니다.<br><br>새로운 권한을 통해 CloudFront는 Elastic Load Balancing의 로드 밸런서와 Amazon EC2의 인스턴스 및 인터넷 게이트웨이에 대한 자세한 정보를 얻을 수 있습니다. | 2024년 11월 20일 |
| AWSCloudFrontVPCOriginServiceRolePolicy – 새 정책 | CloudFront에서 새 정책을 추가했습니다.<br><br>CloudFront가 사용자를 대신하여 EC2 탄력적 네트워크 인터페이스 및 보안 그룹을 관리할 수 있도록 허용합니다. | 2024년 11월 20일 |
| CloudFrontReadOnlyAccess 및 CloudFrontFullAccess - 두 가지 기존 정책에 대한 업데이트입니다. | CloudFront에서 키 값 저장소에 대한 새로운 권한을 추가했습니다.<br><br>새 권한을 통해 사용자는 키 값 저장소에 대한 정보를 얻고 키 값 저장소에 대한 조치를 취할 수 있습니다. | 2023년 12월 19일 |

| 변경 | 설명 | 날짜 |
| --- | --- | --- |
| CloudFrontReadOnlyAccess - 기존 정책에 대한 업데이트 | CloudFront는 CloudFront 함수를 설명하는 새로운 권한을 추가했습니다.<br><br>이 권한을 사용하면 사용자, 그룹 또는 역할이 함수에 대한 정보 및 메타데이터를 읽을 수 있지만 함수 코드는 읽을 수 없습니다. | 2021년 9월 8일 |
| CloudFront, 변경 내용 추적 시작 | CloudFront가 AWS 관리형 정책에 대한 변경 내용 추적을 시작했습니다. | 2021년 9월 8일 |

## CloudFront의 서비스 연결 역할 사용

Amazon CloudFront는 AWS Identity and Access Management(IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 CloudFront에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 CloudFront에서 미리 정의하며 서비스가 사용자를 대신하여 다른 AWS 서비스를 호출하는 데 필요한 모든 권한을 포함합니다.

서비스 연결 역할을 사용하면 필요한 권한을 수동으로 추가할 필요가 없으므로 CloudFront를 더 쉽게 설정할 수 있습니다. CloudFront에서 서비스 연결 역할의 권한을 정의하므로 다르게 정의되지 않은 한, CloudFront만 해당 역할을 수임할 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

먼저 관련 리소스를 삭제한 후에만 서비스 연결 역할을 삭제할 수 있습니다. 이렇게 하면 리소스에 대한 액세스 권한을 부주의로 삭제할 수 없기 때문에 CloudFront 리소스가 보호됩니다.

서비스 연결 역할을 지원하는 기타 서비스에 대한 자세한 내용은 IAM으로 작업하는 AWS 서비스를 참조하고 서비스 연결 역할 열에 예가 있는 서비스를 찾으세요. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예 링크를 선택합니다.

## CloudFront VPC 오리진에 대한 서비스 연결 역할 권한

CloudFront VPC 오리진은 AWSServiceRoleForCloudFrontVPCOrigin이라는 서비스 연결 역할을 사용합니다. CloudFront가 사용자를 대신하여 EC2 탄력적 네트워크 인터페이스 및 보안 그룹을 관리할 수 있습니다.

AWSServiceRoleForCloudFrontVPCOrigin 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- `vpcorigin.cloudfront.amazonaws.com`

이름이 AWSCloudFrontVPCOriginServiceRolePolicy인 연결 권한 정책은 CloudFront VPC 오리진이 지정된 리소스에 대해 다음 작업을 완료하도록 허용합니다.

- 작업: `arn:aws:ec2:*:*:network-interface/*`에 대한 `ec2:CreateNetworkInterface`
- 작업: `arn:aws:ec2:*:*:subnet/*` 및 `arn:aws:ec2:*:*:security-group/*`의 `ec2:CreateNetworkInterface`
- 작업: `arn:aws:ec2:*:*:security-group/*`에 대한 `ec2:CreateSecurityGroup`
- 작업: `arn:aws:ec2:*:*:vpc/*`에 대한 `ec2:CreateSecurityGroup`
- 작업: `supported AWS resources that have the aws:ResourceTag/aws.cloudfront.vpcorigin tag enabled`의 `ec2:ModifyNetworkInterfaceAttribute`, `ec2:DeleteNetworkInterface`, `ec2:DeleteSecurityGroup`, `ec2:AssignIpv6Addresses`, `ec2:UnassignIpv6Addresses`
- 작업: `all AWS resources that the actions support`의 `ec2:DescribeNetworkInterfaces`, `ec2:DescribeSecurityGroups`, `ec2:DescribeInstances`, `ec2:DescribeInternetGateways`, `ec2:DescribeSubnets`, `ec2:DescribeRegions`, `ec2:DescribeAddresses`
- 작업: `arn:aws:ec2:*:*:security-group/*` 및 `arn:aws:ec2:*:*:network-interface/*`의 `ec2:CreateTags`
- 작업: `all AWS resources that the actions support`의 `elasticloadbalancing:DescribeLoadBalancers`, `elasticloadbalancing:DescribeListeners`, `elasticloadbalancing:DescribeTargetGroups`

사용자, 그룹 또는 역할이 서비스 연결 역할을 생성, 편집 또는 삭제할 수 있도록 사용 권한을 구성해야 합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

## CloudFront VPC 오리진에 대한 서비스 연결 역할 생성

서비스 연결 역할은 수동으로 생성할 필요가 없습니다. AWS Management Console, AWS CLI 또는 AWS API에서 VPC 오리진을 만들 때 CloudFront VPC 오리진은 서비스 연결 역할을 만듭니다.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다. VPC 오리진을 만들 때 CloudFront VPC 오리진은 서비스 연결 역할을 다시 만듭니다.

## CloudFront VPC 오리진에 대한 서비스 연결 역할 편집

CloudFront VPC 오리진에서는 AWSServiceRoleForCloudFrontVPCOrigin 서비스 연결 역할을 편집하도록 허용하지 않습니다. 서비스 연결 역할을 생성한 후에는 다양한 개체가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

## CloudFront VPC 오리진에 대한 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 따라서 적극적으로 모니터링하거나 유지하지 않는 미사용 엔터티가 없도록 합니다. 단, 서비스 연결 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

> **Note**
>
> 리소스를 삭제하려 할 때 CloudFront 서비스가 역할을 사용 중이면 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AWSServiceRoleForCloudFrontVPCOrigin에서 사용하는 CloudFront VPC 오리진 리소스를 삭제하려면 다음과 같이 합니다.

- 계정에서 VPC 오리진 리소스를 삭제합니다.
  - CloudFront가 계정에서 리소스 삭제를 완료하는 데 다소 시간이 걸릴 수 있습니다. 서비스 연결 역할을 즉시 삭제할 수 없는 경우 기다렸다가 다시 시도합니다.

### IAM을 사용하여 수동으로 서비스 연결 역할을 삭제하려면

IAM 콘솔, AWS CLI 또는 AWS API를 사용하여 AWSServiceRoleForCloudFrontVPCOrigin 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

## CloudFront VPC 오리진 서비스 연결 역할을 지원하는 리전

CloudFront VPC 오리진에서는 서비스가 제공되는 모든 리전에서 서비스 연결 역할을 사용하도록 지원하지 않습니다. 다음 리전에서 AWSServiceRoleForCloudFrontVPCOrigin 역할을 사용할 수 있습니다.

| 리전 이름 | 리전 자격 증명 | CloudFront에서 지원 |
| --- | --- | --- |
| 미국 동부(버지니아 북부) | us-east-1 | 예 |
| 미국 동부(오하이오) | us-east-2 | 예 |
| 미국 서부(캘리포니아 북부) | us-west-1(AZ usw1-az2 제외) | 예 |
| 미국 서부(오리건) | us-west-2 | 예 |
| 아프리카(케이프타운) | af-south-1 | 예 |
| 아시아 태평양(홍콩) | ap-east-1 | 예 |
| 아시아 태평양(자카르타) | ap-southeast-3 | 예 |
| 아시아 태평양(멜버른) | ap-southeast-4 | 예 |
| 아시아 태평양(뭄바이) | ap-south-1 | 예 |
| 아시아 태평양(하이데라바드) | ap-south-2 | 예 |
| 아시아 태평양(오사카) | ap-northeast-3 | 예 |
| 아시아 태평양(서울) | ap-northeast-2 | 예 |
| 아시아 태평양(싱가포르) | ap-southeast-1 | 예 |
| 아시아 태평양(시드니) | ap-southeast-2 | 예 |
| 아시아 태평양(도쿄) | ap-northeast-1(AZ apne1-az3 제외) | 예 |

| 리전 이름 | 리전 자격 증명 | CloudFront에서 지원 |
| --- | --- | --- |
| 캐나다(중부) | ca-central-1(AZ cac1-az3 제외) | 예 |
| 캐나다 서부(캘거리) | ca-west-1 | 예 |
| 유럽(프랑크푸르트) | eu-central-1 | 예 |
| 유럽(아일랜드) | eu-west-1 | 예 |
| 유럽(런던) | eu-west-2 | 예 |
| 유럽(밀라노) | eu-south-1 | 예 |
| 유럽(파리) | eu-west-3 | 예 |
| 유럽(스페인) | eu-south-2 | 예 |
| 유럽(스톡홀름) | eu-north-1 | 예 |
| 유럽(취리히) | eu-central-2 | 예 |
| 이스라엘(텔아비브) | il-central-1 | 예 |
| 중동(바레인) | me-south-1 | 예 |
| 중동(UAE) | me-central-1 | 예 |
| 남아메리카(상파울루) | sa-east-1 | 예 |

## Amazon CloudFront 자격 증명 및 액세스 문제 해결

다음 정보를 사용하여 CloudFront 및 IAM 작업 시 발생할 수 있는 일반적인 문제를 진단하고 수정할 수 있습니다.

주제

- CloudFront에서 작업을 수행할 권한이 없습니다.
- iam:PassRole을 수행하도록 인증되지 않음
- 내 AWS 계정 외부의 사람이 내 CloudFront 리소스에 액세스할 수 있게 허용하기를 원합니다.

## CloudFront에서 작업을 수행할 권한이 없습니다.

작업을 수행할 권한이 없다는 오류가 표시되면 작업을 수행할 수 있도록 정책을 업데이트해야 합니다.

다음의 예제 오류는 `mateojackson` IAM 사용자가 콘솔을 사용하여 가상 *`my-example-widget`* 리소스에 대한 세부 정보를 보려고 하지만 가상 `cloudfront:`*`GetWidget`* 권한이 없을 때 발생합니다.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cloudfront:GetWidget on resource: my-example-widget
```

이 경우, `cloudfront:`*`GetWidget`* 작업을 사용하여 *`my-example-widget`* 리소스에 액세스할 수 있도록 `mateojackson` 사용자 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

## iam:PassRole을 수행하도록 인증되지 않음

`iam:PassRole` 작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 CloudFront에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다.

일부 AWS 서비스에서는 새로운 서비스 역할 또는 서비스 연결 역할을 생성하는 대신 해당 서비스에 기존 역할을 전달할 수 있습니다. 이렇게 하려면 사용자가 서비스에 역할을 전달할 수 있는 권한을 가지고 있어야 합니다.

다음 예시 오류는 `marymajor`라는 IAM 사용자가 콘솔을 사용하여 CloudFront에서 작업을 수행하려고 하는 경우에 발생합니다. 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 수 있는 권한을 가지고 있지 않습니다.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

이 경우, Mary가 `iam:PassRole`작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

## 내 AWS 계정 외부의 사람이 내 CloudFront 리소스에 액세스할 수 있게 허용하기를 원합니다.

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제

어 목록(ACL)을 지원하는 서비스의 경우, 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세한 내용은 다음을 참조하세요.

- CloudFront에서 이러한 기능을 지원하는지 여부를 알아보려면 Amazon CloudFront와 함께 IAM을 사용하는 방법 섹션을 참조하세요.
- 소유하고 있는 AWS 계정의 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용자 설명서의 자신이 소유한 다른 AWS 계정의 IAM 사용자에 대한 액세스 권한 제공을 참조하세요.
- 리소스에 대한 액세스 권한을 서드 파티 AWS 계정에게 제공하는 방법을 알아보려면 IAM 사용자 설명서의 서드 파티가 소유한 AWS 계정에 대한 액세스 제공을 참조하세요.
- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용자 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(ID 페더레이션)을 참조하세요.
- 크로스 계정 액세스에 대한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용자 설명서의 IAM의 크로스 계정 리소스 액세스를 참조하세요.

# Amazon CloudFront의 로깅 및 모니터링

모니터링은 CloudFront 및 AWS 솔루션의 가용성과 성능을 유지하는 데 중요한 부분입니다. 다중 지점 실패가 발생할 경우 보다 쉽게 디버깅할 수 있도록 AWS 솔루션의 모든 부분으로부터 모니터링 데이터를 수집해야 합니다.AWS는 CloudFront 리소스와 활동을 모니터링하고 잠재적 인시던트에 대응하기 위한 여러 도구를 제공합니다.

Amazon CloudWatch 경보

CloudWatch 경보를 사용하면 지정한 기간 동안 단일 지표를 감시할 수 있습니다. 지표가 지정된 임계값을 초과하면 Amazon SNS 주제 또는 AWS Auto Scaling 정책으로 알림이 전송됩니다. CloudWatch 경보는 지표가 특정 상태에 있다고 해서 작업을 호출하지 않습니다. 대신, 상태가 변경되어 지정된 기간 동안 유지되어야 합니다.

자세한 내용은 Amazon CloudWatch를 사용한 CloudFront 지표 모니터링 단원을 참조하세요.

AWS CloudTrail 로그

CloudTrail은 CloudFront에서 사용자, 역할 또는 AWS 서비스가 수행한 API 작업의 기록을 제공합니다. CloudTrail에서 수집한 정보를 사용하여 CloudFront에 수행된 API 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

자세한 내용은 AWS CloudTrail을 사용하여 Amazon CloudFront API 직접 호출 로깅 섹션을 참조하세요.

CloudFront 표준 로그 및 실시간 액세스 로그

CloudFront 로그는 배포에 대해 이루어진 요청에 대한 상세 레코드를 제공합니다. 이러한 로그는 많은 애플리케이션에 유용합니다. 예를 들어 로그 정보는 보안 및 액세스 감사에 유용할 수 있습니다.

자세한 내용은 액세스 로그(표준 로그) 및 실시간 액세스 로그 구성 생성 및 사용(을)를 참조하세요.

엣지 함수 로그

엣지 함수(CloudFront Functions 및 Lambda@Edge 모두)에서 생성된 로그는 Amazon CloudWatch Logs로 직접 전송되며 CloudFront에서는 어디에도 저장되지 않습니다. CloudFront Functions는 AWS Identity and Access Management(IAM) 서비스 연결 역할을 사용하여 사용자 계정의 CloudWatch Logs로 사용자가 생성한 로그를 직접 전송합니다.

자세한 내용은 엣지 함수 로그 섹션을 참조하세요.

CloudFront 콘솔 보고서

CloudFront 콘솔에는 캐시 통계 보고서, 많이 사용되는 객체 보고서 및 주요 참조 보고서 등의 다양한 보고서가 포함되어 있습니다. 대부분의 CloudFront 콘솔 보고서는 CloudFront 액세스 로그의 데이터를 기반으로 합니다. 이 로그에는 CloudFront가 수신하는 모든 사용자 요청에 대한 세부 정보가 포함되어 있습니다. 그러나 보고서를 보기 위해 액세스 로그를 활성화할 필요는 없습니다.

자세한 내용은 콘솔에서 CloudFront 보고서 보기 단원을 참조하세요.

# Amazon CloudFront에 대한 규정 준수 확인

타사 감사자는 여러 AWS 규정 준수 프로그램의 일환으로 Amazon CloudFront의 보안 및 규정 준수를 평가합니다. 여기에는 SOC, PCI, HIPAA 등이 포함됩니다.

특정 규정 준수 프로그램 범위에 속하는 AWS 서비스의 목록은 규정 준수 프로그램 제공 AWS 범위 내 서비스 페이지에서 확인하세요. 일반 정보는 AWS 규정 준수 프로그램을 참조하세요.

AWS Artifact를 사용하여 제3자 감사 보고서를 다운로드할 수 있습니다. 자세한 설명은 AWS Artifact의 보고서 다운로드를 참조합니다.

CloudFront 사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률과 규정에 따라 결정됩니다. AWS에서는 규정 준수를 지원할 다음과 같은 리소스를 제공합니다.

- 보안 및 규정 준수 빠른 시작 안내서 – 이 배포 안내서에서는 아키텍처 고려 사항에 대해 설명하고 AWS에서 보안 및 규정 준수에 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.
- AWS에 대한 Architecting for HIPAA Security and Compliance - 이 백서는 기업에서 AWS를 사용하여 HIPAA를 준수하는 애플리케이션 생성 방법을 설명합니다.

  AWS HIPAA 규정 준수 프로그램에는 CloudFront가 HIPAA 적격 서비스로 CloudFront(CloudFront 임베디드 POP를 통한 콘텐츠 전송 제외)가 HIPAA 적격 서비스로 포함되어 있습니다. AWS와 BAA(Business Associate Addendum)를 체결한 경우, CloudFront(CloudFront 임베디드 POP를 통한 콘텐츠 전송 제외)를 사용하여 PHI(보호 대상 건강 정보)가 포함된 콘텐츠를 제공할 수 있습니다. 자세한 내용은 HIPAA 규정 준수를 참조하세요.
- AWS 규정 준수 리소스 – 사용자의 업계와 위치에 해당할 수 있는 워크북 및 안내서 모음입니다.
- AWS Config – 이 AWS 서비스로 리소스 구성이 내부 관행, 업계 지침 및 규정을 준수하는 정도를 평가할 수 있습니다.
- AWS Security Hub CSPM - 이 AWS 서비스는 보안 제어를 사용하여 리소스 구성 및 보안 표준을 평가하여 다양한 규정 준수 프레임워크를 준수할 수 있도록 지원합니다. Security Hub CSPM을 사용하여 CloudFront 리소스를 평가하는 방법에 대한 자세한 내용은 AWS Security Hub CSPM 사용 설명서의 Amazon CloudFront 제어를 참조하세요.

## CloudFront 규정 준수 모범 사례

이 단원에서는 Amazon CloudFront를 사용하여 콘텐츠를 제공할 때 규정 준수를 위해 참고할 수 있는 모범 사례 및 권장 사항을 제공합니다.

AWS 공동 책임 모델에 따라 PCI 또는 HIPAA 규정에 맞는 워크로드를 실행하는 경우, 향후 감사를 위해 과거 365일 간의 CloudFront 사용 데이터를 기록해 두는 것이 좋습니다. 사용 데이터를 기록하는 방법은 다음과 같습니다.

- CloudFront 액세스 로그 활성화 자세한 내용은 액세스 로그(표준 로그) 섹션을 참조하세요.
- CloudFront API로 전송된 요청을 캡처합니다. 자세한 내용은 AWS CloudTrail을 사용하여 Amazon CloudFront API 직접 호출 로깅 섹션을 참조하세요.

또한 CloudFront가 PCI DSS 및 SOC 표준을 준수하는 방법에 대한 자세한 내용은 다음을 참조하십시오.

### Payment Card Industry Data Security Standard(PCI DSS)

CloudFront(CloudFront 임베디드 POP를 통한 콘텐츠 전송 제외)에서는 전자 상거래 웹사이트 운영자 또는 서비스 공급자에 의한 신용 카드 데이터의 처리, 저장 및 전송을 지원하며, Payment Card Industry(PCI) Data Security Standard(DSS) 준수를 검증 받았습니다. AWS PCI 규정 준수 패키지의 사본을 요청하는 방법 등 PCI DSS에 대해 자세히 알아보려면 PCI DSS 레벨 1을 참조하세요.

보안을 위해 신용 카드 정보를 CloudFront 엣지 캐시에 캐싱하지 않는 것이 좋습니다. 예를 들어, 신용카드 번호 마지막 4자리와 신용카드 소유자의 연락처 정보와 같은 정보를 담고 있는 응답에 `Cache-Control:no-cache="field-name"` 헤더가 포함되도록 오리진을 구성할 수 있습니다.

### SOC(시스템 및 조직 제어)

CloudFront(CloudFront 임베디드 POP를 통한 콘텐츠 전송 제외)는 SOC 1, SOC 2 및 SOC 3을 포함한 시스템 및 조직 제어(SOC) 측정값을 준수합니다. SOC 보고서는 AWS가 주요 규정 준수 제어 항목 및 제어 대상을 어떻게 준수하고 있는지를 입증하는 독립적인 타사 심사 보고서입니다. 이러한 감사는 고객 및 회사 데이터의 보안, 기밀성 및 가용성에 영향을 미칠 수 있는 위험으로부터 보호하기 위해 적절한 보호 및 절차가 시행되고 있는지 확인합니다. 이러한 타사 감사의 결과는 AWS SOC 규정 준수 웹사이트에서 사용할 수 있습니다. 이 사이트에서는 게시된 보고서를 검토하여 AWS 작업 및 규정 준수를 지원하는 제어에 대한 추가 정보를 확인할 수 있습니다.

## Amazon CloudFront의 복원성

AWS 글로벌 인프라는 AWS리전 및 가용 영역을 중심으로 구축됩니다. AWS 리전은 물리적으로 분리되고 격리된 다수의 가용 리전을 제공하며 이러한 가용 리전은 짧은 지연 시간, 높은 처리량 및 높은 중복성을 갖춘 네트워크에 연결되어 있습니다. 가용 영역을 사용하면 중단 없이 가용 영역 간에 자동으로 장애 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 내용은 AWS 글로벌 인프라를 참조하십시오.

## CloudFront 오리진 장애 조치

Amazon CloudFront는 AWS 글로벌 인프라를 지원할 뿐만 아니라 오리진 장애 조치 기능을 제공하여 데이터 복원성 요구 사항을 지원합니다. CloudFront는 엣지 로케이션 또는 POP(Point of Presence)라고 하는 데이터 센터의 전 세계 네트워크를 통해 콘텐츠를 제공하는 글로벌 서비스입니다. 콘텐츠가 엣지 로케이션에 캐싱되어 있지 않은 경우 CloudFront는 최종 버전의 콘텐츠 소스로 식별한 오리진에서 해당 콘텐츠를 검색합니다.

오리진 장애 조치로 CloudFront를 설정하면 특정 시나리오의 복원력을 향상시키고 가용성을 높일 수 있습니다. 시작하려면 CloudFront의 기본 오리진과 두 번째 오리진을 지정하는 오리진 그룹을 생성합니다. 기본 오리진이 특정 HTTP 상태 코드 실패 응답을 반환하면 CloudFront가 자동으로 두 번째 오리진으로 전환됩니다. 자세한 내용은 CloudFront 오리진 장애 조치를 통한 고가용성 최적화 단원을 참조하십시오.

# Amazon CloudFront의 인프라 보안

관리형 서비스인 Amazon CloudFront는 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스와 AWS의 인프라 보호 방법에 대한 자세한 내용은 AWS Cloud Security을 참조하세요. 인프라 보안에 대한 모범 사례를 사용하여 AWS 환경을 설계하려면 보안 원칙 AWS Well-Architected Framework의 인프라 보호를 참조하세요.

AWS에서 게시한 API 직접 호출을 사용하여 네트워크를 통해 CloudFront에 액세스합니다. 고객은 다음을 지원해야 합니다.

- Transport Layer Security(TLS) TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군 Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

CloudFront Functions는 AWS 계정 간에 매우 안전한 격리 장벽을 사용하므로 고객 환경이 Spectre 및 Meltdown과 같은 부채널 공격으로부터 안전하게 보호됩니다. 함수는 다른 고객에게 속한 데이터에 액세스하거나 이를 수정할 수 없습니다. 함수는 하이퍼스레딩 없이 전용 CPU에서 전용 단일 스레드 프로세스에서 실행됩니다. 지정된 모든 CloudFront 엣지 로케이션 POP(Point of Presence)에서 CloudFront 함수는 한 번에 한 명의 고객에게만 응답하고 모든 고객별 데이터는 함수 실행 사이에 지워집니다.

# 문제 해결

Amazon CloudFront를 설정하여 콘텐츠를 배포할 때 발생할 수 있는 일반적인 문제를 이 섹션을 사용하여 해결하세요.

각 주제는 일반적인 문제의 근본 원인을 식별하기 위한 자세한 지침과 이를 해결하기 위한 단계별 지침을 제공합니다.

주제


- 배포 문제 해결
- CloudFront의 오류 응답 상태 코드 문제 해결
- CloudFront 로드 테스트


## 배포 문제 해결

이 문서의 정보를 사용하여 Amazon CloudFront 배포로 웹 사이트 또는 애플리케이션 설정 시 발생할 수 있는 인증서 오류, 액세스 거부 문제 또는 기타 일반적인 문제를 진단하고 해결할 수 있습니다.

주제


- CloudFront에서 Access Denied 오류가 반환되는 경우
- 대체 도메인 이름을 추가하려고 하면 CloudFront에서 InvalidViewerCertificate 오류가 반환됩니다.
- 새 CNAME을 추가하려고 하면 CloudFront에서 잘못 구성된 DNS 레코드 오류가 반환됩니다.
- 배포에서 파일을 볼 수 없습니다.
- 오류 메시지: Certificate: <certificate-id> is being used by CloudFront


### CloudFront에서 Access Denied 오류가 반환되는 경우

Amazon S3 버킷을 CloudFront 배포의 오리진으로 사용하는 경우 다음 예시에서 액세스 거부됨(403) 오류 메시지가 표시될 수 있습니다.

목차


- Amazon S3 오리진에서 누락된 객체를 지정함
- Amazon S3 오리진에 IAM 권한이 누락됨
- 잘못된 자격 증명을 사용하고 있거나 충분한 권한이 없음

## Amazon S3 오리진에서 누락된 객체를 지정함

요청된 객체가 버킷에 존재하는지 확인하세요. 객체 이름은 대/소문자를 구분합니다. 잘못된 객체 이름을 입력하면 액세스 거부됨 오류 코드가 반환될 수 있습니다.

예를 들어 CloudFront 자습서에 따라 기본 배포를 생성하는 경우 Amazon S3 버킷을 오리진으로 생성하고 예시 `index.html` 파일을 업로드합니다.

웹 브라우저에서 `https://d111111abcdef8.cloudfront.net/index.html` 대신 `https://d111111abcdef8.cloudfront.net/INDEX.HTML`을 입력하면 URL 경로의 `index.html` 파일이 대/소문자를 구분하므로 비슷한 메시지가 표시될 수 있습니다.

```
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestId>22Q367AHT7Y1ABCD</RequestId>
<HostId>
ABCDE/Vg+7PSNa/d/IfFQ8Fb92TGQ0KH0ZwG5iEKbc6+e06DdMS1ZW+ryB9GFRIVtS66rSSy6So=
</HostId>
</Error>
```

## Amazon S3 오리진에 IAM 권한이 누락됨

올바른 Amazon S3 버킷을 오리진 도메인과 이름으로 선택했는지 확인하세요. 오리진(Amazon S3)에 올바른 권한이 있어야 합니다.

올바른 권한을 지정하지 않으면 뷰어에게 `AccessDenied` 메시지가 표시될 수 있습니다.

Amazon S3에서 콘텐츠를 배포하고 AWS Key Management Service(AWS KMS) 서비스 측 암호화(SSE-KMS)도 사용하는 경우 KMS 키와 Amazon S3 버킷에 대해 추가 IAM 권한을 지정해야 합니다. CloudFront 배포에서 오리진 Amazon S3 버킷의 암호화에 사용되는 KMS 키를 사용하려면 이러한 권한이 필요합니다.

Amazon S3 버킷 정책의 구성을 통해 CloudFront 배포가 콘텐츠 전송을 위해 암호화된 객체를 검색할 수 있습니다.

Amazon S3 버킷 및 KMS 키 권한을 확인하려면

1. 사용 중인 KMS 키가 Amazon S3 버킷이 기본 암호화에 사용하는 것과 동일한 키인지 확인하세요. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 AWS KMS(SSE-KMS)로 서버 측 암호화 지정을 참조하세요.

2. 버킷의 객체가 동일한 KMS 키로 암호화되어 있는지 확인합니다. Amazon S3 버킷에서 원하는 객체를 선택하고 서버 측 암호화 설정을 확인하여 KMS 키 ARN을 확인할 수 있습니다.
3. Amazon S3 버킷에서 `GetObject` API 작업을 직접적으로 호출할 권한을 CloudFront에 부여하도록 Amazon S3 버킷 정책을 편집합니다. 오리진 액세스 제어를 사용하는 Amazon S3 버킷 정책의 예시는 CloudFront에 S3 버킷에 액세스할 수 있는 권한 부여 섹션을 참조하세요.
4. `Encrypt`, `Decrypt` 및 `GenerateDataKey*`에 작업을 수행할 권한을 CloudFront에 부여하도록 KMS 키 정책을 편집합니다. 최소 권한에 맞추려면 지정된 CloudFront 배포만 나열된 작업을 수행할 수 있도록 `Condition` 요소를 지정하세요. 기존 AWS KMS 정책에 맞게 정책을 사용자 지정할 수 있습니다. 예시 KMS 키 정책은 SSE-KMS 섹션을 참조하세요.

OAC 대신 오리진 액세스 ID(OAI)를 사용하는 경우 Amazon S3 버킷에 대한 권한은 약간 다릅니다. AWS 서비스 대신 ID에 권한을 부여하기 때문입니다. 자세한 내용은 오리진 액세스 ID에 Amazon S3 버킷의 파일을 읽을 수 있는 권한 부여 섹션을 참조하세요.

배포에 있는 파일을 여전히 볼 수 없는 경우 배포에서 파일을 볼 수 없습니다. 섹션을 참조하세요.

### 잘못된 자격 증명을 사용하고 있거나 충분한 권한이 없음

올바르지 않거나 만료된 AWS SCT 자격 증명(액세스 키 및 비밀 키)을 사용하거나 IAM 역할 또는 사용자에게 CloudFront 리소스에서 작업을 수행하는 데 필요한 권한이 없는 경우 액세스 거부됨 오류 메시지가 나타날 수 있습니다. 액세스 거부됨 오류 메시지에 대한 자세한 내용은 IAM 사용 설명서의 액세스 거부됨 오류 메시지 문제 해결을 참조하세요.

CloudFront에서 IAM이 작동하는 방식에 대한 자세한 내용은 Amazon CloudFront용 Identity and Access Management 섹션을 참조하세요.

## 대체 도메인 이름을 추가하려고 하면 CloudFront에서 InvalidViewerCertificate 오류가 반환됩니다.

배포에 대체 도메인 이름(CNAME)을 추가하려고 하면 CloudFront 에서 `InvalidViewerCertificate` 오류를 반환하는 경우, 다음 정보를 검토하여 문제를 해결하십시오. 이 오류는 대체 도메인 이름을 추가하려면 먼저 다음 문제 중 하나를 해결해야 함을 뜻할 수 있습니다.

다음 오류는 CloudFront에서 대체 도메인 이름을 추가할 권한이 있는지 확인하는 순서대로 나열한 것입니다. 이는 CloudFront가 반환하는 오류를 기반으로 하여 어떤 확인 검사가 성공적으로 완료되었는지 알 수 있으므로 문제를 해결하는 데 도움이 됩니다.

배포에 연결된 인증서가 없습니다.

대체 도메인 이름(CNAME)을 추가하려면 배포에 신뢰할 수 있는 유효한 인증서를 연결해야 합니다. 요구 사항을 검토하고 요구 사항을 충족하는 유효한 인증서를 얻고 배포에 연결한 후 다시 시도하십시오. 자세한 내용은 대체 도메인 이름 사용과 관련된 요구 사항 단원을 참조하세요.

연결한 인증서에 대한 인증서 체인에 너무 많은 인증서가 있습니다.

인증서 체인 하나에 최대 5개의 인증서만 포함할 수 있습니다. 체인의 인증서 수를 줄인 후 다시 시도하십시오.

인증서 체인에 현재 날짜에 유효하지 않은 인증서가 하나 이상 포함되어 있습니다.

추가한 인증서의 인증서 체인에 인증서가 아직 유효하지 않거나 인증서가 만료된 이유로 유효하지 않은 인증서가 하나 이상 있습니다. 인증서 체인의 인증서에 있는 Not Valid Before 및 Not Valid After 필드를 확인하여 모든 인증서가 나열한 날짜를 기반으로 유효한지 확인합니다.

연결한 인증서가 신뢰할 수 있는 인증 기관(CA)에서 서명되지 않았습니다.

대체 도메인 이름을 확인하기 위해 CloudFront에 연결하는 인증서는 자체 서명된 인증서일 수 없습니다. 신뢰할 수 있는 CA에서 서명해야 합니다. 자세한 내용은 대체 도메인 이름 사용과 관련된 요구 사항 단원을 참조하세요.

연결한 인증서가 올바른 형식이 아닙니다.

인증서에 포함된 도메인 이름 및 IP 주소 형식과, 인증서 자체의 형식이 인증서 표준을 따라야 합니다.

CloudFront 내부 오류가 발생했습니다.

CloudFront가 내부 문제로 차단되어 인증서를 검사할 수 없습니다. 이 시나리오에서 CloudFront는 HTTP 500 상태 코드를 반환하고 인증서 연결과 관련된 내부 CloudFront 문제가 있음을 보여 줍니다. 몇 분 정도 기다린 후 인증서와 함께 대체 도메인 이름을 다시 추가해 보십시오.

연결한 인증서가 추가하려는 대체 도메인 이름을 포함하지 않습니다.

CloudFront는 사용자가 추가하는 대체 도메인 이름별로 해당 도메인 이름이 포함된 신뢰할 수 있는 인증 기관(CA)의 유효한 SSL/TLS 인증서를 연결하여 사용 권한을 검증할 것을 요구합니다. 추가하려는 CNAME이 포함된 도메인 이름을 포함하도록 인증서를 업데이트합니다. 와일드카드가 있는 도메인 이름 사용에 대한 자세한 내용과 예는 대체 도메인 이름 사용과 관련된 요구 사항 단원을 참조하세요.

# 새 CNAME을 추가하려고 하면 CloudFront에서 잘못 구성된 DNS 레코드 오류가 반환됩니다.

CloudFront 배포를 가리키는 기존 와일드카드 DNS 항목이 있는 경우 보다 구체적인 이름을 가진 새 CNAME을 추가하려고 하면 다음과 같은 오류가 발생할 수 있습니다.

```
One or more aliases specified for the distribution includes an incorrectly configured
 DNS record that points to another CloudFront distribution. You must update the DNS
 record to correct the problem.
```

이 오류는 CloudFront가 CNAME에 대해 DNS를 쿼리하고 와일드카드 DNS 항목이 다른 배포로 확인되기 때문에 발생합니다.

이 문제를 해결하려면 먼저 다른 배포를 생성한 다음 새 배포를 가리키는 DNS 항목을 생성합니다. 마지막으로 더 구체적인 CNAME을 추가합니다. CNAME를 추가하는 방법에 대한 자세한 내용은 대체 도메인 이름 사용 섹션을 참조하세요.

# 배포에서 파일을 볼 수 없습니다.

CloudFron 배포에서 파일을 볼 수 없는 경우, 몇 가지 일반적인 해결 방법은 다음 항목을 참조하십시오.

## CloudFront와 Amazon S3에 모두 가입했습니까?

Amazon S3 오리진에서 Amazon CloudFront를 사용하려면 CloudFront와 Amazon S3에 각각 별도로 가입해야 합니다. CloudFront 및 Amazon S3 가입에 대한 자세한 내용은 AWS 계정 설정 단원을 참조하십시오.

## Amazon S3 버킷과 객체 권한을 올바르게 설정했습니까?

Amazon S3 오리진에서 CloudFront를 사용하는 경우 콘텐츠의 원래 버전은 S3 버킷에 저장됩니다. 뷰어에게 콘텐츠를 제공하려면 CloudFront 오리진 액세스 제어(OAC)를 사용하여 Amazon S3 버킷 액세스를 보호하는 것이 좋습니다. 즉, S3 버킷은 CloudFront를 통해서만 연결할 수 있습니다. OAC는 CloudFront를 통한 뷰어 액세스 및 보안 전송을 제어합니다. OAC에 대한 자세한 내용은 the section called “Amazon S3 오리진에 대한 액세스 제한”을 참조합니다.

버킷 액세스를 관리하는 방법에 대한 자세한 내용은 Amazon S3 사용 설명서의 Amazon S3 스토리지에 대한 퍼블릭 액세스 차단을 참조하세요.

객체 속성과 버킷 속성은 독립적입니다. Amazon S3의 각 객체에 명시적으로 권한을 부여해야 합니다. 객체는 버킷에서 속성을 상속하지 않으므로 객체 속성을 버킷과 별도로 설정해야 합니다.

## 대체 도메인 이름(CNAME)을 올바르게 구성했습니까?

도메인 이름에 대한 기존 CNAME 레코드가 이미 있는 경우 해당 레코드를 업데이트하거나 배포의 도메인 이름을 가리키는 새 리소스 레코드 세트로 바꿉니다.

또한 CNAME 레코드가 Amazon S3 버킷이 아니라 배포의 도메인 이름을 가리키는지 확인하십시오. DNS 시스템의 CNAME 레코드가 배포의 도메인 이름을 가리키는지 확인할 수 있습니다. 이렇게 하려면 dig와 같은 DNS 도구를 사용합니다.

다음 예는 `images.example.com`이라고 하는 도메인 이름에 대한 dig 요청과 관련 응답 부분을 보여줍니다. `ANSWER SECTION` 아래에서 `CNAME`를 포함하는 줄을 참조하십시오. CNAME 오른쪽에 있는 값이 CloudFront 배포의 도메인 이름이면 도메인 이름의 CNAME 레코드가 올바르게 설정된 것입니다. 이 값이 Amazon S3 오리진 서버 버킷이나 다른 도메인 이름이면 CNAME 레코드가 잘못 설정된 것입니다.

```
[prompt]> dig images.example.com

; <<> DiG 9.3.3rc2 <<> images.example.com
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15917
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 2, ADDITIONAL: 0
;; QUESTION SECTION:
;images.example.com.   IN  A
;; ANSWER SECTION:
images.example.com. 10800 IN CNAME d111111abcdef8.cloudfront.net.
...
...
```

CNAME에 대한 자세한 내용은 대체 도메인 이름(CNAME)을 추가하여 사용자 지정 URL 사용을 참조하십시오.

## CloudFront 배포에 대한 올바른 URL을 참조하고 있습니까?

참조하고 있는 URL에 Amazon S3 버킷 또는 사용자 지정 오리진이 아닌 CloudFront 배포 도메인 이름(또는 CNAME)이 사용되는지 확인하십시오.

### 사용자 지정 오리진 문제 해결과 관련하여 도움이 필요합니까?

사용자 지정 오리진 문제를 해결하는 데 AWS의 도움이 필요한 경우 아마도 요청의 `X-Amz-Cf-Id` 헤더 항목을 검사해야 할 것입니다. 이러한 항목을 로그에 기록하고 있지 않은 경우 향후 참조를 위해 로그에 기록할 수 있습니다. 자세한 내용은 the section called “Amazon EC2(또는 기타 사용자 지정 오리진) 사용” 섹션을 참조하세요. 자세한 도움말은 AWS Support Center를 참조하세요.

## 오류 메시지: Certificate: <certificate-id> is being used by CloudFront

문제: IAM 인증서 스토어에서 SSL/TLS 인증서를 삭제하려고 할 때 "Certificate: <certificate-id> is being used by CloudFront." 메시지가 표시됩니다.

해결 방법: 모든 CloudFront 배포를 기본 CloudFront 인증서 또는 사용자 지정 SSL/TLS 인증서와 연결해야 합니다. SSL/TLS 인증서를 삭제하기 전에 이 인증서를 교체하거나(현재 사용자 지정 SSL/TLS 인증서를 다른 사용자 지정 SSL/TLS 인증서로 바꾸기) 사용자 지정 SSL/TLS 인증서 사용을 기본 CloudFront 인증서 사용으로 되돌려야 합니다. 이를 해결하려면 다음 절차 중 하나의 단계를 수행합니다.

- SSL/TLS 인증서 교체
- 사용자 지정 SSL/TLS 인증서에서 기본 CloudFront 인증서로 되돌리기

# CloudFront의 오류 응답 상태 코드 문제 해결

CloudFront가 오리진에 객체를 요청했는데 오리진에서 HTTP 4xx 또는 5xx 상태 코드를 반환하는 경우, CloudFront와 오리진 사이에 통신 문제가 있는 것입니다.

이 주제에는 Lambda@Edge 또는 CloudFront Functions를 사용할 때 이러한 상태 코드에 대한 문제 해결 단계도 포함되어 있습니다.

다음 주제에서는 이러한 오류 응답의 이면에 있는 잠재적 원인을 자세히 설명하고 근본적인 문제를 진단하고 해결하는 방법에 대한 단계별 지침을 제공합니다.

주제


- HTTP 400 상태 코드(잘못된 요청)
- HTTP 401 상태 코드(승인되지 않음)
- HTTP 403 상태 코드(잘못된 메서드)
- HTTP 403 상태 코드(권한 거부됨)
- HTTP 404 상태 코드(찾을 수 없음)

- HTTP 412 상태 코드(사전 조건 실패)
- HTTP 500 상태 코드(내부 서버 오류)
- HTTP 502 상태 코드(잘못된 게이트웨이)
- HTTP 503 상태 코드(Service Unavailable)
- HTTP 504 상태 코드(게이트웨이 제한 시간)

# HTTP 400 상태 코드(잘못된 요청)

CloudFront는 클라이언트가 요청에서 일부 유효하지 않은 데이터(예: 페이로드 또는 파라미터의 누락되거나 잘못된 콘텐츠)를 전송할 때 400 잘못된 요청을 반환합니다. 이는 일반적인 클라이언트 오류일 수도 있습니다.

## Amazon S3 오리진에서 400 오류를 반환함

CloudFront 배포와 함께 Amazon S3 오리진을 사용하는 경우, 배포가 HTTP 상태 코드 400 잘못된 요청 및 다음과 유사한 메시지와 함께 오류 응답을 보낼 수 있습니다.

권한 부여 헤더 형식이 잘못되었습니다. 리전 '*<AWS ##>*'이 잘못되었습니다. '*<AWS ##>*'이어야 합니다.

예:

권한 부여 헤더 형식이 잘못되었습니다. 리전 'us-east-1'이 잘못되었습니다. 'us-west-2'여야 합니다.

다음과 같은 시나리오에서 이 문제가 발생할 수 있습니다.

1. CloudFront 배포의 오리진은 Amazon S3 버킷입니다.
2. S3 버킷을 특정 AWS 리전에서 다른 리전으로 이전했습니다. 즉, S3 버킷을 삭제한 후 나중에 동일한 버킷 이름을 가진 새 버킷을 만들었지만 원래 S3 버킷이 있던 AWS 리전과 다릅니다.

이 오류를 수정하려면 버킷의 현재 AWS 리전에서 S3 버킷을 찾도록 CloudFront 배포를 업데이트하세요.

CloudFront 배포를 업데이트하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/cloudfront/v4/home에서 CloudFront 콘솔을 엽니다.
2. 이 오류를 일으키는 배포를 선택합니다.

3. Origins and Origin Groups(오리진 및 오리진 그룹)를 선택합니다.
4. 이동한 S3 버킷의 오리진을 찾습니다. 이 오리진 옆의 확인란을 선택한 후 편집을 선택합니다.
5. 예, 편집합니다를 선택합니다. Yes, Edit(예, 편집합니다.)을 선택하기 전 설정을 변경할 필요가 없습니다.

이 단계를 끝내면 CloudFront가 배포를 다시 배포합니다. 배포가 진행되는 동안 마지막 수정일 열 아래에 배포 중 상태가 표시됩니다. 배포가 완료된 후 일정 시간이 지나면 `AuthorizationHeaderMalformed` 오류 응답을 더 이상 받지 않아야 합니다.

### Application Load Balancer 오리진이 400 오류를 반환함

Application Load Balancer 오리진을 CloudFront 배포와 함께 사용하는 경우, 400 오류의 가능한 원인은 다음과 같습니다.

- 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 상태 코드(승인되지 않음)

401 승인되지 않음 응답 상태 코드는 요청된 리소스에 대한 유효한 인증 자격 증명이 없기 때문에 클라이언트 요청이 완료되지 않았음을 나타냅니다. 이 상태 코드는 클라이언트가 사용자에게 인증 자격 증명을 요청한 후 리소스를 다시 요청할 수 있는 방법에 대한 정보가 포함된 HTTP `WWW-Authenticate` 응답 헤더와 함께 전송됩니다. 자세한 내용은 401 Unauthorized를 참조하세요.

CloudFront에서 오리진이 요청을 인증하기 위한 `Authorization` 헤더를 기대하는 경우 CloudFront는 `Authorization` 헤더를 오리진에 전달해야 401 승인되지 않음 오류가 발생하지 않습니다. CloudFront에서 최종 사용자 요청을 오리진에 전달할 때 `Authorization` 헤더를 포함한 일부 최종 사용자 헤더는 기본적으로 제거됩니다. 오리진 요청의 `Authorization` 헤더가 오리진에 항상 수신되도록 하려면 다음 옵션을 선택할 수 있습니다.

- 캐시 정책을 사용하여 캐시 키에 `Authorization` 헤더를 추가합니다. 캐시 키의 모든 헤더가 오리진 요청에 자동으로 포함됩니다. 자세한 내용은 정책으로 캐시 키 제어 섹션을 참조하세요.
- 모든 최종 사용자 헤더를 오리진으로 전달하는 오리진 요청 정책을 사용합니다. 오리진 요청 정책에서 `Authorization` 헤더를 개별적으로 전달할 수는 없지만 모든 뷰어 헤더를 전달하면 CloudFront에서 뷰어 요청에 `Authorization` 헤더를 포함합니다. CloudFront는 이 사용 사례에 `AllViewer`

관리형 오리진 요청 정책을 제공합니다. 자세한 내용은 관리형 오리진 요청 정책 사용 섹션을 참조하세요.

자세한 내용은 권한 부여 헤더를 오리진에 전달하도록 CloudFront를 구성하려면 어떻게 해야 하나요?를 참조하세요.

## HTTP 403 상태 코드(잘못된 메서드)

CloudFront 배포에서 지정하지 않은 HTTP 메서드를 사용하려고 하면 CloudFront는 403(잘못된 메서드) 오류를 반환합니다. 배포에 대해 다음 옵션 중 하나를 지정할 수 있습니다.

- CloudFront가 `GET` 및 `HEAD` 요청만 전달합니다.
- CloudFront가 `GET`, `HEAD`, `OPTIONS` 요청만 전달합니다.
- CloudFront가 `GET`, `HEAD`, `OPTIONS`, `PUT`, `PATCH`, `POST`, `DELETE` 요청을 전달합니다. 이 옵션을 선택할 경우, 허용하지 않을 작업을 사용자가 수행하지 못하도록 Amazon S3 버킷 또는 사용자 지정 오리진에 대한 액세스를 제한해야 할 수 있습니다. 예를 들어, 사용자가 오리진에서 객체를 삭제할 수 있는 권한을 사용자에게 허용하지 않을 수 있습니다.

## HTTP 403 상태 코드(권한 거부됨)

HTTP 403 오류는 클라이언트가 요청된 리소스에 액세스할 권한이 없음을 의미합니다. 클라이언트는 요청을 이해하지만 뷰어 액세스를 승인할 수는 없습니다. 다음은 CloudFront에서 이 상태 코드를 반환하는 일반적인 원인입니다.

주제

- 대체 CNAME이 잘못 구성됨
- AWS WAF가 CloudFront 배포 또는 오리진에서 구성됨
- 사용자 지정 오리진이 403 오류를 반환함
- Amazon S3 오리진에서 403 오류를 반환함
- 지리적 제한으로 인해 403 오류가 반환됨
- 서명된 URL 또는 서명된 쿠키 구성이 403 오류를 반환함
- 누적 배포로 인해 403 오류가 발생함

## 대체 CNAME이 잘못 구성됨

배포에 올바른 CNAME을 지정했는지 확인하세요. 기본 CloudFront URL 대신 대체 CNAME을 사용하려면 다음 단계를 따르세요.

1. DNS에 CNAME 레코드를 생성하여 CNAME이 CloudFront 배포 URL을 가리키도록 합니다.
2. CloudFront 배포 구성에 CNAME을 추가합니다.

DNS 레코드를 생성하지만 CloudFront 배포 구성에 CNAME을 추가하지 않는 경우 요청은 403 오류를 반환합니다. 고객 CNAME 구성에 대한 자세한 정보는 대체 도메인 이름(CNAME)을 추가하여 사용자 지정 URL 사용 섹션을 참조하세요.

## AWS WAF가 CloudFront 배포 또는 오리진에서 구성됨

AWS WAF가 클라이언트와 CloudFront 사이에 위치하는 경우, 오리진에서 반환하는 403 오류 코드와 요청이 차단될 때 AWS WAF에서 반환하는 403 오류 코드를 구별할 수 없습니다.

403 상태 코드의 출처를 찾으려면 AWS WAF 웹 액세스 제어 목록(ACL) 규칙에서 차단된 요청을 확인하세요. 자세한 내용은 다음 항목을 참조하세요.

- AWS WAF 웹 액세스 제어 목록(웹 ACL)
- AWS WAF 보호 기능 테스트 및 튜닝

## 사용자 지정 오리진이 403 오류를 반환함

사용자 지정 오리진을 사용하는 경우 오리진에 사용자 지정 방화벽을 구성하면 403 오류가 표시될 수 있습니다. 문제를 해결하려면 오리진에 직접 요청하세요. CloudFront 없이 오류를 재현할 수 있다면 오리진에서 403 오류가 발생하는 것입니다.

사용자 지정 오리진으로 인해 오류가 발생한 경우 오리진 로그를 확인하여 오류의 원인을 파악하세요. 자세한 내용은 다음 문제 해결 주제를 참조하세요.

- API Gateway의 HTTP 403 오류를 해결하려면 어떻게 해야 하나요?
- Application Load Balancer HTTP 403 금지됨 오류 문제는 어떻게 해결하나요?

## Amazon S3 오리진에서 403 오류를 반환함

다음과 같은 이유로 403 오류가 발생할 수 있습니다.

- CloudFront에 Amazon S3 버킷에 대한 액세스 권한이 없습니다. 배포에 오리진 액세스 ID(OAI) 또는 오리진 액세스 제어(OAC)가 활성화되지 않았고 버킷이 프라이빗인 경우 이런 상황이 발생할 수 있습니다.
- 요청된 URL의 지정된 경로가 올바르지 않습니다.
- 요청된 객체가 존재하지 않습니다.
- 호스트 헤더가 REST API 엔드포인트와 함께 전달되었습니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 HTTP Host header bucket specification을 참조하세요.
- 사용자 지정 오류 페이지를 구성했습니다. 자세한 내용은 사용자 지정 오류 페이지를 구성했을 때 CloudFront에서 오류를 처리하는 방법 섹션을 참조하세요.

### 지리적 제한으로 인해 403 오류가 반환됨

특정 지리적 위치에 있는 사용자가 CloudFront 배포를 통해 배포한 콘텐츠에 액세스하는 것을 차단하기 위해 지리적 차단이라고도 하는 지리적 제한을 활성화한 경우, 차단된 사용자에게 403 오류가 발생합니다.

자세한 내용은 콘텐츠의 지리적 배포 제한 섹션을 참조하세요.

### 서명된 URL 또는 서명된 쿠키 구성이 403 오류를 반환함

배포의 동작 구성에 대해 뷰어 액세스 제한을 활성화한 경우, 서명된 쿠키 또는 서명된 URL을 사용하지 않는 요청에 대해 403 오류가 발생합니다. 자세한 내용은 다음 항목을 참조하세요.

- 서명된 URL과 서명된 쿠키를 사용하여 프라이빗 콘텐츠 제공
- CloudFront의 서명된 URL 또는 서명된 쿠키와 관련된 문제를 해결하려면 어떻게 해야 하나요?

### 누적 배포로 인해 403 오류가 발생함

오리진 엔드포인트에 대한 요청 체인 내에 두 개 이상의 배포가 있는 경우 CloudFront는 403 오류를 반환합니다. 한 배포를 다른 배포 앞에 배치하지 않는 것이 좋습니다.

## HTTP 404 상태 코드(찾을 수 없음)

클라이언트가 존재하지 않는 리소스에 액세스를 시도하면 CloudFront는 404(찾을 수 없음) 오류를 반환합니다. CloudFront 배포에서 이 오류가 발생하는 경우 일반적인 원인은 다음과 같습니다.

- 리소스가 존재하지 않습니다.
- URL이 올바르지 않습니다.

- 사용자 지정 오리진이 404를 반환합니다.
- 사용자 지정 오류 페이지가 404를 반환합니다. (어떤 오류 코드든 404로 변환될 수 있습니다.) 자세한 내용은 사용자 지정 오류 페이지를 구성했을 때 CloudFront에서 오류를 처리하는 방법 섹션을 참조하세요.
- 사용자 지정 오류 페이지가 실수로 삭제되어 요청이 삭제된 사용자 지정 오류 페이지를 찾기 때문에 404가 발생합니다. 자세한 내용은 사용자 지정 오류 페이지를 구성하지 않은 경우 CloudFront에서 오류를 처리하는 방법 섹션을 참조하세요.
- 오리진 경로가 잘못되었습니다. 오리진 경로를 입력하면 요청이 오리진에 전달되기 전에 브라우저의 각 요청 경로에 오리진 값이 추가됩니다. 자세한 내용은 오리진 경로 섹션을 참조하세요.

## HTTP 412 상태 코드(사전 조건 실패)

대상 리소스에 대한 액세스가 거부되면 CloudFront는 412(사전 조건 실패) 오류 코드를 반환합니다. 경우에 따라 서버는 특정 조건이 충족된 후에만 요청을 수락하도록 구성되어 있습니다. 지정된 조건 중 하나라도 충족되지 않으면 서버는 클라이언트가 지정된 리소스에 액세스하는 것을 허용하지 않습니다. 대신 서버는 412 오류 코드로 응답합니다.

CloudFront에서 발생하는 412 오류의 일반적인 원인은 다음과 같습니다.

- `If-Unmodified-Since` 또는 `If-None-Match` 헤더로 정의된 조건이 충족되지 않을 때 `GET` 또는 `HEAD` 외 다른 메서드에 대한 조건부 요청. 이 경우 요청(일반적으로 리소스 업로드 또는 수정)을 수행할 수 없습니다.
- CloudFront UpdateDistribution API 작업의 요청 필드 중 하나 이상에 있는 조건이 거짓으로 평가됨

## HTTP 500 상태 코드(내부 서버 오류)

HTTP 500 상태 코드(내부 서버 오류)는 서버가 요청을 이행하지 못하게 하는 예기치 않은 상황이 발생했음을 나타냅니다. Amazon CloudFront에서 500 오류가 발생하는 일반적인 원인은 다음과 같습니다.

주제

- 오리진 서버가 CloudFront에 500 오류를 반환함

### 오리진 서버가 CloudFront에 500 오류를 반환함

오리진 서버가 CloudFront에 500 오류를 반환하고 있을 수 있습니다. 자세한 내용은 다음 문제 해결 주제를 참조하세요.

- Amazon S3에서 500 오류를 반환하는 경우, Amazon S3에서 발생하는 HTTP 500 또는 503 오류 문제를 해결하려면 어떻게 해야 하나요?를 참조하세요.
- API Gateway에서 500 오류를 반환하는 경우, API Gateway의 5xx 오류를 해결하려면 어떻게 해야 하나요?를 참조하세요.
- Elastic Load Balancing에서 500 오류를 반환하는 경우, 애플리케이션 로드 밸런서 사용 설명서의 HTTP 500: Internal server error를 참조하세요.

위 목록의 방법으로 500 오류가 해결되지 않으면 CloudFront 상호 접속 지점(POP)에서 내부 서버 오류를 반환하는 문제일 수 있습니다. 지원에 지원을 문의할 수 있습니다.

## HTTP 502 상태 코드(잘못된 게이트웨이)

CloudFront는 오리진 서버에 연결할 수 없기 때문에 요청된 객체를 제공할 수 없을 때 HTTP 502 상태 코드(잘못된 게이트웨이)를 반환합니다.

Lambda@Edge를 사용하는 경우 Lambda 검증 오류가 문제일 수 있습니다. `NonS3OriginDnsError` 오류 코드와 함께 HTTP 502 오류가 발생하는 경우 CloudFront가 오리진에 연결할 수 없는 DNS 구성 문제가 있을 가능성이 있습니다.

주제

- CloudFront와 사용자 지정 원본 서버 간의 SSL/TLS 협상 실패
- 오리진이 지원되는 암호화/프로토콜에 응답하지 않음
- 오리진의 SSL/TLS 인증서가 만료되었거나 잘못되었거나 자체 서명되었거나, 인증서 체인의 순서가 잘못됨
- 오리진이 오리진 설정의 지정된 포트에서 응답하지 않음
- Lambda 검증 오류
- CloudFront Functions 검증 오류
- DNS 오류(NonS3OriginDnsError)
- Application Load Balancer 오리진 502 오류
- API Gateway 오리진 502 오류

### CloudFront와 사용자 지정 원본 서버 간의 SSL/TLS 협상 실패

CloudFront와 오리진 사이에 HTTPS가 위치하도록 요구하는 사용자 지정 오리진을 사용하는 경우, 도메인 이름 불일치로 인해 오류가 발생할 수 있습니다. 오리진의 SSL/TLS 인증서에는 CloudFront 배포

용으로 지정한 오리진 도메인 또는 오리진 요청의 `Host` 헤더와 일치하는 도메인 이름이 포함되어야 합니다.

도메인 이름이 일치하지 않으면 SSL/TLS 핸드쉐이크가 실패하고, CloudFront는 HTTP 상태 코드 502(잘못된 게이트웨이)를 반환하고 `X-Cache` 헤더를 `Error from cloudfront`로 설정합니다.

온라인 SSL 확인 프로그램 또는 OpenSSL을 사용하여 인증서의 도메인 이름이 배포의 오리진 도메인 또는 `Host` 헤더와 일치하는지 여부를 파악할 수 있습니다. 도메인 이름이 일치하지 않는 경우, 다음 두 가지 옵션이 있습니다.

- 해당하는 도메인 이름이 포함된 새 SSL/TLS 인증서를 받습니다.

  AWS Certificate Manager(ACM)를 사용하는 경우 새 인증서를 요청하려면 AWS Certificate Manager 사용 설명서에서 퍼블릭 인증서 요청을 참조하세요.
- CloudFront가 더 이상 SSL을 사용하여 오리진 연결을 시도하지 않도록 배포 구성을 변경합니다.

### 온라인 SSL 확인 프로그램

SSL 테스트 도구를 찾으려면 인터넷에서 "online ssl checker"를 검색합니다. 일반적으로, 도메인 이름을 지정하면 SSL/TLS 인증서에 대한 다양한 정보가 도구에 표시됩니다. 인증서의 일반 이름 또는 주제 대체 이름 필드에 도메인 이름이 나와 있는지 확인합니다.

### OpenSSL

CloudFront에서 HTTP 502 오류를 해결하기 위해 OpenSSL을 사용하여 오리진 서버에 SSL/TLS 연결을 시도할 수 있습니다. OpenSSL을 연결할 수 없는 경우 이는 오리진 서버의 SSL/TLS 구성에 문제가 있음을 나타낼 수 있습니다. OpenSSL을 연결할 수 있는 경우 인증서의 일반 이름(`Subject CN` 필드) 및 주체 대체 이름(`Subject Alternative Name` 필드)을 포함하여 오리진 서버의 인증서에 대한 정보를 반환합니다.

다음 OpenSSL 명령을 사용하여 오리진 서버에 대한 연결을 테스트합니다(*`origin domain`*을 example.com과 같은 오리진 서버의 도메인 이름으로 변경).

```
openssl s_client -connect origin domain name:443
```

다음이 참인 경우:

- 오리진 서버는 여러 SSL/TLS 인증서를 사용하여 여러 도메인 이름을 지원합니다.
- 배포가 `Host` 헤더를 오리진에 전달하도록 구성되어 있습니다.

이후 다음 예제와 같이 OpenSSL 명령에 `-servername` 옵션을 추가합니다(*CNAME*을 배포에 구성된 CNAME으로 변경).

```
openssl s_client -connect origin domain name:443 -servername CNAME
```

## 오리진이 지원되는 암호화/프로토콜에 응답하지 않음

CloudFront는 암호화 및 프로토콜을 사용하여 오리진 서버에 연결합니다. CloudFront에서 지원하는 암호화 및 프로토콜 목록은 the section called "CloudFront와 오리진 간에 지원되는 프로토콜 및 암호" 섹션을 참조하세요. 오리진이 SSL/TLS 교환에서 이러한 암호화 또는 프로토콜 중 하나에 응답하지 않는 경우, CloudFront가 연결되지 않은 것입니다. SSL Labs와 같은 온라인 도구를 사용하면, 오리진에서 암호화 및 프로토콜을 지원하는지 확인할 수 있습니다. 호스트 이름 필드에 오리진의 도메인 이름을 입력한 다음 제출을 선택합니다. 테스트에서 일반 이름 및 대체 이름 필드를 검토하여 오리진의 도메인 이름과 일치하는지 확인합니다. 테스트가 완료되면 테스트 결과에서 프로토콜 및 암호 글부 섹션을 찾아 오리진에서 암호화 또는 프로토콜을 지원하는지 확인합니다. 이를 the section called "CloudFront와 오리진 간에 지원되는 프로토콜 및 암호"의 목록과 비교합니다.

## 오리진의 SSL/TLS 인증서가 만료되었거나 잘못되었거나 자체 서명되었거나, 인증서 체인의 순서가 잘못됨

오리진 서버에서 다음을 반환하면, CloudFront는 TCP 연결을 끊고 HTTP 상태 코드 502(잘못된 게이트웨이)를 반환하며, `X-Cache` 헤더를 `Error from cloudfront`로 설정합니다.

- 만료된 인증서
- 잘못된 인증서
- 자체 서명된 인증서
- 인증서 체인의 순서가 잘못됨

> **Note**
>
> 중간 인증서를 포함하여 인증서의 전체 체인이 없는 경우, CloudFront는 TCP 연결을 끊습니다.

사용자 지정 원본 서버에 SSL/TLS 인증서를 설치하는 데 대한 자세한 내용은 the section called "사용자 지정 오리진에 대해 HTTPS 요구" 섹션을 참조하세요.

## 오리진이 오리진 설정의 지정된 포트에서 응답하지 않음

CloudFront 배포에 오리진을 만드는 경우, CloudFront가 HTTP 및 HTTPS 트래픽을 통해 오리진에 연결하는 포트를 설정할 수 있습니다. 기본적으로 TCP 80/443이 있습니다. 이러한 포트를 수정할 수 있습니다. 오리진이 어떤 이유로 이러한 포트의 트래픽을 거부하거나 백엔드 서버가 포트에서 응답하지 않는 경우, CloudFront가 연결되지 않습니다.

이러한 문제를 해결하려면 인프라에서 실행 중인 방화벽을 확인하고, 방화벽이 지원되는 IP 범위를 차단하지 않는지 확인하십시오. 자세한 내용은 Amazon VPC 사용 설명서의 AWS IP 주소 범위를 참조하세요. 또한 웹 서버가 오리진에서 실행되는지 확인하십시오.

## Lambda 검증 오류

Lambda@Edge를 사용하는 경우 HTTP 502 상태 코드는 Lambda 함수 응답이 잘못 구성되어 있거나 잘못된 내용을 포함하고 있음을 나타낼 수 있습니다. Lambda@Edge 오류 문제 해결에 대한 자세한 내용은 Lambda@Edge 함수 테스트 및 디버깅 단원을 참조하십시오.

## CloudFront Functions 검증 오류

CloudFront Functions를 사용하는 경우 HTTP 502 상태 코드는 CloudFront Functions가 읽기 전용 헤더를 추가, 삭제 또는 변경하려고 시도 중임을 나타낼 수 있습니다. 이 오류는 테스트 중에는 표시되지 않지만 함수를 배포하고 요청을 실행한 후에 표시됩니다. 이 오류를 해결하려면 CloudFront Functions를 확인하고 업데이트하세요. 자세한 내용은 함수 업데이트 섹션을 참조하세요.

## DNS 오류(`NonS3OriginDnsError`)

`NonS3OriginDnsError` 오류 코드가 있는 HTTP 502 오류는 CloudFront가 오리진에 연결할 수 없는 DNS 구성 문제가 있음을 나타냅니다. CloudFront에서 이 오류가 발생하는 경우 오리진의 DNS 구성이 올바르고 제대로 작동하는지 확인합니다.

만료되었거나 캐시에 없는 객체에 대한 요청을 수신하면 CloudFront는 객체를 가져오기 위해 오리진에 요청합니다. 성공적으로 오리진에 요청하기 위해 CloudFront는 오리진 도메인에 대해 DNS 확인을 수행합니다. 도메인의 DNS 서비스에 문제가 있는 경우 CloudFront가 도메인 이름을 확인하여 IP 주소를 가져올 수 없어 HTTP 502 오류(`NonS3OriginDnsError`)가 발생합니다. 이 문제를 해결하려면 DNS 공급자에게 문의하세요. 또는 Amazon Route 53을 사용하는 경우 Route 53 DNS 서비스를 사용하는 웹 사이트에 액세스할 수 없는 이유는 무엇입니까?를 참조하세요.

이 문제를 추가로 해결하려면 오리진 루트 도메인 또는 zone apex(예: `example.com`)의 권한 있는 이름 서버가 올바르게 작동 중인지 확인하십시오. dig 또는 nslookup 같은 툴과 함께 다음 명령을 사용하여 apex 오리진의 이름 서버를 찾습니다

```
dig OriginAPEXDomainName NS +short
```

```
nslookup -query=NS OriginAPEXDomainName
```

이름 서버의 이름을 아는 경우 다음 명령을 사용하여 해당 이름에 대해 오리진의 도메인 이름을 쿼리하여 각각 다음 답변으로 응답하는지 확인합니다.

```
dig OriginDomainName @NameServer
```

```
nslookup OriginDomainName NameServer
```

> **Important**
> 퍼블릭 인터넷에 연결된 컴퓨터를 사용하여 이 DNS 문제 해결을 수행해야 합니다. CloudFront는 인터넷의 퍼블릭 DNS를 사용하여 오리진 도메인을 확인하므로 비슷한 컨텍스트에서 문제를 해결하는 것이 중요합니다.

오리진이 루트 도메인이 아닌 다른 이름 서버에 DNS 권한이 위임된 하위 도메인인 경우, 해당 하위 도메인에 대해 이름 서버(NS) 및 권한 시작(SOA) 레코드가 올바르게 구성되었는지 확인합니다. 이전 예와 비슷한 명령을 사용하여 이러한 레코드를 확인할 수 있습니다.

DNS에 대한 자세한 내용은 Amazon Route 53 설명서의 도메인 이름 시스템(DNS) 개념을 참조하세요.

### Application Load Balancer 오리진 502 오류

Application Load Balancer를 오리진으로 사용하고 502 오류가 발생하는 경우 Application Load Balancer HTTP 502 오류를 해결하려면 어떻게 해야 하나요?를 참조하세요.

### API Gateway 오리진 502 오류

API Gateway를 사용하고 502 오류가 발생하는 경우 Lambda 프록시 통합을 사용하는 API Gateway REST API에서 HTTP 502 오류를 해결하려면 어떻게 해야 하나요?를 참조하세요.

## HTTP 503 상태 코드(Service Unavailable)

HTTP 503 상태 코드(Service Unavailable)는 일반적으로 오리진 서버의 성능 문제를 나타냅니다. 드물지만 이 상태 코드가 엣지 로케이션의 리소스 제한 때문에 CloudFront가 일시적으로 요청을 충족할 수 없음을 나타냅니다.

Lambda@Edge 또는 CloudFront Functions를 사용하는 경우 문제는 실행 오류 또는 Lambda@Edge 제한 초과 오류일 수 있습니다.

주제

- Origin server does not have enough capacity to support the request rate
- 엣지 위치에서의 리소스 제약 조건으로 인해 CloudFront에서 오류가 발생함
- Lambda@Edge 또는 CloudFront Functions 실행 오류
- Lambda @Edge 제한 초과

## Origin server does not have enough capacity to support the request rate

오리진 서버를 사용할 수 없거나 수신된 요청을 처리할 수 없는 경우, HTTP 503 상태 코드(서비스 사용 불가)가 반환됩니다. 그런 다음 CloudFront가 다시 사용자에게 오류를 전달합니다. 이 문제를 해결하려면 다음 솔루션을 시도해 보십시오.

- Amazon S3를 오리진 서버로 사용하는 경우:
  - 분할된 Amazon S3 접두사별로 초당 최소 3,500개의 PUT/COPY/POST/DELETE 또는 5,500개의 GET/HEAD 요청을 전송할 수 있습니다. Amazon S3에서 503 Slow Down 응답을 반환하는 경우 이는 일반적으로 특정 Amazon S3 접두사에 대한 요청 빈도가 너무 높음을 나타냅니다.

    요청 요금은 S3 버킷의 접두사별로 적용되므로 객체를 여러 접두사에 분산해야 합니다. 접두사의 요청 빈도가 점차 증가함에 따라 Amazon S3는 각 접두사에 대한 요청을 개별적으로 처리하도록 스케일 업니다. 결과적으로, 버킷이 처리하는 전체 요청 빈도는 접두사 수의 배수입니다.
  - 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 Amazon S3 성능 최적화를 참조하세요.
- Elastic Load Balancing을 오리진 서버로 사용하는 경우:
  - 백엔드 인스턴스가 상태 확인에 응답할 수 있는지 확인합니다.
  - 로드 밸런서와 백엔드 인스턴스가 부하를 처리할 수 있는지 확인합니다.

  자세한 내용은 다음을 참조하세요.
  - Classic Load Balancer 사용 중 반환되는 503 오류를 해결하려면 어떻게 해야 하나요?
  - Application Load Balancer에서 발생하는 503(서비스 사용 불가) 오류를 해결하려면 어떻게 해야 합니까?
- 사용자 지정 오리진을 사용하는 경우:

- 애플리케이션 로그를 검사하여 오리진에 메모리, CPU, 디스크 크기 등의 리소스가 충분한지 확인합니다.
- Amazon EC2를 백엔드로 사용하는 경우 인스턴스 유형에 수신 요청을 충족하는 적절한 리소스가 있는지 확인합니다. 자세한 내용을 알아보려면 Amazon EC2 사용 설명서의 인스턴스 유형을 참조하세요.

- API Gateway를 사용하는 경우:
  - 이 오류는 API Gateway API가 응답을 받을 수 없는 경우의 백엔드 통합과 관련이 있습니다. 백엔드 서버는 다음과 같을 수 있습니다.
    - 용량을 초과하여 과부하가 걸려 새 클라이언트 요청을 처리할 수 없습니다.
    - 임시 유지 관리 중입니다.
  - 이 오류를 해결하려면 API Gateway 애플리케이션 로그를 살펴보고 백엔드 용량, 통합 등에 문제가 있는지 확인합니다.

## 엣지 위치에서의 리소스 제약 조건으로 인해 CloudFront에서 오류가 발생함

드문 경우지만 CloudFront가 사용 가능한 차선의 엣지 로케이션으로 요청을 라우팅할 수 없어서 요청을 충족할 수 없는 경우에 이 오류가 발생합니다. 이 오류는 CloudFront 배포에 대한 로드 테스트를 수행할 때 흔히 발생합니다. 이러한 오류를 방지하려면 503(용량 초과) 오류 방지를 위한 the section called “CloudFront 로드 테스트” 지침을 따르세요.

프로덕션 환경에서 이러한 오류가 발생하면 지원에 문의하세요.

## Lambda@Edge 또는 CloudFront Functions 실행 오류

Lambda@Edge 또는 CloudFront 함수를 사용하는 경우, HTTP 503 상태 코드는 함수가 실행 오류를 반환했음을 나타낼 수 있습니다.

Lambda@Edge 오류를 식별하고 해결하는 방법에 대한 자세한 내용은 Lambda@Edge 함수 테스트 및 디버깅 섹션을 참조하세요.

CloudFront Functions 테스트에 대한 자세한 내용은 함수 테스트 섹션을 참조하세요.

## Lambda @Edge 제한 초과

Lambda@Edge를 사용하는 경우 HTTP 503 상태 코드는 Lambda가 오류를 반환했음을 나타낼 수 있습니다. 오류는 다음 중 하나로 인해 발생할 수 있습니다.

- 함수 실행 수가 AWS 리전에서 실행을 제한하기 위해 Lambda가 설정한 할당량(동시 실행 또는 간접 호출 빈도) 중 하나를 초과했습니다.
- 함수가 Lambda 함수의 제한 시간 할당량을 초과했습니다.

Lambda@Edge 할당량에 대한 자세한 내용은 Lambda@Edge에 대한 할당량 섹션을 참조하세요. Lambda@Edge 오류를 식별하고 해결하는 방법에 대한 자세한 내용은 the section called "테스트 및 디버깅" 섹션을 참조하세요. AWS Lambda 개발자 안내서에서도 Lambda 서비스 할당량을 확인할 수 있습니다.

## HTTP 504 상태 코드(게이트웨이 제한 시간)

HTTP 504 상태 코드(게이트웨이 시간 초과)는 CloudFront가 오리진에 요청을 전달했을 때(요청된 객체가 엣지 캐시에 없었기 때문에) 다음 중 하나가 발생했음을 나타냅니다.

- 오리진이 CloudFront에 HTTP 504 상태 코드를 반환했습니다.
- 요청이 만료되기 전에 오리진이 응답하지 않았습니다.

방화벽 또는 보안 그룹에 의해 오리진에 대해 트래픽이 차단된 경우나 오리진이 인터넷에서 액세스가 불가능한 경우에 CloudFront는 HTTP 504 상태 코드를 반환합니다. 이러한 문제들이 있는지 먼저 확인합니다. 그런 다음, 액세스 문제가 아닌 경우에는 애플리케이션 지연 및 서버 제한 시간을 탐색하여 문제를 확인하고 시정합니다.

주제

- CloudFront 트래픽을 허용하도록 원본 서버에서 방화벽 구성
- CloudFront 트래픽을 허용하도록 원본 서버에서 보안 그룹 구성
- 사용자 지정 원본 서버를 인터넷에서 액세스할 수 있도록 설정
- 원본 서버의 애플리케이션에서 지연된 응답을 찾아서 수정

### CloudFront 트래픽을 허용하도록 원본 서버에서 방화벽 구성

오리진 서버의 방화벽에 의해 CloudFront 트래픽이 차단되는 경우에 CloudFront는 HTTP 504 상태 코드를 반환하는데, 다른 문제를 확인하기 전에 먼저 이 문제가 아닌지 확인하는 것이 좋습니다.

이것이 방화벽 문제인지를 판단하기 위해 사용하는 방법은 오리진 서버가 사용하는 시스템에 따라 다릅니다.

- Linux 서버에서 IPTable 방화벽을 사용하는 경우에는 IPTable에서의 작업에 도움이 되는 도구 및 정보를 검색할 수 있습니다.
- Windows 서버에서 Windows 방화벽을 사용하는 경우에는 Microsoft 설명서의 방화벽 규칙 추가 또는 편집을 참조하세요.

오리진 서버에서 방화벽 구성을 평가할 때는 게시된 IP 주소 범위를 토대로 CloudFront 엣지 로케이션에서의 트래픽을 차단하는 방화벽이나 보안 규칙이 있는지 찾아봅니다. 자세한 내용은 CloudFront 엣지 서버의 위치 및 IP 주소 범위 섹션을 참조하세요.

CloudFront IP 주소 범위가 오리진 서버에 연결하도록 허용되는 경우, 변경 사항이 포함되도록 서버의 보안 규칙을 업데이트합니다. Amazon SNS 주제를 구독하면 IP 주소 범위 파일이 업데이트될 때 알림을 수신할 수 있습니다. 알림을 수신한 후에는 코드를 사용하여 파일을 찾아서 구문 분석하고 로컬 환경에 맞게 조정할 수 있습니다. 자세한 내용은 AWS 뉴스 블로그에서 Amazon SNS를 통한 AWS 퍼블릭 IP 주소 변경 사항 구독을 참조하세요.

## CloudFront 트래픽을 허용하도록 원본 서버에서 보안 그룹 구성

오리진이 Elastic Load Balancing을 사용하는 경우에는 ELB 보안 그룹을 검토해서 보안 그룹이 CloudFront에서의 인바운드 트래픽을 허용하는지 확인합니다.

또한 AWS Lambda를 사용하여 보안 그룹을 자동 업데이트함으로써 CloudFront에서의 인바운드 트래픽을 허용할 수도 있습니다.

## 사용자 지정 원본 서버를 인터넷에서 액세스할 수 있도록 설정

인터넷에서 공개적으로 사용할 수 없기 때문에 CloudFront가 사용자 지정 오리진 서버를 액세스할 수 없는 경우, CloudFront는 HTTP 504 오류를 반환합니다.

CloudFront 엣지 로케이션은 인터넷을 통해 오리진 서버에 연결됩니다. 사용자 지정 오리진이 사설 네트워크에 있는 경우에는 CloudFront가 이를 액세스할 수 없습니다. 따라서 내부 Classic Load Balancer를 포함한 프라이빗 서버를 CloudFront에서 오리진 서버로 사용할 수 없습니다.

인터넷 트래픽을 오리진 서버에 연결할 수 있는지 확인하려면 다음 명령(*OriginDomainName*은 서버의 도메인 이름)을 실행하세요.

HTTPS 트래픽의 경우:

- nc -zv *OriginDomainName* 443
- telnet *OriginDomainName* 443

HTTP 트래픽의 경우:

- nc -zv *OriginDomainName* 80
- telnet *OriginDomainName* 80

## 원본 서버의 애플리케이션에서 지연된 응답을 찾아서 수정

애플리케이션에서 응답에 너무 많은 시간이 소요되거나 제한 시간 값이 너무 낮게 설정된 결과로 인해 서버 제한 시간이 발생하는 경우가 종종 있습니다.

HTTP 504 오류를 방지하기 위한 일시적인 해결책으로 배포 시 CloudFront 제한 시간 값을 더 높게 설정하는 방법이 있기는 합니다. 그러나 애플리케이션 및 오리진 서버에서 성능 및 지연 문제가 해결되었는지 먼저 확인하는 것이 좋습니다. 제한 시간 값을 적당하게 설정하면 HTTP 504 오류를 방지하고 사용자에게 뛰어난 응답성을 제공할 수 있습니다.

여기에는 성능 문제를 찾아서 해결할 때 거치게 되는 단계들이 개략적으로 나와 있습니다.

1. 정상 부하 및 고부하에서 웹 애플리케이션의 지연 시간(응답성)을 측정합니다.
2. 필요할 경우 CPU나 메모리 같은 리소스를 추가합니다. 고부하 상황을 지원하도록 데이터베이스 쿼리를 튜닝하는 등 문제 해결을 위한 기타 조치를 취합니다.
3. 필요할 경우 CloudFront 배포의 제한 시간 값을 조정합니다.

아래에는 각 단계에 대한 세부 정보가 나와 있습니다.

### 정상 부하 및 고부하에서 지연 시간 측정

하나 이상의 백엔드 웹 애플리케이션 서버가 고부하를 경험하고 있는지 판단하려면 각 서버에서 아래의 Linux curl 명령을 실행합니다.

```
curl -w "DNS Lookup Time: %{time_namelookup} \nConnect time: %{time_connect}
 \nTLS Setup: %{time_appconnect} \nRedirect Time: %{time_redirect} \nTime to first
 byte: %{time_starttransfer} \nTotal time: %{time_total} \n" -o /dev/null https://
www.example.com/yourobject
```

> **Note**
>
> 서버에서 Windows를 실행하는 경우에는 유사한 명령을 실행하도록 Windows용 cURL을 검색 및 다운로드할 수 있습니다.

서버에서 실행되는 애플리케이션의 지연 시간을 측정 및 평가할 때는 다음 사항에 유의하십시오.

- 지연 값은 각 애플리케이션에 상대적입니다. 하지만 초 단위가 아니라 밀리초 단위의 TTFB(Time To First Byte)가 적당합니다.
- 정상 부하에서 애플리케이션 지연 시간을 측정한 결과 문제가 없었더라도 고부하에서는 최종 사용자가 제한 시간을 경험할 수 있다는 점에 유의하세요. 수요가 높으면 서버에서 응답이 지연되거나 아예 응답이 없을 수 있습니다. 고부하 지연 문제를 방지하려면 CPU, 메모리, 디스크 읽기/쓰기 같은 서버의 리소스를 확인하여 고부하에 맞게 서버 용량을 확장합니다.

  다음 Linux 명령을 실행하여 Apache 프로세스에서 사용되는 메모리를 확인할 수 있습니다.

  ```
  watch -n 1 "echo -n 'Apache Processes: ' && ps -C apache2 --no-headers |
  wc -l && free -m"
  ```

- 서버에서 CPU 활용도가 높으면 애플리케이션 성능이 크게 낮아질 수 있습니다. 백엔드 서버에서 Amazon EC2 인스턴스를 사용하는 경우에는 서버의 CloudWatch 지표를 검토하여 CPU 활용도를 확인합니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하십시오. 자체 서버를 사용 중인 경우에는 서버 도움말 문서에서 CPU 활용도를 확인할 수 있는 방법에 대한 지침을 참조하세요.
- 요청의 수가 많을 때 데이터베이스 쿼리의 실행 속도가 느려지는 것과 같이 고부하 상황에서 발생할 수 있는 기타 문제들을 확인하세요.

리소스를 추가하고 서버 및 데이터베이스 튜닝

애플리케이션 및 서버의 응답성을 평가한 후에는 정상 부하 및 고부하 상황을 위해 충분한 리소스가 구축되어 있는지 확인합니다.

- 자체 서버를 가지고 있는 경우에는 평가 결과를 토대로 CPU, 메모리 및 디스크 공간이 최종 사용자 요청을 처리하기에 충분한지 확인합니다.
- Amazon EC2 인스턴스를 백엔드 서버로 사용하는 경우 인스턴스 유형에 수신 요청을 이행하기에 적절한 리소스가 있는지 확인합니다. 자세한 내용을 알아보려면 Amazon EC2 사용 설명서의 인스턴스 유형을 참조하세요.

뿐만 아니라, 제한 시간을 방지하려면 다음과 같은 튜닝 조치를 고려하십시오.

- cURL 명령이 반환한 TTFB(Time To First Byte) 값이 높은 경우에는 애플리케이션의 성능을 개선하기 위한 조치를 취합니다. 애플리케이션 응답성을 높이면 제한 시간 오류가 줄어듭니다.
- 성능 저하 없이 많은 양의 요청을 처리할 수 있도록 데이터베이스 쿼리를 튜닝합니다.

- 백엔드 서버에서 연결 유지(영구) 연결을 설정합니다. 이 옵션은 후속 요청이나 사용자에 대해 연결을 다시 설정해야 하는 경우에 발생하는 지연을 방지하는 데 도움이 됩니다.
- Elastic Load Balancing를 오리진으로 사용하는 경우, 504 오류가 발생할 수 있는 원인은 다음과 같습니다.
  - 연결 제한 시간이 만료(10초)되기 전에 로드 밸런서가 대상에 대한 연결을 설정하지 못합니다.
  - 로드 밸런서가 대상에 대한 연결을 설정하지만, 유휴 제한 시간이 끝나기 전에 대상이 응답을 하지 않습니다.
  - 서브넷의 네트워크 액세스 제어 목록(ACL)이 휘발성 포트(1024-65535)에서 대상으로부터 로드 밸런서 노드로의 트래픽을 허용하지 않습니다.
  - 대상이 개체 몸체보다 큰 콘텐츠 길이 헤더를 반환합니다. 로드 밸런서가 놓친 바이트를 기다리는 도중 시간이 초과됩니다.
  - 대상이 Lambda 함수이고 Lambda 서비스가 연결 제한 시간이 만료되기 전에 응답하지 않습니다.

  지연 시간 단축에 대한 자세한 내용은 ELB Classic Load Balancer의 지연 시간이 긴 문제를 해결하려면 어떻게 해야 하나요?를 참조하세요.
- MediaTailor를 오리진으로 사용하는 경우, 504 오류가 발생할 수 있는 원인은 다음과 같습니다.
  - 상대 URL을 잘못 처리하는 경우 MediaTailor는 플레이어로부터 잘못된 형식의 URL을 수신할 수 있습니다.
  - MediaPackage가 MediaTailor의 매니페스트 오리진인 경우 MediaPackage 404 매니페스트 오류로 인해 MediaTailor가 504 오류를 반환할 수 있습니다.
  - MediaTailor 오리진 서버에 대한 요청을 완료하는 데 2초 이상 걸립니다.
- Amazon API Gateway를 오리진으로 사용하는 경우, 504 오류가 발생할 수 있는 원인은 다음과 같습니다.
  - 통합 요청이 API Gateway REST API 최대 통합 제한 시간 파라미터보다 오래 걸립니다. 자세한 내용은 API Gateway의 API HTTP 504 제한 시간 오류를 해결하려면 어떻게 해야 하나요?를 참조하세요.

필요할 경우 CloudFront 제한 시간 값을 조정합니다.

애플리케이션 성능 저하, 오리진 서버 용량 및 기타 문제를 평가하고 해결했지만 최종 사용자가 여전히 HTTP 504 오류를 경험하고 있는 경우에는 오리진 응답 제한 시간에 대해 배포에 지정되어 있는 시간을 변경하는 것을 반드시 고려해봐야 합니다. 자세한 내용은 the section called “응답 제한 시간” 섹션을 참조하세요.

# CloudFront 로드 테스트

기존 로드 테스트 방식은 CloudFront에서 DNS를 사용하여 지리적으로 분산된 엣지 로케이션으로 각 엣지 로케이션 내에 로드를 분배하기 때문에 CloudFront에는 잘 맞지 않았습니다. 클라이언트가 CloudFront에서 콘텐츠를 요청하면 이 클라이언트는 IP 주소 집합이 포함된 DNS 응답을 받습니다. DNS를 반환하는 IP 주소 중 하나에만 요청을 전송하여 테스트하는 경우 단일 CloudFront 엣지 로케이션에서 리소스의 소규모 하위 집합만 테스트하게 되며, 이는 실제 트래픽 패턴을 정확하게 나타내지 않습니다. 요청된 데이터 볼륨에 따라, 이 방법을 통한 테스팅은 CloudFront 서버의 해당 소규모 하위 집합을 오버로드하고 성능을 떨어뜨릴 수 있습니다.

CloudFront는 여러 지리적 리전에서 서로 다른 클라이언트 IP 주소와 DNS 확인 프로그램을 보유한 최종 사용자를 위해 규모를 조정하도록 설계되어 있습니다. CloudFront 성능을 정확하게 평가하는 로드 테스팅을 수행하려면 다음 사항을 모두 수행하는 것이 좋습니다.

- 여러 지리적 리전에서 클라이언트 요청을 보냅니다.
- 각 클라이언트에서 독립적인 DNS 요청을 하도록 테스트를 구성합니다. 그러면 클라이언트별로 DNS에서 서로 다른 IP 주소 세트를 수신하게 됩니다.
- 요청을 수행하는 각 클라이언트의 경우 DNS에서 반환되는 IP 주소 세트 전반에 클라이언트 요청을 분산합니다. 이렇게 하면 CloudFront 엣지 로케이션의 여러 서버에 부하가 분산됩니다.

> **참고**
>
> - Lambda@Edge 뷰어 요청 또는 뷰어 응답 트리거가 있는 캐시 동작에서는 부하 테스트가 허용되지 않습니다.
> - Origin Shield가 사용되는 오리진에서는 부하 테스트가 허용되지 않습니다.

# 할당량

다음 옵션을 사용하여 CloudFront 할당량 증가를 요청할 수 있습니다.

- Service Quotas 콘솔 또는 AWS Command Line Interface를 사용할 수 있습니다. 자세한 내용은 다음 항목을 참조하세요.
  - Service Quotas 사용 설명서의 Requesting a quota increase
  - AWS CLI 명령 참조의 request-service-quota-increase
- Service Quotas에서 CloudFront 할당량을 사용할 수 없는 경우 AWS Support Center Console을 사용하여 서비스 할당량 증가 사례를 생성합니다.

CloudFront에는 다음 할당량이 적용됩니다.

주제


- 일반 할당량
- 배포의 일반 할당량
- 정책에 대한 일반 할당량
- mTLS 및 트러스트 스토어의 할당량
- CloudFront 함수의 할당량
- 연결 함수에 대한 할당량
- 키 값 저장소의 할당량
- Lambda@Edge에 대한 할당량
- SSL 인증서의 할당량
- 무효화에 대한 할당량
- 키 그룹에 대한 할당량
- WebSocket 연결에 대한 할당량
- 필드 레벨 암호화에 대한 할당량
- 쿠키에 대한 할당량(레거시 캐시 설정)
- 쿼리 문자열에 대한 할당량(레거시 캐시 설정)
- 헤더에 대한 할당량
- 다중 테넌트 배포의 할당량
- 관련 정보

# 일반 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| 배포당 데이터 전송 속도<br><br>(이 할당량은 CloudFront 정액 요금제를 구독하는 배포에는 적용되지 않습니다. 자세한 내용은 ??? 섹션을 참조하세요.) | 150Gbps<br><br>더 높은 할당량 요청 |
| 배포당 초당 요청 수<br><br>(이 할당량은 CloudFront 정액 요금제를 구독하는 배포에는 적용되지 않습니다. 자세한 내용은 ??? 섹션을 참조하세요.) | 250,000<br><br>더 높은 할당량 요청 |
| 배포에 추가할 수 있는 태그 | 50<br><br>더 높은 할당량 요청 |
| 배포당 제공할 수 있는 파일 수 | 할당량 없음 |
| 헤더와 쿼리 문자열을 포함한 요청 또는 오리진 응답의 최대 길이(본문 내용 제외) | 20,480바이트 |
| URL의 최대 길이 | 8,192바이트 |
| AWS 계정당 최대 실시간 액세스 로그 전송 구성 수 | 150 |
| 웹 ACL당 최대 연결 수 | 100<br><br>더 높은 할당량 요청 |

# 배포의 일반 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| 배포당 대체 도메인 이름(CNAME) 수<br><br>자세한 내용은 대체 도메인 이름(CNAME)을 추가하여 사용자 지정 URL 사용 단원을 참조하세요. | 100<br><br>더 높은 할당량 요청 |

| 엔터티 | 기본 할당량 |
| --- | --- |
| 배포당 캐시 동작 수 | 75<br><br>더 높은 할당량 요청 |
| 오리진당 연결 시도 횟수<br><br>자세한 내용은 연결 시도 단원을 참조하세요. | 1-3 |
| 오리진당 연결 제한 시간<br><br>자세한 내용은 연결 제한 시간 단원을 참조하세요. | 1-10초 |
| 오리진별 응답 제한 시간<br><br>이를 오리진 읽기 제한 시간 또는 오리진 요청 제한 시간이라고도 합니다. 자세한 내용은 응답 제한 시간 섹션을 참조하세요. | 1~120초<br><br>더 높은 할당량 요청 |
| 오리진별 연결 유지 제한 시간<br><br>자세한 내용은 연결 유지 제한 시간(사용자 지정 및 VPC 오리진만 해당) 섹션을 참조하세요. | 1~120초<br><br>더 높은 할당량 요청 |
| AWS 계정당 배포<br><br>자세한 내용은 배포 생성 섹션을 참조하세요. | 500<br><br>더 높은 할당량 요청 |
| 오리진별 분포 액세스 제어 | 100<br><br>더 높은 할당량 요청 |
| 요청 체인 내에서 오리진 엔드포인트로의 배포<br><br>한 배포를 다른 배포 앞에 배치하지 않는 것이 좋습니다. 이 할당량을 초과하면 403 오류가 발생합니다. | 2 |
| 파일 압축: CloudFront가 압축하는 파일 크기의 범위<br><br>자세한 내용은 압축된 파일 제공 단원을 참조하세요. | 1,000 ~ 10,000,000바이트 |

| 엔터티 | 기본 할당량 |
| --- | --- |
| HTTP GET 응답당 캐시 가능한 최대 파일 크기.<br><br>HTTP GET에 대한 응답만 캐시됩니다. POST 또는 PUT에 대한 응답은 캐시되지 않습니다. | 50GB |
| AWS 계정당 오리진 액세스 제어 | 100<br><br>더 높은 할당량 요청 |
| AWS 계정당 오리진 액세스 ID | 100<br><br>더 높은 할당량 요청 |
| 배포당 오리진 수 | 100<br><br>더 높은 할당량 요청 |
| 배포당 오리진 그룹 수 | 10<br><br>더 높은 할당량 요청 |
| AWS 계정당 스테이징 배포<br><br>자세한 내용은 the section called “CloudFront 지속적 배포를 사용하여 변경을 안전하게 테스트” 섹션을 참조하세요. | 20<br><br>더 높은 할당량 요청 |
| 동일한 VPC 오리진과 연결된 배포 | 50 |
| AWS 계정당 VPC 오리진 | 25<br><br>더 높은 할당량 요청 |
| 단일 애니캐스트 고정 IP 목록과 연결할 수 있는 최대 배포 수입니다. | 100<br><br>더 높은 할당량 요청 |

# 정책에 대한 일반 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| AWS 계정당 사용자 지정 캐시 정책<br><br>(CloudFront 관리형 캐시 정책에는 적용되지 않음) | 20<br><br>더 높은 할당량 요청 |
| 동일한 캐시 정책과 연결된 배포 | 100 |
| 캐시 정책당 쿼리 문자열 | 10<br><br>더 높은 할당량 요청 |
| 캐시 정책당 헤더 | 10<br><br>더 높은 할당량 요청 |
| 캐시 정책당 쿠키 | 10<br><br>더 높은 할당량 요청 |
| 캐시 정책에서 모든 쿼리 문자열, 헤더, 쿠키 이름을 합친 총 길이 | 1024 |
| AWS 계정당 사용자 지정 오리진 요청 정책 수<br><br>(CloudFront 관리형 오리진 요청 정책에는 적용되지 않음) | 20<br><br>더 높은 할당량 요청 |
| 동일한 오리진 요청 정책과 연결된 배포 | 100 |
| 오리진 요청 정책당 쿼리 문자열 | 10<br><br>더 높은 할당량 요청 |
| 오리진 요청 정책당 헤더 | 10<br><br>더 높은 할당량 요청 |
| 오리진 요청 정책당 쿠키 | 10<br><br>더 높은 할당량 요청 |

| 엔터티 | 기본 할당량 |
| --- | --- |
| 오리진 요청 정책에서 모든 쿼리 문자열, 헤더, 쿠키 이름을 합친 총 길이 | 1024 |
| AWS 계정당 사용자 지정 응답 헤더 정책<br><br>(CloudFront 관리형 응답 헤더 정책에는 적용되지 않음) | 20<br><br>더 높은 할당량 요청 |
| 동일한 응답 헤더 정책과 연결된 배포 | 100<br><br>더 높은 할당량 요청 |
| 응답 헤더 정책당 사용자 지정 헤더 | 10<br><br>더 높은 할당량 요청 |
| AWS 계정당 지속적 배포 정책 | 20<br><br>더 높은 할당량 요청 |

# mTLS 및 트러스트 스토어의 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| AWS 계정당 트러스트 스토어 | 20<br><br>더 높은 할당량 요청 |
| 트러스트 스토어당 배포 수 | 25 |
| CA 번들 크기 | 64KB<br><br>더 높은 할당량 요청 |
| CA 번들의 인증서 크기 | 16384<br><br>더 높은 할당량 요청 |
| CA 번들의 인증서 수 | 25 |

| 엔터티 | 기본 할당량 |
| --- | --- |
| 인증서 체인 깊이 | 4 |

## CloudFront 함수의 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| AWS 계정당 기능 | 100 |
| 최대 함수 크기<br><br>이 할당량은 조정할 수 없습니다. CloudFront Functions에 대한 추가 데이터를 저장하려면 키 값 저장소를 생성하고 키 값 페어를 추가합니다. 자세한 내용은 Amazon CloudFront KeyValueStore 섹션을 참조하세요. | 10KB |
| 최대 함수 메모리 | 2MB |
| 동일한 함수와 연결된 배포 | 100 |

CloudFront 함수를 사용할 때 이 할당량 이외에도 다른 제한 사항이 있습니다. 자세한 내용은 CloudFront 함수에 대한 제한 섹션을 참조하세요.

## 연결 함수에 대한 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| AWS 계정당 연결 함수<br><br>자세한 내용은 연결 함수 할당량 증가 요청 섹션을 참조하세요. | 0 |
| 최대 연결 함수 크기<br><br>이 할당량은 조정할 수 없습니다. 연결 함수에 대한 추가 데이터를 저장하려면 키 값 저장소를 생성하고 키 값 페어를 추가합니다. 자세한 내용은 Amazon CloudFront KeyValueStore 섹션을 참조하세요. | 10KB |

| 엔터티 | 기본 할당량 |
| --- | --- |
| 최대 연결 함수 메모리 | 2MB |
| 동일한 Connection Function과 연결된 배포 | 100 |

연결 함수를 사용할 때 이 할당량 이외에도 다른 제한 사항이 있습니다. 자세한 내용은 CloudFront 연결 함수 연결 섹션을 참조하세요.

## 키 값 저장소의 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| 키 값 페어 키의 최대 크기 | 512B |
| 키 값 페어 값의 최대 크기 | 1KB |
| 단일 API 요청으로 업데이트할 수 있는 최대 키 값 페어 | 키 50개 또는 3MB 페이로드(둘 중 먼저 도달한 쪽) |
| 개별 키 값 저장소의 최대 크기 | 5MB |
| 단일 키 값 저장소를 연결할 수 있는 최대 함수 수 | 10 |
| 함수당 최대 키 값 저장소 수 | 1 |
| 계정당 최대 키 값 저장소 수 | 50<br>더 높은 할당량 요청 |

## Lambda@Edge에 대한 할당량

일반 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| Lambda@Edge 함수를 가질 수 있는 AWS 계정당 배포 수 | 500 |

| 엔터티 | 기본 할당량 |
| --- | --- |
| | 더 높은 할당량 요청 |
| 배포당 Lambda@Edge 함수 | 100<br><br>더 높은 할당량 요청 |
| 동시 실행<br><br>참고<br><br>• AWS Lambda는 Lambda@Edge에 대한 동시성 할당량을 관리합니다. AWS 리전의 모든 Lambda 함수는 이 할당량을 공유합니다.<br>• 뷰어 요청이 발생할 것으로 예상되는 모든 AWS 리전에서 동시 실행 할당량을 검토하는 것이 좋습니다. 또한 Lambda@Edge 함수의 각 인스턴스는 초당 최대 10개의 요청을 처리할 수 있습니다. 총 간접 호출 한도는 동시성 한도의 10배입니다.<br><br>자세한 내용은 AWS Lambda 개발자 설명서에서 다음 주제를 참조하세요.<br><br>• Lambda 함수 규모 조정 이행<br>• Lambda API 요청 | 1,000개(AWS 리전당)<br><br>더 높은 할당량 요청 |
| 동일한 함수와 연결된 배포 | 500 |
| Lambda 함수 및 포함된 라이브러리의 최대 압축 크기 | 50MB |
| 초당 Lambda@Edge 요청(지원되는 각 AWS 리전)<br><br>자세한 내용은 AWS Lambda 개발자 안내서의 동시성 할당량을 참조하시기 바랍니다. | 10,000개 |

이벤트 유형에 따라 다른 할당량

| 엔터티 | 뷰어 요청 및 뷰어 응답 이벤트 | 오리진 요청 및 오리진 응답 이벤트 |
| --- | --- | --- |
| 함수 메모리 크기 | 128MB | Lambda 할당량과 동일 |
| 함수 제한 시간. 이 함수는 AWS 리전에서 Amazon S3 버킷, DynamoDB 테이블 또는 Amazon EC2 인스턴스와 같은 리소스에 대한 네트워크 호출을 수행할 수 있습니다. | 30초 | 30초 |
| 헤더 및 본문을 포함하여 Lambda 함수에서 생성되는 응답의 크기입니다. | 40KB | 1MB |

참고

- Service Quotas에서 늘릴 수 있는 추가 Lambda@Edge 할당량 목록은 AWS 일반 참조의 Amazon CloudFront 엔드포인트 및 할당량을 참조하세요.
- Lambda@Edge 함수를 사용할 때 할당량 외에도 기타 제한 사항이 있습니다. 자세한 내용은 Lambda@Edge에 대한 제한 사항 단원을 참조하세요.

# SSL 인증서의 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| 전용 IP 주소를 사용하여 HTTPS 요청을 제공하는 경우 AWS 계정당 SSL 인증서 수(SNI를 사용하여 HTTPS 요청을 제공하는 경우 할당량 없음)<br><br>자세한 내용은 CloudFront에서 HTTPS 사용 섹션을 참조하세요. | 2<br><br>더 높은 할당량 요청 |
| CloudFront 배포와 연결할 수 있는 SSL 인증서 | 1 |

SSL 인증서가 뷰어와 CloudFront 간의 HTTPS 통신을 위한 것이며 인증서를 프로비저닝하거나 가져오기 위해 AWS Certificate Manager(ACM) 또는 IAM 인증서 저장소를 사용한 경우에는 추가 할당량이 적용됩니다. 자세한 내용은 CloudFront에서 SSL/TLS 인증서 사용 시의 할당량(뷰어와 CloudFront 간의 HTTPS만 해당) 섹션을 참조하세요.

또한 AWS Certificate Manager(ACM)로 가져오거나 AWS Identity and Access Management(IAM)에 업로드할 수 있는 SSL 인증서 수에는 할당량이 있습니다. 자세한 내용은 SSL/TLS 인증서에 대한 할당량 증가 섹션을 참조하세요.

## 무효화에 대한 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| 파일 무효화: 와일드카드 무효화를 제외하고 활성 무효화 요청에 허용된 파일의 최대 수<br><br>자세한 내용은 파일을 무효화하여 콘텐츠 제거 단원을 참조하세요. | 3,000 |
| 파일 무효화: 허용된 활성 와일드카드 무효화의 최대 수 | 15 |
| 파일 무효화: 하나의 와일드카드 무효화가 처리할 수 있는 파일의 최대 수 | 할당량 없음 |

## 키 그룹에 대한 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| 단일 키 그룹의 퍼블릭 키 | 5<br><br>더 높은 할당량 요청 |
| 단일 캐시 동작과 연결된 키 그룹 | 4<br><br>더 높은 할당량 요청 |
| AWS 계정당 키 그룹 | 10<br><br>더 높은 할당량 요청 |
| 단일 키 그룹과 연결된 배포 | 100 |

| 엔터티 | 기본 할당량 |
| --- | --- |
| | 더 높은 할당량 요청 |

## WebSocket 연결에 대한 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| 오리진 응답 제한 시간(유휴 제한 시간) | 10분<br><br>CloudFront가 지난 10분 사이에 오리진에서 클라이언트로 전송된 데이터를 전혀 감지하지 못한 경우, 유휴 상태로 간주되어 연결이 종료됩니다. |

## 필드 레벨 암호화에 대한 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| 암호화할 필드의 최대 길이<br><br>자세한 내용은 필드 수준 암호화를 사용하여 민감한 데이터 보호 단원을 참조하세요. | 16KB |
| 필드 레벨 암호화 구성 시 요청 본문의 최대 필드 수 | 10 |
| 필드 레벨 암호화 구성 시 요청 본문의 최대 길이 | 1MB |
| 하나의 AWS 계정과 연결될 수 있는 필드 레벨 암호화 구성의 최대 수 | 10 |
| 하나의 AWS 계정과 연결될 수 있는 필드 레벨 암호화 프로필의 최대 수 | 10 |
| 하나의 AWS 계정에 추가될 수 있는 퍼블릭 키의 최대 수 | 10 |

| 엔터티 | 기본 할당량 |
| --- | --- |
| 하나의 프로필에 지정되도록 암호화할 수 있는 최대 필드 수 | 10 |
| 필드 레벨 암호화 구성과 연결할 수 있는 CloudFront 배포의 최대 수 | 20 |
| 필드 레벨 암호화 구성에 포함될 수 있는 쿼리 인수 프로필 매핑의 최대 수 | 5 |

## 쿠키에 대한 할당량(레거시 캐시 설정)

이러한 할당량은 CloudFront의 레거시 캐시 설정에 적용됩니다. 레거시 설정 대신 캐시 정책 또는 오리진 요청 정책을 사용하는 것이 좋습니다.

| 엔터티 | 기본 할당량 |
| --- | --- |
| 캐시 동작당 쿠키<br><br>자세한 내용은 쿠키 기반의 콘텐츠 캐싱 단원을 참조하세요. | 10<br><br>더 높은 할당량 요청 |
| 쿠키 이름의 총 바이트 수(오리진에 모든 쿠키를 전달하도록 CloudFront를 구성하는 경우에는 적용되지 않음) | 512에서 쿠키 수를 뺀 값 |

## 쿼리 문자열에 대한 할당량(레거시 캐시 설정)

이러한 할당량은 CloudFront의 레거시 캐시 설정에 적용됩니다. 레거시 설정 대신 캐시 정책 또는 오리진 요청 정책을 사용하는 것이 좋습니다.

| 엔터티 | 기본 할당량 |
| --- | --- |
| 쿼리 문자열의 최대 문자 수 | 128자 |
| 동일한 파라미터 내의 모든 쿼리 문자열에 대한 총 최대 문자 수 | 512자 |
| 캐시 동작당 쿼리 문자열<br><br>자세한 내용은 쿼리 문자열 파라미터 기반의 콘텐츠 캐싱 단원을 참조하세요. | 10<br><br>더 높은 할당량 요청 |

# 헤더에 대한 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| 캐시 동작당 헤더(레거시 캐시 설정)<br><br>자세한 내용은 the section called “요청 헤더 기반의 콘텐츠 캐싱” 섹션을 참조하세요. | 10<br><br>더 높은 할당량 요청 |
| 캐시 동작당 전달 헤더 | 25<br><br>더 높은 할당량 요청 |
| 사용자 지정 헤더: 오리진 요청에 추가하도록 CloudFront를 구성할 수 있는 사용자 지정 헤더의 최대 수<br><br>자세한 내용은 the section called “사용자 지정 헤더를 오리진 요청에 추가” 섹션을 참조하세요. | 30<br><br>더 높은 할당량 요청 |
| 사용자 지정 헤더: 응답 헤더 정책에 추가할 수 있는 사용자 지정 헤더의 최대 수 | 10<br><br>더 높은 할당량 요청 |
| 사용자 지정 헤더: 헤더 이름의 최대 길이 | 256자 |
| 사용자 지정 헤더: 헤더 값의 최대 길이 | 1,783자 |
| 사용자 지정 헤더: 헤더 값과 이름을 모두 결합한 최대 길이 | 10,240자 |
| `Content-Security-Policy` 헤더 값의 최대 길이 | 1,783자<br><br>더 높은 할당량 요청 |
| `CORS (Access-Control-Allow-Origin)` 헤더 값의 최대 길이 | 1,783자 |

# 다중 테넌트 배포의 할당량

| 엔터티 | 기본 할당량 |
| --- | --- |
| AWS 계정당 최대 배포 테넌트 수 | 10,000개<br>더 높은 할당량 요청 |
| AWS 계정당 최대 다중 테넌트 배포 수 | 20<br>더 높은 할당량 요청 |
| AWS 계정당 최대 연결 그룹 수 | 100<br>더 높은 할당량 요청 |
| 배포 테넌트당 최대 별칭 수 | 100<br>더 높은 할당량 요청 |
| 배포 테넌트당 최대 파라미터 수 | 5<br>더 높은 할당량 요청 |
| 다중 테넌트 배포당 최대 파라미터 수 | 5<br>더 높은 할당량 요청 |
| 다중 테넌트 배포의 필드에 있는 최대 파라미터 수 | 2<br>더 높은 할당량 요청 |
| 애니캐스트 고정 IP 목록당 최대 연결 그룹 수 | 5<br>더 높은 할당량 요청 |

다중 테넌트 배포에 대한 자세한 내용은 다중 테넌트 배포의 작동 방식 이해 섹션을 참조하시기 바랍니다.

# 관련 정보

자세한 내용은 AWS 일반 참조의 Amazon CloudFront endpoints and quotas를 참조하세요.

# AWS SDK를 사용한 CloudFront 코드 예제

다음 코드 예제에서는 AWS 소프트웨어 개발 키트(SDK)에서 CloudFront를 사용하는 방법을 보여줍니다.

작업은 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 작업은 개별 서비스 함수를 직접 호출하는 방법을 보여주며, 관련 시나리오의 컨텍스트에 맞는 작업을 볼 수 있습니다.

시나리오는 서비스 내에서 또는 다른 AWS 서비스와 결합된 상태에서 여러 함수를 직접적 호출하여 특정 작업을 수행하는 방법을 보여주는 코드 예제입니다.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

코드 예시


- AWS SDK를 사용한 CloudFront 기본 예시
  - AWS SDK를 사용한 CloudFront 작업
    - AWS SDK 또는 CLI와 함께 CreateDistribution 사용
    - AWS SDK와 함께 CreateFunction 사용
    - CLI로 CreateInvalidation 사용
    - AWS SDK와 함께 CreateKeyGroup 사용
    - AWS SDK 또는 CLI와 함께 CreatePublicKey 사용
    - AWS SDK 또는 CLI와 함께 DeleteDistribution 사용
    - CLI로 GetCloudFrontOriginAccessIdentity 사용
    - CLI로 GetCloudFrontOriginAccessIdentityConfig 사용
    - CLI로 GetDistribution 사용
    - AWS SDK 또는 CLI와 함께 GetDistributionConfig 사용
    - CLI로 ListCloudFrontOriginAccessIdentities 사용
    - AWS SDK 또는 CLI와 함께 ListDistributions 사용
    - AWS SDK 또는 CLI와 함께 UpdateDistribution 사용
- AWS SDK를 사용한 CloudFront의 시나리오
  - SaaS Manager 리소스 AWS SDK 생성
  - AWS SDK를 사용하여 CloudFront 서명 리소스 삭제

- CLI를 사용하여 기본 CloudFront 배포 시작하기
- AWS SDK를 사용하여 서명된 URL과 쿠키 생성
- CloudFront의 CloudFront Functions 예제
  - CloudFront Functions 뷰어 응답 이벤트에 HTTP 보안 헤더 추가
  - CloudFront Functions 뷰어 응답 이벤트에 CORS 헤더 추가
  - CloudFront Functions 뷰어 응답 이벤트에 캐시 제어 헤더 추가
  - CloudFront Functions 뷰어 요청 이벤트에 실제 클라이언트 IP 헤더 추가
  - CloudFront Functions 뷰어 요청 이벤트에 오리진 헤더 추가
  - CloudFront Functions 뷰어 요청 이벤트에서 파일 이름 없이 URL을 요청하기 위한 index.html 추가
  - CloudFront Functions 뷰어 요청에서 쿼리 문자열 파라미터 정규화
  - CloudFront Functions 뷰어 요청 이벤트에서 새 URL로 리디렉션
  - CloudFront Functions 뷰어 요청 이벤트에 대한 KeyValueStore 구성을 기반으로 요청 URI 재작성
  - CloudFront Functions 뷰어 요청 이벤트에서 뷰어에게 더 가까운 오리진으로 요청 라우팅
  - CloudFront Functions 뷰어 요청에서 키-값 페어 사용
  - CloudFront Functions 뷰어 요청에서 단순 토큰 검증

# AWS SDK를 사용한 CloudFront 기본 예시

다음 코드 예시에서는 AWS SDK를 사용하여 Amazon CloudFront의 기본 기능을 사용하는 방법을 보여줍니다.

예제

- AWS SDK를 사용한 CloudFront 작업
  - AWS SDK 또는 CLI와 함께 CreateDistribution 사용
  - AWS SDK와 함께 CreateFunction 사용
  - CLI로 CreateInvalidation 사용
  - AWS SDK와 함께 CreateKeyGroup 사용
  - AWS SDK 또는 CLI와 함께 CreatePublicKey 사용
  - AWS SDK 또는 CLI와 함께 DeleteDistribution 사용
  - CLI로 GetCloudFrontOriginAccessIdentity 사용
  - CLI로 GetCloudFrontOriginAccessIdentityConfig 사용

- CLI로 GetDistribution 사용
- AWS SDK 또는 CLI와 함께 GetDistributionConfig 사용
- CLI로 ListCloudFrontOriginAccessIdentities 사용
- AWS SDK 또는 CLI와 함께 ListDistributions 사용
- AWS SDK 또는 CLI와 함께 UpdateDistribution 사용

## AWS SDK를 사용한 CloudFront 작업

다음 코드 예제에서는 AWS SDK에서 개별 CloudFront 작업을 수행하는 방법을 보여줍니다. 각 예제에는 GitHub에 대한 링크가 포함되어 있습니다. 여기에서 코드 설정 및 실행에 대한 지침을 찾을 수 있습니다.

이들 발췌문은 CloudFront API를 직접적으로 호출하며, 컨텍스트에서 실행되어야 하는 더 큰 프로그램에서 발췌한 코드입니다. AWS SDK를 사용한 CloudFront의 시나리오에서 컨텍스트에 맞는 작업을 볼 수 있습니다.

다음 예제에는 가장 일반적으로 사용되는 작업만 포함되어 있습니다. 전체 목록은 Amazon CloudFront API 참조에서 확인하세요.

예제

- AWS SDK 또는 CLI와 함께 CreateDistribution 사용
- AWS SDK와 함께 CreateFunction 사용
- CLI로 CreateInvalidation 사용
- AWS SDK와 함께 CreateKeyGroup 사용
- AWS SDK 또는 CLI와 함께 CreatePublicKey 사용
- AWS SDK 또는 CLI와 함께 DeleteDistribution 사용
- CLI로 GetCloudFrontOriginAccessIdentity 사용
- CLI로 GetCloudFrontOriginAccessIdentityConfig 사용
- CLI로 GetDistribution 사용
- AWS SDK 또는 CLI와 함께 GetDistributionConfig 사용
- CLI로 ListCloudFrontOriginAccessIdentities 사용
- AWS SDK 또는 CLI와 함께 ListDistributions 사용
- AWS SDK 또는 CLI와 함께 UpdateDistribution 사용

## AWS SDK 또는 CLI와 함께 **CreateDistribution** 사용

다음 코드 예시는 `CreateDistribution`의 사용 방법을 보여 줍니다.

작업 예시는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 다중 테넌트 배포 및 배포 테넌트 생성
- CloudFront 시작하기

### CLI

#### AWS CLI

예제 1: CloudFront 배포를 만드는 방법

다음 `create-distribution` 예제에서는 명령줄 인수를 사용하여 이름이 `amzn-s3-demo-bucket`인 S3 버킷에 대한 배포를 생성하고 `index.html`을 기본 루트 객체로 지정합니다.

```
aws cloudfront create-distribution \
    --origin-domain-name amzn-s3-demo-bucket.s3.amazonaws.com \
    --default-root-object index.html
```

출력:

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/
EMLARXS9EXAMPLE",
    "ETag": "E9LHASXEXAMPLE",
    "Distribution": {
        "Id": "EMLARXS9EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EMLARXS9EXAMPLE",
        "Status": "InProgress",
        "LastModifiedTime": "2019-11-22T00:55:15.705Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-example",
```

```
            "Aliases": {
                "Quantity": 0
            },
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                        "OriginPath": "",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        }
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-
example",
                "ForwardedValues": {
                    "QueryString": false,
                    "Cookies": {
                        "Forward": "none"
                    },
                    "Headers": {
                        "Quantity": 0
                    },
                    "QueryStringCacheKeys": {
                        "Quantity": 0
                    }
                },
                "TrustedSigners": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
                "MinTTL": 0,
```

```
                "AllowedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ],
                    "CachedMethods": {
                        "Quantity": 2,
                        "Items": [
                            "HEAD",
                            "GET"
                        ]
                    }
                },
                "SmoothStreaming": false,
                "DefaultTTL": 86400,
                "MaxTTL": 31536000,
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FieldLevelEncryptionId": ""
            },
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
                "Quantity": 0
            },
            "Comment": "",
            "Logging": {
                "Enabled": false,
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "PriceClass": "PriceClass_All",
            "Enabled": true,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": true,
                "MinimumProtocolVersion": "TLSv1",
                "CertificateSource": "cloudfront"
            },
            "Restrictions": {
```

```
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "IsIPV6Enabled": true
        }
    }
}
```

예제 2: JSON 파일을 사용하여 CloudFront 배포를 만드는 방법

다음 `create-distribution` 예제에서는 JSON 파일을 사용하여 이름이 `amzn-s3-demo-bucket`인 S3 버킷에 대한 배포를 생성하고 `index.html`을 기본 루트 객체로 지정합니다.

```
aws cloudfront create-distribution \
    --distribution-config file://dist-config.json
```

`dist-config.json`의 콘텐츠:

```
{
    "CallerReference": "cli-example",
    "Aliases": {
        "Quantity": 0
    },
    "DefaultRootObject": "index.html",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                "OriginPath": "",
                "CustomHeaders": {
                    "Quantity": 0
                },
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                }
            }
        ]
```

```
    },
    "OriginGroups": {
        "Quantity": 0
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
        "ForwardedValues": {
            "QueryString": false,
            "Cookies": {
                "Forward": "none"
            },
            "Headers": {
                "Quantity": 0
            },
            "QueryStringCacheKeys": {
                "Quantity": 0
            }
        },
        "TrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "ViewerProtocolPolicy": "allow-all",
        "MinTTL": 0,
        "AllowedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ],
            "CachedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ]
            }
        },
        "SmoothStreaming": false,
        "DefaultTTL": 86400,
        "MaxTTL": 31536000,
        "Compress": false,
        "LambdaFunctionAssociations": {
            "Quantity": 0
```

```
        },
        "FieldLevelEncryptionId": ""
    },
    "CacheBehaviors": {
        "Quantity": 0
    },
    "CustomErrorResponses": {
        "Quantity": 0
    },
    "Comment": "",
    "Logging": {
        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
```

샘플 출력은 예제 1을 참조하시기 바랍니다.

예제 3: 인증서를 사용하여 CloudFront 다중 테넌트 배포를 생성하는 방법

다음 `create-distribution` 예제에서는 다중 테넌트를 지원하는 CloudFront 배포를 생성하고는 TLS 인증서를 지정합니다.

```
aws cloudfront create-distribution \
    --distribution-config file://dist-config.json
```

dist-config.json의 콘텐츠:

```
{
    "CallerReference": "cli-example-with-cert",
    "Comment": "CLI example distribution",
    "DefaultRootObject": "index.html",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "DomainName": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "OriginPath": "/{{tenantName}}",
                "CustomHeaders": {
                    "Quantity": 0
                },
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                }
            }
        ]
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
        "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
        "ViewerProtocolPolicy": "allow-all",
        "AllowedMethods": {
            "Quantity": 2,
            "Items": ["HEAD", "GET"],
            "CachedMethods": {
                "Quantity": 2,
                "Items": ["HEAD", "GET"]
            }
        }
    },
    "Enabled": true,
    "ViewerCertificate": {
        "ACMCertificateArn": "arn:aws:acm:us-
east-1:123456789012:certificate/191306a1-db01-49ca-90ef-fc414ee5dabc",
        "SSLSupportMethod": "sni-only"
    },
    "HttpVersion": "http2",
    "ConnectionMode": "tenant-only",
    "TenantConfig": {
```

```
        "ParameterDefinitions": [
            {
                "Name": "tenantName",
                "Definition": {
                    "StringSchema": {
                        "Comment": "tenantName parameter",
                        "DefaultValue": "root",
                        "Required": false
                    }
                }
            }
        ]
    }
}
```

출력:

```
{
    "Location": "https://cloudfront.amazonaws.com/2020-05-31/distribution/
E1HVIAU7UABC",
    "ETag": "E20LT7R1BABC",
    "Distribution": {
        "Id": "E1HVIAU7U12ABC",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/E1HVIAU7U12ABC",
        "Status": "InProgress",
        "LastModifiedTime": "2025-07-10T20:33:31.117000+00:00",
        "InProgressInvalidationBatches": 0,
        "DomainName": "example.com",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "ActiveTrustedKeyGroups": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-example-with-cert",
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
```

```
                        "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                        "DomainName": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                        "OriginPath": "/{{tenantName}}",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        },
                        "ConnectionAttempts": 3,
                        "ConnectionTimeout": 10,
                        "OriginShield": {
                            "Enabled": false
                        },
                        "OriginAccessControlId": ""
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                "TrustedKeyGroups": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
                "AllowedMethods": {
                    "Quantity": 2,
                    "Items": ["HEAD", "GET"],
                    "CachedMethods": {
                        "Quantity": 2,
                        "Items": ["HEAD", "GET"]
                    }
                },
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FunctionAssociations": {
                    "Quantity": 0
```

```
                },
                "FieldLevelEncryptionId": "",
                "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
                "GrpcConfig": {
                    "Enabled": false
                }
            },
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
                "Quantity": 0
            },
            "Comment": "CLI example distribution",
            "Logging": {
                "Enabled": false,
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "Enabled": true,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": false,
                "ACMCertificateArn": "arn:aws:acm:us-
east-1:123456789012:certificate/1954f095-11b6-4daf-9952-0c308a00abc",
                "SSLSupportMethod": "sni-only",
                "MinimumProtocolVersion": "TLSv1.2_2021",
                "Certificate": "arn:aws:acm:us-
east-1:123456789012:certificate/1954f095-11b6-4daf-9952-0c308a00abc",
                "CertificateSource": "acm"
            },
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "TenantConfig": {
                "ParameterDefinitions": [
                    {
                        "Name": "tenantName",
                        "Definition": {
```

```
                            "StringSchema": {
                                "Comment": "tenantName parameter",
                                "DefaultValue": "root",
                                "Required": false
                            }
                        }
                    }
                ]
            },
            "ConnectionMode": "tenant-only"
        }
    }
}
```

자세한 내용은 Amazon CloudFront 개발자 안내서의 배포 작업을 참조하세요.

예제 4: 인증서 없이 CloudFront 다중 테넌트 배포를 생성하는 방법

다음 `create-distribution` 예제에서는 TLS 인증서 없이 다중 테넌트를 지원하는 CloudFront 배포를 생성합니다.

```
aws cloudfront create-distribution \
    --distribution-config file://dist-config.json
```

`dist-config.json`의 콘텐츠:

```
{
    "CallerReference": "cli-example",
    "Comment": "CLI example distribution",
    "DefaultRootObject": "index.html",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "DomainName": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "OriginPath": "/{{tenantName}}",
                "CustomHeaders": {
                    "Quantity": 0
                },
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
```

```
                }
            }
        ]
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
        "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
        "ViewerProtocolPolicy": "allow-all",
        "AllowedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ],
            "CachedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ]
            }
        }
    },
    "Enabled": true,
    "HttpVersion": "http2",
    "ConnectionMode": "tenant-only",
    "TenantConfig": {
        "ParameterDefinitions": [
            {
                "Name": "tenantName",
                "Definition": {
                    "StringSchema": {
                        "Comment": "tenantName parameter",
                        "DefaultValue": "root",
                        "Required": false
                    }
                }
            }
        ]
    }
}
```

출력:

```
{
    "Location": "https://cloudfront.amazonaws.com/2020-05-31/distribution/
E2GJ5J9QN12ABC",
    "ETag": "E37YLVVQIABC",
    "Distribution": {
        "Id": "E2GJ5J9QNABC",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/E2GJ5J9QN12ABC",
        "Status": "InProgress",
        "LastModifiedTime": "2025-07-10T20:35:20.565000+00:00",
        "InProgressInvalidationBatches": 0,
        "DomainName": "example.com",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "ActiveTrustedKeyGroups": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-example-no-cert",
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                        "DomainName": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                        "OriginPath": "/{{tenantName}}",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        },
                        "ConnectionAttempts": 3,
                        "ConnectionTimeout": 10,
                        "OriginShield": {
                            "Enabled": false
                        },
                        "OriginAccessControlId": ""
                    }
```

```
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                "TrustedKeyGroups": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
                "AllowedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ],
                    "CachedMethods": {
                        "Quantity": 2,
                        "Items": [
                            "HEAD",
                            "GET"
                        ]
                    }
                },
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FunctionAssociations": {
                    "Quantity": 0
                },
                "FieldLevelEncryptionId": "",
                "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
                "GrpcConfig": {
                    "Enabled": false
                }
            },
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
```

```
                "Quantity": 0
            },
            "Comment": "CLI example distribution",
            "Logging": {
                "Enabled": false,
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "Enabled": true,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": true,
                "SSLSupportMethod": "sni-only",
                "MinimumProtocolVersion": "TLSv1",
                "CertificateSource": "cloudfront"
            },
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "TenantConfig": {
                "ParameterDefinitions": [
                    {
                        "Name": "tenantName",
                        "Definition": {
                            "StringSchema": {
                                "Comment": "tenantName parameter",
                                "DefaultValue": "root",
                                "Required": false
                            }
                        }
                    }
                ]
            },
            "ConnectionMode": "tenant-only"
        }
    }
}
```

자세한 내용은 Amazon CloudFront 개발자 안내서의 배포 구성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 CreateDistribution을 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

다음 예제에서는 Amazon Simple Storage Service(Amazon S3) 버킷을 콘텐츠 오리진으로 사용합니다.

배포를 생성한 후 코드는 CloudFrontWaiter를 생성하여 배포가 완료될 때까지 기다린 후 배포를 반환합니다.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.internal.waiters.ResponseOrException;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
 software.amazon.awssdk.services.cloudfront.model.CreateDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.Distribution;
import software.amazon.awssdk.services.cloudfront.model.GetDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.ItemSelection;
import software.amazon.awssdk.services.cloudfront.model.Method;
import software.amazon.awssdk.services.cloudfront.model.ViewerProtocolPolicy;
import software.amazon.awssdk.services.cloudfront.waiters.CloudFrontWaiter;
import software.amazon.awssdk.services.s3.S3Client;

import java.time.Instant;

public class CreateDistribution {

        private static final Logger logger =
 LoggerFactory.getLogger(CreateDistribution.class);
```

```
        public static Distribution createDistribution(CloudFrontClient
cloudFrontClient, S3Client s3Client,
                        final String bucketName, final String keyGroupId, final
String originAccessControlId) {

                final String region = s3Client.headBucket(b ->
b.bucket(bucketName)).sdkHttpResponse().headers()
                                .get("x-amz-bucket-region").get(0);
                final String originDomain = bucketName + ".s3." + region +
".amazonaws.com";
                String originId = originDomain; // Use the originDomain value for
the originId.

                // The service API requires some deprecated methods, such as
                // DefaultCacheBehavior.Builder#minTTL and #forwardedValue.
                CreateDistributionResponse createDistResponse =
cloudFrontClient.createDistribution(builder -> builder
                                .distributionConfig(b1 -> b1
                                                .origins(b2 -> b2
                                                                .quantity(1)
                                                                .items(b3 -> b3

.domainName(originDomain)

.id(originId)

.s3OriginConfig(builder4 -> builder4

              .originAccessIdentity(

                              ""))

.originAccessControlId(

              originAccessControlId)))
                                                .defaultCacheBehavior(b2 -> b2

.viewerProtocolPolicy(ViewerProtocolPolicy.ALLOW_ALL)

.targetOriginId(originId)
                                                                .minTTL(200L)

.forwardedValues(b5 -> b5
```

```
.cookies(cp -> cp

              .forward(ItemSelection.NONE))

.queryString(true))

.trustedKeyGroups(b3 -> b3

.quantity(1)

.items(keyGroupId)

.enabled(true))

.allowedMethods(b4 -> b4

.quantity(2)

.items(Method.HEAD, Method.GET)

.cachedMethods(b5 -> b5

              .quantity(2)

              .items(Method.HEAD,

                             Method.GET))))
                                                .cacheBehaviors(b -> b
                                                                .quantity(1)
                                                                .items(b2 -> b2

.pathPattern("/index.html")

.viewerProtocolPolicy(

              ViewerProtocolPolicy.ALLOW_ALL)

.targetOriginId(originId)

.trustedKeyGroups(b3 -> b3

              .quantity(1)
```

```
              .items(keyGroupId)

              .enabled(true))

.minTTL(200L)

.forwardedValues(b4 -> b4

              .cookies(cp -> cp

                              .forward(ItemSelection.NONE))

              .queryString(true))

.allowedMethods(b5 -> b5.quantity(2)

              .items(Method.HEAD,

                             Method.GET)

              .cachedMethods(b6 -> b6

                              .quantity(2)

                              .items(Method.HEAD,

                                             Method.GET)))))
                                              .enabled(true)
                                              .comment("Distribution built with
java")

.callerReference(Instant.now().toString())));

               final Distribution distribution =
createDistResponse.distribution();
               logger.info("Distribution created. DomainName: [{}]  Id: [{}]",
distribution.domainName(),
                               distribution.id());
               logger.info("Waiting for distribution to be deployed ...");
               try (CloudFrontWaiter cfWaiter =
CloudFrontWaiter.builder().client(cloudFrontClient).build()) {
                       ResponseOrException<GetDistributionResponse>
responseOrException = cfWaiter
```

```
                                        .waitUntilDistributionDeployed(builder ->
 builder.id(distribution.id()))
                                        .matched();
                        responseOrException.response()
                                        .orElseThrow(() -> new
 RuntimeException("Distribution not created"));
                        logger.info("Distribution deployed. DomainName: [{}]  Id:
 [{}]", distribution.domainName(),
                                        distribution.id());
                }
                return distribution;
        }
}
```

- API에 대한 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateDistribution을 참조하세요.

## PowerShell

### Tools for PowerShell V4

예 1: 로깅 및 캐싱으로 구성된 기본 CloudFront 배포를 생성합니다.

```
$origin = New-Object Amazon.CloudFront.Model.Origin
$origin.DomainName = "amzn-s3-demo-bucket.s3.amazonaws.com"
$origin.Id = "UniqueOrigin1"
$origin.S3OriginConfig = New-Object Amazon.CloudFront.Model.S3OriginConfig
$origin.S3OriginConfig.OriginAccessIdentity = ""
New-CFDistribution `
      -DistributionConfig_Enabled $true `
      -DistributionConfig_Comment "Test distribution" `
      -Origins_Item $origin `
      -Origins_Quantity 1 `
      -Logging_Enabled $true `
      -Logging_IncludeCookie $true `
      -Logging_Bucket amzn-s3-demo-logging-bucket.s3.amazonaws.com `
      -Logging_Prefix "help/" `
      -DistributionConfig_CallerReference Client1 `
      -DistributionConfig_DefaultRootObject index.html `
      -DefaultCacheBehavior_TargetOriginId $origin.Id `
      -ForwardedValues_QueryString $true `
      -Cookies_Forward all `
      -WhitelistedNames_Quantity 0 `
```

```
      -TrustedSigners_Enabled $false `
      -TrustedSigners_Quantity 0 `
      -DefaultCacheBehavior_ViewerProtocolPolicy allow-all `
      -DefaultCacheBehavior_MinTTL 1000 `
      -DistributionConfig_PriceClass "PriceClass_All" `
      -CacheBehaviors_Quantity 0 `
      -Aliases_Quantity 0
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 CreateDistribution을 참조하세요.

Tools for PowerShell V5

예 1: 로깅 및 캐싱으로 구성된 기본 CloudFront 배포를 생성합니다.

```
$origin = New-Object Amazon.CloudFront.Model.Origin
$origin.DomainName = "amzn-s3-demo-bucket.s3.amazonaws.com"
$origin.Id = "UniqueOrigin1"
$origin.S3OriginConfig = New-Object Amazon.CloudFront.Model.S3OriginConfig
$origin.S3OriginConfig.OriginAccessIdentity = ""
New-CFDistribution `
      -DistributionConfig_Enabled $true `
      -DistributionConfig_Comment "Test distribution" `
      -Origins_Item $origin `
      -Origins_Quantity 1 `
      -Logging_Enabled $true `
      -Logging_IncludeCookie $true `
      -Logging_Bucket amzn-s3-demo-logging-bucket.s3.amazonaws.com `
      -Logging_Prefix "help/" `
      -DistributionConfig_CallerReference Client1 `
      -DistributionConfig_DefaultRootObject index.html `
      -DefaultCacheBehavior_TargetOriginId $origin.Id `
      -ForwardedValues_QueryString $true `
      -Cookies_Forward all `
      -WhitelistedNames_Quantity 0 `
      -TrustedSigners_Enabled $false `
      -TrustedSigners_Quantity 0 `
      -DefaultCacheBehavior_ViewerProtocolPolicy allow-all `
      -DefaultCacheBehavior_MinTTL 1000 `
      -DistributionConfig_PriceClass "PriceClass_All" `
      -CacheBehaviors_Quantity 0 `
      -Aliases_Quantity 0
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 CreateDistribution을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## AWS SDK와 함께 **CreateFunction** 사용

다음 코드 예시는 `CreateFunction`의 사용 방법을 보여 줍니다.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

```
import software.amazon.awssdk.core.SdkBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.CloudFrontException;
import software.amazon.awssdk.services.cloudfront.model.CreateFunctionRequest;
import software.amazon.awssdk.services.cloudfront.model.CreateFunctionResponse;
import software.amazon.awssdk.services.cloudfront.model.FunctionConfig;
import software.amazon.awssdk.services.cloudfront.model.FunctionRuntime;
import java.io.InputStream;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateFunction {
```

```
   public static void main(String[] args) {
       final String usage = """

               Usage:
                   <functionName> <filePath>

               Where:
                   functionName - The name of the function to create.\s
                   filePath - The path to a file that contains the application
logic for the function.\s
               """;

       if (args.length != 2) {
           System.out.println(usage);
           System.exit(1);
       }

       String functionName = args[0];
       String filePath = args[1];
       CloudFrontClient cloudFrontClient = CloudFrontClient.builder()
               .region(Region.AWS_GLOBAL)
               .build();

       String funArn = createNewFunction(cloudFrontClient, functionName,
filePath);
       System.out.println("The function ARN is " + funArn);
       cloudFrontClient.close();
   }

   public static String createNewFunction(CloudFrontClient cloudFrontClient,
String functionName, String filePath) {
       try {
           InputStream fileIs =
CreateFunction.class.getClassLoader().getResourceAsStream(filePath);
           SdkBytes functionCode = SdkBytes.fromInputStream(fileIs);

           FunctionConfig config = FunctionConfig.builder()
                   .comment("Created by using the CloudFront Java API")
                   .runtime(FunctionRuntime.CLOUDFRONT_JS_1_0)
                   .build();

           CreateFunctionRequest functionRequest =
CreateFunctionRequest.builder()
                   .name(functionName)
```

```
                    .functionCode(functionCode)
                    .functionConfig(config)
                    .build();

            CreateFunctionResponse response =
 cloudFrontClient.createFunction(functionRequest);
            return response.functionSummary().functionMetadata().functionARN();

        } catch (CloudFrontException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateFunction을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CLI로 `CreateInvalidation` 사용

다음 코드 예시는 `CreateInvalidation`의 사용 방법을 보여 줍니다.

CLI

AWS CLI

CloudFront 배포에 대한 무효화를 생성하려면

다음 `create-invalidation` 예제는 지정된 CloudFront 배포의 지정된 파일에 대한 무효화를 생성합니다.

```
aws cloudfront create-invalidation \
    --distribution-id EDFDVBD6EXAMPLE \
    --paths "/example-path/example-file.jpg" "/example-path/example-file2.png"
```

출력:

```
{
```

```
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/
EDFDVBD6EXAMPLE/invalidation/I1JLWSDAP8FU89",
    "Invalidation": {
        "Id": "I1JLWSDAP8FU89",
        "Status": "InProgress",
        "CreateTime": "2019-12-05T18:24:51.407Z",
        "InvalidationBatch": {
            "Paths": {
                "Quantity": 2,
                "Items": [
                    "/example-path/example-file2.png",
                    "/example-path/example-file.jpg"
                ]
            },
            "CallerReference": "cli-1575570291-670203"
        }
    }
}
```

이전 예제에서는 AWS CLI가 자동으로 랜덤 `CallerReference`를 생성했습니다. 직접 `CallerReference`를 지정하거나 무효화 매개변수를 명령줄 인수로 전달하지 않으려면 JSON 파일을 사용할 수 있습니다. 다음 예제에서는 `inv-batch.json`라는 JSON 파일에 무효화 매개 변수를 제공하여 두 파일에 대한 무효화를 생성합니다

```
aws cloudfront create-invalidation \
    --distribution-id EDFDVBD6EXAMPLE \
    --invalidation-batch file://inv-batch.json
```

`inv-batch.json`의 콘텐츠:

```
{
    "Paths": {
        "Quantity": 2,
        "Items": [
            "/example-path/example-file.jpg",
            "/example-path/example-file2.png"
        ]
    },
    "CallerReference": "cli-example"
}
```

출력:

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/
EDFDVBD6EXAMPLE/invalidation/I2J0I21PCUYOIK",
    "Invalidation": {
        "Id": "I2J0I21PCUYOIK",
        "Status": "InProgress",
        "CreateTime": "2019-12-05T18:40:49.413Z",
        "InvalidationBatch": {
            "Paths": {
                "Quantity": 2,
                "Items": [
                    "/example-path/example-file.jpg",
                    "/example-path/example-file2.png"
                ]
            },
            "CallerReference": "cli-example"
        }
    }
}
```

- API 세부 정보는 AWS CLI 명령 참조의CreateInvalidation 섹션을 참조하세요.

## PowerShell

### Tools for PowerShell V4

예제 1: 이 예제에서는 ID가 EXAMPLENSTXAXE인 배포에 대해 새 무효화를 생성합니다. CallerReference는 사용자가 선택한 고유한 ID입니다. 이 경우 2019년 5월 15일 오전 9시를 나타내는 타임스탬프가 사용됩니다. $Paths 변수는 사용자가 원하지 않는 이미지 및 미디어 파일에 대한 세 가지 경로를 배포 캐시의 일부로 저장합니다. -Paths_Quantity 매개 변수 값은 -Paths_Item 매개 변수에 지정된 총 경로 수입니다.

```
$Paths = "/images/*.gif", "/images/image1.jpg", "/videos/*.mp4"
New-CFInvalidation -DistributionId "EXAMPLENSTXAXE" -
InvalidationBatch_CallerReference 20190515090000 -Paths_Item $Paths -
Paths_Quantity 3
```

출력:

```
Invalidation                          Location
```

```
------------                          --------

Amazon.CloudFront.Model.Invalidation https://cloudfront.amazonaws.com/2018-11-05/
distribution/EXAMPLENSTXAXE/invalidation/EXAMPLE8NOK9H
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 CreateInvalidation을 참조하세요.

Tools for PowerShell V5

예제 1: 이 예제에서는 ID가 EXAMPLENSTXAXE인 배포에 대해 새 무효화를 생성합니다. CallerReference는 사용자가 선택한 고유한 ID입니다. 이 경우 2019년 5월 15일 오전 9시를 나타내는 타임스탬프가 사용됩니다. $Paths 변수는 사용자가 원하지 않는 이미지 및 미디어 파일에 대한 세 가지 경로를 배포 캐시의 일부로 저장합니다. -Paths_Quantity 매개 변수 값은 -Paths_Item 매개 변수에 지정된 총 경로 수입니다.

```
$Paths = "/images/*.gif", "/images/image1.jpg", "/videos/*.mp4"
New-CFInvalidation -DistributionId "EXAMPLENSTXAXE" -
InvalidationBatch_CallerReference 20190515090000 -Paths_Item $Paths -
Paths_Quantity 3
```

출력:

```
Invalidation                          Location

------------                          --------

Amazon.CloudFront.Model.Invalidation https://cloudfront.amazonaws.com/2018-11-05/
distribution/EXAMPLENSTXAXE/invalidation/EXAMPLE8NOK9H
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 CreateInvalidation을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## AWS SDK와 함께 **CreateKeyGroup** 사용

다음 코드 예시는 CreateKeyGroup의 사용 방법을 보여 줍니다.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

키 그룹에는 서명된 URL 또는 쿠키를 확인하는 데 사용되는 공개 키가 최소 하나 이상 필요합니다.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;

import java.util.UUID;

public class CreateKeyGroup {
    private static final Logger logger =
 LoggerFactory.getLogger(CreateKeyGroup.class);

    public static String createKeyGroup(CloudFrontClient cloudFrontClient, String
 publicKeyId) {
        String keyGroupId = cloudFrontClient.createKeyGroup(b ->
 b.keyGroupConfig(c -> c
                .items(publicKeyId)
                .name("JavaKeyGroup" + UUID.randomUUID())))
                .keyGroup().id();
        logger.info("KeyGroup created with ID: [{}]", keyGroupId);
        return keyGroupId;
    }
}
```

- API에 대한 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateKeyGroup을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## AWS SDK 또는 CLI와 함께 **CreatePublicKey** 사용

다음 코드 예시는 CreatePublicKey의 사용 방법을 보여 줍니다.

CLI

AWS CLI

CloudFront 퍼블릭 키를 생성하려면

다음 예제에서는 pub-key-config.json이라는 JSON 파일로 파라미터를 제공하여 CloudFront 퍼블릭 키를 생성합니다. 이 명령을 사용하려면 먼저 PEM으로 인코딩된 퍼블릭 키가 있어야 합니다. 자세한 내용은 Amazon CloudFront 개발자 안내서의 RSA 키 페어 생성을 참조하세요.

```
aws cloudfront create-public-key \
    --public-key-config file://pub-key-config.json
```

pub-key-config.json 파일은 다음을 포함한 현재 문서의 JSON 문서입니다. 참고로 퍼블릭 키는 PEM 형식으로 인코딩됩니다.

```
{
    "CallerReference": "cli-example",
    "Name": "ExampleKey",
    "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks0lnd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1UOWcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAzO3ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95ylUQd9LfYTBzVZYG2tSesplcOkjM3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6ZqlV3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nq
+kGZ2NQ0FyIyT2eiLKOX5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nrwIDAQAB\n-----
END PUBLIC KEY-----\n",
    "Comment": "example public key"
}
```

출력:

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/public-key/
KDFB19YGCR002",
    "ETag": "E2QWRUHEXAMPLE",
```

```
    "PublicKey": {
        "Id": "KDFB19YGCR002",
        "CreatedTime": "2019-12-05T18:51:43.781Z",
        "PublicKeyConfig": {
            "CallerReference": "cli-example",
            "Name": "ExampleKey",
            "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks0lnd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1UOWcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAzO3ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95ylUQd9LfYTBzVZYG2tSesplcOkjM3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6ZqlV3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nq
+kGZ2NQ0FyIyT2eiLKOX5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nrwIDAQAB\n-----
END PUBLIC KEY-----\n",
            "Comment": "example public key"
        }
    }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 CreatePublicKey를 참조하세요.

## Java

### SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

다음 코드 예제는 퍼블릭 키를 읽고 Amazon CloudFront에 업로드합니다.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.CreatePublicKeyResponse;
import software.amazon.awssdk.utils.IoUtils;

import java.io.IOException;
import java.io.InputStream;
import java.util.UUID;
```

```
public class CreatePublicKey {
    private static final Logger logger =
 LoggerFactory.getLogger(CreatePublicKey.class);

    public static String createPublicKey(CloudFrontClient cloudFrontClient,
 String publicKeyFileName) {
        try (InputStream is =
 CreatePublicKey.class.getClassLoader().getResourceAsStream(publicKeyFileName)) {
            String publicKeyString = IoUtils.toUtf8String(is);
            CreatePublicKeyResponse createPublicKeyResponse = cloudFrontClient
                    .createPublicKey(b -> b.publicKeyConfig(c -> c
                            .name("JavaCreatedPublicKey" + UUID.randomUUID())
                            .encodedKey(publicKeyString)
                            .callerReference(UUID.randomUUID().toString())));
            String createdPublicKeyId = createPublicKeyResponse.publicKey().id();
            logger.info("Public key created with id: [{}]", createdPublicKeyId);
            return createdPublicKeyId;

        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}
```

- API에 대한 세부 정보는 AWS SDK for Java 2.x API 참조의 CreatePublicKey을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## AWS SDK 또는 CLI와 함께 **DeleteDistribution** 사용

다음 코드 예시는 DeleteDistribution의 사용 방법을 보여 줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- CloudFront 시작하기

## CLI

### AWS CLI

#### CloudFront 배포를 삭제하려면

다음 예제에서는 ID `EDFDVBD6EXAMPLE`을 사용하여 CloudFront 배포를 삭제합니다. 배포를 삭제하려면 먼저 배포를 비활성화해야 합니다. 배포를 비활성화하려면 update-distribution 명령을 사용하세요. 자세한 정보는 update-distribution 예시를 참조하세요.

배포가 비활성화된 경우 이를 삭제할 수 있습니다. 배포를 삭제하려면 배포의 `ETag`를 제공하는 `--if-match` 옵션을 사용해야 합니다. `ETag`를 가져오려면 get-distribution 또는 get-distribution-config 명령을 사용하세요.

```
aws cloudfront delete-distribution \
    --id EDFDVBD6EXAMPLE \
    --if-match E2QWRUHEXAMPLE
```

이 명령이 제대로 실행되면 출력이 표시되지 않습니다.

- API 세부 정보는 AWS CLI 명령 참조의 DeleteDistribution를 참조합니다.

## Java

### SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

다음 코드 예제는 배포를 비활성화됨으로 업데이트하고, 변경사항이 배포되기를 기다리는 웨이터를 사용한 다음 배포를 삭제합니다.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.internal.waiters.ResponseOrException;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
 software.amazon.awssdk.services.cloudfront.model.DeleteDistributionResponse;
```

```
import software.amazon.awssdk.services.cloudfront.model.DistributionConfig;
import software.amazon.awssdk.services.cloudfront.model.GetDistributionResponse;
import software.amazon.awssdk.services.cloudfront.waiters.CloudFrontWaiter;

public class DeleteDistribution {
        private static final Logger logger =
 LoggerFactory.getLogger(DeleteDistribution.class);

        public static void deleteDistribution(final CloudFrontClient
 cloudFrontClient, final String distributionId) {
                // First, disable the distribution by updating it.
                GetDistributionResponse response =
 cloudFrontClient.getDistribution(b -> b
                                .id(distributionId));
                String etag = response.eTag();
                DistributionConfig distConfig =
 response.distribution().distributionConfig();

                cloudFrontClient.updateDistribution(builder -> builder
                                .id(distributionId)
                                .distributionConfig(builder1 -> builder1

 .cacheBehaviors(distConfig.cacheBehaviors())

 .defaultCacheBehavior(distConfig.defaultCacheBehavior())
                                                .enabled(false)
                                                .origins(distConfig.origins())
                                                .comment(distConfig.comment())

 .callerReference(distConfig.callerReference())

 .defaultCacheBehavior(distConfig.defaultCacheBehavior())

 .priceClass(distConfig.priceClass())
                                                .aliases(distConfig.aliases())
                                                .logging(distConfig.logging())

 .defaultRootObject(distConfig.defaultRootObject())

 .customErrorResponses(distConfig.customErrorResponses())

 .httpVersion(distConfig.httpVersion())

 .isIPV6Enabled(distConfig.isIPV6Enabled())
```

```
 .restrictions(distConfig.restrictions())

 .viewerCertificate(distConfig.viewerCertificate())
                                                .webACLId(distConfig.webACLId())

 .originGroups(distConfig.originGroups()))
                               .ifMatch(etag));

               logger.info("Distribution [{}] is DISABLED, waiting for
 deployment before deleting ...",
                               distributionId);
               GetDistributionResponse distributionResponse;
               try (CloudFrontWaiter cfWaiter =
 CloudFrontWaiter.builder().client(cloudFrontClient).build()) {
                       ResponseOrException<GetDistributionResponse>
 responseOrException = cfWaiter
                                       .waitUntilDistributionDeployed(builder ->
 builder.id(distributionId)).matched();
                       distributionResponse = responseOrException.response()
                                       .orElseThrow(() -> new
 RuntimeException("Could not disable distribution"));
               }

               DeleteDistributionResponse deleteDistributionResponse =
 cloudFrontClient
                               .deleteDistribution(builder -> builder
                                                .id(distributionId)

 .ifMatch(distributionResponse.eTag()));
               if (deleteDistributionResponse.sdkHttpResponse().isSuccessful())
 {
                       logger.info("Distribution [{}] DELETED", distributionId);
               }
       }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 DeleteDistribution를 참조합니다.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CLI로 **GetCloudFrontOriginAccessIdentity** 사용

다음 코드 예시는 GetCloudFrontOriginAccessIdentity의 사용 방법을 보여 줍니다.

CLI

AWS CLI

CloudFront 원본 액세스 ID를 가져오려면

다음 예제는 ID가 E74FTE3AEXAMPLE인 CloudFront 오리진 액세스 ID(OAI) 및 그에 딸린 ETag와 관련된 S3 카노니컬 ID를 가져옵니다. OAI ID는 create-cloud-front-origin-access-identity 및 list-cloud-front-origin-access-identities 명령의 출력에 반환됩니다.

```
aws cloudfront get-cloud-front-origin-access-identity --id E74FTE3AEXAMPLE
```

출력:

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "CloudFrontOriginAccessIdentity": {
        "Id": "E74FTE3AEXAMPLE",
        "S3CanonicalUserId":
 "cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
        "CloudFrontOriginAccessIdentityConfig": {
            "CallerReference": "cli-example",
            "Comment": "Example OAI"
        }
    }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 GetCloudFrontOriginAccessIdentity 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 이 예제는 -Id 파라미터로 지정된 특정 Amazon CloudFront 원본 액세스 ID를 반환합니다. -Id 파라미터가 필수는 아니지만 이 파라미터를 지정하지 않으면 결과가 반환되지 않습니다.

```
Get-CFCloudFrontOriginAccessIdentity -Id E3XXXXXXXXXXRT
```

출력:

```
     CloudFrontOriginAccessIdentityConfig    Id
    S3CanonicalUserId
     ------------------------------------    --
    -----------------
     Amazon.CloudFront.Model.CloudFrontOr... E3XXXXXXXXXXRT
    4b6e...
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 GetCloudFrontOriginAccessIdentity를 참조하세요.

Tools for PowerShell V5

예 1: 이 예제는 -Id 파라미터로 지정된 특정 Amazon CloudFront 원본 액세스 ID를 반환합니다. -Id 파라미터가 필수는 아니지만 이 파라미터를 지정하지 않으면 결과가 반환되지 않습니다.

```
Get-CFCloudFrontOriginAccessIdentity -Id E3XXXXXXXXXXRT
```

출력:

```
     CloudFrontOriginAccessIdentityConfig    Id
    S3CanonicalUserId
     ------------------------------------    --
    -----------------
     Amazon.CloudFront.Model.CloudFrontOr... E3XXXXXXXXXXRT
    4b6e...
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 GetCloudFrontOriginAccessIdentity를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CLI로 **GetCloudFrontOriginAccessIdentityConfig** 사용

다음 코드 예시는 `GetCloudFrontOriginAccessIdentityConfig`의 사용 방법을 보여 줍니다.

## CLI

### AWS CLI

CloudFront 원본 액세스 ID 구성을 가져오려면

다음 예제에서는 ID E74FTE3AEXAMPLE을 사용하여 ETag를 포함한 CloudFront 오리진 액세스 ID(OAI)에 대한 메타데이터를 가져옵니다. OAI ID는 create-cloud-front-origin-access-identity 및 list-cloud-front-origin-access-identities 명령의 출력에 반환됩니다.

```
aws cloudfront get-cloud-front-origin-access-identity-config --id E74FTE3AEXAMPLE
```

출력:

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "CloudFrontOriginAccessIdentityConfig": {
        "CallerReference": "cli-example",
        "Comment": "Example OAI"
    }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 GetCloudFrontOriginAccessIdentityConfig 섹션을 참조하세요.

## PowerShell

### Tools for PowerShell V4

예 1: 이 예제는 -Id 파라미터로 지정된 단일 Amazon CloudFront 원본 액세스 ID에 대한 구성 정보를 반환합니다. -Id 파라미터가 지정되지 않은 경우 오류가 발생합니다.

```
Get-CFCloudFrontOriginAccessIdentityConfig -Id E3XXXXXXXXXXRT
```

출력:

```
      CallerReference                                           Comment
      ---------------                                           -------
      mycallerreference: 2/1/2011 1:16:32 PM                    Caller
 reference: 2/1/2011 1:16:32 PM
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 GetCloudFrontOriginAccessIdentityConfig를 참조하세요.

Tools for PowerShell V5

예 1: 이 예제는 -Id 파라미터로 지정된 단일 Amazon CloudFront 원본 액세스 ID에 대한 구성 정보를 반환합니다. -Id 파라미터가 지정되지 않은 경우 오류가 발생합니다.

```
Get-CFCloudFrontOriginAccessIdentityConfig -Id E3XXXXXXXXXXRT
```

출력:

```
     CallerReference                                            Comment
     ---------------                                            -------
     mycallerreference: 2/1/2011 1:16:32 PM                     Caller
reference: 2/1/2011 1:16:32 PM
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 GetCloudFrontOriginAccessIdentityConfig를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CLI로 **GetDistribution** 사용

다음 코드 예시는 `GetDistribution`의 사용 방법을 보여 줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- CloudFront 시작하기

CLI

AWS CLI

CloudFront 배포를 가져오려면

다음 `get-distribution` 예제에서는 `ETag`를 비롯해 `EDFDVBD6EXAMPLE` ID를 가진 CloudFront 배포를 가져옵니다. 배포 ID는 create-distribution 및 list-distributions 명령에서 반환됩니다.

```
aws cloudfront get-distribution \
    --id EDFDVBD6EXAMPLE
```

출력:

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "Distribution": {
        "Id": "EDFDVBD6EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",
        "Status": "Deployed",
        "LastModifiedTime": "2019-12-04T23:35:41.433Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-example",
            "Aliases": {
                "Quantity": 0
            },
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                        "OriginPath": "",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        }
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
```

```
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-
example",
                "ForwardedValues": {
                    "QueryString": false,
                    "Cookies": {
                        "Forward": "none"
                    },
                    "Headers": {
                        "Quantity": 0
                    },
                    "QueryStringCacheKeys": {
                        "Quantity": 0
                    }
                },
                "TrustedSigners": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
                "MinTTL": 0,
                "AllowedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ],
                    "CachedMethods": {
                        "Quantity": 2,
                        "Items": [
                            "HEAD",
                            "GET"
                        ]
                    }
                },
                "SmoothStreaming": false,
                "DefaultTTL": 86400,
                "MaxTTL": 31536000,
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FieldLevelEncryptionId": ""
            },
```

```
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
                "Quantity": 0
            },
            "Comment": "",
            "Logging": {
                "Enabled": false,
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "PriceClass": "PriceClass_All",
            "Enabled": true,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": true,
                "MinimumProtocolVersion": "TLSv1",
                "CertificateSource": "cloudfront"
            },
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "IsIPV6Enabled": true
        }
    }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 GetDistribution 섹션을 참조하세요.

## PowerShell

### Tools for PowerShell V4

예 1: 특정 배포에 대한 정보를 검색합니다.

```
Get-CFDistribution -Id EXAMPLE0000ID
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 GetDistribution을 참조하세요.

Tools for PowerShell V5

예 1: 특정 배포에 대한 정보를 검색합니다.

```
Get-CFDistribution -Id EXAMPLE0000ID
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 GetDistribution을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## AWS SDK 또는 CLI와 함께 **GetDistributionConfig** 사용

다음 코드 예시는 GetDistributionConfig의 사용 방법을 보여 줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- CloudFront 시작하기

CLI

AWS CLI

CloudFront 배포 구성을 가져오려면

다음 예제에서는 ID EDFDVBD6EXAMPLE을 사용하여 ETag를 포함한 CloudFront 배포에 대한 메타데이터를 가져옵니다. 배포 ID는 create-distribution 및 list-distributions 명령에서 반환됩니다.

```
aws cloudfront get-distribution-config \
    --id EDFDVBD6EXAMPLE
```

출력:

```
{
    "ETag": "E2QWRUHEXAMPLE",
```

```
    "DistributionConfig": {
        "CallerReference": "cli-example",
        "Aliases": {
            "Quantity": 0
        },
        "DefaultRootObject": "index.html",
        "Origins": {
            "Quantity": 1,
            "Items": [
                {
                    "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                    "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                    "OriginPath": "",
                    "CustomHeaders": {
                        "Quantity": 0
                    },
                    "S3OriginConfig": {
                        "OriginAccessIdentity": ""
                    }
                }
            ]
        },
        "OriginGroups": {
            "Quantity": 0
        },
        "DefaultCacheBehavior": {
            "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
            "ForwardedValues": {
                "QueryString": false,
                "Cookies": {
                    "Forward": "none"
                },
                "Headers": {
                    "Quantity": 0
                },
                "QueryStringCacheKeys": {
                    "Quantity": 0
                }
            },
            "TrustedSigners": {
                "Enabled": false,
                "Quantity": 0
            },
            "ViewerProtocolPolicy": "allow-all",
```

```
            "MinTTL": 0,
            "AllowedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ],
                "CachedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ]
                }
            },
            "SmoothStreaming": false,
            "DefaultTTL": 86400,
            "MaxTTL": 31536000,
            "Compress": false,
            "LambdaFunctionAssociations": {
                "Quantity": 0
            },
            "FieldLevelEncryptionId": ""
        },
        "CacheBehaviors": {
            "Quantity": 0
        },
        "CustomErrorResponses": {
            "Quantity": 0
        },
        "Comment": "",
        "Logging": {
            "Enabled": false,
            "IncludeCookies": false,
            "Bucket": "",
            "Prefix": ""
        },
        "PriceClass": "PriceClass_All",
        "Enabled": true,
        "ViewerCertificate": {
            "CloudFrontDefaultCertificate": true,
            "MinimumProtocolVersion": "TLSv1",
            "CertificateSource": "cloudfront"
        },
```

```
        "Restrictions": {
            "GeoRestriction": {
                "RestrictionType": "none",
                "Quantity": 0
            }
        },
        "WebACLId": "",
        "HttpVersion": "http2",
        "IsIPV6Enabled": true
    }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 GetDistributionConfig를 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 특정 배포에 대한 구성을 검색합니다.

```
Get-CFDistributionConfig -Id EXAMPLE0000ID
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 GetDistributionConfig를 참조하세요.

Tools for PowerShell V5

예 1: 특정 배포에 대한 구성을 검색합니다.

```
Get-CFDistributionConfig -Id EXAMPLE0000ID
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 GetDistributionConfig를 참조하세요.

Python

SDK for Python(Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

```
class CloudFrontWrapper:
    """Encapsulates Amazon CloudFront operations."""

    def __init__(self, cloudfront_client):
        """
        :param cloudfront_client: A Boto3 CloudFront client
        """
        self.cloudfront_client = cloudfront_client


    def update_distribution(self):
        distribution_id = input(
            "This script updates the comment for a CloudFront distribution.\n"
            "Enter a CloudFront distribution ID: "
        )

        distribution_config_response =
 self.cloudfront_client.get_distribution_config(
            Id=distribution_id
        )
        distribution_config = distribution_config_response["DistributionConfig"]
        distribution_etag = distribution_config_response["ETag"]

        distribution_config["Comment"] = input(
            f"\nThe current comment for distribution {distribution_id} is "
            f"'{distribution_config['Comment']}'.\n"
            f"Enter a new comment: "
        )
        self.cloudfront_client.update_distribution(
            DistributionConfig=distribution_config,
            Id=distribution_id,
            IfMatch=distribution_etag,
```

```
        )
        print("Done!")
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 GetDistributionConfig를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CLI로 **ListCloudFrontOriginAccessIdentities** 사용

다음 코드 예시는 `ListCloudFrontOriginAccessIdentities`의 사용 방법을 보여 줍니다.

CLI

AWS CLI

CloudFront 오리진 액세스 ID를 나열하려면

다음 예제에서는 AWS 계정의 CloudFront 원본 액세스 ID(OAI) 목록을 가져옵니다.

```
aws cloudfront list-cloud-front-origin-access-identities
```

출력:

```
{
    "CloudFrontOriginAccessIdentityList": {
        "Items": [
            {
                "Id": "E74FTE3AEXAMPLE",
                "S3CanonicalUserId":
 "cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
                "Comment": "Example OAI"
            },
            {
                "Id": "EH1HDMBEXAMPLE",
                "S3CanonicalUserId":
 "1489f6f2e6faacaae7ff64c4c3e6956c24f78788abfc1718c3527c263bf7a17EXAMPLE",
                "Comment": "Test OAI"
```

```
            },
            {
                "Id": "E2X2C9TEXAMPLE",
                "S3CanonicalUserId":
 "cbfeebb915a64749f9be546a45b3fcfd3a31c779673c13c4dd460911ae402c2EXAMPLE",
                "Comment": "Example OAI #2"
            }
        ]
    }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 ListCloudFrontOriginAccessIdentities 섹션을 참조하세요.

## PowerShell

### Tools for PowerShell V4

예 1: 이 예제는 Amazon CloudFront 원본 액세스 ID 목록을 반환합니다. -MaxItem 파라미터는 값 2를 지정하므로 결과에는 두 개의 ID가 포함됩니다.

```
Get-CFCloudFrontOriginAccessIdentityList -MaxItem 2
```

출력:

```
IsTruncated : True
Items       : {E326XXXXXXXXXT, E1YWXXXXXXX9B}
Marker      :
MaxItems    : 2
NextMarker  : E1YXXXXXXXXX9B
Quantity    : 2
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 ListCloudFrontOriginAccessIdentities를 참조하세요.

### Tools for PowerShell V5

예 1: 이 예제는 Amazon CloudFront 원본 액세스 ID 목록을 반환합니다. -MaxItem 파라미터는 값 2를 지정하므로 결과에는 두 개의 ID가 포함됩니다.

```
Get-CFCloudFrontOriginAccessIdentityList -MaxItem 2
```

출력:

```
IsTruncated : True
Items       : {E326XXXXXXXXXT, E1YWXXXXXXX9B}
Marker      :
MaxItems    : 2
NextMarker  : E1YXXXXXXXXX9B
Quantity    : 2
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 ListCloudFrontOriginAccessIdentities를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## AWS SDK 또는 CLI와 함께 **ListDistributions** 사용

다음 코드 예시는 `ListDistributions`의 사용 방법을 보여 줍니다.

CLI

AWS CLI

CloudFront 배포를 나열하려면

다음 예제에서는 AWS 계정의 CloudFront 배포 목록을 가져옵니다.

```
aws cloudfront list-distributions
```

출력:

```
{
    "DistributionList": {
        "Items": [
            {
                "Id": "E23YS8OEXAMPLE",
                "ARN": "arn:aws:cloudfront::123456789012:distribution/
E23YS8OEXAMPLE",
                "Status": "Deployed",
                "LastModifiedTime": "2024-08-05T18:23:40.375000+00:00",
                "DomainName": "abcdefgh12ijk.cloudfront.net",
                "Aliases": {
```

```
                    "Quantity": 0
                },
                "Origins": {
                    "Quantity": 1,
                    "Items": [
                        {
                            "Id": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                            "DomainName": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                            "OriginPath": "",
                            "CustomHeaders": {
                                "Quantity": 0
                            },
                            "S3OriginConfig": {
                                "OriginAccessIdentity": ""
                            },
                            "ConnectionAttempts": 3,
                            "ConnectionTimeout": 10,
                            "OriginShield": {
                                "Enabled": false
                            },
                            "OriginAccessControlId": "EIAP8PEXAMPLE"
                        }
                    ]
                },
                "OriginGroups": {
                    "Quantity": 0
                },
                "DefaultCacheBehavior": {
                    "TargetOriginId": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                    "TrustedSigners": {
                        "Enabled": false,
                        "Quantity": 0
                    },
                    "TrustedKeyGroups": {
                        "Enabled": false,
                        "Quantity": 0
                    },
                    "ViewerProtocolPolicy": "allow-all",
                    "AllowedMethods": {
                        "Quantity": 2,
                        "Items": [
```

```
                            "HEAD",
                            "GET"
                        ],
                        "CachedMethods": {
                            "Quantity": 2,
                            "Items": [
                                "HEAD",
                                "GET"
                            ]
                        }
                    },
                    "SmoothStreaming": false,
                    "Compress": true,
                    "LambdaFunctionAssociations": {
                        "Quantity": 0
                    },
                    "FunctionAssociations": {
                        "Quantity": 0
                    },
                    "FieldLevelEncryptionId": "",
                    "CachePolicyId": "658327ea-f89d-4fab-a63d-7e886EXAMPLE"
                },
                "CacheBehaviors": {
                    "Quantity": 0
                },
                "CustomErrorResponses": {
                    "Quantity": 0
                },
                "Comment": "",
                "PriceClass": "PriceClass_All",
                "Enabled": true,
                "ViewerCertificate": {
                    "CloudFrontDefaultCertificate": true,
                    "SSLSupportMethod": "vip",
                    "MinimumProtocolVersion": "TLSv1",
                    "CertificateSource": "cloudfront"
                },
                "Restrictions": {
                    "GeoRestriction": {
                        "RestrictionType": "none",
                        "Quantity": 0
                    }
                },
                "WebACLId": "",
```

```
                "HttpVersion": "HTTP2",
                "IsIPV6Enabled": true,
                "Staging": false
            }
        ]
    }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 ListDistributions 섹션을 참조하세요.

## PowerShell

### Tools for PowerShell V4

예 1: 배포판을 반환합니다.

```
Get-CFDistributionList
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 ListDistributions를 참조하세요.

### Tools for PowerShell V5

예 1: 배포판을 반환합니다.

```
Get-CFDistributionList
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 ListDistributions를 참조하세요.

## Python

### SDK for Python(Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

```
class CloudFrontWrapper:
    """Encapsulates Amazon CloudFront operations."""

    def __init__(self, cloudfront_client):
        """
        :param cloudfront_client: A Boto3 CloudFront client
        """
        self.cloudfront_client = cloudfront_client


    def list_distributions(self):
        print("CloudFront distributions:\n")
        distributions = self.cloudfront_client.list_distributions()
        if distributions["DistributionList"]["Quantity"] > 0:
            for distribution in distributions["DistributionList"]["Items"]:
                print(f"Domain: {distribution['DomainName']}")
                print(f"Distribution Id: {distribution['Id']}")
                print(
                    f"Certificate Source: "
                    f"{distribution['ViewerCertificate']['CertificateSource']}"
                )
                if distribution["ViewerCertificate"]["CertificateSource"] ==
 "acm":
                    print(
                        f"Certificate: {distribution['ViewerCertificate']
['Certificate']}"
                    )
                print("")
        else:
            print("No CloudFront distributions detected.")
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 ListDistributions를 참조하세요.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   TRY.
       oo_result = lo_fnt->listdistributions( ). " oo_result is returned for
testing purposes. "
       MESSAGE 'Retrieved list of CloudFront distributions.' TYPE 'I'.
     CATCH /aws1/cx_fntinvalidargument.
       MESSAGE 'Invalid argument provided.' TYPE 'E'.
   ENDTRY.
```

- API 세부 정보는 AWS SDK for SAP ABAP API 참조의 ListDistributions를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## AWS SDK 또는 CLI와 함께 **UpdateDistribution** 사용

다음 코드 예시는 `UpdateDistribution`의 사용 방법을 보여 줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- CloudFront 시작하기

CLI

AWS CLI

예제 1: CloudFront 배포의 기본 루트 객체를 업데이트하는 방법

다음 예제에서는 `EDFDVBD6EXAMPLE` ID를 가진 CloudFront 배포에서 기본 루트 객체를 `index.html`로 업데이트합니다.

```
aws cloudfront update-distribution \
    --id EDFDVBD6EXAMPLE \
    --default-root-object index.html
```

출력:

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "Distribution": {
        "Id": "EDFDVBD6EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",
        "Status": "InProgress",
        "LastModifiedTime": "2019-12-06T18:55:39.870Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "6b10378d-49be-4c4b-a642-419ccaf8f3b5",
            "Aliases": {
                "Quantity": 0
            },
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "example-website",
                        "DomainName": "www.example.com",
                        "OriginPath": "",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "CustomOriginConfig": {
                            "HTTPPort": 80,
                            "HTTPSPort": 443,
                            "OriginProtocolPolicy": "match-viewer",
                            "OriginSslProtocols": {
                                "Quantity": 2,
                                "Items": [
                                    "SSLv3",
```

```
                                "TLSv1"
                            ]
                        },
                        "OriginReadTimeout": 30,
                        "OriginKeepaliveTimeout": 5
                    }
                }
            ]
        },
        "OriginGroups": {
            "Quantity": 0
        },
        "DefaultCacheBehavior": {
            "TargetOriginId": "example-website",
            "ForwardedValues": {
                "QueryString": false,
                "Cookies": {
                    "Forward": "none"
                },
                "Headers": {
                    "Quantity": 1,
                    "Items": [
                        "*"
                    ]
                },
                "QueryStringCacheKeys": {
                    "Quantity": 0
                }
            },
            "TrustedSigners": {
                "Enabled": false,
                "Quantity": 0
            },
            "ViewerProtocolPolicy": "allow-all",
            "MinTTL": 0,
            "AllowedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ],
                "CachedMethods": {
                    "Quantity": 2,
                    "Items": [
```

```
                        "HEAD",
                        "GET"
                    ]
                }
            },
            "SmoothStreaming": false,
            "DefaultTTL": 86400,
            "MaxTTL": 31536000,
            "Compress": false,
            "LambdaFunctionAssociations": {
                "Quantity": 0
            },
            "FieldLevelEncryptionId": ""
        },
        "CacheBehaviors": {
            "Quantity": 0
        },
        "CustomErrorResponses": {
            "Quantity": 0
        },
        "Comment": "",
        "Logging": {
            "Enabled": false,
            "IncludeCookies": false,
            "Bucket": "",
            "Prefix": ""
        },
        "PriceClass": "PriceClass_All",
        "Enabled": true,
        "ViewerCertificate": {
            "CloudFrontDefaultCertificate": true,
            "MinimumProtocolVersion": "TLSv1",
            "CertificateSource": "cloudfront"
        },
        "Restrictions": {
            "GeoRestriction": {
                "RestrictionType": "none",
                "Quantity": 0
            }
        },
        "WebACLId": "",
        "HttpVersion": "http1.1",
        "IsIPV6Enabled": true
    }
```

```
    }
}
```

예제 2: CloudFront 배포를 업데이트하는 방법

다음 예제에서는 `dist-config-disable.json`이라는 JSON 파일로 배포 구성을 제공하여 ID `EMLARXS9EXAMPLE`을 사용한 CloudFront 배포를 비활성화합니다. 배포를 업데이트하려면 배포의 `ETag`를 제공하는 `--if-match` 옵션을 사용해야 합니다. `ETag`를 가져오려면 get-distribution 또는 get-distribution-config 명령을 사용하세요. JSON 파일에서 `Enabled` 필드가 `false`로 설정되어 있습니다.

다음 예제를 사용하여 배포를 비활성화한 후 delete-distribution 명령을 사용하여 배포를 삭제할 수 있습니다.

```
aws cloudfront update-distribution \
    --id EMLARXS9EXAMPLE \
    --if-match E2QWRUHEXAMPLE \
    --distribution-config file://dist-config-disable.json
```

`dist-config-disable.json`의 콘텐츠:

```
{
    "CallerReference": "cli-1574382155-496510",
    "Aliases": {
        "Quantity": 0
    },
    "DefaultRootObject": "index.html",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-1574382155-273939",
                "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                "OriginPath": "",
                "CustomHeaders": {
                    "Quantity": 0
                },
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                }
            }
        ]
```

```
    },
    "OriginGroups": {
        "Quantity": 0
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-
bucket.s3.amazonaws.com-1574382155-273939",
        "ForwardedValues": {
            "QueryString": false,
            "Cookies": {
                "Forward": "none"
            },
            "Headers": {
                "Quantity": 0
            },
            "QueryStringCacheKeys": {
                "Quantity": 0
            }
        },
        "TrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "ViewerProtocolPolicy": "allow-all",
        "MinTTL": 0,
        "AllowedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ],
            "CachedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ]
            }
        },
        "SmoothStreaming": false,
        "DefaultTTL": 86400,
        "MaxTTL": 31536000,
        "Compress": false,
        "LambdaFunctionAssociations": {
```

```
            "Quantity": 0
        },
        "FieldLevelEncryptionId": ""
    },
    "CacheBehaviors": {
        "Quantity": 0
    },
    "CustomErrorResponses": {
        "Quantity": 0
    },
    "Comment": "",
    "Logging": {
        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": false,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
```

출력:

```
{
    "ETag": "E9LHASXEXAMPLE",
    "Distribution": {
        "Id": "EMLARXS9EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EMLARXS9EXAMPLE",
        "Status": "InProgress",
```

```
        "LastModifiedTime": "2019-12-06T18:32:35.553Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-1574382155-496510",
            "Aliases": {
                "Quantity": 0
            },
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "amzn-s3-demo-
bucket.s3.amazonaws.com-1574382155-273939",
                        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                        "OriginPath": "",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        }
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-
bucket.s3.amazonaws.com-1574382155-273939",
                "ForwardedValues": {
                    "QueryString": false,
                    "Cookies": {
                        "Forward": "none"
                    },
                    "Headers": {
                        "Quantity": 0
                    },
```

```
                "QueryStringCacheKeys": {
                    "Quantity": 0
                }
            },
            "TrustedSigners": {
                "Enabled": false,
                "Quantity": 0
            },
            "ViewerProtocolPolicy": "allow-all",
            "MinTTL": 0,
            "AllowedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ],
                "CachedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ]
                }
            },
            "SmoothStreaming": false,
            "DefaultTTL": 86400,
            "MaxTTL": 31536000,
            "Compress": false,
            "LambdaFunctionAssociations": {
                "Quantity": 0
            },
            "FieldLevelEncryptionId": ""
        },
        "CacheBehaviors": {
            "Quantity": 0
        },
        "CustomErrorResponses": {
            "Quantity": 0
        },
        "Comment": "",
        "Logging": {
            "Enabled": false,
            "IncludeCookies": false,
            "Bucket": "",
```

```
                "Prefix": ""
            },
            "PriceClass": "PriceClass_All",
            "Enabled": false,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": true,
                "MinimumProtocolVersion": "TLSv1",
                "CertificateSource": "cloudfront"
            },
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "IsIPV6Enabled": true
        }
    }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 UpdateDistribution을 참조하세요.

## Java

### SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.GetDistributionRequest;
import software.amazon.awssdk.services.cloudfront.model.GetDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.Distribution;
import software.amazon.awssdk.services.cloudfront.model.DistributionConfig;
import
 software.amazon.awssdk.services.cloudfront.model.UpdateDistributionRequest;
```

```
import software.amazon.awssdk.services.cloudfront.model.CloudFrontException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ModifyDistribution {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <id>\s

                Where:
                    id - the id value of the distribution.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String id = args[0];
        CloudFrontClient cloudFrontClient = CloudFrontClient.builder()
                .region(Region.AWS_GLOBAL)
                .build();

        modDistribution(cloudFrontClient, id);
        cloudFrontClient.close();
    }

    public static void modDistribution(CloudFrontClient cloudFrontClient, String
 idVal) {
        try {
            // Get the Distribution to modify.
            GetDistributionRequest disRequest = GetDistributionRequest.builder()
                    .id(idVal)
                    .build();
```

```
            GetDistributionResponse response =
 cloudFrontClient.getDistribution(disRequest);
            Distribution disObject = response.distribution();
            DistributionConfig config = disObject.distributionConfig();

            // Create a new DistributionConfig object and add new values to
 comment and
            // aliases
            DistributionConfig config1 = DistributionConfig.builder()
                    .aliases(config.aliases()) // You can pass in new values here
                    .comment("New Comment")
                    .cacheBehaviors(config.cacheBehaviors())
                    .priceClass(config.priceClass())
                    .defaultCacheBehavior(config.defaultCacheBehavior())
                    .enabled(config.enabled())
                    .callerReference(config.callerReference())
                    .logging(config.logging())
                    .originGroups(config.originGroups())
                    .origins(config.origins())
                    .restrictions(config.restrictions())
                    .defaultRootObject(config.defaultRootObject())
                    .webACLId(config.webACLId())
                    .httpVersion(config.httpVersion())
                    .viewerCertificate(config.viewerCertificate())
                    .customErrorResponses(config.customErrorResponses())
                    .build();

            UpdateDistributionRequest updateDistributionRequest =
 UpdateDistributionRequest.builder()
                    .distributionConfig(config1)
                    .id(disObject.id())
                    .ifMatch(response.eTag())
                    .build();

            cloudFrontClient.updateDistribution(updateDistributionRequest);

        } catch (CloudFrontException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 UpdateDistribution을 참조하십시오.

## Python

### SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

```
class CloudFrontWrapper:
    """Encapsulates Amazon CloudFront operations."""

    def __init__(self, cloudfront_client):
        """
        :param cloudfront_client: A Boto3 CloudFront client
        """
        self.cloudfront_client = cloudfront_client


    def update_distribution(self):
        distribution_id = input(
            "This script updates the comment for a CloudFront distribution.\n"
            "Enter a CloudFront distribution ID: "
        )

        distribution_config_response =
 self.cloudfront_client.get_distribution_config(
            Id=distribution_id
        )
        distribution_config = distribution_config_response["DistributionConfig"]
        distribution_etag = distribution_config_response["ETag"]

        distribution_config["Comment"] = input(
            f"\nThe current comment for distribution {distribution_id} is "
            f"'{distribution_config['Comment']}'.\n"
            f"Enter a new comment: "
        )
        self.cloudfront_client.update_distribution(
            DistributionConfig=distribution_config,
```

```
            Id=distribution_id,
            IfMatch=distribution_etag,
        )
        print("Done!")
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 UpdateDistribution을 참조하세요.

## SAP ABAP

### SDK for SAP ABAP

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        " Get the current distribution configuration and ETag "
        DATA(lo_distribution_config_result) = lo_fnt-
>getdistributionconfig( iv_id = iv_distribution_id ).
        DATA(lo_old_config) = lo_distribution_config_result-
>get_distributionconfig( ).
        DATA(lv_etag) = lo_distribution_config_result->get_etag( ).

        " Create a new distribution config with the updated comment "
        " Since the config object is immutable, we need to create a new one with
 all existing values "
        DATA(lo_new_config) = NEW /aws1/cl_fntdistributionconfig(
          iv_callerreference = lo_old_config->get_callerreference( )
          io_aliases = lo_old_config->get_aliases( )
          iv_defaultrootobject = lo_old_config->get_defaultrootobject( )
          io_origins = lo_old_config->get_origins( )
          io_origingroups = lo_old_config->get_origingroups( )
          io_defaultcachebehavior = lo_old_config->get_defaultcachebehavior( )
          io_cachebehaviors = lo_old_config->get_cachebehaviors( )
          io_customerrorresponses = lo_old_config->get_customerrorresponses( )
          iv_comment = iv_comment
          io_logging = lo_old_config->get_logging( )
```

```
          iv_priceclass = lo_old_config->get_priceclass( )
          iv_enabled = lo_old_config->get_enabled( )
          io_viewercertificate = lo_old_config->get_viewercertificate( )
          io_restrictions = lo_old_config->get_restrictions( )
          iv_webaclid = lo_old_config->get_webaclid( )
          iv_httpversion = lo_old_config->get_httpversion( )
          iv_isipv6enabled = lo_old_config->get_isipv6enabled( ) ).

        " Update the distribution with the modified configuration "
        lo_fnt->updatedistribution(
          io_distributionconfig = lo_new_config
          iv_id = iv_distribution_id
          iv_ifmatch = lv_etag ).
        MESSAGE 'CloudFront distribution updated successfully.' TYPE 'I'.
      CATCH /aws1/cx_fntnosuchdistribution.
        MESSAGE 'Distribution does not exist.' TYPE 'E'.
      CATCH /aws1/cx_fntpreconditionfailed.
        MESSAGE 'Precondition failed - ETag mismatch.' TYPE 'E'.
      CATCH /aws1/cx_fntinvalidifmatchvrs.
        MESSAGE 'Invalid If-Match version.' TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 AWS SDK for SAP ABAP API 참조의 UpdateDistribution을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

# AWS SDK를 사용한 CloudFront의 시나리오

다음 코드 예제에서는 AWS SDK로 CloudFront에서 일반적인 시나리오를 구현하는 방법을 보여줍니다. 이러한 시나리오에서는 CloudFront 내에서 또는 다른 AWS 서비스와 결합된 상태에서 여러 함수를 직접 호출하여 특정 작업을 수행하는 방법을 보여줍니다. 각 시나리오에는 전체 소스 코드에 대한 링크가 포함되어 있습니다. 여기에서 코드를 설정 및 실행하는 방법에 대한 지침을 찾을 수 있습니다.

시나리오는 컨텍스트에 맞는 서비스 작업을 이해하는 데 도움이 되도록 중급 수준의 경험을 대상으로 합니다.

예시

- SaaS Manager 리소스 AWS SDK 생성

- AWS SDK를 사용하여 CloudFront 서명 리소스 삭제
- CLI를 사용하여 기본 CloudFront 배포 시작하기
- AWS SDK를 사용하여 서명된 URL과 쿠키 생성

## SaaS Manager 리소스 AWS SDK 생성

다음 코드 예제에서는 다양한 구성으로 다중 테넌트 배포 및 배포 테넌트를 생성하는 방법을 보여줍니다.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

다음 예제에서는 파라미터와 와일드카드 인증서를 사용하여 다중 테넌트 배포를 생성하는 방법을 보여줍니다.

```
import software.amazon.awssdk.core.internal.waiters.ResponseOrException;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.ConnectionMode;
import
 software.amazon.awssdk.services.cloudfront.model.CreateDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.Distribution;
import software.amazon.awssdk.services.cloudfront.model.GetDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.HttpVersion;
import software.amazon.awssdk.services.cloudfront.model.Method;
import software.amazon.awssdk.services.cloudfront.model.SSLSupportMethod;
import software.amazon.awssdk.services.cloudfront.model.ViewerProtocolPolicy;
import software.amazon.awssdk.services.cloudfront.waiters.CloudFrontWaiter;
import software.amazon.awssdk.services.s3.S3Client;

import java.time.Instant;

public class CreateMultiTenantDistribution {
```

```
   public static Distribution
CreateMultiTenantDistributionWithCert(CloudFrontClient cloudFrontClient,
                                                                        S3Client
s3Client,
                                                                        final String
bucketName,
                                                                        final String
certificateArn) {
       // fetch the origin info if necessary
       final String region = s3Client.headBucket(b ->
b.bucket(bucketName)).sdkHttpResponse().headers()
               .get("x-amz-bucket-region").get(0);
       final String originDomain = bucketName + ".s3." + region +
".amazonaws.com";
       String originId = originDomain; // Use the originDomain value for the
originId.

       CreateDistributionResponse createDistResponse =
cloudFrontClient.createDistribution(builder -> builder
               .distributionConfig(b1 -> b1
                       .httpVersion(HttpVersion.HTTP2)
                       .enabled(true)
                       .comment("Template Distribution with cert built with
java")
                       .connectionMode(ConnectionMode.TENANT_ONLY)
                       .callerReference(Instant.now().toString())
                       .viewerCertificate(certBuilder -> certBuilder
                               .acmCertificateArn(certificateArn)
                               .sslSupportMethod(SSLSupportMethod.SNI_ONLY))
                       .origins(b2 -> b2
                               .quantity(1)
                               .items(b3 -> b3
                                       .domainName(originDomain)
                                       .id(originId)
                                       .originPath("/{{tenantName}}")
                                       .s3OriginConfig(builder4 -> builder4
                                               .originAccessIdentity(
                                                       "")))
                       .tenantConfig(b5 -> b5
                               .parameterDefinitions(b6 -> b6
                                       .name("tenantName")
                                       .definition(b7 -> b7
                                               .stringSchema(b8 -> b8
```

```
                                                                .comment("tenantName
 value")
                                                                .defaultValue("root")
                                                                .required(false)))))
                        .defaultCacheBehavior(b2 -> b2

 .viewerProtocolPolicy(ViewerProtocolPolicy.ALLOW_ALL)
                                .targetOriginId(originId)
                                .cachePolicyId("658327ea-f89d-4fab-
a63d-7e88639e58f6") // CachingOptimized Policy
                                .allowedMethods(b4 -> b4
                                        .quantity(2)
                                        .items(Method.HEAD, Method.GET)))
                ));

        final Distribution distribution = createDistResponse.distribution();
        try (CloudFrontWaiter cfWaiter =
 CloudFrontWaiter.builder().client(cloudFrontClient).build()) {
            ResponseOrException<GetDistributionResponse> responseOrException =
 cfWaiter
                    .waitUntilDistributionDeployed(builder ->
 builder.id(distribution.id()))
                    .matched();
            responseOrException.response()
                    .orElseThrow(() -> new RuntimeException("Distribution not
 created"));
        }
        return distribution;
    }

    public static Distribution
 CreateMultiTenantDistributionNoCert(CloudFrontClient cloudFrontClient,
                                                                S3Client s3Client,
                                                                final String
 bucketName) {
        // fetch the origin info if necessary
        final String region = s3Client.headBucket(b ->
 b.bucket(bucketName)).sdkHttpResponse().headers()
                .get("x-amz-bucket-region").get(0);
        final String originDomain = bucketName + ".s3." + region +
 ".amazonaws.com";
        String originId = originDomain; // Use the originDomain value for the
 originId.
```

```
        CreateDistributionResponse createDistResponse =
 cloudFrontClient.createDistribution(builder -> builder
                .distributionConfig(b1 -> b1
                        .httpVersion(HttpVersion.HTTP2)
                        .enabled(true)
                        .comment("Template Distribution with cert built with
 java")
                        .connectionMode(ConnectionMode.TENANT_ONLY)
                        .callerReference(Instant.now().toString())
                        .origins(b2 -> b2
                                .quantity(1)
                                .items(b3 -> b3
                                        .domainName(originDomain)
                                        .id(originId)
                                        .originPath("/{{tenantName}}")
                                        .s3OriginConfig(builder4 -> builder4
                                                .originAccessIdentity(
                                                        "")))
                        .tenantConfig(b5 -> b5
                                .parameterDefinitions(b6 -> b6
                                        .name("tenantName")
                                        .definition(b7 -> b7
                                                .stringSchema(b8 -> b8
                                                        .comment("tenantName
 value")
                                                        .defaultValue("root")
                                                        .required(false)))))
                        .defaultCacheBehavior(b2 -> b2

 .viewerProtocolPolicy(ViewerProtocolPolicy.ALLOW_ALL)
                                .targetOriginId(originId)
                                .cachePolicyId("658327ea-f89d-4fab-
a63d-7e88639e58f6") // CachingOptimized Policy
                                .allowedMethods(b4 -> b4
                                        .quantity(2)
                                        .items(Method.HEAD, Method.GET)))
                ));

        final Distribution distribution = createDistResponse.distribution();
        try (CloudFrontWaiter cfWaiter =
 CloudFrontWaiter.builder().client(cloudFrontClient).build()) {
            ResponseOrException<GetDistributionResponse> responseOrException =
 cfWaiter
```

```
                    .waitUntilDistributionDeployed(builder ->
 builder.id(distribution.id()))
                    .matched();
            responseOrException.response()
                    .orElseThrow(() -> new RuntimeException("Distribution not
 created"));
        }
        return distribution;
    }
}
```

다음 예제에서는 위에서 선언한 파라미터를 사용하는 것을 포함하여 해당 템플릿과 연결된 배포 테넌트를 생성하는 방법을 보여줍니다. 도메인이 이미 상위 템플릿에 포함되어 있기 때문에 여기에 인증서 정보를 추가할 필요가 없습니다.

```
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
 software.amazon.awssdk.services.cloudfront.model.CreateConnectionGroupResponse;
import
 software.amazon.awssdk.services.cloudfront.model.CreateDistributionTenantResponse;
import software.amazon.awssdk.services.cloudfront.model.DistributionTenant;
import
 software.amazon.awssdk.services.cloudfront.model.GetConnectionGroupResponse;
import software.amazon.awssdk.services.cloudfront.model.ValidationTokenHost;
import software.amazon.awssdk.services.route53.Route53Client;
import software.amazon.awssdk.services.route53.model.RRType;

import java.time.Instant;

public class CreateDistributionTenant {

    public static DistributionTenant
 createDistributionTenantNoCert(CloudFrontClient cloudFrontClient,
                                                                    Route53Client
 route53Client,
                                                                    String
 distributionId,
                                                                    String
 domain,
                                                                    String
 hostedZoneId) {
```

```
        CreateDistributionTenantResponse createResponse =
 cloudFrontClient.createDistributionTenant(builder -> builder
                .distributionId(distributionId)
                .domains(b1 -> b1
                        .domain(domain))
                .parameters(b2 -> b2
                        .name("tenantName")
                        .value("myTenant"))
                .enabled(false)
                .name("no-cert-tenant")
        );

        final DistributionTenant distributionTenant =
 createResponse.distributionTenant();

        // Then update the Route53 hosted zone to point your domain at the
 distribution tenant
        // We fetch the RoutingEndpoint to point to via the default connection
 group that was created for your tenant
        final GetConnectionGroupResponse fetchedConnectionGroup =
 cloudFrontClient.getConnectionGroup(builder -> builder
                .identifier(distributionTenant.connectionGroupId()));

        route53Client.changeResourceRecordSets(builder -> builder
                .hostedZoneId(hostedZoneId)
                .changeBatch(b1 -> b1
                        .comment("ChangeBatch comment")
                        .changes(b2 -> b2
                                .resourceRecordSet(b3 -> b3
                                        .name(domain)
                                        .type("CNAME")
                                        .ttl(300L)
                                        .resourceRecords(b4 -> b4

 .value(fetchedConnectionGroup.connectionGroup().routingEndpoint())))
                                .action("CREATE"))
                ));
        return distributionTenant;
    }

}
```

뷰어 인증서가 상위 템플릿에서 생략된 경우 대신 연결된 테넌트(들)에 인증서 정보를 추가해야 합니다. 다음 예제에서는 테넌트에 필요한 도메인을 포함하는 ACM 인증서 ARN을 통해 이 작업을 수행하는 방법을 보여줍니다.

```
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
 software.amazon.awssdk.services.cloudfront.model.CreateConnectionGroupResponse;
import
 software.amazon.awssdk.services.cloudfront.model.CreateDistributionTenantResponse;
import software.amazon.awssdk.services.cloudfront.model.DistributionTenant;
import
 software.amazon.awssdk.services.cloudfront.model.GetConnectionGroupResponse;
import software.amazon.awssdk.services.cloudfront.model.ValidationTokenHost;
import software.amazon.awssdk.services.route53.Route53Client;
import software.amazon.awssdk.services.route53.model.RRType;

import java.time.Instant;

public class CreateDistributionTenant {

    public static DistributionTenant
 createDistributionTenantWithCert(CloudFrontClient cloudFrontClient,

 Route53Client route53Client,
                                                                        String
 distributionId,
                                                                        String
 domain,
                                                                        String
 hostedZoneId,
                                                                        String
 certificateArn) {
        CreateDistributionTenantResponse createResponse =
 cloudFrontClient.createDistributionTenant(builder -> builder
                .distributionId(distributionId)
                .domains(b1 -> b1
                        .domain(domain))
                .enabled(false)
                .name("tenant-with-cert")
                .parameters(b2 -> b2
                        .name("tenantName")
                        .value("myTenant"))
```

```
                .customizations(b3 -> b3
                        .certificate(b4 -> b4
                                .arn(certificateArn))) // NOTE: Cert must be in
 Us-East-1 and cover the domain provided in this request

        );

        final DistributionTenant distributionTenant =
 createResponse.distributionTenant();

        // Then update the Route53 hosted zone to point your domain at the
 distribution tenant
        // We fetch the RoutingEndpoint to point to via the default connection
 group that was created for your tenant
        final GetConnectionGroupResponse fetchedConnectionGroup =
 cloudFrontClient.getConnectionGroup(builder -> builder
                .identifier(distributionTenant.connectionGroupId()));

        route53Client.changeResourceRecordSets(builder -> builder
                .hostedZoneId(hostedZoneId)
                .changeBatch(b1 -> b1
                        .comment("ChangeBatch comment")
                        .changes(b2 -> b2
                                .resourceRecordSet(b3 -> b3
                                        .name(domain)
                                        .type("CNAME")
                                        .ttl(300L)
                                        .resourceRecords(b4 -> b4

 .value(fetchedConnectionGroup.connectionGroup().routingEndpoint())))
                                .action("CREATE"))
                ));
        return distributionTenant;
    }

}
```

다음 예제에서는 CloudFront 호스팅 관리형 인증서 요청을 사용하여 이 작업을 수행하는 방법을 보여줍니다. 이는 도메인으로 향하는 트래픽이 아직 없는 경우에 적합합니다. 이 경우 ConnectionGroup을 만들어 RoutingEndpoint를 생성합니다. 그런 다음 해당 RoutingEndpoint를 사용하여 도메인 소유권을 확인하고 CloudFront를 가리키는 DNS 레코드를 생성합니다. 그러면

CloudFront가 토큰을 자동으로 제공하여 도메인 소유권을 검증하고 관리형 인증서를 생성합니다.

```
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
 software.amazon.awssdk.services.cloudfront.model.CreateConnectionGroupResponse;
import
 software.amazon.awssdk.services.cloudfront.model.CreateDistributionTenantResponse;
import software.amazon.awssdk.services.cloudfront.model.DistributionTenant;
import
 software.amazon.awssdk.services.cloudfront.model.GetConnectionGroupResponse;
import software.amazon.awssdk.services.cloudfront.model.ValidationTokenHost;
import software.amazon.awssdk.services.route53.Route53Client;
import software.amazon.awssdk.services.route53.model.RRType;

import java.time.Instant;

public class CreateDistributionTenant {

    public static DistributionTenant
 createDistributionTenantCfHosted(CloudFrontClient cloudFrontClient,

 Route53Client route53Client,
                                                                        String
 distributionId,
                                                                        String
 domain,
                                                                        String
 hostedZoneId) throws InterruptedException {
        CreateConnectionGroupResponse createConnectionGroupResponse =
 cloudFrontClient.createConnectionGroup(builder -> builder
                .ipv6Enabled(true)
                .name("cf-hosted-connection-group")
                .enabled(true));

        route53Client.changeResourceRecordSets(builder -> builder
                .hostedZoneId(hostedZoneId)
                .changeBatch(b1 -> b1
                        .comment("cf-hosted domain validation record")
                        .changes(b2 -> b2
                                .resourceRecordSet(b3 -> b3
                                        .name(domain)
```

```
                                        .type(RRType.CNAME)
                                        .ttl(300L)
                                        .resourceRecords(b4 -> b4

.value(createConnectionGroupResponse.connectionGroup().routingEndpoint())))
                                .action("CREATE"))
                ));

        // Give the R53 record time to propagate, if it isn't being returned by
servers yet, the following call will fail
        Thread.sleep(60000);

        CreateDistributionTenantResponse createResponse =
cloudFrontClient.createDistributionTenant(builder -> builder
                .distributionId(distributionId)
                .domains(b1 -> b1
                        .domain(domain))

.connectionGroupId(createConnectionGroupResponse.connectionGroup().id())
                .enabled(false)
                .name("cf-hosted-tenant")
                .parameters(b2 -> b2
                        .name("tenantName")
                        .value("myTenant"))
                .managedCertificateRequest(b3 -> b3
                        .validationTokenHost(ValidationTokenHost.CLOUDFRONT)
                )
        );

        return createResponse.distributionTenant();
    }

}
```

다음 예제에서는 자체 호스팅 관리형 인증서 요청을 사용하여 이 작업을 수행하는 방법을 보여줍니다. 이는 도메인으로 향하는 트래픽이 있고 마이그레이션 중에 가동 중지 시간을 허용할 수 없는 경우에 적합합니다. 이 예제가 끝나면 도메인 검증 및 DNS 설정을 기다리는 상태로 테넌트가 생성됩니다. 트래픽을 마이그레이션할 준비가 되면 [여기](https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/managed-cloudfront-certificates.html#complete-domain-ownership)에 나온 단계에 따라 설정을 완료합니다.

```
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
 software.amazon.awssdk.services.cloudfront.model.CreateConnectionGroupResponse;
import
 software.amazon.awssdk.services.cloudfront.model.CreateDistributionTenantResponse;
import software.amazon.awssdk.services.cloudfront.model.DistributionTenant;
import
 software.amazon.awssdk.services.cloudfront.model.GetConnectionGroupResponse;
import software.amazon.awssdk.services.cloudfront.model.ValidationTokenHost;
import software.amazon.awssdk.services.route53.Route53Client;
import software.amazon.awssdk.services.route53.model.RRType;

import java.time.Instant;

public class CreateDistributionTenant {

    public static DistributionTenant
 createDistributionTenantSelfHosted(CloudFrontClient cloudFrontClient,
                                                                        String
 distributionId,
                                                                        String
 domain) {
        CreateDistributionTenantResponse createResponse =
 cloudFrontClient.createDistributionTenant(builder -> builder
                .distributionId(distributionId)
                .domains(b1 -> b1
                        .domain(domain))
                .parameters(b2 -> b2
                        .name("tenantName")
                        .value("myTenant"))
                .enabled(false)
                .name("self-hosted-tenant")
                .managedCertificateRequest(b3 -> b3
                        .validationTokenHost(ValidationTokenHost.SELF_HOSTED)
                        .primaryDomainName(domain)
                )
        );

        return createResponse.distributionTenant();
    }

}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 다음 주제를 참조하세요.
  - :CreateDistribution
  - CreateDistributionTenant

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## AWS SDK를 사용하여 CloudFront 서명 리소스 삭제

다음 코드 예제는 Amazon Simple Storage Service(Amazon S3) 버킷의 제한된 콘텐츠에 액세스하는 데 사용되는 리소스를 삭제하는 방법을 보여줍니다.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.DeleteKeyGroupResponse;
import
 software.amazon.awssdk.services.cloudfront.model.DeleteOriginAccessControlResponse;
import software.amazon.awssdk.services.cloudfront.model.DeletePublicKeyResponse;
import software.amazon.awssdk.services.cloudfront.model.GetKeyGroupResponse;
import
 software.amazon.awssdk.services.cloudfront.model.GetOriginAccessControlResponse;
import software.amazon.awssdk.services.cloudfront.model.GetPublicKeyResponse;

public class DeleteSigningResources {
    private static final Logger logger =
 LoggerFactory.getLogger(DeleteSigningResources.class);
```

```
    public static void deleteOriginAccessControl(final CloudFrontClient
 cloudFrontClient,
            final String originAccessControlId) {
        GetOriginAccessControlResponse getResponse = cloudFrontClient
                .getOriginAccessControl(b -> b.id(originAccessControlId));
        DeleteOriginAccessControlResponse deleteResponse =
 cloudFrontClient.deleteOriginAccessControl(builder -> builder
                .id(originAccessControlId)
                .ifMatch(getResponse.eTag()));
        if (deleteResponse.sdkHttpResponse().isSuccessful()) {
            logger.info("Successfully deleted Origin Access Control [{}]",
 originAccessControlId);
        }
    }

    public static void deleteKeyGroup(final CloudFrontClient cloudFrontClient,
 final String keyGroupId) {

        GetKeyGroupResponse getResponse = cloudFrontClient.getKeyGroup(b ->
 b.id(keyGroupId));
        DeleteKeyGroupResponse deleteResponse =
 cloudFrontClient.deleteKeyGroup(builder -> builder
                .id(keyGroupId)
                .ifMatch(getResponse.eTag()));
        if (deleteResponse.sdkHttpResponse().isSuccessful()) {
            logger.info("Successfully deleted Key Group [{}]", keyGroupId);
        }
    }

    public static void deletePublicKey(final CloudFrontClient cloudFrontClient,
 final String publicKeyId) {
        GetPublicKeyResponse getResponse = cloudFrontClient.getPublicKey(b ->
 b.id(publicKeyId));

        DeletePublicKeyResponse deleteResponse =
 cloudFrontClient.deletePublicKey(builder -> builder
                .id(publicKeyId)
                .ifMatch(getResponse.eTag()));

        if (deleteResponse.sdkHttpResponse().isSuccessful()) {
            logger.info("Successfully deleted Public Key [{}]", publicKeyId);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 다음 주제를 참조하세요.
  - DeleteKeyGroup
  - DeleteOriginAccessControl
  - DeletePublicKey

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CLI를 사용하여 기본 CloudFront 배포 시작하기

다음 코드 예제에서는 다음과 같은 작업을 수행하는 방법을 보여줍니다.

- 콘텐츠 저장용 Amazon S3 버킷 생성
- S3 버킷에 샘플 콘텐츠 업로드
- 보안 S3 액세스를 위한 오리진 액세스 제어(OAC) 생성
- S3를 오리진으로 사용하여 CloudFront 배포 생성
- CloudFront 액세스를 허용하도록 S3 버킷 정책 업데이트
- 배포 완료 대기 및 콘텐츠 액세스 테스트
- 배포, OAC 및 S3 버킷을 포함한 리소스 정리

Bash

Bash 스크립트와 함께 AWS CLI 사용

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. 샘플 개발자 튜토리얼 리포지토리에서 전체 예제를 찾아보고 설정 및 실행 방법을 배워보세요.

```
#!/bin/bash

# CloudFront Getting Started Tutorial Script
# This script creates an S3 bucket, uploads sample content, creates a CloudFront
 distribution with OAC,
```

```
# and demonstrates how to access content through CloudFront.

# Set up logging
LOG_FILE="cloudfront-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting CloudFront Getting Started Tutorial at $(date)"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created before error:"
    if [ -n "$BUCKET_NAME" ]; then
        echo "- S3 Bucket: $BUCKET_NAME"
    fi
    if [ -n "$OAC_ID" ]; then
        echo "- CloudFront Origin Access Control: $OAC_ID"
    fi
    if [ -n "$DISTRIBUTION_ID" ]; then
        echo "- CloudFront Distribution: $DISTRIBUTION_ID"
    fi

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to clean up resources
cleanup() {
    echo "Cleaning up resources..."

    if [ -n "$DISTRIBUTION_ID" ]; then
        echo "Disabling CloudFront distribution $DISTRIBUTION_ID..."

        # Get the current configuration and ETag
        ETAG=$(aws cloudfront get-distribution-config --id "$DISTRIBUTION_ID" --
query 'ETag' --output text)
        if [ $? -ne 0 ]; then
            echo "Failed to get distribution config. Continuing with cleanup..."
        else
            # Create a modified configuration with Enabled=false
            aws cloudfront get-distribution-config --id "$DISTRIBUTION_ID" | \
            jq '.DistributionConfig.Enabled = false' > temp_disabled_config.json
```

```
            # Update the distribution to disable it
            aws cloudfront update-distribution \
                --id "$DISTRIBUTION_ID" \
                --distribution-config file://<(jq '.DistributionConfig'
 temp_disabled_config.json) \
                --if-match "$ETAG"

            if [ $? -ne 0 ]; then
                echo "Failed to disable distribution. Continuing with cleanup..."
            else
                echo "Waiting for distribution to be disabled (this may take
 several minutes)..."
                aws cloudfront wait distribution-deployed --id "$DISTRIBUTION_ID"

                # Delete the distribution
                ETAG=$(aws cloudfront get-distribution-config --id
 "$DISTRIBUTION_ID" --query 'ETag' --output text)
                aws cloudfront delete-distribution --id "$DISTRIBUTION_ID" --if-
match "$ETAG"
                if [ $? -ne 0 ]; then
                    echo "Failed to delete distribution. You may need to delete
 it manually."
                else
                    echo "CloudFront distribution deleted."
                fi
            fi
        fi
    fi

    if [ -n "$OAC_ID" ]; then
        echo "Deleting Origin Access Control $OAC_ID..."
        OAC_ETAG=$(aws cloudfront get-origin-access-control --id "$OAC_ID" --
query 'ETag' --output text 2>/dev/null)
        if [ $? -ne 0 ]; then
            echo "Failed to get Origin Access Control ETag. You may need to
 delete it manually."
        else
            aws cloudfront delete-origin-access-control --id "$OAC_ID" --if-match
 "$OAC_ETAG"
            if [ $? -ne 0 ]; then
                echo "Failed to delete Origin Access Control. You may need to
 delete it manually."
            else
                echo "Origin Access Control deleted."
```

```
            fi
        fi
    fi

    if [ -n "$BUCKET_NAME" ]; then
        echo "Deleting S3 bucket $BUCKET_NAME and its contents..."
        aws s3 rm "s3://$BUCKET_NAME" --recursive
        if [ $? -ne 0 ]; then
            echo "Failed to remove bucket contents. Continuing with bucket
 deletion..."
        fi

        aws s3 rb "s3://$BUCKET_NAME"
        if [ $? -ne 0 ]; then
            echo "Failed to delete bucket. You may need to delete it manually."
        else
            echo "S3 bucket deleted."
        fi
    fi

    # Clean up temporary files
    rm -f temp_disabled_config.json
    rm -rf temp_content
}

# Generate a random identifier for the bucket name
RANDOM_ID=$(openssl rand -hex 6)
BUCKET_NAME="cloudfront-${RANDOM_ID}"
echo "Using bucket name: $BUCKET_NAME"

# Create a temporary directory for content
TEMP_DIR="temp_content"
mkdir -p "$TEMP_DIR/css"
if [ $? -ne 0 ]; then
    handle_error "Failed to create temporary directory"
fi

# Step 1: Create an S3 bucket
echo "Creating S3 bucket: $BUCKET_NAME"
aws s3 mb "s3://$BUCKET_NAME"
if [ $? -ne 0 ]; then
    handle_error "Failed to create S3 bucket"
fi
```

```
# Step 2: Create sample content
echo "Creating sample content..."
cat > "$TEMP_DIR/index.html" << 'EOF'
<!DOCTYPE html>
<html>
<head>
    <title>Hello World</title>
    <link rel="stylesheet" type="text/css" href="css/styles.css">
</head>
<body>
    <h1>Hello world!</h1>
</body>
</html>
EOF

cat > "$TEMP_DIR/css/styles.css" << 'EOF'
body {
    font-family: Arial, sans-serif;
    margin: 40px;
    background-color: #f5f5f5;
}
h1 {
    color: #333;
    text-align: center;
}
EOF

# Step 3: Upload content to the S3 bucket
echo "Uploading content to S3 bucket..."
aws s3 cp "$TEMP_DIR/" "s3://$BUCKET_NAME/" --recursive
if [ $? -ne 0 ]; then
    handle_error "Failed to upload content to S3 bucket"
fi

# Step 4: Create Origin Access Control
echo "Creating Origin Access Control..."
OAC_RESPONSE=$(aws cloudfront create-origin-access-control \
    --origin-access-control-config Name="oac-for-
$BUCKET_NAME",SigningProtocol=sigv4,SigningBehavior=always,OriginAccessControlOriginType=

if [ $? -ne 0 ]; then
    handle_error "Failed to create Origin Access Control"
fi
```

```
OAC_ID=$(echo "$OAC_RESPONSE" | jq -r '.OriginAccessControl.Id')
echo "Created Origin Access Control with ID: $OAC_ID"

# Step 5: Create CloudFront distribution
echo "Creating CloudFront distribution..."

# Get AWS account ID for bucket policy
ACCOUNT_ID=$(aws sts get-caller-identity --query 'Account' --output text)
if [ $? -ne 0 ]; then
    handle_error "Failed to get AWS account ID"
fi

# Create distribution configuration
cat > distribution-config.json << EOF
{
    "CallerReference": "cli-tutorial-$(date +%s)",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "S3-$BUCKET_NAME",
                "DomainName": "$BUCKET_NAME.s3.amazonaws.com",
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                },
                "OriginAccessControlId": "$OAC_ID"
            }
        ]
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "S3-$BUCKET_NAME",
        "ViewerProtocolPolicy": "redirect-to-https",
        "AllowedMethods": {
            "Quantity": 2,
            "Items": ["GET", "HEAD"],
            "CachedMethods": {
                "Quantity": 2,
                "Items": ["GET", "HEAD"]
            }
        },
        "DefaultTTL": 86400,
        "MinTTL": 0,
        "MaxTTL": 31536000,
        "Compress": true,
```

```
        "ForwardedValues": {
            "QueryString": false,
            "Cookies": {
                "Forward": "none"
            }
        }
    },
    "Comment": "CloudFront distribution for tutorial",
    "Enabled": true,
    "WebACLId": ""
}
EOF

DIST_RESPONSE=$(aws cloudfront create-distribution --distribution-config file://
distribution-config.json)
if [ $? -ne 0 ]; then
    handle_error "Failed to create CloudFront distribution"
fi

DISTRIBUTION_ID=$(echo "$DIST_RESPONSE" | jq -r '.Distribution.Id')
DOMAIN_NAME=$(echo "$DIST_RESPONSE" | jq -r '.Distribution.DomainName')

echo "Created CloudFront distribution with ID: $DISTRIBUTION_ID"
echo "CloudFront domain name: $DOMAIN_NAME"

# Step 6: Update S3 bucket policy
echo "Updating S3 bucket policy..."
cat > bucket-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCloudFrontServicePrincipal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudfront.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::$BUCKET_NAME/*",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceArn": "arn:aws:cloudfront::
$ACCOUNT_ID:distribution/$DISTRIBUTION_ID"
                }
```

```
            }
        }
    ]
}
EOF

aws s3api put-bucket-policy --bucket "$BUCKET_NAME" --policy file://bucket-
policy.json
if [ $? -ne 0 ]; then
    handle_error "Failed to update S3 bucket policy"
fi

# Step 7: Wait for distribution to deploy
echo "Waiting for CloudFront distribution to deploy (this may take 5-10
 minutes)..."
aws cloudfront wait distribution-deployed --id "$DISTRIBUTION_ID"
if [ $? -ne 0 ]; then
    echo "Warning: Distribution deployment wait timed out. The distribution may
 still be deploying."
else
    echo "CloudFront distribution is now deployed."
fi

# Step 8: Display access information
echo ""
echo "===== CloudFront Distribution Setup Complete ======"
echo "You can access your content at: https://$DOMAIN_NAME/index.html"
echo ""
echo "Resources created:"
echo "- S3 Bucket: $BUCKET_NAME"
echo "- CloudFront Origin Access Control: $OAC_ID"
echo "- CloudFront Distribution: $DISTRIBUTION_ID"
echo ""

# Ask user if they want to clean up resources
read -p "Do you want to clean up all resources created by this script? (y/n): "
 CLEANUP_RESPONSE
if [[ "$CLEANUP_RESPONSE" =~ ^[Yy] ]]; then
    cleanup
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually delete them later."
    echo "To access your content, visit: https://$DOMAIN_NAME/index.html"
fi
```

```
echo "Tutorial completed at $(date)"
```

- API 세부 정보는 AWS CLI 명령 참조의 다음 토픽을 참조하세요.
  - :CreateDistribution
  - CreateOriginAccessControl
  - :DeleteDistribution
  - DeleteOriginAccessControl
  - GetDistribution
  - GetDistributionConfig
  - GetOriginAccessControl
  - :UpdateDistribution
  - WaitDistributionDeployed

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## AWS SDK를 사용하여 서명된 URL과 쿠키 생성

다음 코드 예제는 제한된 리소스에 액세스를 허용하는 서명된 URL 및 서명된 쿠키를 생성하는 방법을 보여줍니다.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

CannedSignerRequest 클래스를 사용하면 미리 준비된 정책으로 URL 또는 쿠키에 서명할 수 있습니다.

```
import software.amazon.awssdk.services.cloudfront.model.CannedSignerRequest;
```

```
import java.net.URL;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.time.Instant;
import java.time.temporal.ChronoUnit;

public class CreateCannedPolicyRequest {

    public static CannedSignerRequest createRequestForCannedPolicy(String
 distributionDomainName,
            String fileNameToUpload,
            String privateKeyFullPath, String publicKeyId) throws Exception {
        String protocol = "https";
        String resourcePath = "/" + fileNameToUpload;

        String cloudFrontUrl = new URL(protocol, distributionDomainName,
 resourcePath).toString();
        Instant expirationDate = Instant.now().plus(7, ChronoUnit.DAYS);
        Path path = Paths.get(privateKeyFullPath);

        return CannedSignerRequest.builder()
                .resourceUrl(cloudFrontUrl)
                .privateKey(path)
                .keyPairId(publicKeyId)
                .expirationDate(expirationDate)
                .build();
    }
}
```

CustomSignerRequest 클래스를 사용하면 사용자 지정 정책으로 URL 또는 쿠키에 서명할 수 있습니다. `activeDate`및 `ipRange`는 선택적 메서드입니다.

```
import software.amazon.awssdk.services.cloudfront.model.CustomSignerRequest;

import java.net.URL;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.time.Instant;
import java.time.temporal.ChronoUnit;

public class CreateCustomPolicyRequest {
```

```
    public static CustomSignerRequest createRequestForCustomPolicy(String
 distributionDomainName,
            String fileNameToUpload,
            String privateKeyFullPath, String publicKeyId) throws Exception {
        String protocol = "https";
        String resourcePath = "/" + fileNameToUpload;

        String cloudFrontUrl = new URL(protocol, distributionDomainName,
 resourcePath).toString();
        Instant expireDate = Instant.now().plus(7, ChronoUnit.DAYS);
        // URL will be accessible tomorrow using the signed URL.
        Instant activeDate = Instant.now().plus(1, ChronoUnit.DAYS);
        Path path = Paths.get(privateKeyFullPath);

        return CustomSignerRequest.builder()
                .resourceUrl(cloudFrontUrl)
                // .resourceUrlPattern("https://*.example.com/*")  // Optional.
                .privateKey(path)
                .keyPairId(publicKeyId)
                .expirationDate(expireDate)
                .activeDate(activeDate) // Optional.
                // .ipRange("192.168.0.1/24") // Optional.
                .build();
    }
}
```

다음 예제는 CloudFrontUtilities 클래스를 사용하여 서명된 쿠키와 URL을 생성하는 방법을 보여줍니다. GitHub에서 이 코드 예제를 확인하세요.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.cloudfront.CloudFrontUtilities;
import software.amazon.awssdk.services.cloudfront.cookie.CookiesForCannedPolicy;
import software.amazon.awssdk.services.cloudfront.cookie.CookiesForCustomPolicy;
import software.amazon.awssdk.services.cloudfront.model.CannedSignerRequest;
import software.amazon.awssdk.services.cloudfront.model.CustomSignerRequest;
import software.amazon.awssdk.services.cloudfront.url.SignedUrl;

public class SigningUtilities {
    private static final Logger logger =
 LoggerFactory.getLogger(SigningUtilities.class);
```

```
   private static final CloudFrontUtilities cloudFrontUtilities =
CloudFrontUtilities.create();

   public static SignedUrl signUrlForCannedPolicy(CannedSignerRequest
cannedSignerRequest) {
       SignedUrl signedUrl =
cloudFrontUtilities.getSignedUrlWithCannedPolicy(cannedSignerRequest);
       logger.info("Signed URL: [{}]", signedUrl.url());
       return signedUrl;
   }

   public static SignedUrl signUrlForCustomPolicy(CustomSignerRequest
customSignerRequest) {
       SignedUrl signedUrl =
cloudFrontUtilities.getSignedUrlWithCustomPolicy(customSignerRequest);
       logger.info("Signed URL: [{}]", signedUrl.url());
       return signedUrl;
   }

   public static CookiesForCannedPolicy
getCookiesForCannedPolicy(CannedSignerRequest cannedSignerRequest) {
       CookiesForCannedPolicy cookiesForCannedPolicy = cloudFrontUtilities
               .getCookiesForCannedPolicy(cannedSignerRequest);
       logger.info("Cookie EXPIRES header [{}]",
cookiesForCannedPolicy.expiresHeaderValue());
       logger.info("Cookie KEYPAIR header [{}]",
cookiesForCannedPolicy.keyPairIdHeaderValue());
       logger.info("Cookie SIGNATURE header [{}]",
cookiesForCannedPolicy.signatureHeaderValue());
       return cookiesForCannedPolicy;
   }

   public static CookiesForCustomPolicy
getCookiesForCustomPolicy(CustomSignerRequest customSignerRequest) {
       CookiesForCustomPolicy cookiesForCustomPolicy = cloudFrontUtilities
               .getCookiesForCustomPolicy(customSignerRequest);
       logger.info("Cookie POLICY header [{}]",
cookiesForCustomPolicy.policyHeaderValue());
       logger.info("Cookie KEYPAIR header [{}]",
cookiesForCustomPolicy.keyPairIdHeaderValue());
       logger.info("Cookie SIGNATURE header [{}]",
cookiesForCustomPolicy.signatureHeaderValue());
       return cookiesForCustomPolicy;
   }
```

```
}
```

- API에 대한 세부 정보는 AWS SDK for Java 2.x API 참조의 CloudFrontUtilities을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront의 CloudFront Functions 예제

다음 코드 예제에서는 AWS SDK에서 CloudFront를 사용한 방법을 보여줍니다.

예제

- CloudFront Functions 뷰어 응답 이벤트에 HTTP 보안 헤더 추가
- CloudFront Functions 뷰어 응답 이벤트에 CORS 헤더 추가
- CloudFront Functions 뷰어 응답 이벤트에 캐시 제어 헤더 추가
- CloudFront Functions 뷰어 요청 이벤트에 실제 클라이언트 IP 헤더 추가
- CloudFront Functions 뷰어 요청 이벤트에 오리진 헤더 추가
- CloudFront Functions 뷰어 요청 이벤트에서 파일 이름 없이 URL을 요청하기 위한 index.html 추가
- CloudFront Functions 뷰어 요청에서 쿼리 문자열 파라미터 정규화
- CloudFront Functions 뷰어 요청 이벤트에서 새 URL로 리디렉션
- CloudFront Functions 뷰어 요청 이벤트에 대한 KeyValueStore 구성을 기반으로 요청 URI 재작성
- CloudFront Functions 뷰어 요청 이벤트에서 뷰어에게 더 가까운 오리진으로 요청 라우팅
- CloudFront Functions 뷰어 요청에서 키-값 페어 사용
- CloudFront Functions 뷰어 요청에서 단순 토큰 검증

### CloudFront Functions 뷰어 응답 이벤트에 HTTP 보안 헤더 추가

다음 코드 예제에서는 CloudFront Functions 뷰어 응답 이벤트에 HTTP 보안 헤더를 추가하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async function handler(event) {
    var response = event.response;
    var headers = response.headers;

    // Set HTTP security headers
    // Since JavaScript doesn't allow for hyphens in variable names, we use the
 dict["key"] notation
    headers['strict-transport-security'] = { value: 'max-age=63072000;
 includeSubdomains; preload'};
    headers['content-security-policy'] = { value: "default-src 'none'; img-src
 'self'; script-src 'self'; style-src 'self'; object-src 'none'; frame-ancestors
 'none'"};
    headers['x-content-type-options'] = { value: 'nosniff'};
    headers['x-frame-options'] = {value: 'DENY'};
    headers['x-xss-protection'] = {value: '1; mode=block'};
    headers['referrer-policy'] = {value: 'same-origin'};

    // Return the response to viewers
    return response;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 응답 이벤트에 CORS 헤더 추가

다음 코드 예제에서는 CloudFront Functions 뷰어 응답 이벤트에 CORS 헤더를 추가하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async function handler(event)  {
    var request = event.request;
    var response  = event.response;

    // If Access-Control-Allow-Origin CORS header is missing, add it.
    // Since JavaScript doesn't allow for hyphens in variable names, we use the
 dict["key"] notation.
    if (!response.headers['access-control-allow-origin'] &&
 request.headers['origin']) {
        response.headers['access-control-allow-origin'] = {value:
 request.headers['origin'].value};
        console.log("Access-Control-Allow-Origin was missing, adding it now.");
    }

    return response;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 응답 이벤트에 캐시 제어 헤더 추가

다음 코드 예제에서는 CloudFront Functions 뷰어 응답 이벤트에 캐시 제어 헤더를 추가하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async function handler(event) {
    var response = event.response;
    var headers = response.headers;

    if (response.statusCode >= 200 && response.statusCode < 400) {
        // Set the cache-control header
        headers['cache-control'] = {value: 'public, max-age=63072000'};
    }

    // Return response to viewers
    return response;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 요청 이벤트에 실제 클라이언트 IP 헤더 추가

다음 코드 예제에서는 CloudFront Functions 뷰어 요청 이벤트에 실제 클라이언트 IP 헤더를 추가하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async function handler(event) {
    var request = event.request;
    var clientIP = event.viewer.ip;

    //Add the true-client-ip header to the incoming request
    request.headers['true-client-ip'] = {value: clientIP};

    return request;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 요청 이벤트에 오리진 헤더 추가

다음 코드 예제에서는 CloudFront Functions 뷰어 요청 이벤트에 오리진 헤더를 추가하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async function handler(event) {
    var request = event.request;
    var headers = request.headers;
    var host = request.headers.host.value;

   // If origin header is missing, set it equal to the host header.
   if (!headers.origin)
       headers.origin = {value:`https://${host}`};

   return request;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 요청 이벤트에서 파일 이름 없이 URL을 요청하기 위한 index.html 추가

다음 코드 예제에서는 CloudFront Functions 뷰어 요청 이벤트에서 파일 이름 없이 URL을 요청하기 위해 index.html을 추가하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async function handler(event) {
    var request = event.request;
    var uri = request.uri;

    // Check whether the URI is missing a file name.
    if (uri.endsWith('/')) {
        request.uri += 'index.html';
```

```
    }
    // Check whether the URI is missing a file extension.
    else if (!uri.includes('.')) {
        request.uri += '/index.html';
    }

    return request;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 요청에서 쿼리 문자열 파라미터 정규화

다음 코드 예제에서는 CloudFront Functions 뷰어 요청에서 쿼리 문자열 파라미터를 정규화하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
function handler(event) {
     var qs=[];
     for (var key in event.request.querystring) {
         if (event.request.querystring[key].multiValue) {
             event.request.querystring[key].multiValue.forEach((mv) =>
 {qs.push(key + "=" + mv.value)});
         } else {
             qs.push(key + "=" + event.request.querystring[key].value);
         }
     };

     event.request.querystring = qs.sort().join('&');
```

```
    return event.request;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 요청 이벤트에서 새 URL로 리디렉션

다음 코드 예제에서는 CloudFront Functions 뷰어 요청 이벤트에서 새 URL로 리디렉션하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async function handler(event) {
    var request = event.request;
    var headers = request.headers;
    var host = request.headers.host.value;
    var country = 'DE' // Choose a country code
    var newurl = `https://${host}/de/index.html`; // Change the redirect URL to
 your choice

    if (headers['cloudfront-viewer-country']) {
        var countryCode = headers['cloudfront-viewer-country'].value;
        if (countryCode === country) {
            var response = {
                statusCode: 302,
                statusDescription: 'Found',
                headers:
                    { "location": { "value": newurl } }
                }
```

```
            return response;
        }
    }
    return request;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 요청 이벤트에 대한 KeyValueStore 구성을 기반으로 요청 URI 재작성

다음 코드 예제에서는 CloudFront Functions 뷰어 요청 이벤트에 대한 KeyValueStore 구성을 기반으로 요청 URI를 재작성하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import cf from 'cloudfront';

// (Optional) Replace KVS_ID with actual KVS ID
const kvsId = "KVS_ID";
// enable stickiness by setting a cookie from origin or using another edge
 function
const stickinessCookieName = "appversion";
// set to true to enable console logging
const loggingEnabled = false;

// function rewrites the request uri based on configuration in KVS
// example config in KVS in key:value format
// "latest": {"a_weightage": .8, "a_url": "v1", "b_url": "v2"}
// given above key and value in KVS the request uri will be rewritten
```

```
// for example http(s)://domain/latest/something/else will be rewritten as
 http(s)://domain/v1/something/else or http(s)://domain/v2/something/else
 depending on weightage
// if no configuration is found, then the request is returned as is
async function handler(event) {
    // NOTE: This example function is for a viewer request event trigger.
    // Choose viewer request for event trigger when you associate this function
 with a distribution.
    const request = event.request;
    const pathSegments = request.uri.split('/');
    const key = pathSegments[1];

    // if empty path segment or if there is valid stickiness cookie
    // then skip call to KVS and let the request continue.
    if (!key || hasValidStickinessCookie(request.cookies[stickinessCookieName],
 key)) {
        return event.request;
    }

    try {
        // get the prefix replacement from KVS
        const replacement = await getPathPrefixByWeightage(key);
        if (!replacement) {
            return event.request;
        }
        //Replace the first path with the replacement
        pathSegments[1] = replacement;
        log(`using prefix ${pathSegments[1]}`)
        const newUri = pathSegments.join('/');
        log(`${request.uri} -> ${newUri}`);
        request.uri = newUri;

        return request;
    } catch (err) {
        // No change to the path if the key is not found or any other error
        log(`request uri: ${request.uri}, error: ${err}`);
    }
    // no change to path - return request
    return event.request;
}

// function to get the prefix from KVS
async function getPathPrefixByWeightage(key) {
    const kvsHandle = cf.kvs(kvsId);
```

```
    // get the weightage config from KVS
    const kvsResponse = await kvsHandle.get(key);
    const weightageConfig = JSON.parse(kvsResponse);
    // no configuration - return null
    if (!weightageConfig || !isFinite(weightageConfig.a_weightage)) {
        return null;
    }
    // return the url based on weightage
    // return null if no url is configured
    if (Math.random() <= weightageConfig.a_weightage) {
        return weightageConfig.a_url ? weightageConfig.a_url: null;
    } else {
        return weightageConfig.b_url ? weightageConfig.b_url : null;
    }
}

// function to check if the stickiness cookie is valid
function hasValidStickinessCookie(stickinessCookie, pathSegment) {
    // if the value exists and it matches pathSegment
    return (stickinessCookie && stickinessCookie.value === pathSegment)
}

function log(message) {
    if (loggingEnabled) {
        console.log(message);
    }
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 요청 이벤트에서 뷰어에게 더 가까운 오리진으로 요청 라우팅

다음 코드 예제는 CloudFront Functions 뷰어 요청 이벤트에서 뷰어에게 더 가까운 오리진으로 요청을 라우팅하는 방법을 보여줍니다.

## JavaScript

### CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import cf from 'cloudfront';

function handler(event) {
    const request = event.request;
    const headers = request.headers;
    const country = headers['cloudfront-viewer-country'] &&
        headers['cloudfront-viewer-country'].value;

    //List of Regions with S3 buckets containing content
    const countryToRegion = {
        'DE': 'eu-central-1',
        'IE': 'eu-west-1',
        'GB': 'eu-west-2',
        'FR': 'eu-west-3',
        'JP': 'ap-northeast-1',
        'IN': 'ap-south-1'
    };

    const DEFAULT_REGION = 'us-east-1';

    const selectedRegion = (country && countryToRegion[country]) ||
 DEFAULT_REGION;

    const domainName =
        `cloudfront-functions-demo-bucket-in-${selectedRegion}.s3.
${selectedRegion}.amazonaws.com`;

    cf.updateRequestOrigin({
        "domainName": domainName,
        "originAccessControlConfig": {
            "enabled": true,
            "region": selectedRegion,
```

```
            "signingBehavior": "always",
            "signingProtocol": "sigv4",
            "originType": "s3"
        },
    });

    return request;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 요청에서 키-값 페어 사용

다음 코드 예제에서는 CloudFront Functions 뷰어 요청에서 키-값 페어를 사용하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import cf from 'cloudfront';

// This fails if there is no key value store associated with the function
const kvsHandle = cf.kvs();

// Remember to associate the KVS with your function before referencing KVS in
 your code.
// https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/kvs-with-
functions-associate.html
async function handler(event) {
    const request = event.request;
    // Use the first segment of the pathname as key
    // For example http(s)://domain/<key>/something/else
    const pathSegments = request.uri.split('/')
    const key = pathSegments[1]
```

```
    try {
        // Replace the first path of the pathname with the value of the key
        // For example http(s)://domain/<value>/something/else
        pathSegments[1] = await kvsHandle.get(key);
        const newUri = pathSegments.join('/');
        console.log(`${request.uri} -> ${newUri}`)
        request.uri = newUri;
    } catch (err) {
        // No change to the pathname if the key is not found
        console.log(`${request.uri} | ${err}`);
    }
    return request;
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

## CloudFront Functions 뷰어 요청에서 단순 토큰 검증

다음 코드 예제에서는 CloudFront Functions 뷰어 요청에서 단순 토큰을 검증하는 방법을 보여줍니다.

JavaScript

CloudFront Functions를 위한 JavaScript 런타임 2.0

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. CloudFront Functions 예제 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import crypto from 'crypto';
import cf from 'cloudfront';

//Response when JWT is not valid.
const response401 = {
    statusCode: 401,
    statusDescription: 'Unauthorized'
};
```

```
// Remember to associate the KVS with your function before calling the const
 kvsKey = 'jwt.secret'.
// https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/kvs-with-
functions-associate.html
const kvsKey = 'jwt.secret';
// set to true to enable console logging
const loggingEnabled = false;


function jwt_decode(token, key, noVerify, algorithm) {
    // check token
    if (!token) {
        throw new Error('No token supplied');
    }
    // check segments
    const segments = token.split('.');
    if (segments.length !== 3) {
        throw new Error('Not enough or too many segments');
    }

    // All segment should be base64
    const headerSeg = segments[0];
    const payloadSeg = segments[1];
    const signatureSeg = segments[2];

    // base64 decode and parse JSON
    const payload = JSON.parse(_base64urlDecode(payloadSeg));

    if (!noVerify) {
        const signingMethod = 'sha256';
        const signingType = 'hmac';

        // Verify signature. `sign` will return base64 string.
        const signingInput = [headerSeg, payloadSeg].join('.');

        if (!_verify(signingInput, key, signingMethod, signingType,
 signatureSeg)) {
            throw new Error('Signature verification failed');
        }

        // Support for nbf and exp claims.
        // According to the RFC, they should be in seconds.
        if (payload.nbf && Date.now() < payload.nbf*1000) {
            throw new Error('Token not yet active');
```

```
        }

        if (payload.exp && Date.now() > payload.exp*1000) {
            throw new Error('Token expired');
        }
    }

    return payload;
}

//Function to ensure a constant time comparison to prevent
//timing side channels.
function _constantTimeEquals(a, b) {
    if (a.length != b.length) {
        return false;
    }

    let xor = 0;
    for (let i = 0; i < a.length; i++) {
    xor |= (a.charCodeAt(i) ^ b.charCodeAt(i));
    }

    return 0 === xor;
}

function _verify(input, key, method, type, signature) {
    if(type === "hmac") {
        return _constantTimeEquals(signature, _sign(input, key, method));
    }
    else {
        throw new Error('Algorithm type not recognized');
    }
}

function _sign(input, key, method) {
    return crypto.createHmac(method, key).update(input).digest('base64url');
}

function _base64urlDecode(str) {
    return Buffer.from(str, 'base64url')
}

async function handler(event) {
    let request = event.request;
```

```
    //Secret key used to verify JWT token.
    //Update with your own key.
    const secret_key = await getSecret()

    if(!secret_key) {
        return response401;
    }

    // If no JWT token, then generate HTTP redirect 401 response.
    if(!request.querystring.jwt) {
        log("Error: No JWT in the querystring");
        return response401;
    }

    const jwtToken = request.querystring.jwt.value;

    try{
        jwt_decode(jwtToken, secret_key);
    }
    catch(e) {
        log(e);
        return response401;
    }

    //Remove the JWT from the query string if valid and return.
    delete request.querystring.jwt;
    log("Valid JWT token");
    return request;
}

// get secret from key value store
async function getSecret() {
    // initialize cloudfront kv store and get the key value
    try {
        const kvsHandle = cf.kvs();
        return await kvsHandle.get(kvsKey);
    } catch (err) {
        log(`Error reading value for key: ${kvsKey}, error: ${err}`);
        return null;
    }

}
```

```
function log(message) {
    if (loggingEnabled) {
        console.log(message);
    }
}
```

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK와 함께 CloudFront 사용을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

# 문서 기록

다음 테이블은 CloudFront 설명서의 중요한 변경 사항을 설명합니다. 업데이트에 대한 알림을 받으려면 RSS 피드를 구독하시면 됩니다.

| 변경 사항 | 설명 | 날짜 |
| --- | --- | --- |
| 상호 TLS(뷰어) 추가 | CloudFront는 상호 TLS(뷰어)를 지원합니다. | 2025년 11월 24일 |
| 액세스 로그에 대한 로그 필드 추가 | 액세스 로그(표준 로그) 및 실시간 액세스 로그에 대한 `connection-id` 필드를 추가했습니다. | 2025년 11월 24일 |
| 연결 로그 추가 | 상호 TLS(뷰어)에 대한 새로운 로깅 기능으로 연결 로그를 추가했습니다. | 2025년 11월 24일 |
| 연결 함수 추가 | CloudFront는 상호 TLS(뷰어)에 대한 연결 함수를 지원합니다. | 2025년 11월 24일 |
| IPAM을 사용하여 CloudFront로 자체 IP 가져오기에 대한 지원 추가 | CloudFront는 글로벌 서비스에 대한 IPAM의 BYOIP를 통해 자체 IPv4 주소 가져오기를 지원합니다. | 2025년 11월 24일 |
| AWS 관리형 정책 업데이트 | `CloudFrontReadOnlyAccess` 및 `CloudFrontFullAccess` IAM 정책은 이제 `ec2:DescribeIpamPools` 및 `ec2:GetIpamPoolCidrs` 작업을 지원합니다. | 2025년 11월 24일 |
| CloudFront Functions 업데이트 | 이 릴리스에서는 CloudFront Functions의 오리진 수정 헬 | 2025년 11월 20일 |

| | | |
|---|---|---|
| | 퍼 메서드에 대한 파라미터를 추가합니다. `hostHeader`, `sni`, `allowedCertificateNames` 및 `originOverrides` 파라미터를 사용할 수 있습니다. | |
| CloudFront Functions 업데이트 | CloudFront Functions에 대한 CBOR 웹 토큰(CWT) 지원이 추가되었습니다. | 2025년 11월 20일 |
| CloudFront Functions 업데이트 | CloudFront Functions를 위한 일반 헬퍼 메서드가 추가되었습니다. | 2025년 11월 20일 |
| AWS 관리형 정책이 업데이트됨 | `CloudFrontFullAccess` 가 AWS WAF 웹 ACL 리소스를 생성하고 AWS Pricing Plan Manager에 대한 액세스 권한을 생성, 업데이트, 삭제 및 읽을 수 있는 권한을 허용하도록 업데이트되었습니다. | 2025년 11월 18일 |
| AWS 관리형 정책이 업데이트됨 | `CloudFrontReadOnlyAccess` 가 AWS Pricing Plan Manager에 대한 읽기 전용 권한을 허용하도록 업데이트되었습니다. | 2025년 11월 18일 |
| AWS 관리형 정책이 업데이트됨 | `CloudFrontFullAccess` 가 AWS WAF 웹 ACL 리소스를 생성하고 AWS Pricing Plan Manager에 대한 액세스 권한을 생성, 업데이트, 삭제 및 읽을 수 있는 권한을 허용하도록 업데이트되었습니다. | 2025년 11월 18일 |

| | | |
|---|---|---|
| AWS 관리형 정책이 업데이트됨 | `CloudFrontReadOnlyAccess` 가 AWS Pricing Plan Manager에 대한 읽기 전용 권한을 허용하도록 업데이트되었습니다. | 2025년 11월 18일 |
| CloudFront에서 정액 요금제 지원 | 이제 CloudFront 정액 요금제로 배포를 구독할 수 있습니다. | 2025년 11월 18일 |
| 애니캐스트 고정 IP | 이제 IPv4 주소만 선택하거나 IPv4 및 IPv6 주소(듀얼 스택)를 모두 선택할 수 있습니다. | 2025년 11월 5일 |
| AWS 계정에서 VPC 오리진 공유에 대한 지원 추가 | 리소스 공유를 생성하고 여기에 VPC 오리진을 추가할 수 있습니다. 이를 통해 VPC 오리진과 CloudFront 배포를 별도의 AWS 계정에 유지하여 조직의 다중 계정 요구 사항을 유지할 수 있습니다. 다른 AWS 계정는 CloudFront 배포에 대해 공유 VPC 오리진을 연결할 수 있습니다. | 2025년 11월 5일 |
| 뷰어 보안 정책 추가 | TLSv1.2_2025 보안 정책이 추가되었습니다.<br>• 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호<br>• 보안 정책(최소 SSL/TLS 버전) | 2025년 10월 10일 |
| 양자 내성(PQ) 키 교환 알고리즘 추가 | CloudFront TLS 정책에 두 가지 새로운 PQ 키 교환 알고리즘이 추가되었습니다. | 2025년 9월 4일 |

| | | |
|---|---|---|
| IPv6 오리진 요청 | 사용자 지정 오리진(Amazon S3 및 VPC 오리진 제외)을 사용하는 경우 배포의 오리진 설정을 사용자 지정하여 CloudFront가 IPv4 또는 IPv6 주소를 사용하여 오리진에 연결하는 방법을 선택할 수 있습니다. | 2025년 9월 3일 |
| 새 관리형 오리진 요청 정책 추가 | 새 관리형 오리진 요청 정책 `HostHeaderOnly` 가 추가되었습니다. | 2025년 8월 29일 |
| 이제 CloudFront 퍼블릭 엔드포인트에서 IPv6 지원 | AWS 일반 참조의 Amazon CloudFront 엔드포인트 및 할당량과 Amazon VPC 사용 설명서의 IPv6를 지원하는 AWS 서비스를 참조하세요. | 2025년 8월 21일 |
| 뷰어 보안 정책 추가 | 새로운 TLS 1.3 전용 보안 정책인 TLSv1.3_2025가 추가되었습니다. | 2025년 8월 7일 |
| 새 오리진 제한 시간 설정 추가 | 모든 오리진에 대한 응답 완료 제한 시간을 추가하고 S3 오리진에 대한 응답 제한 시간(오리진 읽기 제한 시간)을 추가했습니다. | 2025년 7월 30일 |
| 사전 구성된 표준 배포 설정 추가 | 표준 배포에 대해 사전 구성된 설정이 추가되었습니다. | 2025년 6월 17일 |
| 표준 배포 도메인 설정을 위한 새 콘솔 워크플로 추가 | 표준 배포 도메인 설정을 위한 새 콘솔 워크플로가 추가되었습니다. | 2025년 6월 17일 |

| | | |
|---|---|---|
| 파라미터 예시 추가 | 배포 테넌트에서 도메인 이름 및 오리진 경로에 파라미터를 사용하는 예시가 추가되었습니다. | 2025년 6월 17일 |
| CloudFront SaaS Manager에 대한 CloudFront Functions 지원 추가 | 헬퍼 함수와 `context` 객체의 `endpoint` 필드가 추가되었습니다. | 2025년 5월 2일 |
| 표준 로깅(v2) 업데이트 | AWS Glue로 액세스 로그 전송을 지원하는 `{distributionid}` 파티션 변수를 추가했습니다. | 2025년 5월 1일 |
| CloudFront 관리형 정책 업데이트 | `CloudFrontReadOnlyAccess` 및 `CloudFrontFullAccess` 관리형 정책에 ACM 권한을 추가했습니다. | 2025년 4월 28일 |
| 다중 테넌트 배포 및 배포 테넌트에 대한 지원 추가 | 다중 테넌트 배포를 만들어 오리진 유형에 따라 공통 배포 설정을 지정할 수 있습니다. 그런 다음 다중 테넌트 배포를 재사용하여 해당 설정을 공유하는 여러 배포 테넌트를 만들 수 있습니다. 이어서 추가 웹 사이트 또는 애플리케이션을 추가할 때 특정 배포 테넌트를 사용자 지정할 수 있습니다. | 2025년 4월 28일 |
| Lambda@Edge 함수 업데이트 | Lambda@Edge 함수는 이제 고급 로깅 제어 및 CloudWatch 로그 그룹 이름 사용자 지정을 지원합니다. | 2025년 4월 7일 |

| | | |
|---|---|---|
| 애니캐스트 고정 IP | 애니캐스트 고정 IP를 사용하여 apex 도메인을 CloudFront 배포로 직접 라우팅할 수 있습니다. | 2025년 4월 4일 |
| 오리진 수정용 헬퍼 메서드 추가 | `selectRequestOriginById()` 및 `createRequestOriginGroup()` CloudFront Functions 헬퍼 메서드가 추가되었습니다. | 2025년 4월 2일 |
| 표준 로깅(v2) 업데이트 | Amazon S3에 대한 액세스 로그 전송을 위한 `{accountid}` 파티션 변수 및 예제 접미사 경로가 추가되었습니다. | 2025년 2월 14일 |
| 표준 로깅(v2)에 대한 추가 실시간 액세스 로그 필드 추가 | 표준 로깅(v2)을 사용할 때 `c-country` 및 `cache-behavior-path-pattern` 실시간 액세스 로그 필드를 지정할 수 있습니다. | 2025년 1월 31일 |
| Lambda@Edge에서 최신 런타임 버전 지원 | 이제 Lambda@Edge는 Node.js 22 런타임에서 Lambda 함수를 지원합니다. | 2024년 11월 22일 |
| CloudFront에 대한 미디어 품질 인식 복원력 지원 | CloudFront가 미디어 품질 점수가 가장 높은 오리진 그룹의 오리진을 자동으로 선택하도록 미디어 품질 인식 복원력(MQAR) 기능을 사용할 수 있습니다. | 2024년 11월 21일 |
| 오리진 수정용 헬퍼 메서드 | 오리진 수정용 새로운 CloudFront Functions 헬퍼 메서드가 추가되었습니다. | 2024년 11월 21일 |

| | | |
|---|---|---|
| VPC 오리진 | CloudFront VPC 오리진을 사용하여 Application Load Balancer, Network Load Balancer 또는 EC2 인스턴스 오리진에 대한 액세스를 제한합니다. | 2024년 11월 20일 |
| 관리형 정책 업데이트 | `CloudFrontFullAccess` 관리형 정책이 업데이트되었습니다. | 2024년 11월 20일 |
| 애니캐스트 고정 IP | CloudFront에서 배포에 사용할 애니캐스트 고정 IP를 요청할 수 있습니다. | 2024년 11월 20일 |
| 표준 로깅에 대한 지원 추가 | CloudFront는 표준 로깅(v2)과 Amazon CloudWatch Logs, Amazon Data Firehose, Amazon Simple Storage Service(Amazon S3)로 로그 전송을 지원합니다. | 2024년 11월 20일 |
| gRPC에 대한 지원 추가 | 이제 CloudFront는 배포에 대한 gRPC 요청을 지원합니다. | 2024년 11월 20일 |
| VPC 오리진에 대한 새로운 관리형 정책 추가 | 새로운 관리형 정책 `AWSCloudFrontVPCOriginServiceRolePolicy` 가 추가되었습니다. | 2024년 11월 20일 |
| Lambda@Edge에서 최신 런타임 버전 지원 | 이제 Lambda@Edge가 Python 3.13 런타임을 사용하여 Lambda 함수를 지원합니다. | 2024년 11월 13일 |
| AWS Config 규칙을 사용한 평가 | AWS Config 규칙을 사용하여 CloudFront 구성을 평가합니다. | 2024년 9월 20일 |

| | | |
|---|---|---|
| 더 많은 문제 해결 콘텐츠를 추가함 | HTTP 4xx 및 5xx 오류 응답 상태 코드에 대한 문제 해결 콘텐츠를 더 추가했습니다. | 2024년 8월 26일 |
| 새 관리형 캐시 정책 추가 | 새 관리형 캐시 정책 `UseOriginCacheControlHeaders` 및 `UseOriginCacheControlHeaders-QueryStrings` 가 추가되었습니다. | 2024년 5월 24일 |
| 오리진 액세스 제어 지원 추가 | 이제 AWS Elemental MediaPackage V2 및 AWS Lambda 함수 URL에 대한 오리진 액세스 제어(OAC)를 만들 수 있습니다. | 2024년 4월 11일 |
| CMCD의 실시간 액세스 로그 필드 | 실시간 액세스 로그를 위한 18개의 일반 미디어 클라이언트 데이터(CMCD) 필드를 추가했습니다. | 2024년 4월 9일 |
| CloudFront 표준 배포 시작 | 오리진 액세스 제어(OAC)와 함께 Amazon S3 오리진을 사용하는 표준 배포에 대한 업데이트된 자습서. | 2024년 3월 18일 |
| AWS SDK를 사용한 CloudFront 코드 예제 | AWS 소프트웨어 개발 키트(SDK)로 CloudFront를 사용하는 방법을 보여주는 코드 예제가 추가되었습니다. 예제는 개별 서비스 함수를 호출하는 방법을 보여주는 코드 발췌문과 동일한 서비스 내에서 여러 함수를 호출하여 특정 태스크를 수행하는 방법을 보여주는 예제로 나뉩니다. | 2024년 2월 16일 |

| | | |
|---|---|---|
| AWS 관리형 정책 | `CloudFrontReadOnlyAccess` 및 `CloudFrontFullAccess` IAM 정책이 이제 `KeyValueStore` 작업을 지원합니다. | 2023년 12월 19일 |
| JavaScript 런타임 2.0 | CloudFront Functions을 위한 JavaScript 런타임 2.0 기능이 추가되었습니다. | 2023년 11월 21일 |
| CloudFront KeyValueStore | Amazon CloudFront는 이제 CloudFront KeyValueStore를 지원합니다. 이 기능은 CloudFront Functions 내에서 읽기 액세스를 허용하는 안전한 글로벌 키 값 데이터 스토어입니다. CloudFront 엣지 로케이션에서 고급 사용자 지정 로직을 활성화할 수 있습니다. | 2023년 11월 21일 |
| Lambda@Edge에서 최신 런타임 버전 지원 | 이제 Lambda@Edge는 Node.js 20 런타임에서 Lambda 함수를 지원합니다. | 2023년 11월 15일 |
| 보안 대시보드 | CloudFront는 배포를 생성할 때 보안 대시보드를 생성합니다. AWS WAF를 활성화하고, 지리적 제한을 관리하고, 요청, 봇, 로그에 대한 개괄적인 데이터를 볼 수 있습니다. | 2023년 11월 8일 |
| 함수 내 쿼리 문자열 정렬 | 이제 CloudFront는 CloudFront Functions를 사용한 쿼리 문자열 정렬을 지원합니다. | 2023년 10월 3일 |
| AWS WAF 보안 권장 사항 | 이제 Amazon CloudFront는 CloudFront 콘솔에 AWS WAF 보안 권장 사항을 표시합니다. | 2023년 9월 26일 |

| | | |
|---|---|---|
| 오래된(만료된) 캐시 콘텐츠 제공 지원 | CloudFront는 `Stale-While-Revalidate` 및 `Stale-If-Error` 캐시 제어 지시문을 지원합니다. | 2023년 5월 15일 |
| 클릭 한 번으로 AWS WAF 보호 활성화 | CloudFront 배포에 AWS WAF 보안 보호를 추가하는 간소화된 방법을 제공합니다. | 2023년 5월 10일 |
| 표준 로그에 사용되는 새 S3 버킷에 대한 ACL 활성화 | 새 S3 버킷의 기본 ACL 설정을 설명하는 메모와 링크를 추가했습니다. | 2023년 4월 11일 |
| Amazon S3 객체 Lambda를 사용하여 오리진 생성 | Amazon S3 객체 Lambda 액세스 포인트 별칭을 배포의 오리진으로 사용할 수 있습니다. | 2023년 3월 31일 |
| CloudFront Functions를 사용하여 HTTP 상태 및 본문 사용자 지정 | CloudFront Functions를 사용하여 뷰어 응답 상태 코드를 업데이트하고 응답 본문을 바꾸거나 제거할 수 있습니다. | 2023년 3월 29일 |
| 포트에 CORS 헤더 와일드카드 옵션 추가 | 이제 CORS 액세스 제어 헤더의 포터에 와일드카드 구성을 포함할 수 있습니다. | 2023년 3월 20일 |
| AWS Security Hub CSPM 사용자 가이드에 대한 새 링크 추가 | AWS Security Hub CSPM 사용 설명서의 내용을 업데이트하고 재구성된 Amazon CloudFront 제어에 대한 링크를 추가했습니다. | 2023년 3월 9일 |
| CloudFront가 이제 오리진 요청 정책에서 차단 목록('다음을 제외하고 모두') 지원 | 오리진 요청 정책에서 차단 목록을 사용하여 CloudFront가 오리진에 보내는 요청에서 지정된 것을 제외한 나머지 쿼리 문자열, HTTP 헤더 또는 쿠키를 포함합니다. | 2023년 2월 22일 |

| | | |
|---|---|---|
| CloudFront가 호스트 헤더를 제외한 모든 뷰어 헤더를 전달하는 새로운 관리형 오리진 요청 정책 추가 | CloudFront의 새로운 관리형 오리진 요청 정책을 사용하면 CloudFront가 오리진으로 보내는 요청에서 `Host` 헤더를 제외한 뷰어 요청의 나머지 헤더를 모두 포함할 수 있습니다. | 2023년 2월 22일 |
| Lambda@Edge에 대한 제한 사항 업데이트 | Lambda@Edge는 자동으로 설정된 Lambda 런타임 관리 구성을 지원합니다. | 2023년 2월 16일 |
| CloudFront에 대한 IAM 지침 업데이트 | IAM 모범 사례에 따라 가이드가 업데이트되었습니다. 자세한 내용은 IAM의 보안 모범 사례를 참조하세요. | 2023년 2월 15일 |
| 오리진 액세스 제어로 개선된 보안 | 이제 지정된 CloudFront 배포에만 액세스하도록 허용하여 MediaStore 오리진을 보호할 수 있습니다. | 2023년 2월 9일 |
| 최종 사용자의 헤더 구조를 결정하기 위한 새 헤더 | 이제 헤더 순서와 헤더 수를 추가하여 전송한 헤더를 기반으로 최종 사용자를 식별할 수 있습니다. | 2023년 1월 13일 |
| Lambda@Edge에서 최신 런타임 버전 지원 | 이제 Lambda@Edge는 Node.js 18 런타임에서 Lambda 함수를 지원합니다. | 2023년 1월 12일 |
| 응답 헤더 정책을 사용하여 응답 헤더 제거 | 이제 CloudFront 응답 헤더 정책을 사용하여 CloudFront가 응답에서 수신한 헤더를 오리진에서 제거할 수 있습니다. 지정된 헤더가 CloudFront가 사용자에게 보내는 응답에 포함되지 않습니다. | 2023년 1월 3일 |

| | | |
|---|---|---|
| 구성 변경을 안전하게 테스트하기 위한 지속적 배포 | 이제 프로덕션 트래픽의 하위 집합으로 테스트하여 CDN 구성에 변경 사항을 배포할 수 있습니다. | 2022년 11월 18일 |
| `CloudFront-Viewer-JA3-Fingerprint` 헤더 릴리스 | 이제 JA3 핑거프린트를 사용하여 요청이 알려진 클라이언트에서 온 것인지 확인할 수 있습니다. | 2022년 11월 16일 |
| CORS 헤더 와일드카드 옵션 추가 | 이제 일부 CORS 액세스 제어 헤더에서 다양한 와일드카드 구성을 사용할 수 있습니다. | 2022년 11월 11일 |
| CloudFront 배포에 대한 추가 지표 | CloudFront API에서 및 CloudFormation에서 `MonitoringSubscription` 지원 | 2022년 10월 3일 |
| 오리진 액세스 제어로 개선된 보안 | 이제 지정된 CloudFront 배포에만 액세스하도록 허용하여 Amazon S3 오리진을 보호할 수 있습니다. | 2022년 8월 24일 |
| CloudFront 배포를 위한 HTTP/3 지원 | 이제 CloudFront 배포에서 HTTP/3을 선택할 수 있습니다. | 2022년 8월 15일 |
| CloudFront-Viewer-TLS 헤더에 핸드셰이크 세부 정보 추가 | 사용된 SSL/TLS 핸드셰이크에 대한 정보를 새로 볼 수 있습니다. | 2022년 6월 27일 |
| Server-Timing 헤더의 새로운 지표 | `Server-Timing` 헤더에 새 `cdn-downstream-fbl` 지표를 추가했습니다. | 2022년 6월 13일 |

| | | |
|---|---|---|
| TLS 버전 및 암호에 대한 정보를 얻을 수 있는 새 헤더 | 이제 `CloudFront-Viewer-TLS` 헤더를 사용하여 TLS(또는 SSL) 버전 및 최종 사용자와 CloudFront 간의 연결에 사용된 암호에 대한 정보를 가져올 수 있습니다. | 2022년 5월 23일 |
| CloudFront Functions에 대한 새로운 FunctionThrottles 지표 | Amazon CloudWatch를 사용하면 지정된 기간 동안 CloudFront Function가 제한된 횟수를 모니터링할 수 있습니다. | 2022년 5월 4일 |
| CloudFront의 Lambda 함수 URL 지원 | 함수 URL이 포함된 Lambda 함수를 사용하여 서버리스 웹 애플리케이션을 구축하는 경우 이제 CloudFront를 추가하여 다양한 이점을 얻을 수 있습니다. | 2022년 4월 6일 |
| HTTP 응답의 Server-Timing 헤더 | 이제 CloudFront에서 전송된 HTTP 응답의 `Server-Timing` 헤더를 사용 설정하여 CloudFront의 동작 및 성능에 대한 인사이트를 제공하는 지표를 확인할 수 있습니다. | 2022년 3월 30일 |
| AWS-관리형 접두사 목록을 사용하여 인바운드 트래픽 제한 | 이제 CloudFront의 오리진 대상 서버에 속한 IP 주소에서만 오리진으로 향하는 인바운드 HTTP 및 HTTPS 트래픽을 제한할 수 있습니다. | 2022년 2월 7일 |

| | | |
|---|---|---|
| 새로운 기능 | CloudFront는 응답 헤더 정책에 대한 지원을 추가하여 CloudFront가 뷰어(웹 브라우저 또는 기타 클라이언트)에게 보내는 HTTP 응답에 추가하는 HTTP 헤더를 지정할 수 있습니다. 원본을 변경하거나 코드를 작성하지 않고 원하는 헤더(및 해당 값)를 지정할 수 있습니다. 자세한 내용은 CloudFront 응답에서 HTTP 헤더 추가 또는 제거를 참조하세요. | 2021년 11월 2일 |
| 새 `CloudFront-Viewer-Address` 요청 헤더 | CloudFront는 HTTP 요청을 CloudFront에 보낸 뷰어의 IP 주소가 포함된 새 헤더 `CloudFront-Viewer-Address` 에 대한 지원을 추가합니다. 자세한 내용은 CloudFront 요청 헤더 추가를 참조하세요. | 2021년 10월 25일 |
| Lambda@Edge에서 새 런타임 버전 지원 | 이제 Lambda@Edge가 Python 3.9 런타임을 사용하여 Lambda 함수를 지원합니다. 자세한 내용은 지원되는 런타임을 참조하세요. | 2021년 9월 22일 |
| AWS 관리형 정책 업데이트 | CloudFront는 CloudFrontReadOnlyAccess 정책을 업데이트했습니다. 자세한 내용은 AWS 관리형 정책에 대한 CloudWatch 업데이트를 참조하세요. | 2021년 9월 8일 |

| | | |
|---|---|---|
| 새로운 기능 | CloudFront가 이제 최종 사용자 방향 HTTPS 연결을 위한 ECDSA 인증서를 지원합니다. 자세한 내용은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호 및 CloudFront에서 SSL/TLS 인증서를 사용하기 위한 요구 사항을 참조하세요. | 2021년 7월 14일 |
| 새로운 기능 | 이제 지원에 문의하지 않고도 CloudFront가 한 배포에서 다른 배포로 대체 도메인 이름을 이동할 수 있는 다양한 방법을 지원합니다. 자세한 내용은 대체 도메인 이름을 다른 배포로 이동을 참조하세요. | 2021년 7월 7일 |
| 새 보안 정책 | CloudFront는 이제 더 작은 지원 암호 집합과 함께 새 보안 정책 TLSv1.2_2021을 지원합니다. 자세한 내용은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호를 참조하세요. | 2021년 6월 23일 |
| 새로운 기능 | 이제 Amazon CloudFront는 대기 시간에 민감한 대규모 CDN 사용자 지정을 위해 JavaScript에 경량 함수를 작성할 수 있는 CloudFront의 기본 기능인 CloudFront 함수를 지원합니다. 자세한 내용은 CloudFront 함수를 사용해 엣지에서 사용자 지정을 참조하세요. | 2021년 5월 3일 |

| | | |
|---|---|---|
| Lambda@Edge에서 최신 런타임 버전 지원 | 이제 Lambda@Edge는 Node.js 14 런타임에서 Lambda 함수를 지원합니다. 자세한 내용은 지원되는 런타임을 참조하세요. | 2021년 4월 29일 |
| RTMP 배포에 대한 설명서 제거 | Amazon CloudFront는 2020년 12월 31일에 RTMP(실시간 메시징 프로토콜) 배포를 더 이상 사용하지 않습니다. RTMP 배포에 대한 설명서가 이제 Amazon CloudFront 개발자 안내서에서 제거되었습니다. | 2021년 2월 10일 |
| 새 요금 옵션 | Amazon CloudFront는 AWS 청구서의 CloudFront 요금을 최대 30% 절약할 수 있는 간단한 방법인 CloudFront Security Savings Bundle을 도입했습니다. 자세한 내용은 절감형 번들 FAQ를 참조합니다. | 2021년 2월 5일 |
| 새 자습서 | Amazon CloudFront 개발자 안내서에는 Amazon CloudFront를 사용하여 Elastic Load Balancing에서 Application Load Balancer에 대한 액세스를 제한하는 데 대한 자습서가 포함되어 있습니다. 자세한 내용은 Application Load Balancer에 대한 액세스 제한을 참조하세요. | 2020년 12월 18일 |

| | | |
|---|---|---|
| 퍼블릭 키 관리를 위한 새로운 옵션 | CloudFront에서 이제 AWS 계정 루트 사용자에 액세스할 필요 없이 CloudFront 콘솔 및 API를 통해 서명된 URL 및 쿠키에 대한 퍼블릭 키 관리를 지원합니다. 자세한 내용은 서명된 URL 및 서명된 쿠키를 생성할 수 있는 서명자 지정을 참조하세요. | 2020년 10월 22일 |
| 새로운 기능 – Origin Shield | CloudFront는 이제 오리진의 부하를 최소화하고 가용성을 높이며 운영 비용을 절감하는 데 도움이 되는 CloudFront 캐싱 인프라의 추가 계층인 CloudFront Origin Shield를 지원합니다. 자세한 내용은 Amazon CloudFront Origin Shield 사용을 참조하세요. | 2020년 10월 20일 |
| 새 압축 형식 | CloudFront는 이제 CloudFront 엣지 로케이션에서 객체를 압축하도록 CloudFront를 구성할 때 Brotli 압축 유형을 지원합니다. 정규화된 `Accept-Encoding` 헤더를 사용하여 Brotli 객체를 캐싱하도록 CloudFront를 구성할 수도 있습니다. 자세한 내용은 압축 파일 제공 및 압축 지원을 참조하세요. | 2020년 9월 14일 |

| | | |
|---|---|---|
| 새 TLS 프로토콜 | CloudFront는 이제 최종 사용자와 CloudFront 배포 간의 HTTPS 연결을 위해 TLS 1.3 프로토콜을 지원합니다. TLS 1.3은 기본적으로 모든 CloudFront 보안 정책에서 활성화됩니다. 자세한 내용은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호를 참조하세요. | 2020년 9월 3일 |
| 새로운 실시간 액세스 로그 | CloudFront는 이제 구성 가능한 실시간 액세스 로그를 지원합니다. 실시간 액세스 로그를 사용하면 배포에 대한 요청 관련 정보를 실시간으로 확인할 수 있습니다. 실시간 액세스 로그를 사용하여 콘텐츠 전송 성능을 모니터링 및 분석하고 이에 기초해 조치를 취할 수 있습니다. 자세한 내용은 실시간 로그를 참조하세요. | 2020년 8월 31일 |
| 추가 지표에 대한 API 지원 | CloudFront는 이제 CloudFront API로 추가 실시간 지표 8개를 활성화할 수 있도록 지원합니다. 자세한 내용은 추가 지표 활성화를 참조하세요. | 2020년 8월 28일 |
| 새로운 CloudFront HTTP 헤더 | CloudFront는 디바이스 유형, 지리적 위치 등 최종 사용자에 대한 정보를 확인하기 위한 추가 HTTP 헤더를 추가했습니다. 자세한 내용은 CloudFront 요청 헤더 추가를 참조하세요. | 2020년 7월 23일 |

| | | |
|---|---|---|
| 새로운 기능 | CloudFront는 이제 캐시 정책 및 오리진 요청 정책을 지원하므로 CloudFront 배포에 대한 캐시 키 및 오리진 요청을 보다 세부적으로 제어할 수 있습니다. 자세한 내용은 캐시 키 제어 및 오리진 요청 제어를 참조하세요. | 2020년 7월 22일 |
| 새 보안 정책 | CloudFront는 이제 더 작은 지원 암호 집합과 함께 새 보안 정책 TLSv1.2_2019를 지원합니다. 자세한 내용은 최종 사용자와 CloudFront 간에 지원되는 프로토콜 및 암호를 참조하세요. | 2020년 7월 8일 |
| 오리진 제한 시간 및 시도 횟수를 제어하는 새로운 설정 | CloudFront가 오리진 제한 시간 및 시도 횟수를 제어하는 새로운 설정을 추가했습니다. 자세한 내용은 오리진 시간 제한 및 시도 제어를 참조하세요. | 2020년 6월 5일 |
| 안전한 정적 웹 사이트를 생성하여 CloudFront를 시작하기 위한 새로운 문서 | CloudFormation을 통해 배포된 Amazon S3, CloudFront, Lambda@Edge 등을 사용해 안전한 정적 웹 사이트를 생성하여 CloudFront를 시작하세요. 자세한 내용은 안전한 정적 웹 사이트 시작하기를 참조하십시오. | 2020년 6월 2일 |
| Lambda@Edge에서 최신 런타임 버전 지원 | 이제 Lambda@Edge가 Node.js 12 및 Python 3.8 런타임을 사용하여 Lambda 함수를 지원합니다. 자세한 내용은 지원되는 런타임을 참조하세요. | 2020년 2월 27일 |

| | | |
|---|---|---|
| CloudWatch의 새로운 실시간 지표 | 이제 Amazon CloudFront에서 Amazon CloudWatch에 실시간 지표 8개를 추가로 제공합니다. 자세한 내용은 추가 CloudFront 배포 지표 활성화를 참조하세요. | 2019년 12월 19일 |
| 액세스 로그의 새 필드 | CloudFront가 새로운 필드 7개를 액세스 로그에 추가합니다. 자세한 내용은 표준 로그 파일 필드를 참조하세요. | 2019년 12월 12일 |
| AWS WordPress 플러그인 | AWS WordPress 플러그인을 사용하여 WordPress 웹 사이트 방문자에게 CloudFront를 통해 가속화된 시청 환경을 제공할 수 있습니다. (업데이트: 2022년 9월 30일부터 AWS for WordPress 플러그인은 더 이상 사용되지 않습니다.) | 2019년 10월 30일 |
| 태그 기반 및 리소스 수준 IAM 권한 정책 | 이제 CloudFront는 IAM 권한 정책을 지정하는 두 가지 추가 방법을 지원합니다. 이 두 가지 추가 방법은 태그 기반 및 리소스 수준 정책 권한입니다. 자세한 내용은 리소스에 대한 액세스 관리를 참조하십시오. | 2019년 8월 8일 |
| Python 프로그래밍 언어에 대한 지원 | 이제 Node.js 외에도 Python 프로그래밍 언어를 사용하여 Lambda@Edge에서 함수를 개발할 수 있습니다. 다양한 시나리오에 적용되는 예제 함수는 Lambda@Edge 예제 함수를 참조하십시오. | 2019년 8월 1일 |

| | | |
|---|---|---|
| 모니터링 그래프 업데이트 | CloudFront 콘솔에서 직접 CloudFront 배포와 관련된 Lambda 함수를 모니터링하여 오류를 보다 쉽게 추적하고 디버깅할 수 있는 새로운 방법을 설명하는 내용이 업데이트되었습니다. 자세한 내용은 CloudFront 모니터링을 참조하십시오. | 2019년 6월 20일 |
| 보안 콘텐츠 통합 | 새로운 보안 장에서는 CloudFront의 기능에 대한 정보와 함께 데이터 보호, IAM, 로깅, 규정 준수 등의 구현에 대한 정보가 제공됩니다. 자세한 내용은 보안을 참조하십시오. | 2019년 5월 24일 |
| 도메인 검증이 이제 필수 | CloudFront는 이제 사용자가 SSL 인증서를 사용하여 배포에 대체 도메인 이름을 사용할 권한이 있는지 확인할 것을 요구합니다. 자세한 내용은 대체 도메인 이름 및 HTTPS 사용을 참조하십시오. | 2019년 4월 9일 |
| 업데이트된 PDF 파일 이름 | Amazon CloudFront 개발자 안내서의 새 파일 이름은 AmazonCloudFront_DevGuide 입니다. 이전 이름은 cf-dg였습니다. | 2019년 1월 7일 |

| | | |
|---|---|---|
| 새로운 기능 | CloudFront는 현재 클라이언트와 서버 사이에 수명이 긴 연결이 필요할 때 유용한 TCP 기반 프로토콜인 WebSocket을 지원합니다. 또한 고가용성을 필요로 하는 시나리오에 대한 오리진 장애 조치를 사용하여 이제 CloudFront를 설정할 수 있습니다. 자세한 내용은 CloudFront 배포를 통한 WebSocket 사용 및 CloudFront 오리진 장애 조치를 통한 고가용성 최적화를 참조하십시오. | 2018년 11월 20일 |
| 새로운 기능 | CloudFront는 이제 Lambda 함수를 실행하는 HTTP 요청에 대한 상세 오류 로깅을 지원합니다. 로그를 CloudWatch에 저장하여 함수가 잘못된 응답을 반환할 경우 HTTP 5xx 오류 문제를 해결하는 데 사용할 수 있습니다. 자세한 내용은 Lambda 함수에 대한 CloudWatch 지표 및 CloudWatch Logs를 참조하세요. | 2012년 10월 8일 |

| | | |
|---|---|---|
| 새로운 기능 | 이제 Lambda 함수에서 액세스할 수 있도록 Lambda@Edge가 본문에 쓰기 가능한 HTTP 메서드 (POST, PUT, DELETE 등)에 대한 요청을 노출하도록 선택할 수 있습니다. 읽기 전용 액세스 권한을 선택하거나 본문을 바꾸도록 지정할 수 있습니다. 자세한 내용은 본문 포함 옵션을 선택해 요청 본문에 액세스를 참조하십시오. | 2018년 8월 14일 |
| 새로운 기능 | 이제 CloudFront는 gzip에 추가로, 또는 gzip 대신에 brotli나 기타 압축 알고리즘으로 압축된 콘텐츠의 제공을 지원합니다. 자세한 내용은 압축 파일 서비스를 참조하십시오. | 2018년 7월 25일 |
| 재구성 | Amazon CloudFront 개발자 안내서가 관련 콘텐츠 찾기를 간소화하고 검색 및 검색 기능을 향상하도록 재구성되었습니다. | 2018년 28월 6일 |
| 새로운 기능 | Lambda@Edge를 통해 이제 Amazon S3 버킷에 저장된 콘텐츠 전송을 사용자 지정할 수 있습니다. 이는 오리진 대상 이벤트 내에서 사용자 지정 헤더를 포함한 추가 헤더에 대한 액세스를 허용함으로써 가능합니다. 자세한 내용은 최종 사용자 위치 및 최종 사용자 디바이스 유형을 기반으로 한 콘텐츠 맞춤화를 보여주는 예제를 참조하십시오. | 2018년 3월 20일 |

| | | |
|---|---|---|
| 새로운 기능 | 이제 Amazon CloudFront를 사용하여 Elliptic Curve 디지털 서명 알고리즘(ECDSA)을 사용한 오리진으로의 HTTPS 연결을 협상할 수 있습니다. ECDSA는 더욱 빠르고 이전 RSA 알고리즘처럼 안전한 크기가 작은 키를 사용합니다. 자세한 내용은 CloudFront와 오리진 간 통신에 지원되는 SSL/TLS 프로토콜 및 암호 및 RSA 및 ECDSA 암호 소개를 참조하십시오. | 2018년 3월 15일 |
| 새로운 기능 | Lambda@Edge는 Amazon CloudFront가 오리진으로부터 수신하는 HTTP 오류에 대해 Lambda 함수를 실행함으로써 오리진의 오류 응답을 사용자 지정하는 기능을 제공합니다. 자세한 내용은 다른 위치로의 리디렉션 및 200 상태 코드(OK)와 함께 응답 생성을 보여주는 예제를 참조하십시오. | 2017년 12월 21일 |
| 새로운 기능 | 새로운 CloudFront 기능인 필드 레벨 암호화는 주민등록번호와 같은 신용 카드 번호 또는 개인 식별 정보(PII) 등의 중요한 보안 데이터를 강화하는 데 도움이 됩니다. 자세한 내용은 필드 수준 암호화 사용으로 민감한 데이터 보호를 참조하세요. | 2017년 12월 14일 |
| 문서 기록 아카이빙 | 이전 문서 기록이 아카이빙되었습니다. | 2017년 12월 1일 |