개발자 가이드

Amazon Simple Email Service

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

Amazon Simple Email Service: 개발자 가이드

Copyright © 2025 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 Amazon과 제휴 관계이거나 관련이 있거나 후원 관계와 관계없이 해당 소유자의 자산입니다.

Table of Contents

Amazon SES이란 무엇인가요? ... 1
- 이점 ... 1
- 관련 서비스 ... 1
- 요금 ... 2
- 리전 ... 2
  - SES 리전 및 엔드포인트 ... 3
  - 샌드박스 제거 및 발신 한도 증가 ... 3
  - 이메일 주소 및 도메인 확인 ... 4
  - Easy DKIM ... 4
  - 계정 수준 금지 목록 ... 4
  - 피드백 알림 ... 4
  - SMTP 자격 증명 ... 5
  - 사용자 지정 MAIL FROM 도메인에 사용되는 피드백 엔드포인트 ... 5
  - 전송 권한 부여 ... 6
  - 이메일 수신 ... 6
- 할당량 ... 6
  - 이메일 전송 할당량 ... 6
  - 이메일 수신 할당량 ... 10
  - Mail Manager 할당량 ... 11
  - 일반 할당량 ... 12
- 자격 증명 유형 ... 13
- Amazon SES 작동 방식 ... 16
  - 발신자가 SES로 이메일 요청을 보낸 후 ... 17
  - Amazon SES가 이메일을 전송한 후 ... 18
  - 이메일 형식 ... 20
  - 발송률 이해 ... 24
  - 이메일 모범 사례 ... 29
- AWS SDKs 작업 ... 34

시작 ... 36
- 설정 ... 36
  - 에 가입 AWS ... 36
  - SES 계정 설정 ... 37
  - 프로그래밍 방식 액세스 권한 부여(콘솔 외부에서 SES와 상호 작용하기 위한 목적) ... 37
  - AWS SDK 다운로드(SES APIs 사용용) ... 38

Amazon SES로 마이그레이션 ... 39
1단계. 도메인 확인 ... 39
2단계. 프로덕션 액세스 권한 요청 ... 39
3단계. 도메인 인증 시스템 구성 ... 39
4단계. SMTP 자격 증명 생성 ... 40
5단계. SMTP 엔드포인트에 연결 ... 40
다음 단계 ... 40
프로덕션 액세스 권한 요청 ... 41
발신 한도 ... 44
발신 할당량 높이기 ... 45
자동으로 발신 할당량 높이기 ... 46
사용자의 발신 할당량 증가 요청 ... 46
발신 할당량 모니터링 ... 47
Amazon SES 콘솔을 사용하여 발신 할당량 모니터링 ... 47
Amazon SES API를 사용하여 발신 할당량 모니터링 ... 48
발신 할당량 오류 ... 49
Amazon SES API를 통해 발신량 한도 도달 ... 49
SMTP를 통해 발신 한도 도달 ... 49
이메일 전송 설정 ... 50
SMTP 인터페이스 사용 ... 50
SMTP를 통한 이메일 전송 요구 사항 ... 51
SMTP를 통해 전자 메일을 보내는 방법 ... 51
제공할 이메일 정보 ... 52
SMTP 자격 증명 획득 ... 52
SMTP 엔드포인트에 연결 ... 60
소프트웨어 패키지를 사용하여 이메일 보내기 ... 61
프로그래밍 방식으로 이메일 전송 ... 63
기존 이메일 서버와 통합 ... 73
Amazon SES SMTP 인터페이스와의 연결 테스트 ... 86
API 사용 ... 94
서식이 지정된 이메일 보내기 ... 95
원시 이메일 보내기 ... 96
템플릿을 사용하여 이메일 보내기 ... 107
AWS SDK를 사용하여 이메일 전송 ... 129
콘텐츠 인코딩 ... 148
지원되는 보안 프로토콜 ... 148

이메일 발신자와 Amazon SES 연결 149
Amazon SES와 수신자 연결 149
종단 간 암호화 150
지원되는 헤더 필드 151
이메일 첨부 파일 153
첨부 파일의 작동 방식 154
첨부 파일 객체 구조 154
모범 사례 159
지원되지 않는 연결 형식 159
이메일 수신 161
이메일 수신 개념 및 사용 사례 162
수신 규칙을 사용한 수신자 기반 제어 162
IP 주소 필터를 사용한 IP 기반 제어 164
이메일 수신 프로세스 165
사용 사례 및 제한 165
이메일 인증 및 맬웨어 탐지 168
이메일 수신 설정하기 169
도메인 확인 절차 170
MX 레코드 게시하기 170
권한 부여하기 173
이메일 수신 콘솔 연습 181
수신 규칙 만들기 182
IP 필터 생성 219
이메일 수신 지표 220
테넌트 - 신규 224
테넌트 관리란 무엇입니까? 224
작동 방법 224
리소스 연결 225
평판 모니터링 225
설정 226
테넌트 생성 226
리소스 할당 226
평판 정책 227
이메일 전송 228
SendEmail API 228
SMTP 229

테넌트 상태 229
상태 및 지표 229
일시 중지/일시 중지 해제 230
평판 조사 결과 231
결과 보기 231
결과 이해 232
문제 해결 233
모니터링 233
CloudWatch 지표 233
EventBridge 알림 234
신뢰 및 안전 236
모범 사례 237
제한 사항 237
요금 238
확인된 자격 증명 239
자격 증명 생성 및 확인 239
도메인 자격 증명 생성 242
도메인 자격 증명 확인 246
이메일 주소 자격 증명 생성 250
이메일 주소 자격 증명 확인 252
자격 증명을 생성 및 확인하는 동시에 기본 구성 세트 할당(API) 252
사용자 지정 확인 이메일 템플릿 사용 253
ID 관리 265
콘솔을 사용하여 자격 증명 보기 265
콘솔을 사용하여 ID 삭제 266
콘솔을 사용하여 ID 편집 267
SES API를 사용하여 기본 구성 세트를 사용하도록 ID 편집 268
SES API를 사용하여 ID에 이용되는 기본 구성 세트 검색 269
SES API를 사용하여 ID에 이용되는 현재 기본 구성 세트 재정의 269
자격 증명 구성 270
이메일 인증 방법 271
이메일 알림 설정 315
자격 증명 권한 부여 사용 349
전송 권한 부여 사용 364
시뮬레이터를 사용하여 테스트 이메일 전송 395
콘솔에서 메일박스 시뮬레이터 사용 395

수동으로 메일박스 시뮬레이터 사용 ... 397
구성 세트 ... 401
구성 세트를 생성합니다. ... 402
구성 세트 생성 ... 402
구성 세트를 만듭니다(AWS CLI). ... 406
구성 세트 관리 ... 408
구성 세트 보기, 편집 및 삭제(콘솔) ... 408
구성 세트 나열(AWS CLI) ... 409
구성 세트 세부 정보 가져오기(AWS CLI) ... 409
구성 세트 삭제(AWS CLI) ... 409
구성 세트에서 이메일 전송 중지(AWS CLI) ... 409
기본 구성 세트 이해 ... 410
이벤트 대상 만들기 ... 411
IP 풀 할당 ... 416
사용자 지정 확인 및 클릭 도메인 구성 ... 416
이메일에 구성 세트 지정 ... 422
평판 지표 보기 및 내보내기 ... 422
평판 지표 내보내기 활성화 ... 423
평판 지표 내보내기 비활성화 ... 423
글로벌 엔드포인트 ... 424
글로벌 엔드포인트란 무엇입니까? ... 424
글로벌 엔드포인트 작동 방식 ... 424
글로벌 엔드포인트 설정 ... 424
사전 조건 ... 425
글로벌 엔드포인트 생성 ... 425
글로벌 엔드포인트 상태 ... 426
보조 리전 준비 ... 426
(1) 중복 구성 세트 ... 427
(2) 중복된 확인된 도메인 자격 증명 ... 428
(3) 중복 프로덕션 제한 ... 429
글로벌 엔드포인트 사용 ... 430
애플리케이션과 통합 ... 430
모니터링 및 지표 ... 431
모범 사례 및 고려 사항 ... 431
요금 ... 432
전용 IP 주소 ... 433

설정 용이성 ... 434
평판 관리 ... 435
전송 패턴의 예측 가능성 ... 435
아웃바운드 이메일 볼륨 ... 436
추가 요금 ... 436
발신자 평판 관리 ... 436
발신자 평판 격리 ... 436
변경할 수 없는, 알려진 IP 주소 ... 437
표준 ... 437
요청 및 해제 ... 438
워밍업 ... 439
풀 생성 ... 442
전용 IP 주소(관리형) ... 444
이점 및 특징 ... 445
워밍업의 중요성 ... 446
알아야 할 FAQ ... 447
관리형 IP 풀 만들기 ... 447
풀 전송 및 용량 보기 ... 451
관리형 IP 풀 삭제 ... 452
고유 IP 주소 가져오기 ... 453
요구 사항 ... 454
고려 사항 ... 454
Amazon SES에서 고유 IP 주소 사용 ... 455
가상 배달 가능성 관리자 ... 456
시작 ... 457
시작하기(콘솔) ... 457
시작하기(AWS CLI) ... 459
대시보드 ... 460
대시보드(콘솔) 사용 ... 463
지표 데이터 액세스(AWS CLI) ... 467
지표 데이터 필터링 및 내보내기(AWS CLI) ... 468
메시지 찾기, 상태 확인, 결과 내보내기(AWS CLI) ... 469
내보내기 작업 관리(AWS CLI) ... 473
메시지 세부 정보 보기(AWS CLI) ... 475
대시보드 지표가 계산되는 방법 ... 475
어드바이저 ... 477

어드바이저가 찾는 항목 ... 478
어드바이저 사용(콘솔) ... 480
권장 사항 액세스(AWS CLI) ... 481
설정 ... 482
가상 전달 가능성 관리자 설정 변경(콘솔) ... 482
가상 전달 가능성 관리자 설정 변경(AWS CLI) ... 484
메일 관리자 ... 486
시작 ... 487
시작 ... 487
수신 엔드포인트 ... 488
수신 엔드포인트 구성 ... 489
수신 엔드포인트 만들기(콘솔) ... 493
트래픽 정책 및 정책 스테이트먼트 ... 495
트래픽 정책 및 정책 스테이트먼트 만들기(콘솔) ... 496
정책 스테이트먼트 조건 ... 497
규칙 세트 및 규칙 ... 498
규칙 세트 및 규칙 만들기(콘솔) ... 499
규칙 조건 및 작업 ... 501
SMTP 릴레이 ... 504
SMTP 릴레이 만들기(콘솔) ... 505
Google Workspaces 설정 ... 508
Microsoft Office 365 설정 ... 510
주소 목록 ... 515
주소 목록이란 무엇입니까? ... 515
주소 목록 작동 방식 ... 515
주소 목록 설정 ... 515
주소 목록 사용 ... 519
모범 사례 ... 431
이메일 아카이빙 ... 522
이메일 아카이빙 사용(콘솔) ... 523
이메일 추가 기능 ... 527
추가 기능 구독(콘솔) ... 527
권한 정책 ... 530
수신 엔드포인트 정책 ... 530
SMTP 릴레이 정책 ... 531
이메일 아카이빙 정책 ... 533

규칙 작업 정책 537
로깅 543
로그 전송 설정 544
로그 해석 546
목록 및 구독 552
전역 금지 목록 553
전역 금지 목록 고려 사항 554
계정 수준 금지 목록 사용 555
계정 수준 금지 목록 고려 사항 555
계정 수준 금지 목록 활성화 557
구성 세트에 대한 계정 수준 금지 목록 사용 설정 558
계정 수준 금지 목록에 개별 이메일 주소 추가 559
계정 수준 금지 목록에 이메일 주소 일괄 추가 561
계정 수준 금지 목록에 있는 주소 목록 보기 564
계정 수준 금지 목록에서 개별 이메일 주소 제거 566
계정 수준 금지 목록에서 이메일 주소 일괄 제거 567
계정 가져오기 작업 목록 보기 571
계정 가져오기 작업에 대한 정보 가져오기 573
계정 수준 금지 목록 비활성화 574
구성 세트 수준 금지 목록 사용 574
구성 세트 레벨 금지 사용 577
목록 관리 사용 578
목록 관리 개요 578
목록 관리 구성 579
예제를 사용한 목록 관리 연습 585
구독 관리 사용 587
구독 관리 개요 587
구독 취소 헤더 고려 사항 588
구독 취소 바닥글 링크 추가 589
전송 활동 모니터링 590
콘솔을 사용한 모니터링 595
계정 대시보드 596
평판 지표 597
SMTP 설정 598
콘솔을 사용한 지표 모니터링 599
API를 사용한 모니터링 600

를 사용하여 GetSendStatistics API 작업 호출 AWS CLI 600
프로그래밍 방식의 GetSendStatistics 작업 호출 601
이벤트 게시를 사용하여 이메일 전송 모니터링 604
이벤트 게시가 구성 세트 및 메시지 태그와 작동하는 방법 605
이메일 캠페인에 대한 세분화된 피드백 605
이벤트 게시 사용 방법 607
이벤트 게시 용어 607
이벤트 게시 설정 609
이벤트 데이터 작업 623
발신자 평판 모니터링 692
평판 지표 사용 692
평판 지표 메시지 694
General Status Messages 695
반송 메일 발생률 알림 696
수신 거부 발생률 알림 697
스팸 방지 조직 알림 699
listbombing 알림 700
직접 피드백 알림 701
도메인 차단 목록 알림 702
내부 검토 알림 703
메일박스 제공업체 알림 705
수신자 피드백 알림 706
관련 계정 알림 707
스팸 트랩 알림 708
취약 사이트 알림 709
손상된 보안 인증 정보 알림 710
기타 알림 711
CloudWatch를 사용하여 경보 생성 712
전용 IP에 대한 SNDS 지표 714
문제 해결 716
이메일 전송 자동 일시 중지 716
전체 계정의 경우 717
구성 집합의 겨우 723
EventBridge를 사용하여 모니터링 733
SES 이벤트 733
이벤트 스키마 참조 735

Virtual Deliverability Manager 어드바이저 상태 스키마 736
SES 이메일 전송 상태 스키마 737
EventBridge 사용 739
EventBridge에서 샘플 이벤트 지정 740
SES 이벤트의 이벤트 패턴 740
추가 EventBridge 리소스 743
코드 예제 744
Amazon SES 746
기본 사항 747
시나리오 862
Amazon SES API v2 900
기본 사항 901
시나리오 958
보안 999
데이터 보호 1000
저장 데이터 암호화 1001
전송 중 암호화 1010
개인 데이터 삭제 1010
자격 증명 및 액세스 관리 1017
SES에 액세스하기 위한 IAM 정책 생성 1018
SES에 대한 예제 IAM 정책 1021
AWS 관리형 정책 1027
서비스 연결 역할 사용 1029
로깅 및 모니터링 1032
API 직접 호출 로깅 1032
규정 준수 확인 1043
복원성 1043
SES의 인프라 보안 1044
VPC 엔드포인트 1044
Amazon VPC에서 SES를 설정하는 연습 예제 1045
문제 해결 1049
일반 문제 1050
변경 사항이 즉시 표시되는 것은 아닙니다. 1050
확인 문제 1051
도메인 확인 문제 1051
도메인 확인 설정 확인 1052

이메일 확인 문제 1054
DKIM 문제 1054
전송 문제 1056
수신된 이메일에서 발생하는 문제 1057
알림 문제 1058
이메일 전송 오류 1059
처리량 증가 1061
SMTP 문제 1063
SMTP 응답 코드 1064
FAQ 1070
관리형 DIPS FAQs 1070
관리형 DIPS FAQ Q1 1070
관리형 DIPS FAQ Q2 1070
관리형 DIPS FAQ Q3 1071
관리형 DIPS FAQ Q4 1071
관리형 DIPS FAQ Q5 1071
관리형 DIPS FAQ Q6 1071
전송 검토 프로세스 FAQ 1072
검토 중인 계정 1072
전송 일시 중지 1075
반송 메일 1078
수신 거부 1081
스팸 트랩 1087
수동 조사 1089
DNS 블랙홀 목록(DNSBL) FAQ 1090
DNSBL FAQ Q1 1091
DNSBL FAQ Q2 1091
DNSBL FAQ Q3 1091
DNSBL FAQ Q4 1092
DNSBL FAQ Q5 1092
DNSBL FAQ Q6 1093
이메일 지표 FAQ 1094
일반 1095
열기 추적 1096
클릭 추적 1097
빠른 찾기 인덱스 1100

사용 방법 및 개념 1100
..... mcvii

Amazon SES이란 무엇인가요?

Amazon Simple Email Service(SES)는 사용자의 이메일 주소와 도메인을 사용해 이메일을 보내고 받기 위한 경제적이고 손쉬운 방법을 제공하는 이메일 플랫폼입니다.

예를 들어, 특별 행사 안내 등의 마케팅 이메일, 주문 확인서 등의 거래 이메일, 뉴스레터 등의 기타 통신문을 발송할 수 있습니다. Amazon SES를 사용하여 메일을 수신하면 이메일 자동 응답기, 이메일 구독 해제 시스템, 수신 이메일에서 고객 지원 티켓을 생성하는 애플리케이션과 같은 소프트웨어 솔루션을 개발할 수 있습니다.

Amazon SES에 관련된 다양한 주제에 대한 자세한 내용은 AWS 메시징 및 타겟팅 블로그를 참조하세요.

이점

대규모 이메일 솔루션을 구축하는 것은 비즈니스에 있어서 고비용의 복잡한 작업입니다. 이메일 서버 관리, 네트워크 구성, IP 주소 신뢰도 등의 인프라 문제를 해결해야 하기 때문입니다. 또한 많은 타사 이메일 솔루션에는 계약 및 가격 협상은 물론 상당한 초기 비용이 필요합니다. Amazon SES를 사용하면 이러한 문제들을 해결하여 Amazon.com이 대규모 고객층을 위해 구축한 고급 이메일 인프라와 다년간의 경험을 마음껏 이용할 수 있습니다.

관련 서비스

Amazon SES는 다른 AWS 제품과 원활하게 통합됩니다. 예를 들어, 다음을 수행할 수 있습니다.

- 애플리케이션에 이메일 전송 기능을 추가합니다.
- AWS SDK를 사용하거나, Amazon SES SMTP 인터페이스를 사용하거나, Amazon SES API를 직접 호출하여 Amazon EC2에서 이메일을 보낼 수 있습니다,
- AWS Elastic Beanstalk를 사용하여 이메일 지원 애플리케이션(예를 들어 Amazon SES를 사용하여 고객에게 뉴스레터를 발송하는 프로그램)을 작성할 수 있습니다.
- 반송되었거나 수신 거부가 제출되었거나 수신자의 메일 서버로 성공적으로 전송된 이메일에 대한 알림을 수신하도록 Amazon Simple Notification Service(Amazon SNS)를 설정할 수 있습니다. Amazon SES로 이메일을 수신하면 이메일 콘텐츠를 Amazon SNS 주제에 게시할 수 있습니다.
- AWS Management Console 를 사용하여 이메일을 인증하는 방법인 Easy DKIM을 설정합니다. 어떤 DNS 공급자에서도 Easy DKIM을 사용할 수 있지만, Route 53을 사용하여 도메인을 관리할 경우 설정이 특히 용이합니다.

- AWS Identity and Access Management (IAM)을 사용하여 이메일 전송에 대한 사용자 액세스를 제어할 수 있습니다.
- Amazon Simple Storage Service(Amazon S3)에서 수신한 이메일을 저장합니다.
- AWS Lambda 함수를 트리거하여 수신 이메일에 여러 가지 작업을 수행할 수 있습니다.
- 필요에 따라 AWS Key Management Service (AWS KMS)를 사용해 Amazon S3 버킷에 수신하는 메일을 암호화할 수 있습니다.
- AWS CloudTrail를 사용하여 콘솔 또는 Amazon SES API를 통해 생성한 Amazon SES API 호출을 기록할 수 있습니다.
- 이메일 전송 이벤트를 Amazon CloudWatch 또는 Amazon Data Firehose에 게시할 수 있습니다. 이메일 전송 이벤트를 Firehose에 게시하면 Amazon Redshift, Amazon OpenSearch Service 또는 Amazon S3에서 액세스할 수 있습니다.

요금

Amazon SES를 사용하면 전송 및 수신된 이메일의 볼륨에 따라 비용을 지불할 수 있습니다. 자세한 내용은 Amazon SES 요금을 참조하세요.

리전 및 Amazon SES

SES는 AWS 리전 전 세계 여러에서 사용할 수 있습니다. AWS 는 각 리전에서 여러 가용 영역을 유지합니다. 이러한 가용 영역은 물리적으로 서로 분리되어 있지만, 지연 시간이 짧고 처리량과 중복성이 우수한 프라이빗 네트워크 연결로 통합됩니다. 이러한 가용 영역을 사용하여 아주 높은 수준의 가용성과 중복성을 제공하면서 지연 시간을 최소화할 수 있습니다.

SES 리전 엔드포인트의 전체 목록은 AWS 일반 참조의 Amazon Simple Email Service 엔드포인트 및 할당량을 참조하세요. 각 리전에서 사용할 수 있는 가용 영역의 수를 알아보려면 AWS 글로벌 인프라를 참조하세요.

이 섹션에서는 여러 AWS 리전에서 SES를 사용하기 전에 꼭 알아야 할 내용을 설명합니다. 다음 내용으로 구성됩니다.

- SES 리전 및 엔드포인트
- 샌드박스 제거 및 발신 한도 증가
- 이메일 주소 및 도메인 확인
- Easy DKIM
- 계정 수준 금지 목록

- 피드백 알림
- SMTP 자격 증명
- 전송 권한 부여
- 사용자 지정 MAIL FROM 도메인에 사용되는 피드백 엔드포인트
- 이메일 수신
- (MX) 레코드 설정

에 대한 일반적인 내용은 일반 AWS 리전참조의 AWS 서비스 엔드포인트를 AWS 참조하세요.

SES 리전 및 엔드포인트

SES를 사용하여 이메일을 전송하는 경우 SES API 또는 SMTP 인터페이스에 대한 엔드포인트를 제공하는 URL에 연결합니다. 이 AWS 일반 참조에는 SES를 통해 이메일을 보내고 받는 데 사용하는 엔드포인트의 전체 목록이 포함되어 있습니다. 자세한 내용은 아래 참조된 AWS 일반 참조특정 섹션의 Amazon Simple Email Service 엔드포인트 및 할당량을 참조하세요.

- API 엔드포인트 - SES를 통해 이메일을 보낼 때 이 테이블에 나열된 URL을 사용하여 SES API에 HTTPS 요청을 할 수 있습니다.
- SMTP 엔드포인트 - SMTP 인터페이스를 사용할 때 이 테이블에 나열된 URL을 사용하여 이메일을 보낼 수 있습니다.
- 이메일 수신 엔드포인트 - 도메인으로 전송되는 이메일을 수신하도록 SES를 구성한 경우 도메인의 DNS 설정에서 메일 교환기(MX) 레코드를 설정할 때 이 테이블에 나열된 인바운드 SMTP 엔드포인트 URL을 사용할 수 있습니다.

> **Note**
>
> 인바운드 SMTP URL은 IMAP 서버 주소가 아닙니다. 즉, 이 URL은 Outlook과 같은 애플리케이션을 사용하여 이메일을 수신하는 데 사용할 수 없습니다. 수신 이메일용 IMAP 서버를 제공하는 서비스는 Amazon WorkMail을 참조하세요.

샌드박스 제거 및 발신 한도 증가

계정의 샌드박스 상태는 AWS 리전마다 다를 수 있습니다. 즉, 계정이 미국 서부(오리건) 리전의 샌드박스에서 제거된 경우, 미국 동부(버지니아 북부) 리전의 샌드박스에서도 계정을 제거한 경우가 아니라면 후자 리전의 샌드박스에 계정이 남아 있을 수 있습니다.

발신 한도는 AWS 리전에 따라 다를 수도 있습니다. 예를 들면, 사용 중인 계정이 유럽(아일랜드) 리전에서 초당 10건의 메시지를 발신할 수 있는 경우, 다른 리전에서는 이보다 더 많거나 또는 더 적은 메시지를 보낼 수 있습니다.

샌드박스에서 계정을 제거하라는 요청을 제출하거나 계정의 발신 할당량을 늘리라는 요청을 제출할 때는 해당 요청이 적용되는 모든 AWS 리전 을 선택해야 합니다. 단일 지원 센터 사례에서 여러 건의 요청을 제출할 수 있습니다.

이메일 주소 및 도메인 확인

SES를 사용하여 이메일을 보내려면 먼저 발신 이메일 주소 또는 도메인이 사용자 본인의 소유인지 확인해야 합니다. 이메일 주소 및 도메인의 확인 상태도 AWS 리전마다 다릅니다. 예를 들면, 미국 서부(오리건) 리전의 도메인을 확인하는 경우, 미국 동부(버지니아 북부) 리전의 확인 프로세스를 다시 완료할 때까지 후자의 리전에서 이메일을 전송하기 위해 전자의 도메인을 사용할 수는 없습니다. 이메일 주소 및 도메인 확인에 대한 자세한 내용은 Amazon SES에서 확인된 자격 증명을 참조하세요.

Easy DKIM

Easy DKIM을 사용하려는 각 AWS 리전 에 대해 Easy DKIM 설정 프로세스를 수행해야 합니다. 즉, 각 리전에서 SES 콘솔 또는 SES API를 사용하여 CNAME 레코드를 생성해야 합니다. 그런 다음, 도메인에 대한 DNS 구성에 모든 CNAME 레코드를 추가해야 합니다. Easy DKIM 설정에 대한 자세한 내용은 Amazon SES에서 Easy DKIM 섹션을 참조하세요.

모든 사용자가 기본 SES DKIM 도메인을 AWS 리전 사용하는 것은 아닙니다. 리전에서 리전별 DKIM 도메인을 사용하는지 `dkim.amazonses.com`확인하려면에서 DKIM 도메인 테이블을 확인하세요AWS 일반 참조.

계정 수준 금지 목록

SES 계정 수준 금지 목록은 현재의 AWS 계정 에만 적용됩니다 AWS 리전. SES API v2 또는 콘솔을 사용하여 계정 수준 금지 목록에서 주소를 개별적으로 또는 일괄적으로 수동으로 추가하거나 제거할 수 있습니다. 계정 수준 금지 목록 사용에 대한 자세한 내용은 Amazon SES 계정 수준 금지 목록 사용 섹션을 참조하세요.

피드백 알림

여러 AWS 리전에서 피드백 알림을 설정하는 것에 대한 2가지 중요 사항은 다음과 같습니다.

- 확인된 ID 설정(예: 이메일과 SNS 중 어느 것을 통해 피드백을 받는지 등)은 이를 설정하는 리전에만 적용됩니다. 예를 들어, 미국 서부(오리건)과 미국 동부(버지니아 북부) 리전에서

user@example.com을 확인하고 SNS 알림을 통해 반송된 이메일을 수신하려면 SES API 또는 SES 콘솔을 사용하여 두 리전 모두의 user@example.com에 대해 SNS 피드백 알림을 설정해야 합니다.

- 피드백 전달에 사용하는 SNS 주제는 SES를 사용하는 리전과 동일한 리전에 있어야 합니다.

피드백 알림을 통한 전송 활동 모니터링에 대한 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 섹션을 참조하세요.

SMTP 자격 증명

SES SMTP 인터페이스를 통해 이메일을 전송하는 데 사용하는 자격 증명은 각 AWS 리전마다 고유합니다. SES SMTP 인터페이스를 사용하여 둘 이상의 리전에서 이메일을 전송하는 경우, 각 리전에 대해 SMTP 자격 증명 세트를 생성해야 합니다.

> **Note**
>
> 2019년 1월 10일 이전에 SMTP 자격 증명을 생성한 경우 이전 버전의 AWS 서명을 사용하여 SMTP 자격 증명을 생성했습니다. 보안을 위해 이 날짜 이전에 만든 자격 증명을 삭제하고 새로운 자격 증명으로 교체하세요. IAM 콘솔을 사용하여 이전 자격 증명을 삭제할 수 있습니다.

사용자 지정 MAIL FROM 도메인에 사용되는 피드백 엔드포인트

사용자 지정 MAIL FROM 도메인을 사용하는 경우 SES에서 MX 레코드를 게시하도록 요구하여 해당 도메인이 이메일 공급자가 보내는 반송 메일과 수신 거부 알림을 받을 수 있도록 합니다. 반송 메일 및 수신 거부 알림이 리전별 피드백 엔드포인트로 전송 AWS 리전 되므로 다른에서 확인된 자격 증명에 동일한 사용자 지정 MAIL FROM 도메인을 사용할 수 있습니다.

사용자 지정 MAIL FROM 도메인을 구성하면 SES는 사용자 지정 MAIL FROM이 구성되고 있는 리전에 대해 올바른 피드백 엔드포인트를 자동으로 지정합니다. 이 엔드포인트는 도메인의 DNS 구성에 게시(추가)할 수 있도록 MX 레코드의 값 필드에 제공됩니다.

사용자 지정 MAIL FROM 설정 프로세스에 대한 설명은 사용자 지정 MAIL FROM 도메인 사용에 나와 있습니다. 참고로 SES가 다른에 사용하는 피드백 엔드포인트 AWS 리전 는의 피드백 엔드포인트 표에 나열되어 있습니다 AWS 일반 참조.

전송 권한 부여

위임 발신자는 자격 증명 소유자의 자격 증명이 확인된 AWS 리전 에서만 이메일을 보낼 수 있습니다. 위임 발신자에게 권한을 부여하는 전송 권한 부여 정책이 해당 리전의 ID에 연결되어야 합니다. 권한 부여 전송에 대한 자세한 내용은 Amazon SES에서 전송 권한 부여 사용 섹션을 참조하세요.

이메일 수신

Amazon S3 버킷을 제외하고 SES로 이메일을 수신하는 데 사용하는 모든 AWS 리소스는 AWS 리전 SES 엔드포인트와 동일해야 합니다. 예를 들어, SES를 미국 서부(오리건) 리전에서 사용하는 경우 사용하는 SNS 주제, KMS 키 및 Lambda 함수는 미국 서부(오리건) 리전에도 있어야 합니다. 마찬가지로 어떤 리전 내에서 SES로 이메일을 수신하려면 해당 리전 내에 활성 수신 규칙 세트를 만들어야 합니다. 이메일 수신 개념 및 설정 프로세스는 Amazon SES로 이메일 수신에 설명되어 있습니다.

의 이메일 수신 엔드포인트 테이블에는 SES가 이메일 수신을 지원하는 모든에 대한 이메일 수신 엔드포인트 AWS 리전 가 AWS 일반 참조 나열됩니다.

Amazon SES의 서비스 할당량

다음 단원에서는 Amazon SES 리소스 및 작업에 적용되는 할당량을 나열하고 설명합니다. 일부 할당량만 늘릴 수 있습니다. 할당량 증가를 요청할 수 있는지 여부를 확인하려면 증가 가능 여부 (Adjustable) 열을 참조하세요.

> **Note**
>
> SES 할당량은에서 AWS 리전 사용하는 각에 대한 할당량입니다 AWS 계정.

이메일 전송 할당량

다음 할당량은 SES를 통해 이메일을 보낼 때 적용됩니다.

전송 할당량

할당량은 메시지 수가 아니라 수신자 수를 기준으로 합니다.

리소스	기본 할당량	조정 가능
24시간의 기간당 보낼 수 있는 이메일 수	계정이 샌드박스에 있는 경우 24시간당 최대 200개의 이메일을 보낼 수 있습니다. 계정이 샌드박스 외부에 있는 경우, 이 수는 구체적인 사용 사례에 따라 달라집니다.	예
초당 보낼 수 있는 이메일 수(발신 속도)	계정이 샌드박스에 있는 경우 초당 이메일 1개를 보낼 수 있습니다. 계정이 샌드박스 외부에 있는 경우, 이 속도는 구체적인 사용 사례에 따라 달라집니다.	예

메시지 할당량

리소스	기본 할당량	조정 가능
SES v1 사용 - 최대 메시지 크기(첨부 파일 포함)	메시지당 10MB(base64 인코딩 후)	아니요(메시지 크기가 10MB를 초과하는 워크로드의 경우 SES v2 API로 마이그레이션하는 것이 좋음)
SES v2 API 또는 SMTP 사용 - 최대 메시지 크기(첨부 파일 포함)	메시지당 40MB(base64 인코딩 후).	아니요

> **Note**
>
> 10MB보다 큰 메시지에는 대역폭 제한이 적용되며, 전송 속도에 따라 40MB/s까지 제한될 수 있습니다. 예를 들어 초당 1개의 40MB 메시지 또는 초당 2개의 20MB 메시지를 보낼 수 있습니다.

발신자 및 수신자 할당량

리소스	기본 할당량	조정 가능
메시지당 최대 수신자 수	메시지당 수신자 50명 **Note** 수신자는 모든 "To", "CC" 또는 "BCC" 주소입니다.	아니요.
확인할 수 있는 최대 자격 증명 수	당 10,000개의 자격 증명 AWS 리전 **Note** 자격 증명은 SES를 통해 이메일을 보낼 때 사용하는 도메인 또는 이메일 주소입니다.	사용 사례에 대해 논의하려면 AWS 계정 관리자에게 문의하세요.
전용 IP 풀의 최대 수(관리형 및 표준 IP 풀 모두 포함)	50	아니요

이벤트 게시와 관련된 할당량

리소스	기본 할당량	조정 가능
최대 구성 세트 개수	10,000	아니요
구성 세트 이름의 최대 길이	구성 세트 이름에는 최대 64개의 영숫자가 포함될 수 있습니다. 또한 하이픈(-) 및 밑줄(_)을 포함할 수 있습니다. 이름에는 공백, 억양 표시가 되어 있는 문자 또는 기타 특수 문자를 사용할 수 없습니다.	아니요
구성 세트당 최대 이벤트 대상 개수	10	아니요
CloudWatch 이벤트 대상당 최대 크기 수	10	아니요

이메일 템플릿 할당량

리소스	기본 할당량	조정 가능
각의 최대 이메일 템플릿 수 AWS 리전	20,000건	아니요
최대 템플릿 크기	500KB	아니요
각 템플릿의 최대 대체 값 수	무제한	N/A
각 템플릿 이메일의 최대 수신인 수	대상 50개. 대상은 "To", "CC", "BCC" 줄의 이메일 주소입니다.	아니요

리소스	기본 할당량	조정 가능
	Note 단일 API 호출에서 연락할 수 있는 대상 수는 계정의 최대 전송 속도에 의해 제한될 수 있습니다.	

이메일 수신 할당량

다음 테이블에 SES를 통한 이메일 수신과 관련된 할당량이 나와 있습니다.

리소스	기본 할당량	조정 가능
수신 규칙 세트당 최대 수신 규칙 수	200	아니요
수신 규칙당 최대 작업 수	10	아니요
수신 규칙당 최대 수신자 수	500	아니요
당 최대 수신 규칙 세트 수 AWS 계정	40	아니요
당 최대 IP 주소 필터 수 AWS 계정	100	아니요
Amazon S3 버킷에 저장할 수 있는 최대 이메일 용량(헤더 포함)	40MB	아니요
Amazon SNS 알림에 게시할 수 있는 최대 이메일 용량(헤더 포함)	150KB	아니요

리소스	기본 할당량	조정 가능
Amazon SNS 알림을 사용하여 게시할 수 있는 최대 이메일 헤더 크기	10KB	아니요
AWS Lambda 함수를 사용하여 게시할 수 있는 최대 이메일 헤더 크기	50KB	아니요

Mail Manager 할당량

다음 테이블에는 Mail Manager와 관련된 할당량이 나열되어 있습니다.

리소스	기본 할당량	조정 가능
진행 중인 최대 수신 엔드포인트 수	10	아니요
권한 있는 최대 수신 엔드포인트 수	50	아니요
메시지당 최대 수신자 수	100	아니요
최대 이메일 크기(헤더 포함)	40MB	아니요
최대 트래픽 정책 문 수	20	아니요
최대 트래픽 정책 문 조건 수	10	아니요
리전당 최대 트래픽 정책 수	100	아니요
최대 SMTP 릴레이 수	100	아니요
리전당 최대 주소 목록 수	100	아니요
주소 목록당 최대 주소 수	100,000건	아니요
최대 규칙 세트 수	40	아니요

리소스	기본 할당량	조정 가능
메시지당 최대 규칙 실행 수	40	아니요
규칙당 최대 조건 수	10	아니요
규칙당 최대 작업 수	10	아니요
규칙 세트당 최대 릴레이 또는 전송 작업 수	10	아니요
최대 활성 아카이브 수	10	아니요
최대 아카이브 검색 결과 수	1000	아니요
내보낸 아카이브 검색 결과의 최대 수	250,000	아니요
병렬로 실행되는 최대 검색 요청 수	1	아니요
병렬로 실행되는 최대 내보내기 요청 수	1	아니요
주당 아카이브의 최대 보존 변경 횟수	1	아니요

일반 할당량

다음 테이블에 SES를 통한 이메일 송수신에 모두 적용되는 할당량이 나와 있습니다.

SES API 전송 할당량

리소스	기본 할당량	조정 가능
Amazon SES API 작업을 호출할 수 있는 속도	모든 작업(`SendEmail`, `SendRawEmail` 및 `SendTemplatedEmail` 제	아니요

리소스	기본 할당량	조정 가능
	외)이 초당 요청 하나로 제한됩니다.	
MIME 부분	500	아니요

SES 기타 할당량

리소스	기본 할당량	조정 가능
최대 동시 가져오기 작업 수	20	아니요
최대 동시 내보내기 작업 수	20	아니요

Amazon SES 자격 증명 유형

Amazon Simple Email Service(Amazon SES)와 상호 작용하려면 보안 자격 증명을 사용하여 본인을 확인하고 Amazon SES와 상호 작용할 권한이 있는지 확인합니다. 자격 증명에는 여러 유형이 있으며, 사용하는 자격 증명은 수행할 작업에 따라 다릅니다. 예를 들어, Amazon SES API를 사용하여 이메일을 보낼 때는 AWS 액세스 키를 사용하고, Amazon SES SMTP 인터페이스를 사용하여 이메일을 보낼 때는 SMTP 자격 증명을 사용합니다.

수행하는 작업에 따라 Amazon SES에서 사용할 수 있는 자격 증명 유형이 다음 표에 나와 있습니다.

액세스할 대상	사용할 자격 증명	자격 증명의 구성 요소	자격 증명을 가져오는 방법
Amazon SES API (Amazon SES API에 직접 액세스하거나 AWS SDK, AWS Command Line Interface또는를 통해 간접적으로 액세스할 수	AWS 액세스 키	액세스 키 ID 및 보안 액세스 키	AWS 일반 참조의 액세스 키를 참조하세요. Note 보안 모범 사례를 위해 액세스 키 대신 AWS Identity and Access Management (IAM) 사

액세스할 대상	사용할 자격 증명	자격 증명의 구성 요소	자격 증명을 가져오는 방법
있습니다 AWS Tools for Windows PowerShell.)			용자 AWS 계정 액세스 키를 사용합니다. 자격 AWS 계정 증명은 모든 AWS 리소스에 대한 전체 액세스 권한을 부여하므로 리소스를 안전한 장소에 저장하고 대신 day-to-day 상호 작용에 IAM 사용자 자격 증명을 사용해야 합니다 AWS. 자세한 내용은 AWS 일반 참조의 루트 계정 자격 증명과 IAM 사용자 자격 증명 비교를 참조하세요.

액세스할 대상	사용할 자격 증명	자격 증명의 구성 요소	자격 증명을 가져오는 방법
Amazon SES SMTP 인터페이스	SMTP 자격 증명	사용자 이름 및 암호	Amazon SES SMTP 자격 증명 획득을(를) 참조하세요. **Note** Amazon SES SMTP 자격 증명은 AWS 액세스 키 및 IAM 사용자 액세스 키와 다르지만 Amazon SES SMTP 자격 증명은 사실상 IAM 자격 증명의 일종입니다. IAM 사용자는 Amazon SES SMTP 자격 증명을 생성할 수 있지만 루트 계정 소유자는 IAM 사용자의 정책이 이들에게 "iam:ListUsers", "iam:CreateUser", "iam:CreateAccessKey" 및 "iam:PutUserPolicy" 등의 IAM 작업에 액세스할 권한을 부여하는지 확인해야 합니다.

액세스할 대상	사용할 자격 증명	자격 증명의 구성 요소	자격 증명을 가져오는 방법
Amazon SES 콘솔	IAM 사용자 이름과 암호 OR 이메일 주소와 암호	IAM 사용자 이름과 암호 OR 이메일 주소와 암호	AWS 일반 참조의 IAM 사용자 이름과 암호 및 이메일 주소 및 암호를 참조하세요. **Note** 보안 모범 사례에서는 이메일 주소와 암호 대신 IAM 사용자 이름과 암호를 사용합니다. 이메일 주소와 암호 조합은 용 AWS 계정이므로 day-to-day 상호 작용에 사용하는 대신 안전한 장소에 저장해야 합니다 AWS. 자세한 내용은 AWS 일반 참조의 루트 계정 자격 증명과 IAM 사용자 자격 증명 비교를 참조하세요.

다양한 유형의 AWS 보안 자격 증명(Amazon SES에만 사용되는 SMTP 자격 증명 제외)에 대한 자세한 내용은의 AWS 보안 자격 증명을 참조하세요AWS 일반 참조.

Amazon SES를 통한 이메일 전송 작동 방식

이 주제에서는 SES를 통해 이메일을 전송할 때 어떤 과정을 거치는지 설명하고 이메일 전송 후 발생할 수 있는 다양한 결과에 대해 설명합니다. 다음 그림은 전송 프로세스의 종합적 개요입니다.

1. 이메일 발신자 역할을 담당하는 클라이언트 애플리케이션이 SES에 하나 이상의 수신자에게 이메일을 전송하는 요청을 합니다.
2. 요청이 유효하면 SES가 이메일을 수락합니다.
3. SES는 인터넷을 통해 수신자의 수신 장치에 메시지를 보냅니다. SES로 전달된 메시지는 일반적으로 즉시 전송됩니다. 통상적으로 최초 전송 시도가 몇 밀리초 이내에 이루어집니다.
4. 이 시점에서 다양한 가능성이 존재합니다. 예:
   a. ISP가 성공적으로 메시지를 수신자의 받은 편지함으로 전송합니다.
   b. 수신자의 이메일 주소가 존재하지 않습니다. 따라서 ISP가 SES로 반송 메일 알림을 전송합니다. 그러면 SES가 알림을 발신자에게 전달합니다.
   c. 수신자가 메시지를 수신하지만 이를 스팸으로 여겨 ISP에게 수신 거부를 제기합니다. SES와의 피드백 루프가 설정되어 있는 ISP가 SES로 수신 거부를 전송하고 SES가 이를 발신자에게 전달합니다.

다음 섹션에서는 발신자가 SES로 이메일 요청을 보낸 후, 그리고 SES가 수신자에게 이메일 메시지를 전송한 후 발생 가능한 개별 결과를 알아봅니다.

발신자가 SES로 이메일 요청을 보낸 후

발신자가 SES로 이메일 전송 요청을 할 경우 호출이 성공할 수도 실패할 수도 있습니다. 다음 단원에서 각 경우에 어떤 일이 벌어지는지 설명합니다.

전송 요청 성공

SES에 대한 요청이 성공하면 SES가 발신자에게 성공 응답을 반환합니다. 이 메시지에는 요청을 고유하게 식별하는 문자열인 메시지 ID가 포함됩니다. 메시지 ID를 사용하여 전송된 이메일을 식별하거나

전송 중에 발생한 문제를 추적할 수 있습니다(SES가 이메일을 수락할 때 반환하는 SES 메시지 ID와 식별자 사이의 자체 매핑을 저장해야 함). 그런 다음 SES가 요청 파라미터를 기반으로 이메일 메시지를 수집하고 메시지에서 의심스러운 콘텐츠 및 바이러스를 검사한 다음 간이 전자 우편 전송 프로토콜(SMTP)을 사용하여 인터넷을 통해 전송합니다. 메시지는 일반적으로 즉시 전송됩니다. 통상적으로 최초 전송 시도가 몇 밀리초 이내에 이루어집니다.

> **Note**
>
> SES가 발신자의 요청을 수락한 다음 메시지에 바이러스가 포함된 것을 확인할 경우 SES는 해당 메시지 처리를 중지하고 해당 메시지를 수신자의 메일 서버로 전송하지 않습니다.

전송 요청 실패

발신자가 SES로 보낸 이메일 전송 요청이 실패할 경우 SES가 발신자에게 오류를 반환하고 이메일을 거부합니다. 요청이 실패하는 원인은 여러 가지가 있을 수 있습니다. 예를 들어 요청이 적절한 형식이 아니거나 발신자가 이메일 주소를 확인하지 않았을 수 있습니다.

사용자가 요청 실패 여부를 확인할 수 있는 방법은 SES를 호출한 방법에 따라 다릅니다. 다음 예제는 오류 및 예외가 반환되는 방식입니다.

- 쿼리(HTTPS) API(`SendEmail` 또는 `SendRawEmail`)를 통해 SES를 호출하는 경우 작업이 오류를 반환합니다. 자세한 내용은 Amazon Simple Email Service API 참조 단원을 참조하십시오.
- 예외를 사용하는 프로그래밍 언어에 AWS SDK를 사용하는 경우 SES 호출 시 MessageRejectedException이 발생합니다. (예외의 이름은 SDK에 따라 약간 다를 수 있습니다.)
- SMTP 인터페이스를 사용하는 경우 발신자가 SMTP 응답 코드를 수신하지만, 오류가 전달되는 방식은 발신자의 클라이언트에 따라 달라집니다. 클라이언트에 따라 오류 코드가 표시될 수도, 표시되지 않을 수도 있습니다.

SES를 통해 이메일을 전송할 때 발생할 수 있는 오류에 대한 자세한 내용은 Amazon SES 이메일 전송 오류 섹션을 참조하세요.

Amazon SES가 이메일을 전송한 후

발신자가 SES로 보낸 요청이 성공할 경우 SES가 이메일을 전송하고 다음 결과 중 하나가 발생합니다.

- 이메일이 성공적으로 전송되고 수신자가 이메일을 거부하지 않음 – ISP가 이메일을 수락하고 ISP가 이메일을 수신자에게 전송합니다. 다음 그림에 전송 성공이 나와 있습니다.

- 하드 바운스 – 영구적 조건으로 인해 ISP가 거부했거나 이메일 주소가 SES 금지 목록에 있어 SES가 거부한 이메일입니다. SES 금지 목록에는 최근 SES 고객에게 하드 바운스를 발생시킨 이메일 주소가 들어 있습니다. ISP에서의 하드 바운스는 수신자 주소가 유효하지 않기 때문에 발생할 수 있습니다. 하드 바운스 알림은 ISP에서 SES로 다시 전송되고, SES는 발신자의 설정에 따라 이메일 또는 Amazon Simple Notification Service(Amazon SNS)를 통해 발신자에게 이를 알립니다. SES는 동일한 방법으로 발신자에게 금지 목록 반송 메일 알림을 전송합니다. 다음 그림에는 ISP로부터 시작하는 하드 바운스의 경로가 나와 있습니다.

- 소프트 바운스 – ISP는 일시적 상태(예: ISP가 처리할 요청이 너무 많거나 수신자의 메일박스가 가득 차 있음)로 인해 수신자에게 이메일을 전송하지 못할 수 있습니다. 도메인이 존재하지 않는 경우에도 소프트 바운스가 발생할 수 있습니다 ISP는 SES로 소프트 바운스 알림을 다시 전송합니다. 또는 도메인이 없는 경우 SES가 해당 도메인의 이메일 서버를 찾을 수 없습니다. 두 경우 모두 SES는 이메일 전송을 장시간 재시도합니다. SES가 재시도에서도 이메일을 전송할 수 없는 경우에는 이메일 또는 Amazon SNS를 통해 반송 메일 알림을 전송합니다. 재시도에서 SES가 수신자에게 이메일을 전송할 수 있으면 전송은 성공입니다. 다음 그림에는 소프트 바운스가 나와 있습니다. 이 경우, SES가 이메일 전송을 재시도하여, 결국 ISP가 수신자에게 이메일을 전송합니다.

- 수신 거부 – ISP가 수락하여 수신자에게 전송되었지만 수신자가 이 이메일을 스팸으로 간주하여 이메일 클라이언트에서 '스팸으로 표시'와 같은 버튼을 클릭한 이메일입니다. SES에 ISP와의 피드백 루프가 설정되어 있는 경우, 수신 거부 알림이 SES에 전송되고 SES는 이 수신 거부 알림을 발신자에게 전달합니다. 대부분의 ISP가 수신 거부를 제출한 수신자의 이메일 주소를 제공하지 않으므로, SES가 전달하는 수신 거부 알림은 원래 메시지의 수신자 그리고 SES에 수신 거부를 보낸 ISP를 기준으로 수신 거부를 제출했을 수 있는 수신자의 목록을 발신자에게 제공합니다. 다음 그림에는 수신 거부의 경로가 나와 있습니다.

- 자동 응답 – 이 이메일은 ISP가 수락하여 수신자에게 전송한 이메일입니다. 그런 다음 ISP가 부재중(OOTO) 메시지와 같은 자동 응답을 SES로 전송합니다. 그러면 SES가 자동 응답 알림을 발신자에게 전달합니다. 다음 그림에는 자동 응답이 나와 있습니다.

자동 응답을 생성하는 메시지는 SES 지원 프로그램이 전송을 재시도하지 않도록 하세요.

> **Tip**
>
> SES 메일박스 시뮬레이터를 사용하여 전송 성공, 반송 메일, 수신 거부, OOTO 또는 주소가 금지 목록에 포함된 경우의 결과를 테스트할 수 있습니다. 자세한 내용은 수동으로 메일박스 시뮬레이터 사용 단원을 참조하십시오.

Amazon SES의 이메일 형식

클라이언트가 Amazon SES로 요청을 보내면 Amazon SES가 인터넷 메시지 형식 사양(RFC 5322)을 준수하여 이메일 메시지를 구성합니다. 이메일은 아래에 설명된 대로 헤더, 본문 및 엔벌로프로 구성됩니다.

- 헤더 - 라우팅 지침과 메시지에 대한 정보를 포함합니다. 발신자 주소, 수신자 주소, 제목, 날짜 등이 그 예입니다. 헤더는 우편 서신의 상단에 기재되는 정보에 비유할 수 있지만, 메시지 형식 등 다른 많은 유형의 정보를 포함할 수 있습니다.
- 본문 - 메시지 텍스트 자체를 포함합니다.
- 엔벨로프 - SMTP 세션에서 이메일 클라이언트와 메일 서버 사이에 통신되는 실제 라우팅 정보를 포함합니다. 이 이메일 엔벌로프 정보는 편지 봉투에 기재되는 정보에 비유할 수 있습니다. 이메일 엔벌로프의 라우팅 정보는 일반적으로 이메일 헤더의 라우팅 정보와 동일하지만 반드시 그런 것은 아닙니다. 예를 들어 숨은 참조 수신자(BCC)에게 전송할 경우, 실제 수신자 주소(엔벌로프로부터 파생)가 수신자의 이메일 클라이언트에 표시되는 "To" 주소(헤더로부터 파생)와 다릅니다.

다음은 간단한 이메일의 예입니다. 헤더 다음에 한 줄이 건너뛴 후 이메일 본문이 이어집니다. 엔벌로프는 이메일 자체의 일부가 아니라 SMTP 세션 도중 클리언트와 메일 서버 사이에서 통신되므로 표시되지 않습니다.

```
Received: from abc.smtp-out.amazonses.com (123.45.67.89) by in.example.com
 (87.65.43.210); Fri, 17 Dec 2010 14:26:22
From: "Andrew" <andrew@example.com>;
To: "Bob" <bob@example.com>
Date: Fri, 17 Dec 2010 14:26:21 -0800
Subject: Hello
Message-ID: <61967230-7A45-4A9D-BEC9-87CBCF2211C9@example.com>
Accept-Language: en-US
Content-Language: en-US
Content-Type: text/plain; charset="us-ascii"
Content-Transfer-Encoding: quoted-printable
MIME-Version: 1.0

Hello, I hope you are having a good day.

-Andrew
```

다음 단원에서는 이메일 헤더 및 본문에 대해 살펴보고 Amazon SES를 사용할 때 제공해야 하는 정보를 알아봅니다.

이메일 헤더

이메일 메시지당 하나의 헤더가 있습니다. 헤더의 각 줄은 필드를 포함하며 필드와 필드 본문 사이는 콜론으로 구분됩니다. 이메일 클라이언트에서 이메일을 읽을 때, 통상적으로 이메일 클라이언트가 다음 헤더 필드의 값을 표시합니다.

- To - 메시지 수신자의 이메일 주소입니다.
- CC - 메시지의 참조 수신자의 이메일 주소입니다.
- From - 이메일이 전송된 이메일 주소입니다.
- Subject - 메시지 주제의 요약입니다.
- Date - 이메일이 전송된 날짜 및 시간입니다.

라우팅 정보를 제공하고 메시지 내용을 설명하는 많은 추가 헤더 필드가 있습니다. 보통의 경우 이메일 클라이언트는 이러한 필드를 사용자에게 표시하지 않습니다. Amazon SES에서 사용 가능한 헤더 필드의 전체 목록은 Amazon SES 헤더 필드 단원을 참조하십시오. Amazon SES를 사용할 때 특히 "From," "Reply-To," 및 "Return-Path" 헤더 필드의 차이를 이해해야 합니다. 앞서 언급한 대로, "From" 주소는 메시지 발신자의 이메일 주소이고, "Reply-To" 및 "Return-Path"의 정의는 다음과 같습니다.

- Reply-To - 회신을 전송할 이메일 주소입니다. 기본적으로 회신은 원래 발신자 이메일 주소로 전송됩니다.
- Return-Path - 반송 메일 및 수신 거부를 전송해야 할 이메일 주소입니다. "Return-Path"는 때때로 "envelope from", "envelope sender" 또는 "MAIL FROM"으로 불립니다.

> **Note**
>
> Amazon SES를 사용할 경우 항상 ‘Return-Path’ 파라미터를 반송 메일을 모니터링하고 반송 메일 발생 시 교정 조치를 취할 수 있도록 설정하는 것이 좋습니다.

반송된 메시지를 의도된 수신자와 간편하게 매칭하려면 VERP(Variable Envelope Return Path)를 사용할 수 있습니다. VERP에서는 각 수신자별로 다른 "Return-Path"를 설정합니다. 따라서 메시지가 반송될 경우 반송 메일 메시지를 열고 파싱할 필요 없이 자동으로 어느 수신자로부터 반송되었는지 알 수 있습니다.

이메일 본문

이메일 본문은 메시지의 텍스트를 포함합니다. 본문은 다음 형식으로 전송할 수 있습니다.

- HTML - 수신자의 이메일 클라이언트가 HTML을 해석할 수 있는 경우 본문이 서식 지정된 텍스트 및 하이퍼링크를 포함할 수 있습니다.
- 일반 텍스트 - 수신자의 이메일 클라이언트가 텍스트 기반일 경우 본문이 인쇄되지 않는 문자를 포함할 수 없습니다.
- HTML 및 일반 텍스트 모두 - 두 형식을 모두 사용하여 동일한 콘텐츠를 단일 메시지로 전송할 경우 수신자의 이메일 클라이언트가 기능에 따라 어느 형식을 표시할지 결정합니다.

다수의 수신자에게 이메일 메시지를 전송하는 경우 HTML 및 텍스트 모두 형식으로 이메일을 전송하는 것이 나을 수 있습니다. HTML 지원 이메일 클라이언트를 사용하는 수신자라면 메시지에서 포함된 하이퍼링크를 직접 클릭할 수 있습니다. 텍스트 기반 이메일 클라이언트를 사용하는 수신자를 위해서는 URL을 복사하여 웹 브라우저를 사용하여 열 수 있도록 URL을 포함시켜야 합니다.

Amazon SES로 제공해야 하는 이메일 정보

Amazon SES에서 이메일을 전송할 때 제공해야 하는 이메일 정보는 Amazon SES를 호출하는 방식에 따라 달라집니다. 사용자는 최소한의 정보만 제공하면 Amazon SES가 대신하여 형식 설정을 수행합니다. 또는, 첨부 파일 전송과 같은 고급 작업을 수행하려면 직접 원시 메시지를 제공할 수 있습니다. 다음

단원에서는 Amazon SES API, Amazon SES SMTP 인터페이스 또는 Amazon SES 콘솔을 사용하여 이메일을 전송할 때 무엇이 필요한지 알아봅니다.

Amazon SES API

직접 Amazon SES API를 호출하는 경우 `SendEmail` 또는 `SendRawEmail` API를 호출할 수 있습니다. 사용자가 제공해야 하는 정보는 어느 API를 호출하는가에 따라 다릅니다.

- `SendEmail API`을(를) 호출하는 경우 발신 주소, 수신 주소, 메시지 제목과 본문만 입력하면 됩니다. 선택 사항으로 "Reply-To" 주소를 제공할 수 있습니다. 이 API를 호출하면 Amazon SES가 이메일 클라이언트 소프트웨어를 통한 디스플레이에 최적화된 적절한 서식의 멀티파트 다목적 인터넷 전자 우편(MIME) 이메일 메시지를 자동으로 수집합니다. 자세한 내용은 Amazon SES API를 사용하여 서식이 지정된 이메일 보내기 단원을 참조하세요.
- `SendRawEmail` API는 헤더, MIME 파트, 콘텐츠 유형 등 원시 이메일 메시지의 형식을 사용자가 원하는 대로 지정하여 전송할 수 있습니다. `SendRawEmail`은(는) 대개 고급 사용자가 사용합니다. 메시지 본문과 인터넷 메시지 형식 사양(RFC 5322)에서 필수로 지정된 모든 헤더 필드를 제공해야 합니다. 자세한 내용은 Amazon SES API v2를 사용하여 원시 이메일 보내기 단원을 참조하십시오.

AWS SDK를 사용하여 Amazon SES API를 호출하는 경우 위에 나열된 정보를 해당 함수(예: Java용 `SendEmail` 및)`SendRawEmail`에 제공합니다.

Amazon SES API를 사용한 이메일 전송에 대한 자세한 내용은 Amazon SES API를 사용하여 이메일 보내기 단원을 참조하십시오.

Amazon SES SMTP 인터페이스

SMTP 인터페이스를 통해 Amazon SES에 액세스하는 경우 SMTP 클라이언트 애플리케이션이 메시지를 수집합니다. 따라서 제공해야 할 정보는 어떤 애플리케이션을 사용하는가에 따라 달라집니다. 클라이언트와 서버 간의 SMTP 교환에는 최소한 원본 주소, 대상 주소 및 메시지 데이터가 필요합니다.

Amazon SES SMTP 인터페이스를 사용한 이메일 전송에 대한 자세한 내용은 Amazon SES SMTP 인터페이스를 사용하여 이메일 보내기 단원을 참조하십시오.

Amazon SES 콘솔

Amazon SES 콘솔을 사용하여 이메일을 전송하는 경우 제공해야 할 정보는 서식 지정된 이메일 또는 원시 이메일을 전송하는지에 따라 달라집니다.

- 서식이 지정된 이메일을 전송하려면 원본 주소, 수신 주소, 메시지 제목과 본문을 입력해야 합니다. Amazon SES가 이메일 클라이언트 소프트웨어를 통한 디스플레이에 최적화된 적절한 서식의 멀티파트 MIME) 이메일 메시지를 자동으로 수집합니다. 회신 및 반송 경로 필드도 지정할 수 있습니다.
- 원시 이메일을 전송하려면 발신 주소, 수신 주소 및 메시지 내용을 입력해야 합니다. 메시지 내용은 메시지 본문과 인터넷 메시지 형식 사양(RFC 5322)에서 필수로 지정된 모든 헤더 필드를 포함해야 합니다.

Amazon SES를 통한 이메일 발송률 이해

발신자는 수신자가 이메일을 읽고, 이메일에서 가치를 발견하고, 이메일을 스팸으로 표시하지 않기를 원합니다. 즉, 이메일 발송률(수신자의 받은 편지함에 도착한 이메일의 비율)을 최대화하려 합니다. 이 주제에서는 Amazon SES를 사용할 때 숙지해야 할 이메일 발송률 개념에 대해 알아봅니다.

이메일 발송률을 극대화하기 위해서는 이메일 전송 문제를 이해하고, 이러한 문제를 방지하기 위해 사전에 조치를 취하고, 전송한 이메일의 상태를 추적하고, 필요한 경우 이메일 전송 프로그램을 개선하여 성공적인 전송의 가능성을 더 높여야 합니다. 다음 단원에서는 이러한 조치의 배경이 되는 개념과 Amazon SES가 프로세스에서 어떤 도움을 주는지 알아봅니다.

이메일 전송 문제의 이해

대부분의 경우, 메시지를 기대하는 수신자에게는 메시지가 성공적으로 전송됩니다. 하지만 전송이 실패하거나 수신자가 발송된 이메일을 수신하기를 원치 않는 경우도 간혹 있을 수 있습니다. 반송 메일, 수신 거부, 금지 목록이 이러한 전송 문제와 관련되며, 다음 섹션에서 이들 문제에 대해 설명합니다.

반송 메일

수신자의 수신기(예: 이메일 공급자)가 수신자에게 메시지를 전달하지 못하는 경우 수신기는 메시지를 Amazon SES로 반송합니다. 그러면 Amazon SES는 사용자가 시스템을 설정한 방식에 따라 이메일 또는 Amazon Simple Notification Service(Amazon SNS)를 통해 반송된 이메일을 알립니다. 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하세요.

반송 메일에는 하드 바운스와 소프트 바운스가 있으며, 정의는 다음과 같습니다.

- 하드 바운스 – 지속적인 이메일 전송 실패입니다. 예를 들어 메일박스가 존재하지 않습니다. Amazon SES는 하드 바운스를 재시도하지 않습니다(DNS 조회 실패는 예외). 하드 바운스가 발생한 이메일 주소는 전송 시도를 반복하지 않는 것이 좋습니다.
- 소프트 바운스 – 일시적인 이메일 전송 실패입니다. 예를 들어 메일박스가 가득 찼거나 연결이 너무 많거나(병목 현상이라고도 함) 연결이 시간 초과된 경우입니다. Amazon SES는 소프트 바운스를 여러 번 재시도합니다. 그래도 이메일을 전송할 수 없을 경우 Amazon SES가 재시도를 중지합니다.

Amazon SES는 사용자에게 더 이상 전송을 재시도하지 않을 소프트 바운스 및 하드 바운스에 대해 알립니다. 단, 하드 바운스만 Amazon SES 콘솔 또는 `GetSendStatistics` API를 사용하여 검색하는 반송 메일 발생률 및 반송 메일 측정치에 반영됩니다.

또한 반송 메일은 동기식 또는 비동기식일 수 있습니다. 동기식 반송 메일은 발신자 및 받는 사람의 이메일 서버가 능동적으로 통신하는 동안 발생합니다. 비동기식 반송 메일은 받는 사람이 처음에는 이메일 메시지를 수락했다가 나중에 수신자에게 전송하지 못하는 경우 발생합니다.

불만 제기

대부분의 이메일 클라이언트 프로그램은 "스팸으로 표시" 등으로 표시된 버튼을 제공합니다. 이 버튼을 누르면 메시지가 스팸 폴더로 이동한 후 이메일 공급자로 전달됩니다. 또한, 대부분의 이메일 공급자는 침해 주소(예: abuse@example.net)를 유지합니다. 이 주소를 통해 사용자는 이메일 공급자에게 원치 않는 이메일 메시지를 전달하고 이러한 메시지를 방지하는 작업을 할 것을 요청할 수 있습니다. 두 경우 모두, 수신자가 수신 거부를 제기하는 것입니다. 이메일 공급자가 발신자를 스패머라고 판단하고 Amazon SES가 해당 이메일 공급자와 피드백 루프를 설정한 경우, 이메일 공급자가 수신 거부를 Amazon SES로 전송합니다. 그러한 수신 거부를 수신한 Amazon SES는 사용자가 시스템을 설정한 방

식에 따라 이메일 또는 Amazon SNS 알림을 통해 사용자에게 수신 거부 알림을 전달합니다. 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하세요. 수신 거부가 발생한 이메일 주소는 전송 시도를 반복하지 않는 것이 좋습니다.

전역 금지 목록

SES 공유 IP 풀에서 주소의 평판을 보호하기 위해 SES가 소유하고 관리하는 Amazon SES 전역 금지 목록에는 최근에 모든 SES 고객에게 하드 바운스를 발생시킨 수신자 이메일 주소가 포함되어 있습니다. SES를 통해 금지 목록에 포함된 주소로 이메일을 전송할 경우 SES 호출은 성공하지만, SES가 해당 이메일을 전송하는 대신 하드 바운스로 취급합니다. 일반적인 반송과 마찬가지로 발송 금지 목록 반송은 발신 할당량과 반송률에 포함됩니다. 이메일 주소는 최대 14일까지 발송 금지 목록에 남아 있을 수 있습니다. 전송할 이메일 주소가 유효하다고 확신하는 경우 해당 주소가 계정 수준 금지 목록에 있지 않은지 확인하여 전역 금지 목록을 재정의할 수 있습니다. SES가 계속 전달을 시도하지만 반송되는 경우 반송 메일이 사용자의 평판에 영향을 미치지만, 고유한 계정 수준 금지 목록을 사용하지 않는 경우 해당 이메일 주소로 보낼 수 없기 때문에 아무도 반송을 받지 않습니다. 계정 수준 금지 목록에 대한 자세한 내용을 보려면 Amazon SES 계정 수준 금지 목록 사용 섹션을 참조하세요.

사전 예방

인터넷 상에서 가장 심각한 이메일 문제 중 하나가 원치 않는 대량 메일(스팸)입니다. 이메일 공급자는 고객이 스팸을 수신하지 않도록 광범위한 조치를 취하고 있습니다. 또한 Amazon SES는 이메일 제공업체가 귀하의 이메일을 스팸으로 간주할 가능성을 줄이기 위한 조치를 취합니다. Amazon SES는 확인, 인증, 전송 할당량 및 콘텐츠 필터링을 사용합니다. 또한 Amazon SES는 이메일 제공업체와 신뢰할 수 있는 평판을 유지하며 고품질 이메일을 보내야 합니다. Amazon SES는 이러한 작업 중 일부를 자동으로 처리합니다(예: 컨텐츠 필터링). 다른 경우에는 인증 등의 도구를 제공하거나 올바른 방향(할당량 전송)을 안내합니다. 다음 단원에서는 각 개념에 대한 자세한 내용을 제공합니다.

확인

불행히도 스패머가 이메일을 다른 소스로부터 발송된 것처럼 보이기 위해 이메일 헤더를 조작하고 발신 이메일 주소를 스푸핑하는 것이 가능합니다. 이메일 공급자와 Amazon SES 사이에서 신뢰를 유지하기 위해, Amazon SES는 발신자가 사용자 본인이라는 것을 보증해야 합니다. 그러므로 사용자는 Amazon SES를 통해 이메일을 전송하는 모든 이메일 주소를 확인하여 전송 자격 증명을 보호해야 합니다. Amazon SES 콘솔을 사용하거나 Amazon SES API를 사용하여 이메일 주소를 확인할 수 있습니다. 전체 도메인을 확인할 수도 있습니다. 자세한 내용은 이메일 주소 자격 증명 생성 및 도메인 자격 증명 생성 섹션을 참조하세요.

계정이 아직 Amazon SES 샌드박스에 있는 경우, Amazon SES 메일박스 시뮬레이터에서 제공하는 주소를 제외한 모든 수신자 이메일 주소 또한 확인해야 합니다. 샌드박스 해제에 대한 자세한 내용은 프

로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하세요. 메일박스 시뮬레이터에 대한 자세한 내용은 수동으로 메일박스 시뮬레이터 사용 단원을 참조하세요.

인증

인증은 이메일 공급자에게 발신자가 사용자 본인이라는 것을 표시하는 또 하나의 방법입니다. 이메일을 인증하면 발신자가 계정의 소유자이고 발신자가 보낸 이메일이 전송 중에 수정되지 않았다는 증거를 제공하는 것입니다. 경우에 따라 이메일 공급자는 인증되지 않은 이메일의 전달을 거부합니다. Amazon SES는 두 가지 인증 방법, 즉 발신자 정책 프레임워크(SPF) 및 도메인키 식별 메일(DKIM)을 지원합니다. 자세한 내용은 Amazon SES의 자격 증명 구성 단원을 참조하십시오.

전송 할당량

예상치 못한 갑작스러운 이메일 볼륨 또는 속도 급증이 감지되면 이메일 공급자는 발신자를 스패머로 의심하여 이메일을 차단할 수 있습니다. 따라서 모든 Amazon SES 계정에는 발신 할당량이 있습니다. 이러한 할당량은 24시간 동안 보낼 수 있는 이메일 수와 초당 보낼 수 있는 수를 제한합니다. 이러한 발신 할당량은 이메일 공급자와의 신뢰를 보호하는 데 도움이 됩니다.

대부분의 경우 신규 사용자에게는 Amazon SES가 매일 소량의 이메일을 전송하도록 허용합니다. 사용자가 전송하는 메일이 이메일 공급자가 허용 가능한 수준이라면 이 할당량이 자동으로 증가합니다. 사용자가 더 많은 이메일을 더 빠른 속도로 전송할 수 있도록 발신 할당량이 꾸준히 증가합니다. SES 전송 제한 증가 사례를 생성하여 추가 할당량 증가를 요청할 수도 있습니다.

발신 할당량 및 할당량을 높이는 방법에 대한 자세한 내용은 Amazon SES 발신 한도 관리 단원을 참조하세요.

콘텐츠 필터링

많은 이메일 공급자가 콘텐츠 필터링을 사용하여 수신 이메일이 스팸인지 여부를 결정합니다. 콘텐츠 필터는 의심스러운 콘텐츠를 검색하여 이메일이 스팸의 프로필에 해당하는 경우 해당 이메일을 차단합니다. Amazon SES는 콘텐츠 필터도 사용합니다. 애플리케이션이 Amazon SES로 요청을 전송하면 Amazon SES가 사용자를 대신해 이메일 메시지를 수집한 후 메시지 헤더 및 본문을 스캔하여 이메일 공급자가 스팸으로 간주할 수 있는 콘텐츠가 포함되었는지 판단합니다. 사용자의 메시지가 Amazon SES의 콘텐츠 필터에 스팸처럼 보일 경우 Amazon SES에서의 사용자 평판에 악영향을 주게 됩니다.

또한 Amazon SES는 모든 메시지에서 바이러스를 검사합니다. 메시지에 바이러스가 포함되는 경우 Amazon SES는 해당 메시지를 수신자의 메일 서버로 전송하지 않습니다.

신뢰도

이 이메일 전송과 관련하여, 평판(IP 주소, 이메일 주소 또는 전송 도메인이 스팸의 출처가 아니라는 확신을 나타내는 척도)이 중요합니다. Amazon SES는 수신자의 받은 편지함으로 이메일을 전송할 수

있도록 이메일 공급자에 대한 강력한 평판을 유지하고 있습니다. 마찬가지로, 사용자는 Amazon SES에 대해 높은 평판을 유지해야 합니다. 사용자는 품질이 높은 콘텐츠를 전송함으로써 Amazon SES에서 평판을 쌓을 수 있습니다. 사용자가 품질이 높은 콘텐츠를 전송하면 장기적으로 평판이 상승하여 Amazon SES가 사용자의 발신 할당량을 늘립니다. 과도한 반송 메일 및 수신 거부는 사용자의 평판에 악영향을 미치며 Amazon SES가 계정의 발신 할당량을 낮추거나 사용자의 Amazon SES 계정이 종료되는 원인이 될 수 있습니다.

평판을 유지하는 한 방법은 시스템을 테스트할 때 사용자가 직접 생성한 이메일 주소로 이메일을 전송하는 대신 사서함 시뮬레이터를 사용하는 것입니다. 메일박스 시뮬레이터로 보낸 이메일은 반송 메일 및 수신 거부 지표에 영향을 미치지 않습니다. 메일박스 시뮬레이터에 대한 자세한 내용은 수동으로 메일박스 시뮬레이터 사용 단원을 참조하세요.

품질이 높은 이메일

품질이 높은 이메일은 수신자가 가치를 발견하고 수신을 희망하는 이메일입니다. 가치는 수신자마다 다른 것을 의미하며 제안, 주문 확인, 영수증, 뉴스레터 등의 형태를 띨 수 있습니다. 궁극적으로, 사용자의 발송률은 사용자가 전송하는 이메일의 품질에 달려 있습니다. 이메일 공급자는 낮은 품질로 간주되는 이메일을 차단하기 때문입니다.

최신 정보 파악

이메일 전송이 실패했는지, 수신자가 이메일에 대해 수신 거부를 제기했는지, Amazon SES가 성공적으로 이메일을 수신자의 메일 서버로 전달했는지 여부에 대해 Amazon SES는 알림을 제공하고 간편한 사용량 통계치 모니터링을 제공하여 사용자가 문제를 추적할 수 있게 지원합니다.

알림

이메일이 반송되면 이메일 공급자는 Amazon SES에 알리고 Amazon SES는 사용자에게 알립니다. Amazon SES는 사용자에게 Amazon SES가 더 이상 전송을 재시도하지 않을 소프트 바운스 및 하드 바운스에 대해 알립니다. 많은 이메일 공급자가 수신 거부를 전달하며, Amazon SES가 주요 이메일 공급자에게 수신 거부 피드백 루프를 설정하므로 사용자가 할 필요가 없습니다. Amazon SES는 사용자에게 반송 메일, 수신 거부 및 전송 성공을 알릴 수 있는 두 가지 방법, 즉 Amazon SNS를 통해 알림을 수신하도록 계정을 설정하거나 이메일(반송 메일과 수신 거부만 해당)을 통해 알림을 수신할 수 있습니다. 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하세요.

사용량 통계

Amazon SES는 사용자가 실패한 전송을 통해 근본 원인을 파악하고 해결할 수 있도록 사용량 통계치를 제공합니다. 사용량 통계치는 Amazon SES 콘솔을 사용하거나 Amazon SES API를 호출하여 확인

할 수 있습니다. 전송, 반송 메일, 수신 거부 및 바이러스 감염 거부 이메일 수를 확인할 수 있고, 또한 발신 할당량을 초과하지 않도록 발신 할당량을 확인할 수 있습니다.

이메일 전송 프로그램 개선

반송 메일 및 수신 거부 수가 증가하고 있다면 이메일 전송 전략을 재평가할 때입니다. 과도한 반송 메일, 수신 거부 및 저품질 이메일을 보내려는 시도는 악용을 구성하고를 종료 위험에 빠뜨린다 AWS 계정 는 점을 기억하세요. 궁극적으로, 사용자는 Amazon SES를 사용하여 중요 이메일을 전송하고 수신자가 수신을 희망하는 이메일만 전송해야 합니다.

최소 1회 전송

Amazon SES는 중복성과 고가용성을 위해 여러 대의 서버에 메시지 사본을 저장합니다. 드물게는 메시지 사본을 받거나 삭제할 때 메시지 사본을 저장하는 서버 중 하나를 사용할 수 없을 수도 있습니다.

이 문제가 발생할 경우 사용 불가능한 해당 서버에서 메시지의 사본이 삭지되지 않으며, 메시지를 받을 때 해당 메시지 사본을 다시 가져올 수 있습니다. 따라서 애플리케이션이 idempotent가 되도록 설계해야 합니다(다시 말해 동일한 메시지를 두 번 이상 처리할 경우 부정적인 영향을 받지 않아야 함).

Amazon SES를 사용한 이메일 전송 모범 사례

이메일을 사용한 고객 커뮤니케이션을 관리하는 방법을 이메일 프로그램이라고 합니다. 이러한 이메일 프로그램은 성패를 결정 짓는 요인이 몇 가지 있지만 처음에는 혼란스럽거나 이해가 되지 않을 수도 있습니다. 하지만 이메일 전송 방식에 대해 이해하고 몇 가지 모범 사례를 따르다 보면 이메일이 고객의 메일 수신함까지 이르는 가능성을 높일 수 있습니다.

주제

- 이메일 프로그램을 위한 성공 지표
- 긍정적인 발신자 평판 유지

이메일 프로그램을 위한 성공 지표

이메일 프로그램의 성공 측정에 도움이 되는 여러 지표가 있습니다.

이번 단원에서는 다음 주제에 대한 정보에 대해서 살펴봅니다.

- 반송 메일
- 수신 거부
- 메시지 품질

반송 메일

반송은 이메일이 원하는 수신자에게 전송되지 않았을 때 발생합니다. 반송 메일에는 하드 바운스 및 소프트 바운스라는 두 가지 유형이 있습니다. 하드 바운스는 이메일 주소가 존재하지 않는 것과 같은 영구적인 문제로 인해 이메일이 전송되지 않을 때 발생합니다. 소프트 바운스는 일시적인 문제로 인해 이메일을 전송할 수 없을 때 발생합니다. 소프트 바운스는 수신자의 받은 편지함이 가득 찼거나 수신 서버가 일시적으로 중단되었을 때 발생할 수 있습니다. Amazon SES는 일정 기간 동안 소프트 바운스 이메일을 다시 전송하려고 시도합니다.

사용자는 이메일 프로그램에서 하드 바운스의 수를 모니터링하고 수신자 목록에서 하드 바운스를 일으키는 이메일 주소를 제거해야 합니다. 이메일 수신기가 높은 하드 바운스 발생률을 감지하면 사용자가 수신자에 대해서 잘 모른다고 가정합니다. 결과적으로 하드 바운스 발생률이 높으면 이메일 메시지의 발송률에 부정적인 영향을 미칠 수 있습니다.

다음은 반송 메일을 방지하여 발신자 평판을 높이는 데 도움이 될 수 있는 지침입니다.

- 하드 바운스 발생률을 5% 미만으로 유지하세요. 이메일 프로그램의 하드 바운스 수가 적을수록 ISP가 메시지의 적합성과 가치를 인정할 가능성이 높습니다. 이러한 비율은 합리적이고 달성 가능한 목표로 간주해야 하지만 모든 ISP에게 동일하게 적용되는 규칙은 아닙니다.
- 이메일 목록은 빌리거나 구매하지 마세요. 이메일 목록에는 잘못된 주소가 상당수 포함되어 하드 바운스 발생률이 크게 높아지는 원인이 될 수 있습니다. 또한 이러한 목록에는 스팸 트랩(불법 발신자를 포착하는 데 특별히 사용되는 이메일 주소)이 포함될 수 있습니다. 메시지가 이러한 스팸 트랩으로 전송되면 전송 완료율과 발신자 평판은 돌이킬 수 없을 정도로 손상될 수 있습니다.
- 목록을 최신 상태로 유지하세요. 일부 수신자에게 장기간 이메일을 보내지 않았다면 몇 가지 수단(웹사이트 로그인 활동, 구매 이력 등)을 통해 고객 상태의 유효성을 검증하는 것이 좋습니다.
- 고객 상태를 검증할 방법이 없다면 윈백(win-back) 이메일을 전송하는 것도 좋은 방법입니다. 윈백 이메일이란 오랫동안 소식을 듣지 못한 고객이 여전히 이메일 수신에 동의하는지 확인하는 것을 말합니다. 윈백 이메일을 전송한 후에도 응답이 없는 수신자는 모두 목록에서 제거하세요.

반송 메일을 받으면 다음 규칙을 준수하여 적절하게 대응해야 합니다.

- 하드 바운스를 일으키는 이메일 주소는 즉시 목록에서 제거하세요. 또한 하드 바운스를 일으킨 주소로 메시지를 재전송하지 마세요. 하드 바운스가 반복되면서 누적되면 결국 수신자의 ISP가 판단하는 평판이 떨어지게 됩니다.
- 반송 메일 알림을 받는 데 사용하는 주소가 이메일을 수신할 수 있는지 확인합니다. 반송 메일 및 수신 거부 알림 설정에 대한 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하세요.

- 인바운드 이메일이 자체 내부 서버가 아닌 ISP를 통해서 수신되는 경우에는 반송 메일 알림이 스팸 폴더로 도착하거나, 혹은 완전히 삭제될 수도 있습니다. 따라서 반송 메일을 수신할 때는 호스팅 이메일 주소를 사용하지 않는 것이 바람직합니다. 하지만 호스팅 이메일 주소를 사용해야 한다면 자주 스팸 폴더를 확인하여 반송 메일 메시지가 스팸으로 표시되지 않도록 하세요. Amazon SES에서는 반송 메일 알림이 전송되는 주소를 지정할 수 있습니다.
- 일반적으로 반송 메일에는 전송을 거부하는 메일 수신함의 주소가 포함되어 있습니다. 하지만 수신자 주소를 특정 이메일 캠페인과 연계할 수 있는 세부 데이터가 추가로 필요하다면 X-헤더에 내부 추적 시스템까지 밝혀낼 수 있는 값을 추가하세요. 자세한 내용은 Amazon SES 헤더 필드 단원을 참조하십시오.

수신 거부

수신 거부는 수신자가 자신의 웹 기반 이메일 클라이언트에서 "Mark as Spam" 또는 이와 비슷한 버튼을 클릭할 때 발생합니다. 이러한 수신 거부가 상당수 누적되면 ISP가 스팸을 전송하는 것으로 간주하여 발송률과 발신자 평판에 부정적인 영향을 미칠 수 있습니다. 전부는 아니지만 일부 ISP는 수신 거부가 보고되면 알림을 전송합니다. 이를 피드백 루프라고 합니다. Amazon SES는 피드백 루프를 제공하는 ISP로부터 수신 거부를 자동으로 전달합니다.

다음은 수신 거부를 방지하여 발신자 평판을 높이는 데 도움이 될 수 있는 지침입니다.

- 수신 거부 발생률을 0.1% 미만으로 유지하세요. 이메일 프로그램의 수신 거부 수가 적을수록 ISP가 메시지의 적합성과 가치를 인정할 가능성이 높습니다. 이러한 비율은 합리적이고 달성 가능한 목표로 간주해야 하지만 모든 ISP에게 동일하게 적용되는 규칙은 아닙니다.
- 고객이 마케팅 이메일에 수신 거부하는 경우 해당 고객에게 더 이상 마케팅 이메일을 보내서는 안 됩니다. 하지만 이메일 프로그램에 다른 유형의 이메일, 예를 들어 알림 메시지나 거래 이메일 등이 포함되는 경우에는 수신 거부를 선택한 수신자라고 해도 이러한 유형의 메시지는 계속해서 보낼 수 있습니다.
- 하드 바운스와 마찬가지로 오랫동안 이메일을 전송하지 않은 목록이 있다면 수신자가 메시지를 받는 이유에 대해서 잘 알 수 있도록 해야 합니다. 이러한 경우에는 인사말 메시지를 보내서 본인이 누구이고, 왜 이메일을 보내는지 알려주는 것이 좋습니다.

수신 거부 메일을 받으면 다음 규칙을 준수하여 적절하게 대응해야 합니다.

- 수신 거부 알림을 받는 데 사용하는 주소가 이메일을 수신할 수 있는지 확인합니다. 반송 메일 및 수신 거부 알림 설정에 대한 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하세요.
- 수신 거부 알림이 ISP 또는 메일 시스템에서 스팸으로 표시되지 않도록 하세요.

- 수신 거부 알림에는 일반적으로 이메일 본문이 포함되므로, 이메일 헤더만 포함되는 반송 메일 알림과는 다릅니다. 하지만 수신 거부 알림에서는 수신을 거부한 개인의 이메일 주소가 제거됩니다. 수신을 거부한 이메일 주소를 식별할 수 있도록 이메일 본문에 포함된 사용자 지정 X-헤더나 특수 식별자를 사용하세요. 이 방법을 사용하면 수신을 거부한 이메일을 쉽게 식별하여 해당 이메일을 수신자 목록에서 제거할 수 있습니다.

메시지 품질

이메일 수신기는 콘텐츠 필터를 사용하여 메시지의 속성을 감지함으로써 메시지의 적합성 여부를 식별합니다. 이러한 콘텐츠 필터는 메시지 내용을 자동으로 검토하여 불필요하거나 악의적인 메시지의 공통 특성을 찾아냅니다. Amazon SES는 콘텐츠 필터링 기술을 사용하여 맬웨어가 포함된 메시지를 감지하여 발신 전에 미리 차단합니다.

이메일 수신기의 콘텐츠 필터가 메시지에 스팸 또는 악성 이메일의 특성이 포함되어 있다고 판단할 경우 해당 메시지는 플래그 처리되어 수신자의 메일 수신함이 아닌 다른 곳으로 전송될 가능성이 높습니다.

이메일을 설계할 때는 다음 사항에 주의하세요.

- 최신 콘텐츠 필터는 지능적이어서 상황에 따라 적응하며 계속해서 바뀝니다. 사전 정의된 규칙에 의존하지도 않습니다. ReturnPath 또는 Litmus와 같은 타사 서비스는 이메일에서 콘텐츠 필터를 트리거하는 내용을 식별하는 데 도움이 될 수 있습니다.
- 이메일에 링크가 포함되어 있으면 URIBL.com이나 SURBL.org 등의 DNS 기반 블랙홀 목록(DNSBL)과 대조하여 해당 링크 URL의 유무를 확인하십시오.
- 링크 단축 서비스는 사용하지 마세요. 악의적인 발신자는 링크 단축 서비스를 사용하여 실제 링크 목적지를 숨기기도 합니다. ISP는 링크 단축 서비스(심지어 가장 평판이 좋은 서비스조차도)가 악의적인 목적으로 사용되고 있음을 알게 되면 해당 서비스에 대한 액세스를 모두 거부할 수 있습니다. 이메일에 거부 목록에 등록된 링크 단축 서비스에 대한 링크가 포함되어 있는 경우 고객의 메일 수신함까지 이르지 못하기 때문에 성공적인 이메일 캠페인도 어렵게 됩니다.
- 이메일의 링크가 모두 원하는 페이지를 가리키고 있는지 테스트하세요.
- 웹사이트에는 개인정보 보호정책 및 이용 약관 문서가 포함되어야 하는 동시에 이러한 문서들이 최신 상태를 유지해야 합니다. 전송하는 이메일마다 이러한 문서에 대한 링크를 추가하는 것도 좋은 방법입니다. 이러한 문서 링크를 제공하면 고객에게 숨길 것도 없다는 것을 보여주면서 신뢰 관계를 형성하는 데 도움이 될 수 있습니다.
- 잦은 빈도 수의 콘텐츠("일별 거래" 메시지 등)를 전송할 계획이라면 배포할 때마다 이메일 내용이 달라야 합니다. 잦은 횟수로 메시지를 보낼 때는 각 메시지가 일정한 시간에 적합성을 유지해야만 반복적이고 귀찮게 느껴지지 않습니다.

긍정적인 발신자 평판 유지

Amazon SES에서 발신자 평판은 이메일 공급자 및 스팸 필터가 인식하는 이메일 발신자의 진실성과 신뢰성을 나타냅니다. 이는 이메일이 합법적인 것으로 간주되어 수신자의 받은 편지함에 성공적으로 전달될 가능성을 측정하는 기준입니다.

다음 섹션에서는 발신자의 평판을 양호하게 유지하면서 이메일 커뮤니케이션이 의도한 대상에 도달하도록 하려면 염두에 두어야 하는 핵심 이메일 전송 보안 주체를 소개합니다.

도메인 및 "From(발신)" 주소 주의사항

- 이메일을 보내는 주소에 대해서 주의 깊게 생각하세요. "발신" 주소는 수신자가 가장 먼저 보게 되는 정보 중 하나이기 때문에 마지막까지 첫 인상으로 남을 수 있습니다. 또한 일부 ISP는 평판을 "발신" 주소와 연계하기도 합니다.
- 커뮤니케이션 유형에 따라 하위 도메인을 사용하는 것도 좋은 방법입니다. 예를 들어 도메인 example.com에서 마케팅 메시지와 거래 메시지를 모두 이메일로 전송할 계획이라고 가정하겠습니다. 이때는 두 메시지를 모두 example.com에서 전송하기 보다는 오히려 마케팅 메시지는 marketing.example.com에서, 그리고 거래 메시지는 orders.example.com 같은 하위 도메인에서 보내는 것이 좋습니다. 고유성을 지닌 하위 도메인이라면 평판을 높일 수 있습니다. 또한 예를 들어 마케팅 커뮤니케이션이 스팸 트랩으로 수신되거나 콘텐츠 필터를 트리거하는 경우에도 평판을 떨어뜨릴 위험을 완화하는 효과가 있습니다.
- 다수의 메시지를 보낼 계획이라면 sender@hotmail.com 같은 ISP 기반 주소에서 메시지를 보내지 마세요. ISP가 sender@hotmail.com에서 대량의 메시지가 전송되는 것을 알아차리면 해당 이메일이 본인 소유의 아웃바운드 이메일 전송 도메인에서 발송되는 이메일과 다르게 처리됩니다.
- 도메인의 WHOIS 정보가 정확할 수 있도록 도메인 등록 기관과 정보를 공유하세요. WHOIS 레코드를 거짓 없이 최신 상태로 유지할 경우 투명성을 중요하게 생각한다는 사실을 잘 드러낼 뿐만 아니라 사용자들은 도메인의 적합성 여부를 빠르게 판단할 수 있습니다.
- no-reply@example.com과 같은 발신 전용(no-reply) 주소를 '발신' 또는 '회신' 주소로 사용하지 마십시오. no-reply@ 이메일 주소를 사용하면 수신자가 따로 연락할 방법은 없으며, 보내주는 피드백에도 관심이 없다는 메시지를 분명하게 보여주는 셈입니다.

인증

- SPF 및 SenderID를 사용하여 도메인을 인증합니다. 이러한 인증 방법은 전송하는 이메일 하나하나가 실제로 명시된 도메인에서 발송되었다는 사실을 이메일 수신자에게 입증하는 역할을 합니다.
- DKIM을 사용하여 발신 메일에 서명합니다. 이 단계를 통해 발신자로부터 수신자에게 전송되는 도중 변경된 내용이 없다는 사실을 수신자에게 보장할 수 있습니다.

- 개인용 Gmail이나 Hotmail 계정 같은 본인 소유의 ISP 기반 이메일 주소로 이메일을 보낸 다음 메시지 헤더를 확인하면서 SPF 인증 설정과 DKIM 인증 설정을 테스트할 수 있습니다. 헤더는 메시지 인증 및 서명의 성공 여부를 나타냅니다.

목록 작성 및 유지

- 더블 옵트인 전략을 사용하세요. 사용자가 이메일 수신에 동의할 때는 확인 링크가 포함된 메시지를 보내세요. 그리고 사용자가 해당 링크를 클릭하여 주소를 확인할 때까지는 이메일을 보내지 마세요. 더블 옵트인 전략은 오타로 인한 하드 바운스 수를 줄이는 데 효과적입니다.
- 웹 기반 형식으로 이메일 주소를 수집할 때는 제출 직후 실시하는 주소의 유효성 검증을 최소화하세요. 예를 들어 수집하는 주소가 올바른 형식인지(즉, recipient@example.com 형식 여부), 그리고 참조 도메인이 유효한 MX 레코드로 구성되어 있는지 확인하면 됩니다.
- 사용자 정의 입력 데이터가 확인되지 않은 채로 Amazon SES로 전달될 때는 주의하세요. 포럼 등록이나 포럼 제출은 모두 사용자가 작성한 콘텐츠일 뿐만 아니라 스패머가 자신이 원하는 내용으로 포럼을 작성할 수도 있기 때문에 항상 고유 위험이 존재합니다. 고품질의 콘텐츠가 포함된 이메일만 보내도록 확인하는 것은 사용자의 책임입니다.
- 표준 별칭(postmaster@, abuse@, noc@ 등)은 의도적으로 이메일에 가입할 가능성이 매우 낮습니다. 메시지는 실제로 수신을 원하는 사람들에게만 보내야 합니다. 이러한 규칙은 특히 관례상 이메일 워치독 전용으로 사용되는 표준 별칭에서 더욱 그렇습니다. 이러한 별칭은 고의적 방해 행위로서 평판을 떨어뜨릴 목적으로 목록에 악의적으로 추가되기도 합니다.

규정 준수

- 이메일을 전송하는 대상 국가와 리전의 이메일 마케팅 및 스팸 방지 법률과 규정에 대해서 잘 알고 있어야 합니다. 사용자는 전송하는 이메일이 이러한 법률을 준수하는지 확인할 책임이 있습니다. 본 안내서에서는 이러한 법률을 다루지 않으므로 사용자가 직접 조사해야 합니다. 법률 목록은 Wikipedia의 Email Spam Legislation by Country를 참조하세요.
- 항상 변호사에게 문의하여 법률 자문을 받으세요.

AWS SDK에서 Amazon SES 사용

AWS 소프트웨어 개발 키트(SDKs)는 널리 사용되는 많은 프로그래밍 언어에 사용할 수 있습니다. 각 SDK는 개발자가 선호하는 언어로 애플리케이션을 쉽게 구축할 수 있도록 하는 API, 코드 예제 및 설명서를 제공합니다.

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

Amazon SES 관련 예는 AWS SDKs를 사용하는 Amazon SES의 코드 예제 섹션을 참조하세요.

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

Amazon Simple Email Service 시작하기

이 장에서는 Amazon SES의 초기 설정에 필요한 작업과 시작하는 데 도움이 되는 자습서를 설명합니다.

주제

- Amazon Simple Email Service 설정
- 다른 이메일 전송 솔루션에서 Amazon SES로 마이그레이션
- 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동)

Amazon Simple Email Service 설정

Amazon SES를 사용하기 전에, 먼저 다음 작업을 완료합니다.

업무

- 에 가입 AWS
- SES 계정 설정
- 프로그래밍 방식 액세스 권한 부여(콘솔 외부에서 SES와 상호 작용하기 위한 목적)
- AWS SDK 다운로드(SES APIs 사용용)

에 가입 AWS

이 없는 경우 다음 단계를 AWS 계정완료하여 생성합니다.

에 가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 엽니다.
2. 온라인 지시 사항을 따릅니다.

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

   에 가입하면 AWS 계정AWS 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.

SES 계정 설정

SES를 통해 이메일을 보내고 SES 계정 설정 마법사를 사용하여 계정에 대한 프로덕션 액세스를 요청할 수 있도록 이메일 주소와 전송 도메인을 확인하여 SES를 시작하세요.

SES 계정 설정 마법사를 사용하여 계정 설정

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. SES 콘솔 홈 페이지에서 시작하기를 선택하면 마법사가 SES 계정 설정 단계를 안내합니다.

SES 계정 설정 마법사는 SES에서 ID(이메일 주소 또는 도메인)를 아직 만들지 않은 경우에만 표시됩니다.

프로그래밍 방식 액세스 권한 부여(콘솔 외부에서 SES와 상호 작용하기 위한 목적)

사용자는 AWS 외부에서와 상호 작용하려는 경우 프로그래밍 방식으로 액세스해야 합니다 AWS Management Console. 프로그래밍 방식 액세스를 부여하는 방법은에 액세스하는 사용자 유형에 따라 다릅니다 AWS.

사용자에게 프로그래밍 방식 액세스 권한을 부여하려면 다음 옵션 중 하나를 선택합니다.

프로그래밍 방식 액세스가 필요한 사용자는 누구인가요?	To	액세스 권한을 부여하는 사용자
작업 인력 ID (IAM Identity Center가 관리하는 사용자)	임시 자격 증명을 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • 자세한 AWS CLI내용은 AWS Command Line Interface 사용 설명서의 AWS CLI 를 사용하도록 구성을 AWS IAM Identity Center 참조하세요. • AWS SDKs, 도구 및 AWS APIs의 경우 SDK 및 도구

프로그래밍 방식 액세스가 필요한 사용자는 누구인가요?	To	액세스 권한을 부여하는 사용자
		참조 안내서의 IAM Identity Center 인증을 참조하세요. AWS SDKs
IAM	임시 자격 증명을 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	IAM 사용 설명서의 AWS 리소스에서 임시 자격 증명 사용의 지침을 따릅니다.
IAM	(권장되지 않음) 장기 자격 증명을 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • 자세한 AWS CLI내용은 AWS Command Line Interface 사용 설명서의 IAM 사용자 자격 증명을 사용하여 인증을 참조하세요. • AWS SDKs 및 도구의 경우 SDK 및 도구 참조 안내서의 장기 자격 증명을 사용하여 인증을 참조하세요. AWS SDKs • AWS APIs 경우 IAM 사용 설명서의 IAM 사용자의 액세스 키 관리를 참조하세요.

AWS SDK 다운로드(SES APIs 사용용)

원시 HTTP 요청 수집과 같은 하위 수준 세부 정보를 처리할 필요 없이 SES APIs를 호출하려면 AWS SDK를 사용하면 됩니다. AWS SDKs는 SES 및 기타 AWS 서비스의 기능을 캡슐화하는 함수 및 데이터 형식을 제공합니다. AWS SDK를 다운로드하려면 SDKs로 이동합니다. SDK를 다운로드한 후 공유 자격 증명 파일을 생성하고 AWS 액세스 키를 지정합니다.

다른 이메일 전송 솔루션에서 Amazon SES로 마이그레이션

이 주제에서는 온프레미스에서 호스팅하는 솔루션 또는 Amazon EC2 인스턴스에서 호스팅하는 솔루션에서 Amazon SES로 이메일 전송 솔루션을 이동하려는 경우 수행해야 하는 단계에 대한 개요를 다룹니다.

이 섹션의 주제:

- 1단계. 도메인 확인
- 2단계. 프로덕션 액세스 권한 요청
- 3단계. 도메인 인증 시스템 구성
- 4단계. SMTP 자격 증명 생성
- 5단계. SMTP 엔드포인트에 연결
- 다음 단계

1단계. 도메인 확인

Amazon SES를 사용하여 이메일을 전송하기 전에 이메일을 보내려는 ID를 확인해야 합니다. Amazon SES에서 ID는 이메일 주소 또는 전체 도메인이 될 수 있습니다. 도메인을 확인하면 Amazon SES를 사용하여 해당 도메인의 어떤 주소에서든 이메일을 보낼 수 있습니다. 도메인 확인에 대한 자세한 내용은 도메인 자격 증명 생성 섹션을 참조하세요.

2단계. 프로덕션 액세스 권한 요청

Amazon SES를 처음 사용하기 시작하면 계정이 샌드박스 환경에 있습니다. 계정이 샌드박스에 있으면 확인된 주소에만 이메일을 전송할 수 있습니다. 또한 일별 전송할 수 있는 메시지 수와 초당 전송할 수 있는 수에 제한이 있습니다. 프로덕션 액세스 권한 요청에 대한 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 섹션을 참조하세요.

3단계. 도메인 인증 시스템 구성

DKIM 및 SPF와 같은 인증 시스템을 사용하도록 도메인을 구성할 수 있습니다. 이 단계는 기술적인 선택 사항입니다. 도메인에 DKIM 또는 SPF(또는 둘 모두)를 설정하면 이메일 발송률을 개선하고 고객의 평판을 높일 수 있습니다. SPF 설정에 대한 자세한 내용은 Amazon SES에서 SPF를 사용하여 이메일 인증 섹션을 참조하세요. DKIM 설정에 대한 자세한 내용은 Amazon SES에서 DKIM을 사용하여 이메일 인증 섹션을 참조하세요.

4단계. SMTP 자격 증명 생성

SMTP를 사용하는 애플리케이션을 통해 이메일을 보내려는 경우 SMTP 자격 증명을 생성해야 합니다. SMTP 자격 증명은 일반 AWS 자격 증명과 다릅니다. 이러한 자격 증명은 각 AWS 리전에서 고유합니다. SMTP 자격 증명을 생성하는 방법에 대한 자세한 내용은 Amazon SES SMTP 자격 증명 획득 섹션을 참조하세요.

5단계. SMTP 엔드포인트에 연결

postfix 또는 sendmail과 같은 메시지 전송 에이전트를 사용하는 경우 Amazon SES SMTP 엔드포인트를 참조하도록 해당 애플리케이션의 구성을 업데이트해야 합니다. SMTP 엔드포인트의 전체 목록은 Amazon SES SMTP 엔드포인트에 연결 섹션을 참조하세요. 이전 단계에서 생성한 SMTP 자격 증명은 특정 AWS 리전과 연결됩니다. SMTP 자격 증명을 만든 리전에 있는 SMTP 엔드포인트에 연결해야 합니다.

다음 단계

이제 Amazon SES를 사용하여 이메일 전송을 시작할 준비가 되었습니다. 수행할 수 있는 몇 가지 선택적 단계가 있습니다.

- 보내는 이메일에 적용되는 규칙 세트인 구성 세트를 생성할 수 있습니다. 예를 들어 이메일을 전송할 때, 수신자가 메시지를 열거나 메시지에 있는 링크를 클릭할 때, 이메일을 반송할 때, 그리고 수신자가 이메일을 스팸으로 표시할 때 구성 세트를 사용하여 알림을 보낼 위치를 지정할 수 있습니다. 자세한 내용은 Amazon SES에서 구성 세트 사용 단원을 참조하세요.
- Amazon SES를 통해 이메일을 전송할 때 계정에 대해 반송 및 수신 거부를 모니터링하는 것이 중요합니다. Amazon SES에는 계정에 대한 반송 및 수신 거부를 계속 추적하는 데 사용할 수 있는 평판 지표 콘솔 페이지가 포함되어 있습니다. 자세한 내용은 평판 지표를 사용하여 반송 메일 및 수신 거부률 추적 단원을 참조하십시오. 또한 이러한 요금이 너무 높은 경우 사용자에게 알려주는 CloudWatch 경보를 만들 수도 있습니다. CloudWatch 경보 만들기에 대한 자세한 내용은 CloudWatch를 사용하여 평판 모니터링 경보 생성 섹션을 참조하세요.
- 대량의 이메일을 보내는 고객 또는 단순히 IP 주소의 평판을 완벽하게 제어하려는 고객은 추가 월별 요금을 지불하고 전용 IP 주소를 임대할 수 있습니다. 자세한 내용은 Amazon SES를 위한 전용 IP 주소 단원을 참조하십시오.

프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동)

도용 및 침해를 방지하고 발신자의 평판을 보호하기 위해 신규 Amazon SES 계정에 대해 특정 제한을 적용합니다.

모든 신규 계정은 Amazon SES 샌드박스로 배치됩니다. 계정의 샌드박스 상태는 각 계정마다 고유합니다 AWS 리전. 계정이 샌드박스에 배치된 동안에는 Amazon SES의 모든 기능을 사용할 수 있습니다. 하지만 계정이 샌드박스에 있을 때는 다음과 같은 제한이 계정에 적용됩니다.

- 확인된 이메일 주소 및 도메인으로만 또는 Amazon SES 메일박스 시뮬레이터로만 메일을 전송할 수 있습니다.
- 24시간 동안 최대 200개의 메시지를 보낼 수 있습니다.
- 초당 최대 1개의 메시지를 보낼 수 있습니다.
- 발신 권한 부여의 경우, 본인 및 위임 발신자 모두 확인되지 않은 이메일 주소로 이메일을 보낼 수 없습니다.
- 계정 수준 금지의 경우, 금지 목록 관리와 관련된 대량 작업 및 SES API 호출이 비활성화됩니다.

계정이 샌드박스 외부에 있고 프로덕션에 있는 경우 수신자의 주소 또는 도메인이 확인되었는지 여부와 상관없이 모든 수신자에게 이메일을 보낼 수 있습니다. 하지만 "From", "Source", "Sender", 또는 "Return-Path" 주소로 사용하는 모든 ID는 계속 확인해야 합니다.

샌드박스에서 계정을 삭제하고 프로덕션에 배치해 줄 것을 요청하려면 이 섹션의 절차를 수행합니다.

> **Tip**
>
> - 신규 고객이고 아직 ID를 만들지 않은 경우 콘솔의 SES 계정 설정 마법사가 사용 설정되어 있으면 시작하는 데 도움이 됩니다. 마법사에 액세스하는 방법에 대한 설명은 SES 계정 설정을 참조하세요.
> - 하나 이상의 ID를 이미 만든 경우 계정 설정 마법사 대신 설정 시작 페이지가 표시됩니다.
> - ID 중 하나가 확인된 도메인인 경우 설정 시작 페이지에서 직접 프로덕션 액세스를 요청할 수도 있습니다. 프로덕션 액세스를 요청하기 전에 SES로 도메인을 확인하는 것이 프로덕션 액세스 요청을 더 빨리 승인하는 것이므로 즉시 이메일 전송을 시작하는 데 도움이 되는 모범 사례에 해당합니다.

> **Note**
>
> - Amazon SES를 사용하여 Amazon EC2 인스턴스에서 이메일을 보내는 경우 Amazon EC2 인스턴스의 포트 25에서 병목 현상을 제거하도록 요청해야 할 수도 있습니다. 자세한 내용은 AWS 지식 센터의 EC2 인스턴스에서 포트 25의 스로틀을 제거하려면 어떻게 해야 합니까?를 참조하세요.

AWS Management Console을 사용하여 프로덕션 액세스를 요청하는 방법(샌드박스에서 계정 제거)

1. https://console.aws.amazon.com/ses/에서 Amazon SES 콘솔을 엽니다.
2. 탐색 패널에서 계정 대시보드를 선택합니다.
3. 콘솔의 맨 위에 있는 경고 상자, "Amazon SES 계정이 샌드박스에 있습니다"에서 오른쪽의 설정 시작 보기 페이지를 선택한 후 프로덕션에 액세스 요청을 선택합니다.
4. 계정 세부 정보 모달에서 보낼 메일의 대부분을 가장 잘 설명하는 Marketing(마케팅) 또는 Transitional(전환) 라디오 버튼을 선택합니다.
   - 마케팅 이메일 - 구매, 다운로드 정보 등과 같은 마케팅 및 홍보 콘텐츠를 포함하여 대상 잠재 고객 또는 고객 목록에 일대다 방식으로 전송됩니다.
   - 트랜잭션 이메일 - 일반적으로 웹 사이트 구매, 암호 재설정 요청 등과 같은 사용자 작업에 의해 트리거되어 각 수신자에게 고유한 일대일 방식으로 전송됩니다.
5. 웹사이트 URL에서 전송하려는 콘텐츠의 종류를 더 잘 이해할 수 있도록 웹 사이트의 URL을 입력합니다.
6. 추가 연락처에서, 계정에 대한 커뮤니케이션을 수신할 위치를 지정합니다. 최대 4개의 이메일 주소를 쉼표로 구분하여 사용할 수 있습니다.
7. 기본 연락처 언어에서 커뮤니케이션을 영어 또는 일본어로 수신할지 여부를 선택합니다.
8. 승인에서 명시적으로 요청한 사람에게만 이메일을 보내는 데 동의하는 확인란을 선택하고 반송 메일 및 수신 거부 알림을 처리하는 프로세스가 마련되어 있는지 확인합니다.
9. Submit request(요청 제출) 버튼을 선택하면 요청이 제출되었으며 현재 검토 중임을 확인하는 배너가 표시됩니다.

계정 세부 정보에 대한 검토를 제출한 후에는 검토가 완료될 때까지 세부 정보를 편집할 수 없습니다. AWS Support 팀은 24시간 이내에 요청에 대한 초기 응답을 제공합니다.

시스템이 원치 않는 콘텐츠 또는 악성 콘텐츠를 전송하지 않도록 하기 위해 각 요청을 신중하게 고려해야 합니다. 가능한 경우 이 24시간 이내에 요청에 대한 권한을 부여합니다. 그러나 사용자의 추가 정보가 필요한 경우 요청을 해결하는 데 시간이 오래 걸릴 수도 있습니다.

선택적으로를 사용하여 프로덕션 액세스 요청을 제출할 수도 있습니다 AWS CLI. 를 사용하여 요청을 제출하면 많은 자격 증명에 대한 프로덕션 액세스를 요청하거나 Amazon SES 설정 프로세스를 자동화하려는 경우에 AWS CLI 유용합니다.

AWS CLI을(를) 사용하여 Amazon SES 샌드박스에서 계정을 삭제해 줄 것을 요청하려면

1. 사전 조건: AWS CLI을(를) 설치하고 구성해야 합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.
2. 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 put-account-details \
--production-access-enabled \
--mail-type TRANSACTIONAL \
--website-url https://example.com \
--additional-contact-email-addresses info@example.com \
--contact-language EN
```

   위의 명령에서 다음을 수행합니다.

   a. *TRANSACTIONAL*을 Amazon SES를 통해 보내려는 이메일 유형으로 바꿉니다. `TRANSACTIONAL` 또는 `MARKETING`를 지정할 수 있습니다. 1개 이상의 값이 해당될 경우, 전송하려는 대부분의 이메일에 적용되는 옵션을 선택하세요.
   b. *https://example.com*를 웹 사이트의 URL로 바꿉니다. 이 정보를 제공하면 보내려는 콘텐츠 유형을 지원 센터에서 쉽게 파악할 수 있습니다.
   c. *info@example.com*을 계정에 대한 커뮤니케이션을 수신하려는 이메일 주소로 바꿉니다. 최대 4개의 이메일 주소를 쉼표로 구분하여 사용할 수 있습니다.
   d. *EN*을 선호하는 언어로 바꿉니다. 영어는 `EN`, 일본어는 `JA`을(를) 지정할 수 있습니다.

계정 세부 정보에 대한 검토를 제출한 후에는 검토가 완료될 때까지 세부 정보를 편집할 수 없습니다. AWS Support 팀은 24시간 이내에 요청에 대한 초기 응답을 제공합니다.

시스템이 원치 않는 콘텐츠 또는 악성 콘텐츠를 전송하지 않도록 하기 위해 각 요청을 신중하게 고려해야 합니다. 가능한 경우 이 24시간 이내에 요청에 대한 권한을 부여합니다. 그러나 사용자의 추가 정보가 필요한 경우 요청을 해결하는 데 시간이 오래 걸릴 수도 있습니다.

Amazon SES 발신 한도 관리

Amazon SES 계정에는 보낼 수 있는 이메일 메시지 수와 속도를 규제하는 몇 가지 발신 할당량이 적용됩니다. 발신 할당량은 Amazon SES와 이메일 공급자 간에 신뢰 관계를 유지하는 데 도움이 되기 때문에 모든 Amazon SES 고객에게 유용합니다. 발신 할당량을 이용하여 전송 활동을 점차적으로 늘리고 이메일 전송 볼륨 또는 속도의 갑작스러운 증가로 인해 이메일 공급자가 이메일을 차단할 가능성을 줄일 수 있습니다.

다음 할당량은 Amazon SES를 통해 이메일을 보낼 때 적용됩니다.

- 발신 할당량 - 24시간 내에 보낼 수 있는 최대 이메일 수입니다. 이 할당량은 롤링 기간을 반영하여 계산됩니다. 이메일 전송을 시도할 때마다 Amazon SES에서 지난 24시간 내에 보낸 이메일 수를 확인합니다. 지난 24시간 동안 보낸 이메일의 총 수가 이 일일 최대치보다 낮으면 전송 요청이 수락되고 이메일이 전송됩니다.

  메시지 전송이 계정의 일일 최대치를 초과하면 Amazon SES 호출이 거부됩니다.
- 발신 속도 — Amazon SES가 계정에서 초당 수락할 수 있는 최대 이메일 수입니다. 잠깐 동안 이 할당량을 초과할 수 있지만 장기적으로는 초과할 수 없습니다.

> **Note**
>
> Amazon SES에서 메시지를 수락하는 비율은 계정의 최대 전송률보다 적을 수 있습니다.

- 최대 메시지 크기(MB) - 보낼 수 있는 최대 이메일 크기입니다. 여기에는 MIME 인코딩 후 이메일의 일부가 되는 이미지 및 첨부 파일이 포함됩니다. 예를 들어 5MB 파일을 첨부하면 MIME 인코딩 후 이메일의 첨부 파일 크기는 ~6.85MB(원본 파일 크기의 약 137%)가 됩니다.

> **Note**
>
> 클라우드 드라이브에 첨부 파일을 업로드하고 클라우드 드라이브 첨부 파일의 URL을 포함시켜 이메일 크기를 줄이고 전달률을 높이는 것이 좋습니다. SES는 다른 메일 서버가 다양한 크기 기반 정책을 규정하므로 대용량 이메일이 수신자 사서함에 도달한다고 보장할 수 없습니다.

Amazon SES 전송 할당량은 AWS 리전마다 다릅니다. 여러 AWS 리전에서 Amazon SES를 사용하는 방법에 대한 자세한 내용은 섹션을 참조하세요리전 및 Amazon SES.

계정이 Amazon SES 샌드박스에 있는 경우 24시간 동안 메시지 200개만 보낼 수 있으며 최대 전송 속도는 초당 메시지 한 개입니다. 샌드박스에서 계정을 제거하도록 요청을 제출할 때 할당량을 동시에 늘리도록 요청할 수도 있습니다. 샌드박스에서 계정을 제거하는 방법에 대한 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하세요.

계정이 샌드박스에서 제거되면 AWS 지원 센터에서 새 사례를 생성하여 언제든지 추가 할당량 증가를 요청할 수 있습니다. 자세한 내용은 Amazon SES 발신 할당량 높이기 단원을 참조하십시오.

> **Note**
>
> 발신 할당량은 메시지 수가 아닌 수신자 수를 기준으로 합니다. 예를 들어, 수신자가 10명인 이메일은 할당량 계산에서 10개로 간주됩니다. 그러나 호출이 실패하면 전체 이메일이 거부되기 때문에 `SendEmail` API 작업을 한 번 호출하여 여러 수신자에게 이메일을 보내지 않는 것이 좋습니다. 따라서 모든 수신자에 대해 한 번씩 `SendEmail`을(를) 호출하는 것이 좋습니다.

- 발신 할당량을 늘리려면 Amazon SES 발신 할당량 높이기 단원을 참조하세요.
- Amazon SES 콘솔 또는 Amazon SES API를 사용하여 발신 할당량을 모니터링하려면 Amazon SES 발신 할당량 모니터링단원을 참조하십시오.
- 발신 할당량에 도달하는 경우 애플리케이션에서 받는 오류에 대한 자세한 내용은 Amazon SES 계정의 발신 할당량과 관련된 오류 단원을 참조하세요.

Amazon SES 발신 할당량 높이기

귀하의 계정에는 현재 리전에 따라 다음과 같은 할당량이 존재하며 이러한 할당량을 늘릴 수 있습니다.

리소스	기본 할당량	설명
발신 할당량	200	현재 AWS 리전의 이 계정에 대하여 24시간 동안 보낼 수 있는 이메일의 최대 개수입니다.
전송 속도	1	Amazon SES가 현재 AWS 리전의 이 계정에 대해 초당 수락할 수 있는 최대 이메일 개수입니다.

자동으로 발신 할당량 높이기

계정이 샌드박스 외부에 있을 때 고품질 프로덕션 이메일을 전송하는 경우, 계정의 발신 할당량이 자동으로 증가할 수 있습니다. 실제로 이러한 할당량을 높여야 할 필요가 생기기 전에 이 할당량은 자동으로 증가하는 경우가 많습니다.

송신률을 자동으로 높이려면 다음과 같은 문들이 모두 true여야 합니다.

- 수신자가 수신하려는 고품질의 콘텐츠를 전송합니다 – 수신자가 원하고 기대하는 콘텐츠를 전송하세요. 이메일을 열지 않은 고객들을 대상으로 이메일 전송을 중지하세요.
- 실제 프로덕션 콘텐츠를 전송합니다 – 가짜 이메일 주소로 테스트 메시지를 전송하면 반송 메일 및 수신 거부 발생률에 부정적인 영향을 미칠 수 있습니다. 또한 내부 수신자들에게만 메시지를 전송하면 고객이 받고 싶은 콘텐츠를 보내고 있는지 판단하기가 어렵습니다. 그러나 내부 수신자가 아닌 수신자에게 프로덕션 메시지를 전송하면 이메일 발송 사례를 정확하게 평가할 수 있습니다.
- 현재 할당량에 가까운 수준으로 전송합니다 – 자동으로 할당량을 높이려면 일일 이메일 볼륨이 할당량을 초과하지 않고 주기적으로 계정의 일일 최대 할당량에 근접해야 합니다.
- 반송 메일 및 수신 거부 발생률이 낮습니다 – 수신 중인 반송 메일 및 수신 거부의 건수를 최소화하세요. 반송 메일 및 수신 거부의 건수가 많을 경우, 발신 할당량에 부정적인 영향을 미칠 수 있습니다.

사용자의 발신 할당량 증가 요청

현재 전송 할당량이 필요한 수준에 미치지 못하고 있으며 이 할당량이 자동으로 증가하지 않았다면 할당량 증가를 요청할 수 있습니다.

- 전송 할당량 또는 전송 속도 - 이 중 하나에 대한 증가 요청은 AWS Service Quotas 콘솔을 통해 제출할 수 있습니다.

Service Quotas 콘솔을 사용하여 Amazon SES 발신 할당량 증가를 요청하려면

1. Service Quotas Console을 엽니다.
2. 콘솔의 오른쪽 상단 모서리에 있는 드롭다운(계정 번호 옆)을 사용하여 증가시킬 리전을 선택합니다.
3. 탐색 창에서 AWS services(서비스)를 선택합니다.
4. Amazon Simple Email Service(Amazon SES)를 선택합니다.
5. 할당량을 선택한 다음, 지침에 따라 할당량 증가를 요청합니다.

> **AWS Support 증가 요청 유형에 대한 팀 SLA**
>
> 시스템이 원치 않는 콘텐츠 또는 악성 콘텐츠를 전송하지 않도록 하기 위해 각 요청을 신중하게 고려해야 합니다. 이렇게 할 수 있는 경우 요청된 증가 유형에 대해 아래에 나열된 지정된 시간 내에 요청을 승인합니다. 그러나 사용자의 추가 정보가 필요한 경우 요청을 해결하는 데 시간이 오래 걸릴 수도 있습니다. 정책에 맞지 않는 사용 사례인 경우, 요청이 승인되지 않을 수 있습니다.
>
> - 전송 할당량 또는 전송 속도: 최대 24시간.

> **Note**
>
> Service Quotas 콘솔은 다양한 언어로 제공되지만 실제 지원은 영어로만 제공됩니다.

Amazon SES 발신 할당량 모니터링

Amazon SES 콘솔 또는 Amazon SES API를 사용하여 AWS SDK, AWS Command Line Interface 또는 AWS Tools for Windows PowerShell을(를) 통해 직접 또는 간접적으로 쿼리(HTTPS) 인터페이스를 호출함으로써 발신 할당량을 모니터링할 수 있습니다.

> **Important**
>
> 발신 할당량에 근접하지 않았는지 전송 통계를 자주 확인하는 것이 좋습니다. 발신 할당량에 근접한 경우 발신 할당량을 높이는 방법에 대한 자세한 내용은 Amazon SES 발신 할당량 높이기 단원을 참조하세요. 발신 할당량에 도달할 때까지 기다렸다가 발신 할당량을 높이지 마세요.

Amazon SES 콘솔을 사용하여 발신 할당량 모니터링

다음 절차에서는 Amazon SES 콘솔을 사용하여 발신 할당량을 보는 방법을 보여 줍니다.

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.

2. 탐색 창에서 Account dashboard(계정 대시보드)를 선택합니다. 발신 할당량이 Sending Limits(발신 한도) 아래에 표시됩니다. 전송된 총 이메일, 남은 전송 이메일 및 사용된 전송 할당량의 백분율은 Daily email usage(일별 이메일 사용량)에 표시됩니다.

3. 표시를 업데이트하려면 Daily email usage(일별 이메일 사용량) 상자의 오른쪽 상단 모서리에 있는 새로 고침 아이콘을 선택합니다.

Amazon SES API를 사용하여 발신 할당량 모니터링

Amazon SES API는 발신 할당량을 반환하는 `GetSendQuota` 작업을 제공합니다. `GetSendQuota` 작업을 호출하면 다음과 같은 정보를 받습니다.

- 지난 24시간 동안 전송한 이메일 수
- 앞으로 24시간 동안 발신 할당량
- 최대 전송 속도

> **Note**
>
> `GetSendQuota`에 대한 자세한 설명은 Amazon Simple Email Service API 참조를 참조하십시오.

Amazon SES 계정의 발신 할당량과 관련된 오류

일일 발신 할당량(24시간 기준으로 전송할 수 있는 최대 이메일 수) 또는 최대 송신률(초당 보낼 수 있는 최대 메시지 수)에 도달한 후에 이메일을 보내려고 하면 Amazon SES가 메시지를 거부하고 재전송을 시도하지 않습니다. 또한 Amazon SES는 문제를 설명하는 오류 메시지를 제공합니다. Amazon SES가 이 오류 메시지를 생성하는 방법은 이메일을 보내려 할 때 사용한 방법에 따라 달라집니다. 이 주제에는 Amazon SES API 및 SMTP 인터페이스를 통해 수신하는 메시지에 대한 정보가 포함됩니다.

최대 송신률에 도달할 때 사용할 수 있는 기술에 대해서는 AWS 메시징 및 타깃팅 블로그에서 "조절(Throttling) — 최대 송신률 초과" 오류를 처리하는 방법을 참조하십시오.

Amazon SES API를 통해 발신량 한도 도달

Amazon SES API(또는 AWS SDK)를 사용하여 이메일을 보내려고 하지만 이미 계정의 전송 한도를 초과한 경우 API에서 `ThrottlingException` 오류가 발생합니다. 오류 메시지에는 다음 중 하나의 메시지가 포함되어 있습니다.

- `Daily message quota exceeded`
- `Maximum sending rate exceeded`

조절 오류가 발생한 경우 애플리케이션을 프로그래밍하고 최대 10분간 기다렸다가 전송 요청을 다시 시도하세요.

SMTP를 통해 발신 한도 도달

Amazon SES SMTP 인터페이스를 사용하여 이메일을 보내려 하지만 이미 계정의 발신 한도를 초과했다면 SMTP 클라이언트가 다음 오류 중 하나를 표시할 수 있습니다.

- `454 Throttling failure: Maximum sending rate exceeded`
- `454 Throttling failure: Daily message quota exceeded`

SMTP 클라이언트마다 이러한 오류를 다르게 처리합니다.

Amazon SES를 사용하여 이메일 전송 설정

Amazon SES 콘솔, 간이 전자 우편 전송 프로토콜(SMTP) 인터페이스 또는 Amazon SES API를 사용하여Amazon Simple Email Service (Amazon SES)로 이메일을 전송할 수 있습니다. 일반적으로 콘솔을 사용하여 테스트 이메일을 보내고 전송 활동을 관리합니다. 대용량 이메일을 보내려면 SMTP 인터페이스 또는 API를 사용합니다. Amazon SES 이메일 요금에 대한 자세한 내용은 Amazon SES 요금을 참조하세요.

- SMTP 지원 소프트웨어 패키지, 애플리케이션 또는 프로그래밍 언어를 사용하여 Amazon SES를 통해 이메일을 보내거나 Amazon SES를 기존 메일 서버와 통합하려는 경우 Amazon SES SMTP 인터페이스를 사용합니다. 자세한 내용은 프로그래밍 방식으로 Amazon SES SMTP 인터페이스를 통해 이메일 전송 단원을 참조하십시오.
- 원시 HTTP 요청을 사용하여 Amazon SES를 호출하려는 경우 Amazon SES API를 사용합니다. 자세한 내용은 Amazon SES API를 사용하여 이메일 보내기 단원을 참조하십시오.

> **Important**
>
> 여러 수신자에게 이메일을 보내는 경우(수신자는 "To", "CC" 및 "BCC" 주소) Amazon SES에 대한 호출이 실패하면 전체 이메일이 거부되고 모든 수신자에게 의도한 이메일이 수신되지 않습니다. 따라서 한 번에 한 명의 수신자에게 이메일을 보내는 것이 좋습니다.

Amazon SES SMTP 인터페이스를 사용하여 이메일 보내기

Amazon SES를 통해 프로덕션 이메일을 보내려면 SMTP(Simple Mail Transfer Protocol) 인터페이스 또는 Amazon SES API를 사용할 수 있습니다. Amazon SES API에 대한 자세한 내용은 Amazon SES API를 사용하여 이메일 보내기 섹션을 참조하세요. 이 섹션에서는 SMTP 인터페이스에 대해 설명합니다.

Amazon SES는 인터넷에서 가장 일반적인 이메일 프로토콜인 SMTP를 사용하여 이메일을 전송합니다. 다양한 SMTP 지원 프로그래밍 언어 및 소프트웨어에서 Amazon SES SMTP 인터페이스에 연결하여 Amazon SES를 통해 이메일을 보낼 수 있습니다. 이 섹션에서는 Amazon SES SMTP 자격 증명을 받는 방법, Amazon SES SMTP 인터페이스를 사용하여 이메일을 보내는 방법, 이메일 전송에 Amazon SES를 사용하도록 몇 가지 소프트웨어 및 메일 서버를 구성하는 방법을 설명합니다.

SMTP 인터페이스를 통해 Amazon SES를 사용할 때 발생할 수 있는 일반적인 문제에 대한 해결책은 Amazon SES SMTP 문제 섹션을 참조하세요.

SMTP를 통한 이메일 전송 요구 사항

Amazon SES SMTP 인터페이스를 사용하여 이메일을 보내려면 다음 항목이 필요합니다.

- SMTP 엔드포인트 주소입니다. Amazon SES SMTP 엔드포인트 목록은 Amazon SES SMTP 엔드포인트에 연결 단원을 참조하십시오.
- SMTP 인터페이스 포트 번호. 포트 번호는 연결 방법에 따라 다릅니다. 자세한 내용은 Amazon SES SMTP 엔드포인트에 연결 단원을 참조하세요.
- SMTP 사용자 이름 및 암호. SMTP 자격 증명은 각 AWS 리전마다 고유합니다. SMTP 인터페이스를 사용하여 여러 AWS 리전에서 이메일을 전송하려면 리전별로 SMTP 보안 인증 정보가 필요합니다.

> **Important**
>
> SMTP 자격 증명은 AWS 액세스 키 또는 Amazon SES 콘솔에 로그인하는 데 사용하는 자격 증명과 동일하지 않습니다. SMTP 보안 인증 정보를 생성하는 방법에 대한 자세한 내용은 Amazon SES SMTP 자격 증명 획득 섹션을 참조하세요.

- TLS(전송 계층 보안)를 사용하여 통신할 수 있는 클라이언트 소프트웨어. 자세한 내용은 Amazon SES SMTP 엔드포인트에 연결 단원을 참조하세요.
- Amazon SES에서 확인한 이메일 주소. 자세한 내용은 Amazon SES에서 확인된 자격 증명 단원을 참조하세요.
- 대량의 이메일을 보내려는 경우 높은 발신 할당량. 자세한 내용은 Amazon SES 발신 한도 관리 단원을 참조하십시오.

SMTP를 통해 전자 메일을 보내는 방법

다음 방법 중 하나를 통해 SMTP를 통해 이메일을 보낼 수 있습니다.

- Amazon SES SMTP 인터페이스를 통해 이메일을 보내도록 이러한 SMTP 지원 소프트웨어를 구성하려면 소프트웨어 패키지를 사용하여 Amazon SES를 통해 이메일 보내기 섹션을 참조하세요.
- Amazon SES를 통해 이메일을 보내도록 애플리케이션을 프로그래밍하는 방법은 프로그래밍 방식으로 Amazon SES SMTP 인터페이스를 통해 이메일 전송 섹션을 참조하세요.
- Amazon SES를 통해 발신 메일을 보내도록 기존 이메일 서버를 구성하는 방법은 기존 이메일 서버와 Amazon SES 통합 섹션을 참조하세요.
- 명령줄을 사용하여 Amazon SES SMTP 인터페이스와 상호 작용하는 방법(테스트에 유용할 수 있음)은 명령줄을 사용하여 Amazon SES SMTP 인터페이스에 대한 연결 테스트 섹션을 참조하세요.

SMTP 응답 코드의 목록은 Amazon SES에서 반환하는 SMTP 응답 코드을(를) 참조하세요.

제공할 이메일 정보

SMTP 인터페이스를 통해 Amazon SES에 액세스하는 경우 SMTP 클라이언트 애플리케이션이 메시지를 수집합니다. 따라서 제공해야 할 정보는 어떤 애플리케이션을 사용하는가에 따라 달라집니다. 클라이언트와 서버 간의 SMTP 교환에는 최소한 다음이 필요합니다.

- 소스 주소
- 대상 주소
- 메시지 데이터

SMTP 인터페이스를 사용 중이고 피드백 전달이 활성화된 상태라면 반송 메일, 수신 거부 및 전송 알림이 "MAIL FROM" 주소로 전송됩니다. 사용자가 지정한 "Reply-To" 주소는 사용되지 않습니다.

Amazon SES SMTP 자격 증명 획득

SES SMTP 인터페이스에 액세스하려면 Amazon SES SMTP 보안 인증 정보가 필요합니다.

SES SMTP 인터페이스를 통해 이메일을 보내는 데 사용하는 자격 증명은 각 AWS 리전에 고유합니다. SES SMTP 인터페이스를 사용하여 둘 이상의 리전에서 이메일을 전송하는 경우, 사용할 계획 중인 각 리전에 대해 SMTP 자격 증명 세트를 생성해야 합니다.

SMTP 암호는 AWS 보안 액세스 키와 다릅니다. 자격 증명에 대한 자세한 내용은 Amazon SES 자격 증명 유형 단원을 참조하세요.

> **Note**
>
> 현재 사용 가능한 SMTP 엔드포인트 목록은의 SMTP 엔드포인트를 참조하세요AWS 일반 참조.

SES 콘솔을 사용하여 SES SMTP 보안 인증 정보 받기

요구 사항

IAM 사용자는 SES SMTP 보안 인증 정보를 생성할 수 있지만 IAM을 사용하여 SES SMTP 보안 인증 정보를 생성하기 때문에 사용자의 정책이 사용자에게 IAM 자체를 사용할 권한을 부여해야 합니다. IAM 정책은 사용자가 `iam:ListUsers`, `iam:CreateUser`, `iam:CreateAccessKey`

및 `iam:PutUserPolicy`(이)라는 IAM 작업을 수행하도록 허용해야 합니다. 콘솔을 사용하여 SES SMTP 자격 증명을 생성하려고 하는데 IAM 사용자에게 이러한 권한이 없는 경우 계정이 "iam:ListUsers."

> **Important**
>
> 위에서 참조한 IAM 작업에는 서비스에서 리소스 권한을 부여하거나 수정할 수 있는 권한을 부여하기 때문에 IAM 수준이 가장 높은 권한 관리 액세스 수준이 있습니다. 따라서 AWS 계정의 보안을 개선하려면 권한 관리 액세스 수준 분류를 포함하는 이러한 정책을 제한하거나 정기적으로 모니터링하는 것이 좋습니다.

SMTP 자격 증명을 만들려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 SMTP settings(SMTP 설정)를 선택합니다. 그러면 Simple Mail Transfer Protocol (SMTP) settings(SMTP(Simple Mail Transfer Protocol) 설정) 페이지가 열립니다.
3. 오른쪽 상단에서 Create SMTP Credentials(SMTP 보안 인증 생성)를 선택합니다. IAM 콘솔이 열립니다.
4. (선택 사항) 이미 생성한 SMTP 사용자를 보거나 편집하거나 삭제해야 하는 경우 오른쪽 하단에서 Manage my existing SMTP credentials(기존 SMTP 보안 인증 관리)를 선택합니다. IAM 콘솔이 열립니다. SMTP 보안 인증 관리에 대한 세부 정보는 다음 절차에 따라 제공됩니다.
5. SMTP 사용자 생성의 사용자 이름 필드에 SMTP 사용자의 이름을 입력합니다. 또는 이 필드에 입력된 기본값을 사용할 수 있습니다. 마쳤으면 오른쪽 하단에서 사용자 생성을 선택합니다.
6. SMTP 암호 아래의 표시를 선택하면 SMTP 보안 인증 정보가 화면에 표시됩니다.
7. 이 대화 상자를 닫은 후에는 보안 인증 정보를 보거나 저장할 수 없으므로 .csv 파일 다운로드를 선택하여 이러한 보안 인증 정보를 다운로드하거나 복사하여 안전한 장소에 저장하세요.
8. SES 콘솔로 돌아가기를 선택합니다.

IAM 콘솔의 액세스 관리(Access management)에서 이 절차를 사용하여 생성한 SMTP 보안 인증 정보 목록을 보고 사용자(Users)를 선택한 다음 검색 창을 사용하여 SMTP 보안 인증 정보를 할당한 모든 사용자를 찾을 수 있습니다.

IAM 콘솔을 사용하여 기존 SMTP 사용자를 삭제할 수도 있습니다. 사용자 삭제에 대한 자세한 내용은 IAM 시작 안내서의 IAM 사용자 관리를 참조하세요.

SMTP 암호를 변경하려면 IAM 콘솔에서 기존 SMTP 사용자를 삭제합니다. 그런 다음 이전 절차를 완료하여 새 SMTP 자격 증명 집합을 만듭니다.

기존 자격 증명을 변환하여 SES SMTP AWS 자격 증명 획득

IAM 인터페이스를 사용하여 설정한 사용자가 있는 경우 자격 증명에서 사용자의 SES SMTP AWS 자격 증명을 추출할 수 있습니다.

> **Important**
>
> 임시 AWS 자격 증명을 사용하여 SMTP 자격 증명을 추출하지 마십시오. SES SMTP 인터페이스는 임시 보안 인증 정보에서 생성된 SMTP 보안 인증 정보를 지원하지 않습니다.

IAM 사용자가 SES SMTP 인터페이스를 사용하여 이메일을 보내도록 하는 방법

1. 다음 절차에 따라이 섹션에 제공된 알고리즘을 사용하여 자격 증명에서 사용자의 SMTP AWS 자격 증명을 도출합니다.

   자격 AWS 증명부터 시작하므로 SMTP 사용자 이름은 AWS 액세스 키 ID와 동일하므로 SMTP 암호만 생성하면 됩니다.
2. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/iam/ IAM 콘솔을 엽니다.
3. 액세스 관리에서 정책을 선택한 다음 정책 만들기를 선택합니다.
4. 정책 편집기에서 JSON을 선택하고 편집기에서 예제 코드를 제거합니다.
5. 다음 권한 정책을 편집기에 붙여넣습니다.

   JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
"Effect": "Allow",
            "Action": "ses:SendRawEmail",
            "Resource": "*"
        }
    ]
}
```

6. 다음을 선택하고 정책 이름 필드에 `AmazonSesSendingAccess`를 입력한 다음 정책 만들기를 입력합니다.
7. 액세스 관리에서 사용자 그룹을 선택한 다음 그룹 만들기를 선택합니다.
8. 사용자 그룹 이름 필드에 `AWSSESSendingGroupDoNotRename`을 입력합니다.
9. 그룹에 사용자 추가 테이블에서 SMTP 사용자를 선택하여 그룹에 이를 추가합니다.
10. 권한 정책 연결 테이블에서 정책을 선택한 다음 사용자 그룹 만들기를 선택하여 이전에 만든 `AmazonSesSendingAccess` 정책을 연결합니다.

IAM에서 SES를 사용하는 방법에 대한 자세한 내용은 Amazon SES의 Identity and Access Management 섹션을 참조하세요.

> **Note**
>
> IAM 사용자의 SES SMTP 보안 인증 정보를 생성할 수 있지만 SMTP 보안 인증 정보를 생성할 때 별도의 IAM 사용자를 생성하는 것이 좋습니다. 특정 용도로 사용자를 생성하는 것이 좋은 이유에 대한 자세한 내용은 IAM 모범 사례를 참조하세요.

다음 의사 코드는 AWS 보안 액세스 키를 SES SMTP 암호로 변환하는 알고리즘을 보여줍니다.

```
// Modify this variable to include your AWS secret access key
key = "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY";

// Modify this variable to refer to the AWS Region that you want to use to send email.
region = "us-west-2";

// The values of the following variables should always stay the same.
date = "11111111";
service = "ses";
terminal = "aws4_request";
message = "SendRawEmail";
version = 0x04;

kDate = HmacSha256(date, "AWS4" + key);
kRegion = HmacSha256(region, kDate);
kService = HmacSha256(service, kRegion);
kTerminal = HmacSha256(terminal, kService);
kMessage = HmacSha256(message, kTerminal);
signatureAndVersion = Concatenate(version, kMessage);
```

```
smtpPassword = Base64(signatureAndVersion);
```

일부 프로그래밍 언어에는 IAM 보안 액세스 키를 SMTP 암호로 변환하는 데 사용할 수 있는 라이브러리가 포함되어 있습니다. 이 섹션에는 Python을 사용하여 AWS 보안 액세스 키를 SES SMTP 암호로 변환하는 데 사용할 수 있는 코드 예제가 포함되어 있습니다.

> **Note**
>
> - 다음 예에는 Python 3.6에 도입된 f-strings가 사용됩니다. 이전 버전을 사용하면 작동하지 않습니다.
> - 다음 예에서 SMTP_REGIONS 목록은 단순한 예입니다. 실제 리전 목록은 각각에 대해 SMTP 자격 증명이 필요하므로 이메일을 보내려는 리전에 따라 더 짧거나 길어질 수 있습니다 AWS 리전.

Python

```
#!/usr/bin/env python3

import hmac
import hashlib
import base64
import argparse

SMTP_REGIONS = [
    "us-east-2",  # US East (Ohio)
    "us-east-1",  # US East (N. Virginia)
    "us-west-2",  # US West (Oregon)
    "ap-south-1",  # Asia Pacific (Mumbai)
    "ap-northeast-2",  # Asia Pacific (Seoul)
    "ap-southeast-1",  # Asia Pacific (Singapore)
    "ap-southeast-2",  # Asia Pacific (Sydney)
    "ap-northeast-1",  # Asia Pacific (Tokyo)
    "ca-central-1",  # Canada (Central)
    "eu-central-1",  # Europe (Frankfurt)
    "eu-west-1",  # Europe (Ireland)
    "eu-west-2",  # Europe (London)
    "eu-south-1",  # Europe (Milan)
    "eu-north-1",  # Europe (Stockholm)
    "sa-east-1",  # South America (Sao Paulo)
    "us-gov-west-1",  # AWS GovCloud (US)
```

```
    "us-gov-east-1",  # AWS GovCloud (US)
]

# These values are required to calculate the signature. Do not change them.
DATE = "11111111"
SERVICE = "ses"
MESSAGE = "SendRawEmail"
TERMINAL = "aws4_request"
VERSION = 0x04


def sign(key, msg):
    return hmac.new(key, msg.encode("utf-8"), hashlib.sha256).digest()


def calculate_key(secret_access_key, region):
    if region not in SMTP_REGIONS:
        raise ValueError(f"The {region} Region doesn't have an SMTP endpoint.")

    signature = sign(("AWS4" + secret_access_key).encode("utf-8"), DATE)
    signature = sign(signature, region)
    signature = sign(signature, SERVICE)
    signature = sign(signature, TERMINAL)
    signature = sign(signature, MESSAGE)
    signature_and_version = bytes([VERSION]) + signature
    smtp_password = base64.b64encode(signature_and_version)
    return smtp_password.decode("utf-8")


def main():
    parser = argparse.ArgumentParser(
        description="Convert a Secret Access Key to an SMTP password."
    )
    parser.add_argument("secret", help="The Secret Access Key to convert.")
    parser.add_argument(
        "region",
        help="The AWS Region where the SMTP password will be used.",
        choices=SMTP_REGIONS,
    )
    args = parser.parse_args()
    print(calculate_key(args.secret, args.region))


if __name__ == "__main__":
```

```
    main()
```

이 스크립트를 사용하여 SMTP 암호를 얻으려면 이전 코드를 `smtp_credentials_generate.py`(으)로 저장합니다. 그런 다음 명령줄에서 다음 명령을 실행합니다.

```
python path/to/smtp_credentials_generate.py wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY us-east-1
```

위의 명령에서 다음을 수행합니다.

- *`path/to/`*를 `smtp_credentials_generate.py`을(를) 저장한 위치의 경로로 바꿉니다.
- *`wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`*를 SMTP 암호로 전환하려는 시크릿 액세스 키로 대체합니다.
- *`us-east-1`*을 SMTP 자격 증명을 사용하려는 AWS 리전으로 바꿉니다.

이 스크립트가 성공적으로 실행되면 SMTP 암호만 출력됩니다.

기존 인라인 정책에서 그룹 정책으로 SMTP 사용자 마이그레이션(보안 권장 사항)

> **Important**
>
> 2024년 9월 6일 이전에 SES SMTP 자격 증명을 만든 경우 인라인 정책과 태그가 SMTP 사용자에게 연결되어 있습니다. SES는 인라인 정책에서 벗어나 보안 권장 사항과 동일하게 수행할 것을 권장합니다.

기존 인라인 정책의 SMTP 사용자를 그룹 정책으로 마이그레이션하기 전에 먼저 SES 권한 정책으로 IAM 사용자 그룹을 만들어 인라인 정책을 대신해야 합니다. 이 IAM 사용자 그룹을 이미 만들었거나 2024년 9월 6일 이후부터 만들어진 SMTP 자격 증명이 자동으로 만들어진 경우 다음 절차의 10단계로 바로 건너뛸 수 있습니다.

기존 인라인 정책에서 관리형 그룹으로 마이그레이션하는 방법

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/iam/ IAM 콘솔을 엽니다.
2. 액세스 관리에서 정책을 선택한 다음 정책 만들기를 선택합니다.

3. 정책 편집기에서 JSON을 선택하고 편집기에서 예제 코드를 제거합니다.
4. 다음 권한 정책을 편집기에 붙여넣습니다.

   JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
"Effect": "Allow",
            "Action": "ses:SendRawEmail",
            "Resource": "*"
        }
    ]
}
```

5. 다음을 선택하고 정책 이름 필드에 `AmazonSesSendingAccess`를 입력한 다음 정책 만들기를 입력합니다.
6. 액세스 관리에서 사용자 그룹을 선택한 다음 그룹 만들기를 선택합니다.
7. 사용자 그룹 이름 필드에 `AWSSESSendingGroupDoNotRename`을 입력합니다.
8. 그룹에 사용자 추가 테이블에서 SMTP 사용자를 선택하여 그룹에 이를 추가합니다.
9. 권한 정책 연결 테이블에서 정책을 선택한 다음 사용자 그룹 만들기를 선택하여 이전에 만든 `AmazonSesSendingAccess` 정책을 연결합니다.

   이제 SES 권한 정책을 사용하여 IAM 사용자 그룹을 만들었으므로 나머지 단계에 설명된 대로 현재 인라인 정책에서 이 그룹 정책으로 SMTP 사용자를 마이그레이션할 수 있습니다.
10. 액세스 관리에서 사용자를 선택한 다음, 마이그레이션하려는 SMTP 사용자를 선택합니다.
11. 그룹 탭을 선택하고 그룹에 사용자 추가를 선택합니다.
12. `AWSSESSendingGroupDoNotRename` 그룹을 선택한 다음 그룹에 사용자 추가를 선택합니다.
13. 권한 탭을 선택하고 정책 이름 열에 `AmazonSesSendingAccess`와 함께 나열된 행 두 개가 있는지 확인합니다. 하나는 인라인 행이고 다른 하나는 연결 방식: 열에 나열된 *`AWSSESSendingGroupDoNotRename`* 그룹입니다.
14. 정책 이름 열에 `AmazonSesSendingAccess`가 포함된 행과 연결 방식: 열의 인라인 행만 선택한 다음, 제거를 선택하고 정책 제거를 확인합니다.

    연결 방식: 열에 *`AWSSESSendingGroupDoNotRename`* 그룹이 있는 행이 남아 있는지 확인합니다.

15. 태그 탭을 선택한 다음 태그 관리를 선택합니다.
16. 키 열의 InvokedBy와 값 열의 SESConsole이 포함된 행 옆에 있는 제거를 선택한 다음 변경 사항 저장을 선택합니다.

> **Important**
>
> `AmazonSesSendingAccess` 정책(인라인 또는 그룹 정책 또는 둘 다)은 SMTP 사용자에게 연결되어 있어야 전송에 영향을 주지 않습니다. 그룹 정책이 사용자에게 연결된 후에만 인라인 정책을 제거합니다.

Amazon SES SMTP 엔드포인트에 연결

Amazon SES SMTP 인터페이스를 사용해 이메일을 전송하려면 SMTP 엔드포인트에 연결합니다. Amazon SES SMTP 엔드포인트의 전체 목록은 AWS 일반 참조의 Amazon Simple Email Service 엔드포인트 및 할당량을 참조하세요.

Amazon SES SMTP 엔드포인트에서는 TLS(전송 계층 보안)를 사용하여 모든 연결을 암호화해야 합니다. (TLS는 종종 이전 프로토콜 이름인 SSL로 불립니다.) Amazon SES는 TLS로 암호화된 연결 설정을 위한 두 가지 메커니즘인 STARTTLS 및 TLS 래퍼를 지원합니다. 사용 중인 소프트웨어의 설명서를 참조하여 소프트웨어가 STARTTLS, TLS 래퍼 또는 둘 다 지원하는지 여부를 확인하세요.

Amazon Elastic Compute Cloud(Amazon EC2)는 기본적으로 포트 25를 통한 이메일 트래픽을 제한합니다. EC2에서 SMTP 엔드포인트를 통해 이메일을 전송할 때 시간 초과를 방지하려면 이메일 전송 제한 제거 요청을 제출하여 제한을 제거합니다. 또는 다른 포트를 사용하여 이메일을 보내거나 Amazon VPC 엔드포인트를 사용할 수 있습니다.

SMTP 연결 문제에 대해서는 SMTP 문제 섹션을 참조하세요.

STARTTLS

STARTTLS는 암호화되지 않은 연결을 암호화된 연결로 업그레이드하는 방법입니다. 다양한 프로토콜을 위한 여러 버전의 STARTTLS이 있으며, SMTP 버전은 RFC 3207에서 정의됩니다.

STARTTLS 연결을 설정하기 위해 SMTP 클라이언트는 포트 25, 587 또는 2587에서 Amazon SES SMTP 엔드포인트에 연결하고, EHLO 명령을 실행한 다음, 서버가 STARTTLS SMTP 확장을 지원한다고 공지할 때까지 대기합니다. 그런 다음 클라이언트는 STARTTLS 명령을 실행하여 TLS 협상을 시작합니다. 협상이 완료되면 클라이언트는 새로운 암호화된 연결을 통해 EHLO 명령을 실행하고 SMTP 세션이 정상적으로 진행됩니다.

TLS 래퍼

TLS 래퍼(SMTPS 또는 핸드셰이크 프로토콜이라고도 함)는 암호화되지 않은 연결을 먼저 설정하지 않고 암호화된 연결을 시작하는 방법입니다. TLS 래퍼에서는 Amazon SES SMTP 엔드포인트가 TLS 협상을 수행하지 않습니다. TLS를 사용하여 엔드포인트와 연결한 후 전체 대화에 걸쳐 TLS를 계속 사용하는 것이 클라이언트의 책임입니다. TLS 래퍼는 더 오래된 프로토콜이지만 많은 클라이언트가 여전히 이 프로토콜을 지원합니다.

TLS 래퍼 연결을 설정하기 위해 SMTP 클라이언트는 포트 465 또는 2465에서 Amazon SES SMTP 엔드포인트에 연결합니다. 서버가 인증서를 제공하고, 클라이언트가 EHLO 명령을 실행한 후, SMTP 세션이 정상적으로 진행됩니다.

소프트웨어 패키지를 사용하여 Amazon SES를 통해 이메일 보내기

SMTP를 통한 이메일 전송을 지원하는 수많은 상업용 및 오픈 소스 소프트웨어 패키지가 있습니다. 여기 몇 가지 예가 있습니다:

- 블로그 플랫폼
- RSS 집계기
- 목록 관리 소프트웨어
- 워크플로우 시스템

Amazon SES SMTP 인터페이스를 통해 이메일을 보내도록 이러한 SMTP 지원 소프트웨어를 구성할 수 있습니다. 특정 소프트웨어 패키지에서 SMTP를 구성하는 방법에 대한 지침은 해당 소프트웨어의 설명서를 참조하세요.

다음 절차에서는 인기 있는 문제 추적 솔루션인 JIRA에서 Amazon SES 전송을 설정하는 방법을 보여줍니다. 이 구성을 사용하면 JIRA는 소프트웨어 문제의 상태가 변경될 때마다 이메일을 통해 사용자에게 알릴 수 있습니다.

Amazon SES를 사용하여 이메일을 보내도록 JIRA를 구성하려면

1. 웹 브라우저를 사용하여 관리자 자격 증명으로 JIRA에 로그인합니다.
2. 브라우저 창에서 [Administration]을 선택합니다.
3. [System] 메뉴에서 [Mail]을 선택합니다.
4. [Mail administration] 페이지에서 [Mail Servers]를 선택합니다.
5. [Configure new SMTP mail server]를 선택합니다.

6. [Add SMTP Mail Server] 양식에서 다음 필드에 값을 입력합니다.

   a. 이름 - 이 서버에 대한 설명이 포함된 이름입니다.

   b. 발신 주소 - 이메일이 전송될 주소입니다. 이 주소에서 전송하려면 먼저 Amazon SES에서 이 이메일 주소를 확인해야 합니다. 확인에 대한 자세한 내용은 Amazon SES에서 확인된 자격 증명을(를) 참조하세요.

   c. 이메일 접두사 - JIRA에서 전송 전에 각 제목 줄 앞에 추가하는 문자열입니다.

   d. 프로토콜 - SMTP를 선택합니다.

   > **Note**
   >
   > 이 설정을 사용하여 Amazon SES에 연결할 수 없는 경우 SECURE_SMTP를 선택해 보세요.

   e. 호스트 이름 - Amazon SES SMTP 엔드포인트 목록은 Amazon SES SMTP 엔드포인트에 연결 섹션을 참조하세요. 예를 들어 미국 서부(오레곤) 리전에서 Amazon SES 엔드포인트를 사용하려는 경우 호스트 이름은 email-smtp.us-west-2.amazonaws.com이 됩니다.

   f. SMTP 포트 - 25, 587 또는 2587(STARTTLS를 사용하여 연결하는 경우) 혹은 465 또는 2465(TLS 래퍼를 사용하여 연결하는 경우)입니다.

   g. TLS - 이 확인란을 선택합니다.

   h. 사용자 이름 - SMTP 사용자 이름입니다.

   i. 암호 - SMTP 암호입니다.

   다음 이미지에서 TLS 래퍼에 대한 설정을 볼 수 있습니다.

7. [Test Connection]을 선택합니다. JIRA에서 Amazon SES를 통해 보내는 테스트 이메일이 성공적으로 도착하면 구성이 완료됩니다.

프로그래밍 방식으로 Amazon SES SMTP 인터페이스를 통해 이메일 전송

Amazon SES SMTP 인터페이스를 사용하여 이메일을 전송하기 위해 SMTP 지원 프로그래밍 언어, 이메일 서버 또는 애플리케이션을 사용할 수 있습니다. 시작하기 전에 Amazon Simple Email Service 설정의 작업을 완료해야 합니다. 또한 다음과 같은 정보가 필요합니다.

- Amazon SES SMTP 엔드포인트와 연결하기 위한 Amazon SES SMTP 보안 인증 정보. Amazon SES SMTP 보안 인증 정보를 받으려면 Amazon SES SMTP 자격 증명 획득 섹션을 참조하세요.

> **Important**
>
> SMTP 자격 증명은 자격 AWS 증명과 다릅니다. 자격 증명에 대한 자세한 내용은 Amazon SES 자격 증명 유형 단원을 참조하세요.

- SMTP 엔드포인트 주소입니다. Amazon SES SMTP 엔드포인트 목록은 Amazon SES SMTP 엔드포인트에 연결 섹션을 참조하세요.

- 연결 방법에 따른 Amazon SES SMTP 인터페이스 포트 번호. 자세한 내용은 Amazon SES SMTP 엔드포인트에 연결 단원을 참조하십시오.

코드 예제

SMTP 지원 프로그래밍 언어를 사용하여 Amazon SES SMTP 인터페이스에 액세스할 수 있습니다. Amazon SES SMTP 호스트 이름 및 포트 번호와 사용자의 SMTP 자격 증명을 입력한 후 프로그래밍 언어의 일반 SMTP 함수를 사용하여 이메일을 전송합니다.

Amazon Elastic Compute Cloud(Amazon EC2)는 기본적으로 포트 25를 통한 이메일 트래픽을 제한합니다. Amazon EC2의 SMTP 엔드포인트를 통해 이메일을 전송할 때 시간 초과를 방지하려면 이러한 제한을 제거하도록 요청할 수 있습니다. 자세한 내용은 AWS 지식 센터의 Amazon EC2 인스턴스 또는 AWS Lambda 함수에서 포트 25에 대한 제한을 제거하려면 어떻게 해야 합니까?를 참조하세요.

Java 및 PHP에 대해 이 섹션의 코드 예제에서는 포트 587을 사용하여 이 문제를 방지합니다.

> **Note**
>
> 이러한 자습서에서는 수신 여부를 확인할 수 있도록 자신에게 이메일을 발송합니다. 추가적인 실험 또는 로드 테스트는 Amazon SES 메일박스 시뮬레이터를 사용하십시오. 메일박스 시뮬레이터로 전송되는 이메일은 전송 할당량이나 반송 메일 및 수신 거부 발생률에 포함되지 않습니다. 자세한 내용은 수동으로 메일박스 시뮬레이터 사용 단원을 참조하십시오.

프로그래밍 언어를 선택하여 해당 언어에 대한 예제를 봅니다.

> **Warning**
>
> Amazon SES는 정적 자격 증명 사용을 권장하지 않습니다. 소스 코드에서 하드 코딩된 자격 증명을 제거하여 보안 태세를 개선하는 방법을 알아보려면 AWS Secrets Manager 섹션을 참조하세요. 이 자습서는 프로덕션이 아닌 환경에서 Amazon SES SMTP 인터페이스를 테스트하기 위한 목적으로만 제공됩니다.

Java

이 예제에서는 SMTP 인터페이스를 사용하여 Amazon SES 를 통해 이메일을 보내기 위해 Eclipse IDE 및 JavaMail API를 사용합니다.

다음 절차를 수행하기 전에 Amazon Simple Email Service 설정의 작업을 완료합니다.

Java를 사용하여 Amazon SES SMTP 인터페이스를 통해 이메일을 전송하려면

1. 웹 브라우저에서 JavaMail GitHub 페이지로 이동합니다. 자산에서 javax.mail.jar를 선택하여 최신 버전의 JavaMail을 다운로드합니다.

   > **Important**
   >
   > 이 자습서에서는 JavaMail 버전 1.5 이상이 필요합니다. 이 절차는 JavaMail 버전 1.6.1을 사용하여 테스트되었습니다.

2. 웹 브라우저에서 Jakarta Activation GitHub 페이지로 이동하고 JavaBeans Activation Framework 1.2.1 Final Release에서 jakarta.activation.jar를 다운로드합니다.
3. 다음 단계를 수행하여 Eclipse에서 프로젝트를 만듭니다.

   a. Eclipse를 시작합니다.

   b. Eclipse에서, [File], [New], [Java Project]를 차례로 선택합니다.

   c. [Create a Java Project] 대화 상자에서, 프로젝트 이름을 입력하고 [Next]를 선택합니다.

   d. [Java Settings] 대화 상자에서, [Libraries] 탭을 선택합니다.

   e. Classpath를 선택하고 외부 JAR 추가 버튼을 사용하여 두 개의 외부 jar 파일 javax.mail.jar 및 jakarta.activation.jar를 추가합니다.

   f. [Add External JARs]를 선택합니다.

   g. JavaMail을 다운로드한 폴더로 이동합니다. `javax.mail.jar` 파일과 [Open]을 차례대로 선택합니다.

   h. [Java Settings] 대화 상자에서, [Finish]를 선택합니다.

4. Eclipse에서, [Package Explorer] 창에서 프로젝트를 확장합니다.
5. 프로젝트에서 [src] 디렉터리를 마우스 오른쪽 버튼으로 클릭한 후 [New], [Class]를 차례로 선택합니다.
6. [New Java Class] 대화 상자에서, [Name] 필드에 `AmazonSESSample`을 입력한 후 [Finish]를 선택합니다.
7. AmazonSESSample.java의 전체 내용을 다음 코드로 바꿉니다.

```
import java.util.Properties;

import javax.mail.Message;
import javax.mail.Session;
import javax.mail.Transport;
```

```
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

public class AmazonSESSample {

    // Replace sender@example.com with your "From" address.
    // This address must be verified.
    static final String FROM = "sender@example.com";
    static final String FROMNAME = "Sender Name";

    // Replace recipient@example.com with a "To" address. If your account
    // is still in the sandbox, this address must be verified.
    static final String TO = "recipient@example.com";

    // Replace smtp_username with your Amazon SES SMTP user name.
    static final String SMTP_USERNAME = "smtp_username";

    // The name of the Configuration Set to use for this message.
    // If you comment out or remove this variable, you will also need to
    // comment out or remove the header below.
    static final String CONFIGSET = "ConfigSet";

    // Amazon SES SMTP host name. This example uses the US West (Oregon) region.
    // See https://docs.aws.amazon.com/ses/latest/DeveloperGuide/
regions.html#region-endpoints
    // for more information.
    static final String HOST = "email-smtp.us-west-2.amazonaws.com";

    // The port you will connect to on the Amazon SES SMTP endpoint.
    static final int PORT = 587;

    static final String SUBJECT = "Amazon SES test (SMTP interface accessed
 using Java)";

    static final String BODY = String.join(
            System.getProperty("line.separator"),
            "<h1>Amazon SES SMTP Email Test</h1>",
            "<p>This email was sent with Amazon SES using the ",
            "<a href='https://github.com/javaee/javamail'>Javamail Package</a>",
            " for <a href='https://www.java.com'>Java</a>."
        );

    public static void main(String[] args) throws Exception {
```

```
        // Create a Properties object to contain connection configuration
information.
        Properties props = System.getProperties();
        props.put("mail.transport.protocol", "smtp");
        props.put("mail.smtp.port", PORT);
        props.put("mail.smtp.starttls.enable", "true");
        props.put("mail.smtp.auth", "true");

        // Create a Session object to represent a mail session with the
specified properties.
        Session session = Session.getDefaultInstance(props);

        // Create a message with the specified information.
        MimeMessage msg = new MimeMessage(session);
        msg.setFrom(new InternetAddress(FROM,FROMNAME));
        msg.setRecipient(Message.RecipientType.TO, new InternetAddress(TO));
        msg.setSubject(SUBJECT);
        msg.setContent(BODY,"text/html");

        // Add a configuration set header. Comment or delete the
        // next line if you are not using a configuration set
        msg.setHeader("X-SES-CONFIGURATION-SET", CONFIGSET);

        // Create a transport.
        Transport transport = session.getTransport();

        // Get the password
        String SMTP_PASSWORD = fetchSMTPPasswordFromSecureStorage();

        // Send the message.
        try
        {
            System.out.println("Sending...");

            // Connect to Amazon SES using the SMTP username and password you
specified above.
            transport.connect(HOST, SMTP_USERNAME, SMTP_PASSWORD);

            // Send the email.
            transport.sendMessage(msg, msg.getAllRecipients());
            System.out.println("Email sent!");
        }
        catch (Exception ex) {
            System.out.println("The email was not sent.");
```

```
            System.out.println("Error message: " + ex.getMessage());
        }
        finally
        {
            // Close and terminate the connection.
            transport.close();
        }
    }

    static String fetchSMTPPasswordFromSecureStorage() {
        /* IMPLEMENT THIS METHOD */
        // For example, you might fetch it from a secure location or AWS Secrets
 Manager: https://aws.amazon.com/secrets-manager/
    }
}
```

8. AmazonSESSample.java에서 다음 이메일 주소를 자체 값으로 대체합니다.

> **Important**
>
> 이메일 주소는 대/소문자를 구분합니다. 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.

- *sender@example.com* - 'From' 이메일 주소로 대체합니다. 이 프로그램을 실행하기 전에 이 주소를 확인해야 합니다. 자세한 내용은 Amazon SES에서 확인된 자격 증명 단원을 참조하십시오.
- *recipient@example.com* - 'To' 이메일 주소로 대체합니다. 계정이 아직 샌드박스에 있는 경우, 이 주소를 확인해야 계정을 사용할 수 있습니다. 자세한 내용은 프로덕션 액세스 요청 (Amazon SES 샌드박스에서 이동) 단원을 참조하십시오.

9. AmazonSESSample.java에서 다음을 자체 값으로 대체합니다.

- *smtp_username* – SMTP 사용자 이름 자격 증명으로 대체합니다. SMTP 사용자 이름 자격 증명은 식별 가능한 이름이 아니라 문자/숫자를 포함하는 20자 길이 문자열입니다.
- *smtp_password* – `fetchSMTPPasswordFromSecureStorage`를 구현하여 암호를 가져옵니다.

10. (선택 사항) *email-smtp.us-west-2.amazonaws.com* 이외의 AWS 리전 에서 Amazon SES SMTP 엔드포인트를 사용하려면 HOST 변수 값을 사용하려는 엔드포인트로 바꿉니

다. Amazon SES를 사용할 수 있는 리전 목록은 AWS 일반 참조의 Amazon Simple Email Service(Amazon SES)를 참조하세요.

11. (선택 사항) 이 이메일을 전송할 때 구성 세트를 사용하려면 `ConfigSet` 변수 값을 구성 세트의 이름으로 바꿉니다. 구성 세트에 대한 자세한 내용은 Amazon SES에서 구성 세트 사용 단원을 참조하세요.
12. AmazonSESSample.java를 저장합니다.
13. 프로젝트를 빌드하려면 [Project]를 선택한 후 [Build Project]를 선택합니다. (이 옵션이 비활성화되어 있는 경우 자동 빌딩이 활성화된 것일 수 있습니다.)
14. 프로그램을 시작하고 이메일을 전송하려면 [Run]을 선택하고 다시 [Run]을 선택합니다.
15. 출력 결과를 검토합니다. 이메일이 성공적으로 전송되었으면 콘솔에 '이메일이 발송되었습니다!'가 표시됩니다. 완료되지 않은 경우 오류 메시지가 표시됩니다.
16. 수신자 주소의 이메일 클라이언트에 로그인합니다. 보낸 메시지가 도착해 있을 것입니다.

PHP

이 예제에서는 SMTP 인터페이스를 사용하여 Amazon SES를 통해 이메일을 전송하기 위해 PHPMailer 클래스를 사용합니다.

다음 절차를 수행하기 전에 Amazon Simple Email Service 설정의 작업을 완료해야 합니다. Amazon SES 를 설정하는 것 외에도 PHP로 이메일을 보내려면 다음과 같은 사전 요구 사항을 완료해야 합니다.

사전 조건:

- PHP 설치 – PHP는 http://php.net/downloads.php에서 다운로드할 수 있습니다. PHP를 설치한 후, 원하는 모든 명령 프롬프트에서 PHP를 실행할 수 있도록 PHP에 대한 경로를 환경 변수에 추가합니다.
- Composer 종속성 관리자 설치 – Composer 종속성 관리자를 설치한 후 PHPMailer 클래스와 클래스의 종속성을 다운로드하여 설치할 수 있습니다. Composer를 설치하려면 https://getcomposer.org/download의 설치 지침을 따르세요.
- PHPMailer 클래스 설치 – Composer를 설치한 후 다음 명령을 실행하여 다음과 같이 PHPMailer를 설치합니다.

```
path/to/composer require phpmailer/phpmailer
```

위의 명령에서 `path/to/`를 Composer를 설치한 경로로 바꿉니다.

PHP를 사용하여 Amazon SES SMTP 인터페이스를 통해 이메일을 전송하려면

1. amazon-ses-smtp-sample.php라는 파일을 만듭니다. 텍스트 편집기로 파일을 열고 다음 코드를 붙여 넣습니다.

```
<?php

// Import PHPMailer classes into the global namespace
// These must be at the top of your script, not inside a function
use PHPMailer\PHPMailer\PHPMailer;
use PHPMailer\PHPMailer\Exception;

// If necessary, modify the path in the require statement below to refer to the
// location of your Composer autoload.php file.
require 'vendor/autoload.php';

// Replace sender@example.com with your "From" address.
// This address must be verified with Amazon SES.
$sender = 'sender@example.com';
$senderName = 'Sender Name';

// Replace recipient@example.com with a "To" address. If your account
// is still in the sandbox, this address must be verified.
$recipient = 'recipient@example.com';

// Replace smtp_username with your Amazon SES SMTP user name.
$usernameSmtp = 'smtp_username';

// Specify a configuration set. If you do not want to use a configuration
// set, comment or remove the next line.
$configurationSet = 'ConfigSet';

// If you're using Amazon SES in a region other than US West (Oregon),
// replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP
// endpoint in the appropriate region.
$host = 'email-smtp.us-west-2.amazonaws.com';
$port = 587;

// The subject line of the email
$subject = 'Amazon SES test (SMTP interface accessed using PHP)';

// The plain-text body of the email
$bodyText =  "Email Test\r\nThis email was sent through the
```

```
    Amazon SES SMTP interface using the PHPMailer class.";

// The HTML-formatted body of the email
$bodyHtml = '<h1>Email Test</h1>
    <p>This email was sent through the
    <a href="https://aws.amazon.com/ses">Amazon SES</a> SMTP
    interface using the <a href="https://github.com/PHPMailer/PHPMailer">
    PHPMailer</a> class.</p>';

$mail = new PHPMailer(true);

try {
    // Specify the SMTP settings.
    $mail->isSMTP();
    $mail->setFrom($sender, $senderName);
    $mail->Username   = $usernameSmtp;
    $mail->Password   = fetchSMTPPasswordFromSecureStorage();
    $mail->Host       = $host;
    $mail->Port       = $port;
    $mail->SMTPAuth   = true;
    $mail->SMTPSecure = 'tls';
    $mail->addCustomHeader('X-SES-CONFIGURATION-SET', $configurationSet);

    // Specify the message recipients.
    $mail->addAddress($recipient);
    // You can also add CC, BCC, and additional To recipients here.

    // Specify the content of the message.
    $mail->isHTML(true);
    $mail->Subject    = $subject;
    $mail->Body       = $bodyHtml;
    $mail->AltBody    = $bodyText;
    $mail->Send();
    echo "Email sent!" , PHP_EOL;
} catch (phpmailerException $e) {
    echo "An error occurred. {$e->errorMessage()}", PHP_EOL; //Catch errors from
 PHPMailer.
} catch (Exception $e) {
    echo "Email not sent. {$mail->ErrorInfo}", PHP_EOL; //Catch errors from
 Amazon SES.
}
function fetchSMTPPasswordFromSecureStorage() {
/* IMPLEMENT THIS METHOD */
```

```
// For example, you might fetch it from a secure location or AWS Secrets
 Manager: https://aws.amazon.com/secrets-manager/
}

?>
```

2. amazon-ses-smtp-sample.php에서 다음을 자체 값으로 대체합니다.

   - *sender@example.com* – Amazon SES에서 확인한 이메일 주소로 바꿉니다. 자세한 내용은 확인된 자격 증명 단원을 참조하십시오. Amazon SES에서 이메일 주소는 대/소문자를 구분합니다. 입력하는 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.
   - *recipient@example.com* – 수신자의 주소로 바꿉니다. 계정이 아직 샌드박스에 있는 경우, 이 주소를 확인해야 계정을 사용할 수 있습니다. 자세한 내용은 프로덕션 액세스 요청 (Amazon SES 샌드박스에서 이동) 단원을 참조하세요. 입력하는 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.
   - *smtp_username* – SMTP 사용자 이름 자격 증명으로 대체합니다. 이 자격 증명은 Amazon SES 콘솔의 SMTP 설정 페이지에서 확인할 수 있습니다. 이 이름은 AWS 액세스 키 ID와 동일하지 않습니다. SMTP 사용자 이름 자격 증명은 식별 가능한 이름이 아니라 문자/숫자를 포함하는 20자 길이 문자열입니다.
   - *smtp_password* – `fetchSMTPPasswordFromSecureStorage`를 구현하여 암호를 가져옵니다.
   - (선택 사항) *ConfigSet* – 이 이메일을 전송할 때 구성 세트를 사용하려면 이 값을 구성 세트의 이름으로 바꿉니다. 구성 세트에 대한 자세한 내용은 Amazon SES에서 구성 세트 사용 단원을 참조하세요.
   - (선택 사항) *email-smtp.us-west-2.amazonaws.com* – 미국 서부(오리건) 이외의 리전에서 Amazon SES SMTP 엔드포인트를 사용하려면 사용하려는 리전의 Amazon SES SMTP 엔드포인트로 대체합니다. Amazon SES를 사용할 수 AWS 리전 있는의 SMTP 엔드포인트 URLs 목록은의 Amazon Simple Email Service(Amazon SES)를 참조하세요AWS 일반 참조.
3. amazon-ses-smtp-sample.php를 저장합니다.
4. 프로그램을 실행하려면 amazon-ses-smtp-sample.php와 동일한 디렉터리에서 명령 프롬프트를 열고 php amazon-ses-smtp-sample.php를 입력합니다.
5. 출력 결과를 검토합니다. 이메일이 성공적으로 전송되었으면 콘솔에 '이메일이 발송되었습니다!'가 표시됩니다. 완료되지 않은 경우 오류 메시지가 표시됩니다.
6. 수신자 주소의 이메일 클라이언트에 로그인합니다. 보낸 메시지가 도착해 있을 것입니다.

기존 이메일 서버와 Amazon SES 통합

현재 자신의 이메일 서버를 관리하고 있는 경우 Amazon SES SMTP 엔드포인트를 사용하여 모든 발신 이메일을 Amazon SES로 보낼 수 있습니다. 기존 이메일 클라이언트와 애플리케이션을 수정할 필요는 없으며 모든 프로그램에 투명하게 Amazon SES로 전환됩니다.

여러 메일 전송 에이전트(MTA)가 SMTP 릴레이를 통한 이메일 전송을 지원합니다. 이 섹션에서는 Amazon SES SMTP 인터페이스를 사용하여 이메일을 보내도록 몇 가지 인기 있는 MTA를 구성하는 방법에 대해 일반적인 지침을 제공합니다.

Amazon SES SMTP 엔드포인트에서는 TLS(전송 계층 보안)를 사용하여 모든 연결을 암호화해야 합니다.

주제

- Amazon SES와 Postfix 통합
- Amazon SES와 Sendmail 통합
- Microsoft Windows Server IIS SMTP와 Amazon SES 통합

Amazon SES와 Postfix 통합

Postfix는 광범위하게 사용되는 Sendmail Message Transfer Agent(MTA)의 대체 프로그램입니다. Postfix에 대한 자세한 내용은 http://www.postfix.org를 참조하세요. 이 주제의 절차는 Linux, macOS 또는 Unix에 적용됩니다.

> **Note**
>
> Postfix는 타사 애플리케이션이며 Amazon Web Services에서 개발하거나 지원하지 않습니다. 이 섹션의 절차는 정보 제공이 목적이며 사전 통지 없이 변경될 수 있습니다.

사전 조건

이 섹션의 절차를 완료하려면 다음 작업을 수행해야 합니다.

- 시스템에 설치되어 있는 경우 Sendmail 애플리케이션을 제거합니다. 이 단계를 완료하는 절차는 사용하는 운영 체제에 따라 다릅니다.

> **Important**
>
> sendmail에 대한 다음 참조는 Sendmail 애플리케이션과 혼동하지 않도록 Postfix 명령 `sendmail`을 참조합니다.

- Postfix를 설치합니다. 이 단계를 완료하는 절차는 사용하는 운영 체제에 따라 다릅니다.
- SASL 인증 패키지를 설치합니다. 이 단계를 완료하는 절차는 사용하는 운영 체제에 따라 다릅니다. 예를 들어 RedHat 기반 시스템을 사용할 경우 `cyrus-sasl-plain` 패키지를 설치해야 합니다. Debian 또는 Ubuntu 기반 시스템을 사용할 경우에는 `libsasl2-modules` 패키지를 설치해야 합니다.
- 이메일 전송에 사용할 이메일 주소 또는 도메인을 확인합니다. 자세한 내용은 이메일 주소 자격 증명 생성 단원을 참조하십시오.
- 계정이 아직 샌드박스에 있으면 확인된 이메일 주소에만 이메일을 전송할 수 있습니다. 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하세요.

Postfix 구성

Postfix를 사용하여 Amazon SES를 통해 이메일을 전송하도록 메일 서버를 구성하는 다음 절차를 완료하세요.

Postfix를 구성하려면

1. 명령줄 프롬프트에 다음 명령을 입력합니다.

```
sudo postconf -e "relayhost = [email-smtp.us-west-2.amazonaws.com]:587" \
"smtp_sasl_auth_enable = yes" \
"smtp_sasl_security_options = noanonymous" \
"smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd" \
"smtp_use_tls = yes" \
"smtp_tls_security_level = secure" \
"smtp_tls_note_starttls_offer = yes"
```

> **Note**
>
> 미국 서부(오레곤) 이외의 AWS 리전에서 Amazon SES를 사용하는 경우 이전 명령의 *email-smtp.us-west-2.amazonaws.com* 해당 리전의 SMTP 엔드포인트로 바꿉니다. 자세한 내용은 the section called "리전" 단원을 참조하십시오.

2. 텍스트 편집기에서 `/etc/postfix/master.cf` 파일을 엽니다. 다음 항목 검색:

```
-o smtp_fallback_relay=
```

이 항목이 있는 경우 열의 시작 부분에 #(해시) 문자를 붙여 주석을 남깁니다. 파일을 저장하고 닫습니다.

그렇지 않은 경우 이 항목이 없으면 다음 단계로 계속 진행하세요.

3. 텍스트 편집기에서 `/etc/postfix/sasl_passwd` 파일을 엽니다. 파일이 없으면 새로 만듭니다.
4. 다음 열을 `/etc/postfix/sasl_passwd`에 추가합니다.

```
[email-smtp.us-west-2.amazonaws.com]:587 SMTPUSERNAME:SMTPPASSWORD
```

> **Note**
>
> *SMTPUSERNAME* 및 *SMTPPASSWORD*를 해당 SMTP 로그인 보안 인증 정보로 바꿉니다. SMTP 로그인 보안 인증 정보는 AWS 액세스 키 ID 및 비밀 액세스 키와 동일하지 않습니다. 자격 증명에 대한 자세한 내용은 the section called "SMTP 자격 증명 획득" 단원을 참조하세요.
> 미국 서부(오레곤) 이외의 AWS 리전에서 Amazon SES를 사용하는 경우 이전 예제의 *email-smtp.us-west-2.amazonaws.com* 해당 리전의 SMTP 엔드포인트로 바꿉니다. 자세한 내용은 the section called "리전" 단원을 참조하십시오.

`sasl_passwd`를 저장하고 닫습니다.

5. 명령 프롬프트에서 다음 명령을 입력하여 SMTP 자격 증명이 포함된 해시 맵 데이터베이스 파일을 만듭니다.

```
sudo postmap hash:/etc/postfix/sasl_passwd
```

6. (선택 사항) 이전 단계에서 만든 `/etc/postfix/sasl_passwd` 및 `/etc/postfix/sasl_passwd.db` 파일은 암호화되지 않습니다. 이 파일에는 SMTP 자격 증명이 포함되기 때문에 액세스를 제한하기 위해 파일의 소유권 및 권한을 수정하는 것이 좋습니다. 이 파일에 대한 액세스를 제한하려면:

   a. 명령 프롬프트에서 다음 명령을 입력하여 파일의 소유권을 변경합니다.

   ```
   sudo chown root:root /etc/postfix/sasl_passwd /etc/postfix/sasl_passwd.db
   ```

   b. 명령 프롬프트에서 다음 명령을 사용하여 루트 사용자만 읽거나 쓸 수 있도록 파일 권한을 변경합니다.

   ```
   sudo chmod 0600 /etc/postfix/sasl_passwd /etc/postfix/sasl_passwd.db
   ```

7. CA 인증서를 찾을 위치를 Postfix에 지정합니다(Amazon SES 서버 인증서 확인 필요). 이 단계에서 사용하는 명령은 운영 체제에 따라 다릅니다.

   - Amazon Linux, Red Hat Enterprise Linux 또는 관련 배포를 사용하는 경우 다음 명령을 입력합니다.

     ```
     sudo postconf -e 'smtp_tls_CAfile = /etc/ssl/certs/ca-bundle.crt'
     ```

   - Ubuntu 또는 관련 배포를 사용하는 경우 다음 명령을 입력합니다.

     ```
     sudo postconf -e 'smtp_tls_CAfile = /etc/ssl/certs/ca-certificates.crt'
     ```

   - macOS를 사용하는 경우 시스템 키 체인에서 인증서를 생성할 수 있습니다. 인증서를 생성하려면 명령줄에 다음 명령을 입력합니다.

     ```
     sudo security find-certificate -a -p /System/Library/Keychains/
     SystemRootCertificates.keychain > /etc/ssl/certs/ca-bundle.crt
     ```

     인증서를 생성한 후에 다음 명령을 입력합니다.

     ```
     sudo postconf -e 'smtp_tls_CAfile = /etc/ssl/certs/ca-bundle.crt'
     ```

8. Postfix 서버를 시작하려면(또는 서버가 이미 실행 중인 경우 구성 설정을 다시 로드하려면) 다음 명령을 입력합니다.

```
sudo postfix start; sudo postfix reload
```

9. 명령줄에 다음을 입력하고 각 줄 이후에 Enter를 눌러 테스트 이메일을 보냅니다. *sender@example.com*을 From 이메일 주소로 바꿉니다. From 주소는 Amazon SES에 사용할 수 있는지 확인해야 합니다. *recipient@example.com*을 대상 주소로 바꿉니다. 계정이 아직 샌드박스 환경에 있을 경우 수신자 주소도 확인해야 합니다. 마지막으로 메시지의 마지막 열에는 다른 콘텐츠가 없는 상태로 하나의 마침표(.)가 포함되어 있어야 합니다.

```
sendmail -f sender@example.com recipient@example.com
From: Sender Name <sender@example.com>
Subject: Amazon SES Test
This message was sent using Amazon SES.
.
```

10. 수신자 주소와 연결된 메일박스를 확인합니다. 이메일이 도착하지 않으면 정크 메일 폴더를 확인합니다. 그래도 이메일을 찾을 수 없으면 자세한 내용은 이메일을 전송하기 위해 사용한 시스템의 메일 로그(일반적으로 `/var/log/maillog`에 있음)를 확인합니다.

고급 사용 예

이 예는 구성 세트를 사용하는 이메일 및 일반 텍스트와 HTML 버전의 메시지를 첨부 파일과 함께 모두 보내기 위해 MIME 멀티파트 인코딩을 사용하는 이메일의 전송 방법을 보여 줍니다. 또한 클릭 이벤트를 분류하는 데 사용할 수 있는 링크 태그도 포함되어 있습니다. 이메일의 콘텐츠는 외부 파일에 지정되므로, Postfix 세션에서 수동으로 명령을 입력할 필요가 없습니다.

Postfix를 사용하여 멀티파트 MIME 이메일을 전송하려면

1. 텍스트 편집기에서 `mime-email.txt`라는 새로운 파일을 만듭니다.
2. 텍스트 파일에서 빨간색으로 표시된 값을 계정의 해당 값으로 바꾸고 다음 콘텐츠를 붙여 넣습니다.

```
X-SES-CONFIGURATION-SET: ConfigSet
From:Sender Name <sender@example.com>
Subject:Amazon SES Test
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="YWVhZDFlY2QzMGQ2N2U0YTZmODU"
```

```
--YWVhZDFlY2QzMGQ2N2U0YTZmODU
Content-Type: multipart/alternative; boundary="3NjM0N2QwMTE4MWQ0ZTg2NTYxZQ"

--3NjM0N2QwMTE4MWQ0ZTg2NTYxZQ
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: quoted-printable

Amazon SES Test

This message was sent from Amazon SES using the SMTP interface.

For more information, see:
http://docs.aws.amazon.com/ses/latest/DeveloperGuide/send-email-smtp.html

--3NjM0N2QwMTE4MWQ0ZTg2NTYxZQ
Content-Type: text/html; charset=UTF-8
Content-Transfer-Encoding: quoted-printable

<html>
  <head>
</head>
  <body>
    <h1>Amazon SES Test</h1>
      <p>This message was sent from Amazon SES using the SMTP interface.</p>
      <p>For more information, see
      <a ses:tags="samplekey0:samplevalue0;samplekey1:samplevalue1;"
      href="http://docs.aws.amazon.com/ses/latest/DeveloperGuide/send-email-
smtp.html">
      Using the Amazon SES SMTP Interface to Send Email</a> in the <em>Amazon SES
      Developer Guide</em>.</p>
  </body>
</html>
--3NjM0N2QwMTE4MWQ0ZTg2NTYxZQ--
--YWVhZDFlY2QzMGQ2N2U0YTZmODU
Content-Type: application/octet-stream
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="customers.txt"

SUQsRmlyc3ROYW1lLExhc3ROYW1lLENvdW50cnkKMzQ4LEpvaG4sU3RpbGVzLENh
bmFkYQo5MjM4OSxKaWUsTGl1LENoaW5hCjczNCxTaGlybGV5LFJvZHJpZ3VleixV
bml0ZWQgU3RhdGVzCjI4OTMsQW5heWEsSXllbmdhcixJbmRpYQ==
--YWVhZDFlY2QzMGQ2N2U0YTZmODU--
```

파일을 저장하고 닫습니다.

3. 명령줄 프롬프트에 다음 명령을 입력합니다. *sender@example.com*을 사용자의 이메일 주소로 바꾸고 *recipient@example.com*을 수신자의 이메일 주소로 바꿉니다.

```
sendmail -f sender@example.com recipient@example.com < mime-email.txt
```

명령이 성공적으로 실행되면 출력을 제공하지 않고 종료됩니다.

4. 이메일의 받은 편지함을 확인합니다. 메시지가 전송되지 않은 경우 시스템의 메일 로그를 확인합니다.

Amazon SES와 Sendmail 통합

Sendmail은 1980년대 초반에 릴리스되었으며 이후 지속적으로 개선되었습니다. 대규모 사용자 커뮤니티를 보유한 유연하고 구성 가능한 메시지 전송 에이전트(MTA)입니다. Sendmail은 2013년에 Proofpoint에서 인수했지만, Proofpoint는 Sendmail의 오픈 소스 버전을 계속 제공하고 있습니다. Proofpoint 웹사이트에서, 또는 대다수 Linx 배포판에 포함된 패키지 관리자를 통해 Sendmail의 오픈 소스 버전을 다운로드하세요.

이 섹션의 절차는 Amazon SES를 통해 이메일을 전송하도록 Sendmail을 구성하는 방법을 보여줍니다. 이 절차는 Ubuntu 18.04.2 LTS를 실행하는 서버 상에서 테스트를 거쳤습니다.

> **Note**
>
> Sendmail은 타사 애플리케이션이며 Amazon Web Services에서 개발하거나 지원하지 않습니다. 이 섹션의 절차는 정보 제공이 목적이며 사전 통지 없이 변경될 수 있습니다.

사전 조건

이 섹션의 절차를 완료하려면 먼저 다음 작업을 수행해야 합니다.

- 서버에 Sendmail 패키지를 설치합니다.

> **Note**
>
> 사용하는 운영 체제 배포판에 따라 `sendmail-cf`, `m4`, `cyrus-sasl-plain` 중 해당되는 패키지를 설치해야 할 수도 있습니다.

- 발신 주소로 사용할 ID를 확인합니다. 자세한 내용은 이메일 주소 자격 증명 생성 단원을 참조하십시오.

  계정이 Amazon SES 샌드박스에 있는 경우 이메일을 전송하려는 주소들도 확인해야 합니다. 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하십시오.

Amazon SES를 사용하여 Amazon EC2 인스턴스로부터 이메일을 전송 중인 경우에는 다음 단계도 완료해야 합니다.

- 이메일을 수신하는 공급자가 이메일을 수락하도록 하려면 Amazon EC2 인스턴스에 탄력적 IP 주소를 할당해야 합니다. 자세한 내용은 Amazon EC2 사용 설명서에서 Amazon EC2 탄력적 IP 주소를 참조하세요.
- Amazon Elastic Compute Cloud(Amazon EC2)는 기본적으로 포트 25를 통한 이메일 트래픽을 제한합니다. Amazon EC2의 SMTP 엔드포인트를 통해 이메일을 전송할 때 시간 초과를 방지하려면 이러한 제한을 제거하도록 요청할 수 있습니다. 자세한 내용은 AWS 지식 센터의 Amazon EC2 인스턴스 또는 AWS Lambda 함수에서 포트 25에 대한 제한을 제거하려면 어떻게 해야 합니까?를 참조하세요.

  또는 이 섹션의 절차를 수정하여 포트 25가 아닌 포트 587을 사용할 수 있습니다.

Sendmail 구성

Amazon SES를 사용해 이메일을 전송하도록 Sendmail을 구성하려면 이 섹션에 나온 단계를 완료하세요.

> **Important**
>
> 이 섹션의 절차에서는 미국 서부(오레곤)에서 Amazon SES를 사용한다고 가정합니다 AWS 리전. 다른 리전을 사용하려는 경우 이 절차에 있는 email-smtp.us-west-2.amazonaws.com의 모든 인스턴스를 원하는 리전의 SMTP 엔드포인트로 바꾸세요. Amazon SES를 사용할 수 있는 AWS 리전 의 SMTP 엔드포인트 URL 목록은 AWS 일반 참조의 Amazon Simple Email Service(Amazon SES)를 참조하세요.

Sendmail을 설정하려면

1. 텍스트 편집기에서 `/etc/mail/authinfo` 파일을 엽니다. 파일이 없으면 새로 만듭니다.

다음 줄을 /etc/mail/authinfo에 추가합니다.

```
AuthInfo:email-smtp.us-west-2.amazonaws.com "U:root" "I:smtpUsername"
 "P:smtpPassword" "M:PLAIN"
```

이전 예제에서 다음과 같이 변경합니다.

- *email-smtp.us-west-2.amazonaws.com*을 사용하려는 Amazon SES SMTP 엔드포인트로 바꿉니다.
- *smtpUsername*을 Amazon SES SMTP 사용자 이름으로 바꿉니다.
- *smtpPassword*를 Amazon SES SMTP 암호로 바꿉니다.

> **Note**
>
> SMTP 로그인 자격 증명은 AWS 액세스 키 ID 및 보안 액세스 키와 다릅니다. SMTP 로그인 보안 인증 정보를 받는 방법에 대한 자세한 내용은 Amazon SES SMTP 자격 증명 획득 섹션을 참조하세요.

완료되면 `authinfo`를 저장합니다.

2. 명령줄에 다음 명령을 입력하여 `/etc/mail/authinfo.db` 파일을 만듭니다.

```
sudo sh -c 'makemap hash /etc/mail/authinfo.db < /etc/mail/authinfo'
```

3. 명령줄에 다음 명령을 입력하여 Amazon SES SMTP 엔드포인트에 릴레이에 대한 지원을 추가합니다.

```
sudo sh -c 'echo "Connect:email-smtp.us-west-2.amazonaws.com RELAY" >> /etc/mail/
access'
```

위의 명령에서 *email-smtp.us-west-2.amazonaws.com*을 사용하려는 Amazon SES SMTP 엔드포인트의 주소로 바꿉니다.

4. 명령 줄에 다음 명령을 입력하여 /etc/mail/access.db를 다시 생성합니다.

```
sudo sh -c 'makemap hash /etc/mail/access.db < /etc/mail/access'
```

5. 명령줄에 다음 명령을 입력하여 다음과 같이 `sendmail.cf` 및 `sendmail.mc` 파일의 백업을 생성합니다.

```
sudo sh -c 'cp /etc/mail/sendmail.cf /etc/mail/sendmail_cf.backup && cp /etc/mail/
sendmail.mc /etc/mail/sendmail_mc.backup'
```

6. 다음 줄을 /etc/mail/sendmail.mc 파일의 `MAILER()` 정의 앞에 추가합니다.

```
define(`SMART_HOST', `email-smtp.us-west-2.amazonaws.com')dnl
define(`RELAY_MAILER_ARGS', `TCP $h 25')dnl
define(`confAUTH_MECHANISMS', `LOGIN PLAIN')dnl
FEATURE(`authinfo', `hash -o /etc/mail/authinfo.db')dnl
MASQUERADE_AS(`example.com')dnl
FEATURE(masquerade_envelope)dnl
FEATURE(masquerade_entire_domain)dnl
```

위의 텍스트에서 다음을 수행합니다.

- *`email-smtp.us-west-2.amazonaws.com`*을 사용하려는 Amazon SES SMTP 엔드포인트로 바꿉니다.
- *`example.com`*을 이메일 전송에 사용할 도메인으로 바꿉니다.

작업을 마치고 파일을 저장합니다.

> **Note**
>
> Amazon EC2는 기본적으로 포트 25를 통한 통신을 제한합니다. Amazon EC2 인스턴스에서 Sendmail을 사용 중인 경우 이메일 전송 제한 제거 요청을 작성해야 합니다.

7. 명령줄에 다음 명령을 입력하여 sendmail.cf를 쓰기 가능한 상태로 만듭니다.

```
sudo chmod 666 /etc/mail/sendmail.cf
```

8. 명령줄에 다음 명령을 입력하여 sendmail.cf를 다시 생성합니다.

```
sudo sh -c 'm4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf'
```

> **Note**
>
> '명령을 찾을 수 없음' 및 '해당 파일 또는 디렉터리가 없음' 오류가 발생하면 `m4` 및 `sendmail-cf` 패키지가 시스템에 설치되어 있는지 확인합니다.

9. 명령줄에 다음 명령을 입력하여 sendmail.cf 권한을 읽기 전용으로 재설정합니다.

```
sudo chmod 644 /etc/mail/sendmail.cf
```

10. 명령줄에 다음 명령을 입력하여 Sendmail을 다시 시작합니다.

```
sudo /etc/init.d/sendmail restart
```

Linux 또는 Sendmail 버전에 따라 위의 내용이 작동하지 않으면 다음을 시도하세요.

```
sudo su service sendmail restart
```

11. 테스트 이메일을 전송하려면 다음 작업을 완료합니다.

    a. 명령줄에 다음 명령을 입력합니다.

    ```
    /usr/sbin/sendmail -vf sender@example.com recipient@example.com
    ```

    *sender@example.com*을 From 이메일 주소로 바꿉니다. *recipient@example.com*을 수신 주소로 바꿉니다. 완료되면 Enter를 누릅니다.

    b. 다음과 같은 메시지 콘텐츠를 입력합니다. 줄이 끝날 때마다 Enter를 누릅니다.

    ```
    From: sender@example.com
    To: recipient@example.com
    Subject: Amazon SES test email

    This is a test message sent from Amazon SES using Sendmail.
    ```

    이메일 콘텐츠의 입력이 완료되면 Ctrl+D를 눌러서 전송합니다.

12. 해당 이메일의 수신자 이메일 클라이언트를 확인합니다. 이메일을 찾을 수 없으면 스팸 메일 폴더를 확인합니다. 그래도 이메일을 찾을 수 없는 경우, 메일 서버의 Sendmail 로그를 살펴봅니다. 로그는 대개 /var/log/mail.log 또는 /var/log/maillog에 있습니다.

Microsoft Windows Server IIS SMTP와 Amazon SES 통합

Amazon SES를 통해 이메일을 보내도록 Microsoft Windows Server의 IIS SMTP 서버를 구성할 수 있습니다. 이 지침은 Amazon EC2 인스턴스에서 Microsoft Windows Server 2022를 사용하여 작성되었습니다. Microsoft Windows Server 2016에서 동일한 구성을 사용할 수 있습니다.

> **Note**
>
> Windows Server는 타사 애플리케이션이며 Amazon Web Services에서 개발하거나 지원하지 않습니다. 이 섹션의 절차는 정보 제공이 목적이며 사전 통지 없이 변경될 수 있습니다.

Microsoft Windows Server IIS SMTP 서버를 Amazon SES와 통합하려면

1. 먼저 다음 지침에 따라 Microsoft Windows Server 2022를 설정합니다.
   a. Amazon EC2 관리 콘솔에서 새 Microsoft Windows Server 2022 Base Amazon EC2 인스턴스를 시작합니다.
   b. 인스턴스에 연결하고 Amazon EC2 Windows 인스턴스 시작하기의 지침에 따라 원격 데스크톱을 사용하여 인스턴스에 로그인합니다.
   c. 서버 관리자 대시보드를 시작합니다.
   d. [웹 서버] 역할을 설치합니다. IIS 10 관리 호환성 도구(웹 서버 확인란 아래의 옵션)를 포함해야 합니다.
   e. [SMTP 서버] 기능을 설치합니다.
2. 이제 다음 지침에 따라 IIS SMTP 서비스를 구성합니다.
   a. 서버 관리자 대시보드로 돌아갑니다.
   b. 도구 메뉴에서 인터넷 정보 서비스(IIS) 10.0 관리자를 선택합니다.
   c. [SMTP 가상 서버 #1]을 마우스 오른쪽 버튼으로 클릭하고 [속성]을 선택합니다.
   d. [액세스] 탭에서 [릴레이 제한] 아래 있는 [릴레이]를 선택합니다.
   e. [릴레이 제한] 대화 상자에서 [추가]를 선택합니다.
   f. [단일 컴퓨터] 아래에서 IP 주소에 127.0.0.1을 입력합니다. 이제 이 서버에 IIS SMTP 서비스를 통해 Amazon SES로 이메일을 릴레이하기 위한 액세스를 부여했습니다.

      이 절차에서는 이메일이 이 서버에서 생성된다고 가정합니다. 이메일을 생성하는 애플리케이션이 별도의 서버에서 실행되는 경우 IIS SMTP에서 해당 서버에 릴레이 액세스를 부여해야 합니다.

> **Note**
>
> SMTP 릴레이를 프라이빗 서브넷으로 확장하려면 [릴레이 제한]에 대해 [단일 컴퓨터] 127.0.0.1 및 [컴퓨터 그룹] 172.1.1.0 - 255.255.255.0(넷마스크 섹션에서)을 사용합니다. [연결]에 대해 [단일 컴퓨터] 127.0.0.1 및 [컴퓨터 그룹] 172.1.1.0 - 255.255.255.0(넷마스크 섹션에서)을 사용합니다.

3. 마지막으로, 다음 지침에 따라 Amazon SES를 통해 이메일을 보내도록 서버를 구성합니다.

   a. [SMTP 가상 서버 #1 속성] 대화 상자로 돌아간 다음 [배달] 탭을 선택합니다.

   b. [배달] 탭에서 [아웃바운드 보안]을 선택합니다.

   c. Basic Authentication(기본 인증)을 선택한 후 Amazon SES SMTP 보안 인증 정보를 입력합니다. Amazon SES SMTP 자격 증명 획득의 절차에 따라 Amazon SES 콘솔에서 이 자격 증명을 받을 수 있습니다.

   > **Important**
   >
   > SMTP 자격 증명은 AWS 액세스 키 ID 및 보안 액세스 키와 동일하지 않습니다. 자격 AWS 증명을 사용하여 SMTP 엔드포인트에 대해 자신을 인증하려고 하지 마십시오. 자격 증명에 대한 자세한 내용은 Amazon SES 자격 증명 유형 단원을 참조하세요.

   d. [TLS 암호화]가 선택되어 있는지 확인합니다.

   e. [배달] 탭으로 돌아갑니다.

   f. [아웃바운드 연결]을 선택합니다.

   g. [아웃바운드 연결] 대화 상자에서 포트가 25 또는 587인지 확인합니다.

   h. 고급을 선택합니다.

   i. 스마트 호스트 이름에 사용할 Amazon SES 엔드포인트(예:email-smtp.us-west-2.amazonaws.com)를 입력합니다. Amazon SES를 사용할 수 AWS 리전 있는의 엔드포인트 URLs 목록은의 Amazon Simple Email Service(Amazon SES)를 참조하세요AWS 일반 참조.

   j. 서버 관리자 대시보드로 돌아갑니다.

   k. 서버 관리자 대시보드에서 [SMTP 가상 서버 #1]을 마우스 오른쪽 버튼으로 클릭한 다음 서버를 다시 시작하여 새 구성을 적용합니다.

   l. 이 서버를 통해 이메일을 보냅니다. 메시지 헤더를 조사하여 Amazon SES를 통해 배달되었는지 확인할 수 있습니다.

명령줄을 사용하여 Amazon SES SMTP 인터페이스에 대한 연결 테스트

이 섹션에서 설명하는 방법을 사용하여 명령줄에서 Amazon SES SMTP 엔드포인트에 대한 연결을 테스트하고, SMTP 자격 증명을 확인하고, 연결 문제를 해결할 수 있습니다. 이러한 절차에서는 가장 일반적인 운영 체제에 포함된 도구 및 라이브러리를 사용합니다.

SMTP 연결 문제 해결에 대한 자세한 내용은 Amazon SES SMTP 문제 섹션을 참조하세요.

사전 조건

Amazon SES SMTP 인터페이스에 연결할 때 일련의 SMTP 자격 증명을 제공해야 합니다. 이러한 SMTP 자격 증명은 표준 AWS 자격 증명과 다릅니다. 두 유형의 자격 증명을 서로 바꿔 사용할 수 없습니다. SMTP 자격 증명을 받는 방법에 대한 자세한 내용은 the section called “SMTP 자격 증명 획득” 섹션을 참조하세요.

Amazon SES SMTP 인터페이스와의 연결 테스트

명령줄을 사용하여 메시지를 인증하거나 보내지 않고 Amazon SES SMTP 인터페이스와의 연결을 테스트할 수 있습니다. 기본적인 연결 문제를 해결하는 데 이 절차가 유용합니다. 테스트 연결에 실패하는 경우 SMTP 문제 섹션을 참조하세요.

이 섹션에는 OpenSSL(대부분의 Linux, macOS 및 Unix 배포에 포함되고 Windows에서도 사용 가능함)과 PowerShell의 `Test-NetConnection` cmdlet(대부분의 최신 Windows 버전에 포함됨)을 둘 다 사용하여 연결을 테스트하는 절차가 나와 있습니다.

Linux, macOS, or Unix

Amazon SES SMTP 인터페이스를 OpenSSL과 연결하는 두 가지 방법이 있습니다. 포트 587을 통해 명시적 SSL을 사용하거나 포트 465를 통해 암시적 SSL을 사용하는 것입니다.

명시적 SSL을 사용하여 SMTP 인터페이스에 연결하려면

- 명령줄에 다음 명령을 입력하여 Amazon SES SMTP 서버에 연결합니다.

  ```
  openssl s_client -crlf -quiet -starttls smtp -connect email-smtp.us-
  west-2.amazonaws.com:587
  ```

  앞의 명령에서 *email-smtp.us-west-2.amazonaws.com* AWS 리전의 Amazon SES SMTP 엔드포인트 URL로 바꿉니다. 자세한 내용은 the section called “리전” 단원을 참조하십시오.

연결이 성공하면 다음과 비슷한 출력이 보입니다.

```
depth=2 C = US, O = Amazon, CN = Amazon Root CA 1
verify return:1
depth=1 C = US, O = Amazon, OU = Server CA 1B, CN = Amazon
verify return:1
depth=0 CN = email-smtp.us-west-2.amazonaws.com
verify return:1
250 Ok
```

비활성 시간이 약 10초 이상 지속되면 연결이 자동으로 닫힙니다.

또는 암시적 SSL을 사용하여 포트 465를 통해 SMTP 인터페이스에 연결할 수 있습니다.

암시적 SSL을 사용하여 SMTP 인터페이스에 연결하려면

- 명령줄에 다음 명령을 입력하여 Amazon SES SMTP 서버에 연결합니다.

```
openssl s_client -crlf -quiet -connect email-smtp.us-west-2.amazonaws.com:465
```

앞의 명령에서 *email-smtp.us-west-2.amazonaws.com* AWS 리전에 대한 Amazon SES SMTP 엔드포인트의 URL로 바꿉니다. 자세한 내용은 the section called “리전” 단원을 참조하십시오.

연결이 성공하면 다음과 비슷한 출력이 보입니다.

```
depth=2 C = US, O = Amazon, CN = Amazon Root CA 1
verify return:1
depth=1 C = US, O = Amazon, OU = Server CA 1B, CN = Amazon
verify return:1
depth=0 CN = email-smtp.us-west-2.amazonaws.com
verify return:1
220 email-smtp.amazonaws.com ESMTP SimpleEmailService-d-VCSHDP1YZ
 A1b2C3d4E5f6G7h8I9j0
```

비활성 시간이 약 10초 이상 지속되면 연결이 자동으로 닫힙니다.

PowerShell

PowerShell에서 Test-NetConnection cmdlet을 사용하여 Amazon SES SMTP 서버에 연결할 수 있습니다.

> **Note**
>
> `Test-NetConnection` cmdlet를 통해 컴퓨터가 Amazon SES SMTP 엔드포인트에 연결할 수 있는지 여부를 확인할 수 있습니다. 하지만 컴퓨터가 SMTP 엔드포인트에 대한 암시적 또는 명시적 SSL 연결을 만들 수 있는지 여부는 테스트하지 않습니다. SSL 연결을 테스트하려면 Windows용 OpenSSL을 설치하거나 테스트 이메일을 보낼 수 있습니다.

`Test-NetConnection` cmdlet를 사용하여 SMTP 인터페이스에 연결하려면

- PowerShell에 다음 명령을 입력하여 Amazon SES SMTP 서버에 연결합니다.

```
Test-NetConnection -Port 587 -ComputerName email-smtp.us-west-2.amazonaws.com
```

  앞의 명령에서 *`email-smtp.us-west-2.amazonaws.com`* 리전의 Amazon SES SMTP 엔드포인트 URL로 바꾸고 AWS *`587`*을 포트 번호로 바꿉니다. Amazon SES의 리전 엔드포인트에 대한 자세한 내용은 the section called “리전” 섹션을 참조하세요.

  연결이 성공하면 다음 예제와 유사한 출력이 표시됩니다.

```
ComputerName     : email-smtp.us-west-2.amazonaws.com
RemoteAddress    : 198.51.100.126
RemotePort       : 587
InterfaceAlias   : Ethernet
SourceAddress    : 203.0.113.46
TcpTestSucceeded : True
```

명령줄을 사용하여 Amazon SES SMTP 인터페이스를 통해 이메일 전송

명령줄을 사용하여 Amazon SES SMTP 인터페이스를 통해 메시지를 보낼 수도 있습니다. SMTP 자격 증명을 테스트할 때나, Amazon SES를 사용하여 보내는 메시지를 특정 수신자가 받을 수 있는지 테스트할 때 이 절차가 유용합니다.

Linux, macOS, or Unix

이메일 발신자가 SMTP 서버에 연결하면 클라이언트가 표준 요청 세트를 발행하고 서버가 각 요청에 표준 응답으로 회신합니다. 이러한 일련의 요청 및 응답을 SMTP 대화라고 합니다. OpenSSL을 사용하여 Amazon SES SMTP 서버에 연결하면 서버가 SMTP 대화가 발생할 것을 예상합니다.

OpenSSL을 사용하여 SMTP 인터페이스에 연결하려면 base64 인코딩을 사용하여 SMTP 자격 증명을 인코딩해야 합니다. 이 섹션에는 base64를 사용하여 자격 증명을 인코딩하는 절차가 나와 있습니다.

SMTP 인터페이스를 통해 명령줄에서 이메일을 보내려면

1. 다음을 명령줄에 입력하고 *email-smtp.us-west-2.amazonaws.com*을 AWS 리전의 Amazon SES SMTP 엔드포인트의 URL로 대체합니다. 자세한 내용은 단원을 참조하십시오the section called “리전”.

```
#!/bin/bash

# Prompt user to provide following information
read -p "Configuration set: " CONFIGSET
read -p "Enter SMTP username: " SMTPUsername
read -p "Enter SMTP password: " SMTPPassword
read -p "Sender email address: " MAILFROM
read -p "Receiver email address: " RCPT
read -p "Email subject: " SUBJECT
read -p "Message to send: " DATA

echo

# Encode SMTP username and password using base64
EncodedSMTPUsername=$(echo -n "$SMTPUsername" | openssl enc -base64)
EncodedSMTPPassword=$(echo -n "$SMTPPassword" | openssl enc -base64)

# Construct the email
Email="EHLO example.com
AUTH LOGIN
$EncodedSMTPUsername
$EncodedSMTPPassword
MAIL FROM: $MAILFROM
RCPT TO: $RCPT
DATA
X-SES-CONFIGURATION-SET: $CONFIGSET
From: $MAILFROM
```

```
To: $RCPT
Subject: $SUBJECT

$DATA
.
QUIT"

echo "$Email" | openssl s_client -crlf -quiet -starttls smtp -connect email-smtp.us-west-2.amazonaws.com:587
```

2. 각 변수에 대한 프롬프트에 값을 입력합니다.
3. • 포트 465를 통해 암시적 SSL을 사용하여 전송하려면 다음을 사용합니다.

```
openssl s_client -crlf -quiet -connect email-smtp.us-west-2.amazonaws.com:465
```

Amazon SES에서 메시지를 수락하면 다음 예제와 유사한 출력이 표시됩니다.

```
250 Ok 01010160d7de98d8-21e57d9a-JZho-416c-bbe1-8ebaAexample-000000
```

`250 Ok` 뒤에 나오는 숫자 및 텍스트 문자열이 이메일의 메시지 ID입니다.

> **Note**
> 비활성 시간이 약 10초 이상 지속되면 연결이 자동으로 닫힙니다.

PowerShell

Net.Mail.SmtpClient 클래스를 이용하면 포트 587을 통해 명시적 SSL을 사용하여 이메일을 보낼 수 있습니다.

> **Note**
> 이 `Net.Mail.SmtpClient` 클래스는 공식적으로 사용되지 않으므로 타사 라이브러리를 사용하는 것이 좋습니다. 이 코드는 테스트 목적으로만 사용되며 프로덕션 워크로드에 사용해서는 안 됩니다.

명시적 SSL을 사용하여 PowerShell을 통해 이메일을 보내려면

1. 텍스트 편집기에서 새로운 파일을 생성합니다. 다음 코드를 파일에 붙여넣습니다.

```
function SendEmail($Server, $Port, $Sender, $Recipient, $Subject, $Body) {
    $Credentials = [Net.NetworkCredential](Get-Credential)

    $SMTPClient = New-Object Net.Mail.SmtpClient($Server, $Port)
    $SMTPClient.EnableSsl = $true
    $SMTPClient.Credentials = New-Object
 System.Net.NetworkCredential($Credentials.Username, $Credentials.Password);

    try {
        Write-Output "Sending message..."
        $SMTPClient.Send($Sender, $Recipient, $Subject, $Body)
        Write-Output "Message successfully sent to $($Recipient)"
    } catch [System.Exception] {
        Write-Output "An error occurred:"
        Write-Error $_
    }
}

function SendTestEmail(){
    $Server = "email-smtp.us-west-2.amazonaws.com"
    $Port = 587

    $Subject = "Test email sent from Amazon SES"
    $Body = "This message was sent from Amazon SES using PowerShell (explicit
 SSL, port 587)."

    $Sender = "sender@example.com"
    $Recipient = "recipient@example.com"

    SendEmail $Server $Port $Sender $Recipient $Subject $Body
}

SendTestEmail
```

작업을 마치면 파일 이름을 `SendEmail.ps1`(으)로 저장합니다.

2. 이전 단계에서 생성한 파일을 다음과 같이 변경합니다.

   - *`sender@example.com`*을 메시지 발신자 이메일 주소로 바꿉니다.

- *recipient@example.com*을 메시지 수신자 이메일 주소로 바꿉니다.
- *email-smtp.us-west-2.amazonaws.com*을 AWS 리전의 Amazon SES SMTP 엔드포인트 URL로 바꿉니다. 자세한 내용은 리전 및 Amazon SES 단원을 참조하십시오.

3. PowerShell에 다음 명령을 입력합니다.

```
.\path\to\SendEmail.ps1
```

위의 명령에서 *path\to\SendEmail.ps1*을 1단계에서 생성한 파일의 경로로 바꿉니다.

4. 메시지가 표시되면 SMTP 사용자 이름과 암호를 입력합니다.

또는 System.Web.Mail.SmtpMail 클래스를 이용하면 포트 465를 통해 암시적 SSL을 사용하여 이메일을 보낼 수 있습니다.

> **Note**
>
> 이 `System.Web.Mail.SmtpMail` 클래스는 공식적으로 사용되지 않으므로 타사 라이브러리를 사용하는 것이 좋습니다. 이 코드는 테스트 목적으로만 사용되며 프로덕션 워크로드에 사용해서는 안 됩니다.

암시적 SSL을 사용하여 PowerShell을 통해 이메일을 보내려면

1. 텍스트 편집기에서 새로운 파일을 생성합니다. 다음 코드를 파일에 붙여넣습니다.

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web") > $null

function SendEmail($Server, $Port, $Sender, $Recipient, $Subject, $Body) {
    $Credentials = [Net.NetworkCredential](Get-Credential)

    $mail = New-Object System.Web.Mail.MailMessage
    $mail.Fields.Add("http://schemas.microsoft.com/cdo/configuration/
smtpserver", $Server)
    $mail.Fields.Add("http://schemas.microsoft.com/cdo/configuration/
smtpserverport", $Port)
    $mail.Fields.Add("http://schemas.microsoft.com/cdo/configuration/
smtpusessl", $true)
    $mail.Fields.Add("http://schemas.microsoft.com/cdo/configuration/
sendusername", $Credentials.UserName)
```

```
    $mail.Fields.Add("http://schemas.microsoft.com/cdo/configuration/
sendpassword", $Credentials.Password)
    $mail.Fields.Add("http://schemas.microsoft.com/cdo/configuration/
smtpconnectiontimeout", $timeout / 1000)
    $mail.Fields.Add("http://schemas.microsoft.com/cdo/configuration/sendusing",
 2)
    $mail.Fields.Add("http://schemas.microsoft.com/cdo/configuration/
smtpauthenticate", 1)

    $mail.From = $Sender
    $mail.To = $Recipient
    $mail.Subject = $Subject
    $mail.Body = $Body

    try {
        Write-Output "Sending message..."
        [System.Web.Mail.SmtpMail]::Send($mail)
        Write-Output "Message successfully sent to $($Recipient)"
    } catch [System.Exception] {
        Write-Output "An error occurred:"
        Write-Error $_
    }
}

function SendTestEmail(){
    $Server = "email-smtp.us-west-2.amazonaws.com"
    $Port = 465

    $Subject = "Test email sent from Amazon SES"
    $Body = "This message was sent from Amazon SES using PowerShell (implicit
 SSL, port 465)."

    $Sender = "sender@example.com"
    $Recipient = "recipient@example.com"

    SendEmail $Server $Port $Sender $Recipient $Subject $Body
}

SendTestEmail
```

작업을 마치면 파일 이름을 `SendEmail.ps1`(으)로 저장합니다.

2. 이전 단계에서 생성한 파일을 다음과 같이 변경합니다.

- *sender@example.com*을 메시지 발신자 이메일 주소로 바꿉니다.
- *recipient@example.com*을 메시지 수신자 이메일 주소로 바꿉니다.
- *email-smtp.us-west-2.amazonaws.com*을 AWS 리전의 Amazon SES SMTP 엔드포인트 URL로 바꿉니다. 자세한 내용은 리전 및 Amazon SES 단원을 참조하십시오.

3. PowerShell에 다음 명령을 입력합니다.

```
.\path\to\SendEmail.ps1
```

위의 명령에서 *path\to\SendEmail.ps1*을 1단계에서 생성한 파일의 경로로 바꿉니다.

4. 메시지가 표시되면 SMTP 사용자 이름과 암호를 입력합니다.

Amazon SES API를 사용하여 이메일 보내기

Amazon SES를 통해 프로덕션 이메일을 보내려면 SMTP(Simple Mail Transfer Protocol) 인터페이스 또는 Amazon SES API를 사용할 수 있습니다. SMTP 인터페이스에 대한 자세한 내용은 Amazon SES SMTP 인터페이스를 사용하여 이메일 보내기 섹션을 참조하세요. 이 섹션에서는 API를 사용하여 이메일을 보내는 방법을 설명합니다.

Amazon SES API를 사용하여 이메일을 보낼 때 메시지 내용을 지정하면 Amazon SES에서 MIME 이메일을 수집합니다. 또는 이메일을 직접 수집하여 메시지 내용을 완전히 제어할 수 있습니다. Amazon Simple Email Service API 참조를 참조하세요. Amazon SES를 사용할 수 AWS 리전 있는의 엔드포인트 URLs 목록은의 Amazon Simple Email Service 엔드포인트 및 할당량을 참조하세요AWS 일반 참조.

다음과 같은 방식으로 API를 호출할 수 있습니다.

- 직접 HTTPS 요청 만들기 - 이는 가장 고급 방법이며, 요청의 인증 및 서명을 수동으로 처리한 다음 요청을 수동으로 구성해야 합니다. Amazon SES API에 대한 자세한 내용은 API v2 참조의 환영 페이지를 참조하세요.
- AWS SDK 사용 -AWS SDKs 사용하면 Amazon SES를 비롯한 여러 AWS 서비스의 APIs에 쉽게 액세스할 수 있습니다. SDK를 사용할 경우 인증, 요청 서명, 재시도 로직, 오류 처리 및 기타 하위 수준 기능이 알아서 처리되므로 고객을 만족시킬 애플리케이션 구축에 집중할 수 있습니다.
- 명령줄 인터페이스 사용 - AWS Command Line Interface은(는) Amazon SES용 명령줄 도구입니다. 또한 PowerShell 환경에서 스크립팅하는 사용자를 위해 AWS Tools for PowerShell도 제공합니다.

SDK, AWS Command Line Interface 또는 AWS Tools for PowerShell을 통해 AWS 직접 또는 간접적으로 Amazon SES API에 액세스하는지 여부에 관계없이 Amazon SES API는 이메일 메시지 구성에 대해 원하는 제어 정도에 따라 이메일을 보내는 두 가지 방법을 제공합니다.

- 서식 있음 - Amazon SES에서 적절하게 서식이 지정된 이메일 메시지를 작성하고 전송합니다. 사용자는 "From(발신):" 및 "To(수신):" 주소, 제목, 메시지 본문만 제공하면 됩니다. 나머지는 모두 Amazon SES가 처리합니다. 자세한 내용은 Amazon SES API를 사용하여 서식이 지정된 이메일 보내기 단원을 참조하세요.
- 원시 - 이메일 헤더와 MIME 형식을 직접 지정하여 이메일 메시지를 수동으로 작성하고 전송합니다. 이메일 서식을 직접 설정하는 데 익숙한 경우 원시 인터페이스를 사용하면 메시지 작성을 더 세부적으로 제어할 수 있습니다. 자세한 내용은 Amazon SES API v2를 사용하여 원시 이메일 보내기 단원을 참조하십시오.

내용

- Amazon SES API를 사용하여 서식이 지정된 이메일 보내기
- Amazon SES API v2를 사용하여 원시 이메일 보내기
- 템플릿을 사용하여 Amazon SES API를 통해 맞춤형 이메일 전송
- AWS SDK를 사용하여 Amazon SES를 통해 이메일 전송
- Amazon SES에서 지원하는 콘텐츠 인코딩

Amazon SES API를 사용하여 서식이 지정된 이메일 보내기

를 사용하거나 애플리케이션을 통해 Amazon SES API를 직접 AWS Management Console 호출하거나 AWS SDK, AWS Command Line Interface또는를 통해 간접적으로 형식이 지정된 이메일을 보낼 수 있습니다 AWS Tools for Windows PowerShell.

Amazon SES API는 `SendEmail` 작업을 통해 서식이 지정된 이메일을 작성하고 전송할 수 있습니다. `SendEmail`에는 보낸 사람: 주소, 받는 사람: 주소, 메시지 제목 및 메시지 본문(텍스트, HTML 또는 둘 다)이 필요합니다. 자세한 내용은 SendEmail(API 참조) 또는 SendEmail(API v2 참조)를 참조하세요.

> **Note**
>
> 이메일 주소 문자열은 7비트 ASCII여야 합니다. 주소의 도메인 부분에 유니코드 문자가 포함된 이메일 주소와 메시지를 주고받으려면 퓨니코드를 사용하여 도메인을 인코딩해야 합니다. 자세한 내용은 RFC 3492를 참조하세요.

다양한 프로그래밍 언어를 사용하여 서식이 지정된 메시지를 작성하는 방법에 대한 예제는코드 예제 섹션을 참조하세요.

`SendEmail`을 여러 번 호출할 때 이메일 전송 속도를 향상하는 방법에 대한 팁은 Amazon SES 처리량 증가를 참조하세요.

Amazon SES API v2를 사용하여 원시 이메일 보내기

`raw`로 지정된 콘텐츠 유형으로 Amazon SES API v2 `SendEmail` 작업을 사용하여 원시 이메일 형식으로 수신자에게 사용자 지정 메시지를 보낼 수 있습니다.

이메일 헤더 필드 정보

SMTP(Simple Mail Transfer Protocol)는 이메일 봉투와 몇 가지 해당 파라미터를 정의하여 이메일 메시지를 전송하는 방법을 지정하지만, 메시지 콘텐츠에는 관련되지 않습니다. 그 대신, 인터넷 메시지 형식(RFC 5322)이 메시지 작성 방법을 정의합니다.

인터넷 메시지 형식 지정을 사용할 경우 모든 이메일 메시지는 헤더와 본문으로 구성됩니다. 헤더는 메시지 메타데이터로 구성되며, 본문에는 메시지 자체가 포함됩니다. 이메일 헤더와 본문에 대한 자세한 내용은 Amazon SES의 이메일 형식 섹션을 참조하세요.

원시 이메일 MIME 메시지 구성 사용

SMTP 프로토콜은 원래 7비트 ASCII 문자만 포함된 이메일 메시지를 전송하기 위해 설계되었습니다. 이 사양으로 인해 비 ASCII 텍스트 인코딩(예: 유니코드), 바이너리 콘텐츠 또는 첨부 파일에는 SMTP가 부족합니다. MIME(Multipurpose Internet Mail Extensions) 표준은 SMTP를 사용하여 다른 많은 종류의 콘텐츠를 전송할 수 있도록 하기 위해 개발되었습니다.

MIME 표준은 메시지 본문을 여러 부분으로 분류한 다음 각 부분에 수행할 작업을 지정하는 방식으로 작동합니다. 예를 들어, 이메일 메시지 본문의 한 부분은 일반 텍스트이고 다른 한 부분은 HTML일 수 있습니다. 또한 MIME는 이메일 메시지에 하나 이상의 첨부 파일이 포함될 수 있도록 허용합니다. 메시지 수신자는 이메일 클라이언트에서 첨부 파일을 보거나 첨부 파일을 저장할 수 있습니다.

메시지 헤더와 콘텐츠는 빈 줄로 분리됩니다. 이메일의 각 부분은 각 부분의 시작과 종료를 나타내는 문자열인 경계로 분리됩니다.

다음 예제의 여러 부분으로 구성된 메시지에는 텍스트 및 HTML 부분과 첨부 파일이 포함되어 있습니다. 첨부 파일은 첨부 파일 헤더 바로 아래에 위치해야 하며 대부분 이 예제와 같이 `base64`로 인코딩됩니다.

```
From: "Sender Name" <sender@example.com>
```

```
To: recipient@example.com
Subject: Customer service contact info
Content-Type: multipart/mixed;
    boundary="a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a"

--a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a
Content-Type: multipart/alternative;
    boundary="sub_a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a"

--sub_a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a
Content-Type: text/plain; charset=iso-8859-1
Content-Transfer-Encoding: quoted-printable

Please see the attached file for a list of customers to contact.

--sub_a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a
Content-Type: text/html; charset=iso-8859-1
Content-Transfer-Encoding: quoted-printable

<html>
<head></head>
<body>
<h1>Hello!</h1>
<p>Please see the attached file for a list of customers to contact.</p>
</body>
</html>

--sub_a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a--

--a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a
Content-Type: text/plain; name="customers.txt"
Content-Description: customers.txt
Content-Disposition: attachment;filename="customers.txt";
    creation-date="Sat, 05 Aug 2017 19:35:36 GMT";
Content-Transfer-Encoding: base64

SUQsRmlyc3ROYW1lLExhc3ROYW1lLENvdW50cnkKMzQ4LEpvaG4sU3RpbGVzLENhbmFkYQo5MjM4
OSxKaWUsTGl1LENoaW5hCjczNCxTaGlybGV5LFJvZHJpZ3VleixVbml0ZWQgU3RhdGVzCjI4OTMs
QW5heWEsSXllbmdhcixJbmRpYQ==

--a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a--
```

이 메시지의 콘텐츠 유형은 `multipart/mixed`로, 이는 메시지에 여러 부분(이 예제에서는 본문과 첨부 파일)이 있고 수신 클라이언트가 각 부분을 개별적으로 처리해야 함을 나타냅니다.

본문 섹션 내에서 중첩된 두 번째 부분은 `multipart/alternative` 콘텐츠 유형을 사용합니다. 이 콘텐츠 유형은 각 부분에 동일한 콘텐츠의 대체 버전이 포함되어 있음을 나타냅니다(이 경우에는 텍스트 버전과 HTML 버전). 수신자의 이메일 클라이언트가 HTML 콘텐츠를 표시할 수 있으면 메시지 본문의 HTML 버전이 표시됩니다. 수신자의 이메일 클라이언트가 HTML 콘텐츠를 표시할 수 없으면 메시지 본문의 일반 텍스트 버전이 표시됩니다.

두 메시지 버전에는 첨부 파일도 포함됩니다(이 경우 일부 고객 이름이 포함된 짧은 텍스트 파일).

이 예제에서처럼 다른 부분 내에 MIME 부분을 중첩하는 경우, 중첩된 부분은 상위 부분의 `boundary` 파라미터와 다른 `boundary` 파라미터를 사용해야 합니다. 이러한 경계는 고유한 문자열이어야 합니다. MIME 부분 간에 경계를 정의하려면 하이픈 두 개(--) 뒤에 경계 문자열을 입력합니다 MIME 부분의 마지막에는 경계 문자열의 시작과 끝 모두에 하이픈 두 개를 넣습니다.

> **Note**
>
> 메시지의 MIME 부분은 500개를 초과할 수 없습니다.

MIME 인코딩

이전 시스템과의 호환성을 유지하기 위해 Amazon SES는 RFC 2821에 정의된 대로 SMTP의 7비트 ASCII 제한을 준수합니다. 비 ASCII 문자가 포함된 콘텐츠를 보내려면 해당 문자를 7비트 ASCII 문자를 사용하는 형식으로 인코딩해야 합니다.

이메일 주소

이메일 주소 문자열은 7비트 ASCII여야 합니다. 주소의 도메인 부분에 유니코드 문자가 포함된 이메일 주소와 메시지를 주고받으려면 퓨니코드를 사용하여 도메인을 인코딩해야 합니다. 이메일 주소의 로컬 부분(@ 기호 앞부분)은 물론 "대화명"에도 퓨니코드를 사용할 수 없습니다. '친숙한 사람' 이름에 유니코드 문자를 사용하려면이 섹션에 설명된 대로 MIME 인코딩된 단어 구문을 사용하여 '친숙한 사람' 이름을 인코딩해야 합니다. 퓨니코드에 대한 자세한 내용은 RFC 3492를 참조하세요.

> **Note**
>
> 이 규칙은 메시지 헤더가 아닌 메시지 봉투에 지정하는 이메일 주소에만 적용됩니다. Amazon SES API v2 `SendEmail` 작업을 사용하면 `Source` 및 `Destinations` 파라미터에 지정하는 주소가 봉투 발신자 및 수신자를 각각 정의합니다.

이메일 헤더

메시지 헤더를 인코딩하려면 MIME 인코딩된 단어 구문을 사용하세요. MIME 인코딩된 단어 구문에는 다음 형식이 사용됩니다.

```
=?charset?encoding?encoded-text?=
```

*encoding*의 값은 Q 또는 B가 될 수 있습니다. 인코딩의 값이 Q이면 *encoded-text* 값은 Q 인코딩을 사용해야 합니다. 인코딩의 값이 B이면 *encoded-text* 값은 base64 인코딩을 사용해야 합니다.

예를 들어 이메일의 제목줄에 문자열 “Як ти поживаєш?”을 사용하려면 다음 인코딩 중 하나를 사용할 수 있습니다.

- Q 인코딩

```
=?utf-8?Q?
=D0=AF=D0=BA_=D1=82=D0=B8_=D0=BF=D0=BE=D0=B6=D0=B8=D0=B2=D0=B0=D1=94=D1=88=3F?=
```

- Base64 인코딩

```
=?utf-8?B?0K/QuiDRgtC4INC/0L7QttC40LLQsNGU0Yg/?=
```

Q 인코딩에 대한 자세한 내용은 RFC 2047을 참조하세요. base64 인코딩에 대한 자세한 내용은 RFC 2045를 참조하세요.

메시지 본문

메시지의 본문을 인코딩하려면 QP(Quoted-Printable) 인코딩 또는 base64 인코딩을 사용할 수 있습니다. 그런 다음 `Content-Transfer-Encoding` 헤더를 사용하여 사용한 인코딩 체계를 나타냅니다.

예를 들어 메시지의 본문에 다음 텍스트가 포함되어 있다고 가정해보세요.

१९७२ मे रे टॉमलंसिन ने पहला ई-मेल सेंदश भेजा | रे टॉमलंसिन ने ही स्रव्परथम @ चिह्न का चयन कयिा और इनही को ईमल का आवषिकारक माना जाता है

base64 인코딩을 사용하여 이 텍스트를 인코딩하려면 먼저 다음 헤더를 지정하세요.

```
Content-Transfer-Encoding: base64
```

그 다음, 이메일의 본문 섹션에 base64로 인코딩된 텍스트를 포함시키세요.


```
4KWn4KWv4KWt4KWoIOCkruClhyDgpLDgpYcg4KSf4KWJ4KSu4KSy4KS/4KSC4KS44KSoIOCkqOCl
hyDgpKrgpLngpLLgpL4g4KSILeCkruClh+CksiDgpLjgpILgpKbgpYfgpLYg4KSt4KWH4KSc4KS+
IHwg4KSw4KWHIOCkn+ClieCkruCksuCkv+CkguCkuOCkqCDgpKjgpYcg4KS54KWAIOCkuOCksOCl
jeCkteCkquCljeCksOCkpeCkriBAIOCkmuCkv+CkqOCljeCkuSDgpJXgpL4g4KSa4KSv4KSoIOCk
leCkv+Ckr+CkviDgpJTgpLAg4KSH4KSo4KWN4KS54KWAIOCkleCliyDgpIjgpK7gpYfgpLIg4KSV
4KS+IOCkhuCkteCkv+Ckt+CljeCkleCkvuCksOCklSDgpK7gpL7gpKjgpL4g4KSc4KS+4KSk4KS+
IOCkueCliAo=
```


> **Note**
>
> 경우에 따라 Amazon SES를 사용하여 전송하는 메시지에 8비트 `Content-Transfer-Encoding`을(를) 사용할 수 있습니다. 그러나 Amazon SES가 메시지를 변경해야 하는 경우 (예: 열기 및 클릭 추적을 사용하는 경우) 8비트로 인코딩된 콘텐츠가 수신자의 받은 편지함에 도착할 때 올바르게 표시되지 않을 수 있습니다. 이러한 이유로 7비트 ASCII가 아닌 콘텐츠는 항상 인코딩해야 합니다.

첨부 파일

이메일에 파일을 첨부하려면 base64 인코딩을 사용하여 첨부 파일을 인코딩해야 합니다. 첨부 파일은 일반적으로 다음 헤더를 포함하는 전용 MIME 메시지 부분에 배치됩니다.

- Content-Type – 첨부 파일의 유형입니다. 다음은 일반적인 MIME Content-Type 선언의 예제입니다.
  - 일반 텍스트 파일 – `Content-Type: text/plain; name="sample.txt"`
  - Microsoft Word 문서 – `Content-Type: application/msword; name="document.docx"`
  - JPG 이미지 – `Content-Type: image/jpeg; name="photo.jpeg"`
- Content-Disposition – 수신자의 이메일 클라이언트가 콘텐츠를 처리하는 방법을 지정합니다. 첨부 파일의 경우 이 값은 `Content-Disposition: attachment`입니다.
- Content-Transfer-Encoding – 첨부 파일을 인코딩하는 데 사용된 체계입니다. 첨부 파일의 경우 이 값은 거의 항상 `base64`입니다.
- 인코딩된 첨부 파일 – 실제 첨부 파일을 인코딩하여 예제에서와 같이 첨부 파일 헤더 아래의 본문에 포함해야 합니다.

Amazon SES는 가장 일반적인 파일 유형을 허용합니다. Amazon SES에서 허용하지 않는 파일 유형 목록은 지원되지 않는 SES 연결 유형 섹션을 참조하세요.

Amazon SES API v2를 사용하여 원시 이메일 보내기

Amazon SES API v2는 사용자가 콘텐츠 유형(단순, 원시, 템플릿 형식 중 하나)을 설정하면 사용자가 지정하는 형식으로 이메일 메시지를 작성하고 전송할 수 있는 `SendEmail` 작업을 제공합니다. 전체 설명을 보려면 `SendEmail` 섹션을 참조하세요. 다음 예제에서는 원시 이메일 형식을 사용하여 메시지를 보낼 콘텐츠 유형을 `raw`로 지정합니다.

> **Note**
>
> `SendEmail`을(를) 여러 번 호출할 때 이메일 전송 속도를 향상하는 방법에 대한 팁은 Amazon SES 처리량 증가을(를) 참조하세요.

메시지 본문에는 적절한 헤더 필드 및 메시지 본문 인코딩과 함께 적절하게 서식 설정된 원시 이메일 메시지가 포함되어야 합니다. 애플리케이션 내에서 원시 메시지를 수동으로 작성할 수 있지만, 기존 메일 라이브러리를 사용하여 작성하면 훨씬 더 간편합니다.

Java

다음 코드 예제는 JavaMail 라이브러리와 AWS SDK for Java를 사용하여 원시 이메일을 작성하고 전송하는 방법을 보여줍니다.

```
package com.amazonaws.samples;

import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.nio.ByteBuffer;
import java.util.Properties;

// JavaMail libraries. Download the JavaMail API
// from https://javaee.github.io/javamail/
import javax.activation.DataHandler;
import javax.activation.DataSource;
import javax.activation.FileDataSource;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.Session;
import javax.mail.internet.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeBodyPart;
import javax.mail.internet.MimeMessage;
```

```
import javax.mail.internet.MimeMultipart;

// AWS SDK libraries. Download the AWS SDK for Java // from https://aws.amazon.com/
sdk-for-java
import com.amazonaws.regions.Regions;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailService;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClientBuilder;
import com.amazonaws.services.simpleemail.model.RawMessage;
import com.amazonaws.services.simpleemail.model.SendRawEmailRequest;

public class AmazonSESSample {

 // Replace sender@example.com with your "From" address.
 // This address must be verified with Amazon SES.
 private static String SENDER = "Sender Name <sender@example.com>";

 // Replace recipient@example.com with a "To" address. If your account
 // is still in the sandbox, this address must be verified.
 private static String RECIPIENT = "recipient@example.com";

 // Specify a configuration set. If you do not want to use a configuration
 // set, comment the following variable, and the
 // ConfigurationSetName=CONFIGURATION_SET argument below.
 private static String CONFIGURATION_SET = "ConfigSet";

 // The subject line for the email.
 private static String SUBJECT = "Customer service contact info";

 // The full path to the file that will be attached to the email.
 // If you're using Windows, escape backslashes as shown in this variable.
 private static String ATTACHMENT = "C:\\Users\\sender\\customers-to-contact.xlsx";

 // The email body for recipients with non-HTML email clients.
 private static String BODY_TEXT = "Hello,\r\n"
                                    + "Please see the attached file for a list "
                                    + "of customers to contact.";

 // The HTML body of the email.
 private static String BODY_HTML = "<html>"
                                    + "<head></head>"
                                    + "<body>"
                                    + "<h1>Hello!</h1>"
                                    + "<p>Please see the attached file for a "
                                    + "list of customers to contact.</p>"
```

```
                                        + "</body>"
                                        + "</html>";

   public static void main(String[] args) throws AddressException,
MessagingException, IOException {

    Session session = Session.getDefaultInstance(new Properties());

       // Create a new MimeMessage object.
       MimeMessage message = new MimeMessage(session);

       // Add subject, from and to lines.
       message.setSubject(SUBJECT, "UTF-8");
       message.setFrom(new InternetAddress(SENDER));
       message.setRecipients(Message.RecipientType.TO,
InternetAddress.parse(RECIPIENT));

       // Create a multipart/alternative child container.
       MimeMultipart msg_body = new MimeMultipart("alternative");

       // Create a wrapper for the HTML and text parts.
       MimeBodyPart wrap = new MimeBodyPart();

       // Define the text part.
       MimeBodyPart textPart = new MimeBodyPart();
       textPart.setContent(BODY_TEXT, "text/plain; charset=UTF-8");

       // Define the HTML part.
       MimeBodyPart htmlPart = new MimeBodyPart();
       htmlPart.setContent(BODY_HTML,"text/html; charset=UTF-8");

       // Add the text and HTML parts to the child container.
       msg_body.addBodyPart(textPart);
       msg_body.addBodyPart(htmlPart);

       // Add the child container to the wrapper object.
       wrap.setContent(msg_body);

       // Create a multipart/mixed parent container.
       MimeMultipart msg = new MimeMultipart("mixed");

       // Add the parent container to the message.
       message.setContent(msg);
```

```
        // Add the multipart/alternative part to the message.
        msg.addBodyPart(wrap);

        // Define the attachment
        MimeBodyPart att = new MimeBodyPart();
        DataSource fds = new FileDataSource(ATTACHMENT);
        att.setDataHandler(new DataHandler(fds));
        att.setFileName(fds.getName());

        // Add the attachment to the message.
        msg.addBodyPart(att);

        // Try to send the email.
        try {
            System.out.println("Attempting to send an email through Amazon SES "
                              +"using the AWS SDK for Java...");

            // Instantiate an Amazon SES client, which will make the service
            // call with the supplied AWS credentials.
            AmazonSimpleEmailService client =
                    AmazonSimpleEmailServiceClientBuilder.standard()
                    // Replace US_WEST_2 with the AWS Region you're using for
                    // Amazon SES.
                    .withRegion(Regions.US_WEST_2).build();

            // Print the raw email content on the console
            PrintStream out = System.out;
            message.writeTo(out);

            // Send the email.
            ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
            message.writeTo(outputStream);
            RawMessage rawMessage =
              new RawMessage(ByteBuffer.wrap(outputStream.toByteArray()));

            SendRawEmailRequest rawEmailRequest =
              new SendRawEmailRequest(rawMessage)
                  .withConfigurationSetName(CONFIGURATION_SET);

            client.sendRawEmail(rawEmailRequest);
            System.out.println("Email sent!");
        // Display an error if something goes wrong.
        } catch (Exception ex) {
          System.out.println("Email Failed");
```

```
            System.err.println("Error message: " + ex.getMessage());
            ex.printStackTrace();
        }
    }
}
```

Python

다음 코드 예제는 Python email.mime 패키지와 AWS SDK for Python (Boto)을 사용하여 원시 이메일을 작성하고 전송하는 방법을 보여줍니다.

```
import json
import boto3
from botocore.exceptions import ClientError
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.application import MIMEApplication
import os

def boto3_rawemailv2():
    SENDER = "Sender <sender@example.com>"
    RECIPIENT = "recipient@example.com"
    CONFIGURATION_SET = "ConfigSet"
    AWS_REGION = "us-east-1"
    SUBJECT = "Customer service contact info"
    ATTACHMENT = "path/to/customers-to-contact.xlsx"
    BODY_TEXT = "Hello,\r\nPlease see the attached file for a list of customers to
 contact."

    # The HTML body of the email.
    BODY_HTML = """\
    <html>
    <head/>
    <body>
    <h1>Hello!</h1>
    <p>Please see the attached file for a list of customers to contact.</p>
    </body>
    </html>
    """

    # The character encoding for the email.
    CHARSET = "utf-8"
    msg = MIMEMultipart('mixed')
```

```
    # Add subject, from and to lines.
    msg['Subject'] = SUBJECT
    msg['From'] = SENDER
    msg['To'] = RECIPIENT

    # Create a multipart/alternative child container.
    msg_body = MIMEMultipart('alternative')

    # Encode the text and HTML content and set the character encoding. This step is
    # necessary if you're sending a message with characters outside the ASCII range.
    textpart = MIMEText(BODY_TEXT.encode(CHARSET), 'plain', CHARSET)
    htmlpart = MIMEText(BODY_HTML.encode(CHARSET), 'html', CHARSET)

    # Add the text and HTML parts to the child container.
    msg_body.attach(textpart)
    msg_body.attach(htmlpart)

    # Define the attachment part and encode it using MIMEApplication.
    att = MIMEApplication(open(ATTACHMENT, 'rb').read())

    # Add a header to tell the email client to treat this part as an attachment,
    # and to give the attachment a name.
    att.add_header('Content-
Disposition','attachment',filename=os.path.basename(ATTACHMENT))

    # Attach the multipart/alternative child container to the multipart/mixed
    # parent container.
    msg.attach(msg_body)
    msg.attach(att)

    #changes start from here
    strmsg = str(msg)
    body = bytes (strmsg, 'utf-8')




    client = boto3.client('sesv2')
    response = client.send_email(
    FromEmailAddress=SENDER,
    Destination={
        'ToAddresses': [RECIPIENT]
    },
    Content={
```

```
        'Raw': {
            'Data': body
        }
    }
    )
    print(response)
boto3_rawemailv2 ()
```

템플릿을 사용하여 Amazon SES API를 통해 맞춤형 이메일 전송

Amazon SES에서는 저장된 템플릿을 사용하거나 인라인 템플릿을 사용하여 템플릿화된 이메일을 보낼 수 있습니다.

- 저장된 템플릿 - Amazon SES v2 API의 `CreateEmailTemplate` 작업을 사용하여 SES에서 생성 및 저장된 `Template` 리소스를 나타냅니다. Amazon SES 템플릿에는 작성된 콘텐츠와 일치하는 변수(자리 표시자)가 포함된 이메일의 제목과 본문이 포함되어 있습니다. 저장된 템플릿의 이름과 템플릿의 자리 표시자 변수에 대한 동적 데이터는 `SendEmail` 또는 `SendBulkEmail` v2 API 작업을 호출할 때 제공됩니다.

  저장된 템플릿은 쉽게 재사용할 수 있으며 유사한 유형의 이메일을 보낼 때 시간과 노력을 절약할 수 있습니다. 처음부터 각 이메일을 생성하는 대신 기본 구조와 설계를 한 번만 생성한 다음 템플릿 내의 동적 콘텐츠를 업데이트하면 됩니다.
- 인라인 템플릿 - `Template` 리소스가 사용되지 않고, `SendEmail` 또는 `SendBulkEmail` v2 API 작업을 호출할 때 해당 자리 표시자 변수의 값과 함께 작성된 콘텐츠와 인라인인 변수(자리 표시자)가 포함된 이메일의 제목과 본문이 제공됩니다.

  인라인 템플릿은 SES 계정에서 템플릿 리소스를 관리할 필요가 없으므로 대량 이메일 전송 프로세스를 간소화하고 애플리케이션 로직 내에 템플릿 콘텐츠를 직접 포함할 수 있으므로 통합 프로세스를 간소화합니다. 당 20,000개 템플릿 제한에 포함되지 않습니다 AWS 리전.

저장된 템플릿을 사용할 때는 다음 제한이 적용됩니다.

- 각각 최대 20,000개의 이메일 템플릿을 생성할 수 있습니다 AWS 리전.
- 각 템플릿의 최대 크기는 텍스트 부분과 HTML 부분을 포함하여 500KB입니다.

인라인 템플릿을 사용할 때는 다음 제한이 적용됩니다.

- 각 입력 JSON 파일의 크기는 텍스트 부분과 HTML 부분을 모두 포함하여 최대 1MB일 수 있습니다.

다음은 저장 템플릿과 인라인 템플릿 모두에 적용됩니다.

- 사용할 수 있는 대체 변수 수에는 제한이 없습니다.
- `SendBulkEmail` 작업을 호출할 때마다 최대 50개의 대상 객체로 이메일을 보낼 수 있습니다. `Destination` 객체에는 ToAddresses, CcAddresses 및 BccAddresses. v2 API에 대한 단일 호출에서 연락할 수 있는 대상 수는 계정의 최대 전송 속도에 따라 제한될 수 있습니다. 자세한 내용은 Amazon SES 발신 한도 관리 단원을 참조하십시오.

이 장에는 저장된 템플릿과 인라인 템플릿을 모두 사용하는 예제가 포함된 절차가 포함되어 있습니다.

> **Note**
>
> 이 섹션의 절차는 또한 AWS CLI을(를) 이미 설치하여 구성한 상태를 전제로 설명합니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

(선택 사항) 1부: 렌더링 실패 이벤트 알림 설정

잘못된 맞춤형 콘텐츠가 포함된 이메일을 전송하는 경우 Amazon SES에서 메시지를 허용할 수는 있지만 전달할 수 없게 됩니다. 이러한 이유로 개인 맞춤형 이메일을 보내려는 경우 Amazon SNS를 통해 렌더링 실패 이벤트 알림을 보내도록 SES를 구성해야 합니다. 렌더링 오류 이벤트 알림을 수신할 때 어떤 메시지에 잘못된 콘텐츠가 있는지 식별하고, 문제를 해결한 다음 메시지를 다시 전송할 수 있습니다.

이 섹션의 절차는 선택 사항이지만 강력히 권장됩니다.

렌더링 오류 이벤트 알림을 구성하려면

1. Amazon SNS 주제를 생성합니다. 절차는 Amazon Simple Notification Service 개발자 가이드의 주제 생성을 참조하세요.
2. Amazon SNS 주제를 구독하세요. 예를 들어 렌더링 오류 알림을 이메일로 수신하고자 하는 경우 이메일 엔드포인트(이메일 주소)를 주제에 구독시킵니다.

   자세한 내용은 Amazon Simple Notification Service 개발자 가이드의 주제 구독을 참조하세요.
3. the section called “Amazon SNS 대상 설정”의 절차를 완료하여 구성 세트를 설정해 렌더링 오류 이벤트를 Amazon SNS 주제에 게시합니다.

(선택 사항) 2부: 이메일 템플릿 생성

저장된 템플릿을 사용하려는 경우이 섹션에서는 `CreateEmailTemplate` SES v2 API 작업을 사용하여 템플릿을 생성하는 방법을 보여줍니다. 인라인 템플릿을 사용하려면이 단계를 건너뛸 수 있습니다.

이 절차는 AWS CLI를 이미 설치하여 구성한 상태를 전제로 설명합니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

템플릿을 생성하려면

1. 텍스트 편집기에서 새 파일을 생성하고 필요에 따라 사용자 지정하여 다음 코드를 붙여 넣습니다.

```
{
    "TemplateName": "MyTemplate",
    "TemplateContent": {
        "Subject": "Greetings, {{name}}!",
        "Text": "Dear {{name}},\r\nYour favorite animal is {{favoriteanimal}}.",
        "Html": "<h1>Hello {{name}},</h1><p>Your favorite animal is
 {{favoriteanimal}}.</p>"
    }
}
```

   이 코드는 다음 속성을 포함하고 있습니다.

   - TemplateName - `Template` 리소스의 이름입니다. 이메일을 보낼 때 이 이름을 참조합니다.
   - TemplateContent - 다음 속성을 위한 컨테이너입니다.
     - SubjectPart – 이메일의 제목 줄입니다. 이 속성에는 대체 태그가 포함될 수 있습니다. 이 태그는 `{{tagname}}` 형식을 사용합니다. 이메일을 보낼 때 각 대상에 대해 `tagname`의 값을 지정할 수 있습니다.
     - HtmlPart – 이메일의 HTML 본문입니다. 이 속성에는 대체 태그가 포함될 수 있습니다. 이전 예에는 `{{name}}` 및 `{{favoriteanimal}}`이라는 두 가지 태그가 포함됩니다.
     - TextPart – 이메일의 텍스트 본문입니다. 이메일 클라이언트에 HTML 콘텐츠가 표시되지 않는 수신자는이 버전의 이메일을 볼 수 있습니다. 이 속성에는 대체 태그도 포함될 수 있습니다.

2. 위의 예제를 필요에 맞게 사용자 지정한 다음 파일을 *`mytemplate.json`*으로 저장합니다.
3. 명령줄에 다음 명령을 입력하여 `CreateEmailTemplate` v2 API 작업을 사용하여 새 템플릿을 생성합니다.

```
aws sesv2 create-email-template --cli-input-json file://mytemplate.json
```

3부: 맞춤형 이메일 보내기

다음 두 가지 SES v2 API 작업을 사용하여 저장된 템플릿 또는 인라인 템플릿을 사용하여 이메일을 보낼 수 있습니다.

- `SendEmail` 작업은 사용자 지정 이메일을 단일 대상 객체로 보내는 데 유용합니다. v2 API `Destination` 객체에는 ToAddresses, CcAddresses 및 BccAddresses 속성이 포함될 수 있습니다. 이러한 주소는 어떤 조합으로든 사용할 수 있으며 동일한 이메일을 수신할 하나 이상의 이메일 주소를 포함할 수 있습니다.
- `SendBulkEmail` 작업은 v2 API를 한 번 호출하여 여러 대상 객체에 고유한 이메일을 보내는 데 유용합니다.

이 섹션에서는 AWS CLI 를 사용하여 이러한 두 전송 작업을 모두 사용하여 템플릿이 지정된 이메일을 보내는 방법의 예를 제공합니다.

단일 대상 객체로 템플릿 이메일 전송

`SendEmail` 작업을 사용하여 단일 대상 객체에 정의된 하나 이상의 수신자에게 이메일을 보낼 수 있습니다. `Destination` 객체의 모든 수신자는 동일한 이메일을 수신합니다.

템플릿 이메일을 단일 대상 객체로 보내려면

1. 저장된 템플릿을 사용할지 인라인 템플릿을 사용할지에 따라 텍스트 편집기에 붙여넣을 각 코드 예제를 선택하여 필요에 따라 사용자 지정합니다.

   Stored template code example

   이전 단계인 MyTemplate에서 생성한 템플릿이 `TemplateName` 파라미터 값으로 참조되고 있습니다.

```
{
    "FromEmailAddress": "Mary Major <mary.major@example.com>",
    "Destination": {
        "ToAddresses": [
            "alejandro.rosalez@example.com", "jimmy.jet@example.com"
        ]
```

```
    },
    "Content": {
        "Template": {
            "TemplateName": "MyTemplate",
            "TemplateData": "{ \"name\":\"Alejandro\", \"favoriteanimal\":
 \"alligator\" }"
        }
    },
    "ConfigurationSetName": "ConfigSet"
}
```

이 코드는 다음 속성을 포함하고 있습니다.

- FromEmailAddress - 발신자의 이메일 주소입니다.
- 대상 - ToAddresses, CcAddresses 및 BccAddresses 속성에 정의된 이메일 수신자가 포함된 객체입니다. 이러한 주소는 어떤 조합으로든 사용할 수 있으며 동일한 이메일을 수신할 하나 이상의 이메일 주소를 포함할 수 있습니다.
- TemplateName - 이메일에 적용할 `Template` 리소스의 이름입니다.
- TemplateData – 키-값 페어를 포함하는 이스케이프된 JSON 문자열입니다. 키는와 같이 저장된 템플릿의 `TemplateContent` 속성에 정의된 변수에 해당합니다`{{name}}`. 값은 변수를 대체하는 콘텐츠를 나타냅니다.
- ConfigurationSetName – 이메일 전송 시 사용할 구성 세트의 이름입니다.

> **Note**
>
> 렌더링 오류 이벤트를 Amazon SNS에 게시하도록 구성된 구성 세트를 사용하는 것이 좋습니다. 자세한 내용은 the section called “(선택 사항) 1부: 알림 설정” 단원을 참조하십시오.

Inline template code example

`TemplateContent` 속성(일반적으로 저장된 템플릿에 정의됨)은 인라인 템플릿으로 만드는 `TemplateData` 속성과 함께 인라인으로 정의됩니다.

```
{
    "FromEmailAddress": "Mary Major <mary.major@example.com>",
    "Destination": {
        "ToAddresses": [
```

```
            "alejandro.rosalez@example.com", "jimmy.jet@example.com"
        ]
    },
    "Content": {
        "Template": {
            "TemplateContent": {
                "Subject": "Greetings, {{name}}!",
                "Text": "Dear {{name}},\r\nYour favorite animal is
 {{favoriteanimal}}.",
                "Html": "<h1>Hello {{name}},</h1><p>Your favorite animal is
 {{favoriteanimal}}.</p>"
            },
            "TemplateData": "{ \"name\":\"Alejandro\", \"favoriteanimal\":
 \"alligator\" }"
        }
    },
    "ConfigurationSetName": "ConfigSet"
}
```

이 코드는 다음 속성을 포함하고 있습니다.

- FromEmailAddress - 발신자의 이메일 주소입니다.
- 대상 - ToAddresses, CcAddresses 및 BccAddresses 속성에 정의된 이메일 수신자가 포함된 객체입니다. 이러한 주소는 어떤 조합으로든 사용할 수 있으며 동일한 이메일을 수신할 하나 이상의 이메일 주소를 포함할 수 있습니다.
- TemplateContent - 다음 속성을 위한 컨테이너입니다.
  - SubjectPart – 이메일의 제목 줄입니다. 이 속성에는 대체 태그가 포함될 수 있습니다. 이 태그는 `{{tagname}}` 형식을 사용합니다. 이메일을 보낼 때 각 대상에 대해 `tagname`의 값을 지정할 수 있습니다.
  - HtmlPart – 이메일의 HTML 본문입니다. 이 속성에는 대체 태그가 포함될 수 있습니다. 이전 예에는 `{{name}}` 및 `{{favoriteanimal}}`이라는 두 가지 태그가 포함됩니다.
  - TextPart – 이메일의 텍스트 본문입니다. 이메일 클라이언트에 HTML 콘텐츠가 표시되지 않는 수신자는이 버전의 이메일을 볼 수 있습니다. 이 속성에는 대체 태그도 포함될 수 있습니다.
- TemplateData – 키-값 페어를 포함하는 이스케이프된 JSON 문자열입니다. 키는와 같이이 파일의 `TemplateContent` 속성에 정의된 변수에 해당합니다`{{name}}`. 값은 변수를 대체하는 콘텐츠를 나타냅니다.
- ConfigurationSetName – 이메일 전송 시 사용할 구성 세트의 이름입니다.

> **Note**
>
> 렌더링 오류 이벤트를 Amazon SNS에 게시하도록 구성된 구성 세트를 사용하는 것이 좋습니다. 자세한 내용은 the section called "(선택 사항) 1부: 알림 설정" 단원을 참조하십시오.

2. 위의 예제를 필요에 맞게 사용자 지정한 다음 파일을 *myemail.json*으로 저장합니다.
3. 명령줄에 다음 v2 API 명령을 입력하여 이메일을 보냅니다.

```
aws sesv2 send-email --cli-input-json file://myemail.json
```

여러 대상 객체로 템플릿 이메일 전송

`SendBulkEmail` 작업을 사용하여 SES v2 API를 한 번 호출하여 여러 대상 객체에 이메일을 보낼 수 있습니다. SES는 각 `Destination` 객체의 수신자 또는 수신자에게 고유한 이메일을 보냅니다.

템플릿 이메일을 여러 대상 객체로 보내려면

1. 저장된 템플릿을 사용할지 인라인 템플릿을 사용할지에 따라 텍스트 편집기에 붙여넣을 각 코드 예제를 선택하여 필요에 따라 사용자 지정합니다.

   Stored template code example

   이전 단계인 MyTemplate에서 생성한 템플릿이 `TemplateName` 파라미터 값으로 참조되고 있습니다.

```
{
    "FromEmailAddress": "Mary Major <mary.major@example.com>",
    "DefaultContent": {
        "Template": {
            "TemplateName": "MyTemplate",
            "TemplateData": "{ \"name\":\"friend\", \"favoriteanimal\":\"unknown
\" }"
        }
    },
    "BulkEmailEntries": [
        {
            "Destination": {
                "ToAddresses": [
```

```
                    "anaya.iyengar@example.com"
                ]
            },
            "ReplacementEmailContent": {
                "ReplacementTemplate": {
                    "ReplacementTemplateData": "{ \"name\":\"Anaya\",
\"favoriteanimal\":\"angelfish\" }"
                }
            }
        },
        {
            "Destination": {
                "ToAddresses": [
                    "liu.jie@example.com"
                ]
            },
            "ReplacementEmailContent": {
                "ReplacementTemplate": {
                    "ReplacementTemplateData": "{ \"name\":\"Liu\",
\"favoriteanimal\":\"lion\" }"
                }
            }
        },
        {
            "Destination": {
                "ToAddresses": [
                    "shirley.rodriguez@example.com"
                ]
            },
            "ReplacementEmailContent": {
                "ReplacementTemplate": {
                    "ReplacementTemplateData": "{ \"name\":\"Shirley\",
\"favoriteanimal\":\"shark\" }"
                }
            }
        },
        {
            "Destination": {
                "ToAddresses": [
                    "richard.roe@example.com"
                ]
            },
            "ReplacementEmailContent": {
                "ReplacementTemplate": {
```

```
                    "ReplacementTemplateData": "{}"
                }
            }
        }
    ],
    "ConfigurationSetName": "ConfigSet"
}
```

이 코드는 다음 속성을 포함하고 있습니다.

- FromEmailAddress - 발신자의 이메일 주소입니다.
- DefaultContent - `TemplateName` 및 객체를 포함하는 JSON `TemplateData` 객체입니다.
- TemplateName - 이메일에 적용할 `Template` 리소스의 이름입니다.
- TemplateData - `ReplacementEmailContent` 객체에 `ReplacementTemplateData` 속성`{}`에 빈 JSON 객체가 포함된 경우 사용할 키-값 페어를 포함합니다.
- BulkEmailEntries - 하나 이상의 `Destination` 객체를 포함하는 배열입니다.
- 대상 - ToAddresses, CcAddresses 및 BccAddresses 속성에 정의된 이메일 수신자가 포함된 객체입니다. 이러한 주소는 어떤 조합으로든 사용할 수 있으며 동일한 이메일을 수신할 하나 이상의 이메일 주소를 포함할 수 있습니다.
- ReplacementTemplateData – 키-값 페어가 포함된 이스케이프 처리된 JSON 문자열입니다. 키는 템플릿의 변수에 해당합니다. 예: `{{name}}`. 값은 이메일의 변수를 대체하는 콘텐츠를 나타냅니다. (여기서 JSON 문자열이 로 표시된 비어 있는 경우 `DefaultContent` 객체 내의 `TemplateData` 속성에 정의된 `{}`키-값 페어가 사용됩니다.)
- ConfigurationSetName – 이메일 전송 시 사용할 구성 세트의 이름입니다.

> **Note**
>
> 렌더링 오류 이벤트를 Amazon SNS에 게시하도록 구성된 구성 세트를 사용하는 것이 좋습니다. 자세한 내용은 the section called “(선택 사항) 1부: 알림 설정” 단원을 참조하십시오.

Inline template code example

`TemplateContent` 속성(일반적으로 저장된 템플릿에 정의됨)은 인라인 템플릿으로 만드는 `TemplateData` 속성과 함께 인라인으로 정의됩니다.

```
{
    "FromEmailAddress": "Mary Major <mary.major@example.com>",
    "DefaultContent": {
        "Template": {
            "TemplateContent": {
                "Subject": "Greetings, {{name}}!",
                "Text": "Dear {{name}},\r\nYour favorite animal is
 {{favoriteanimal}}.",
                "Html": "<h1>Hello {{name}},</h1><p>Your favorite animal is
 {{favoriteanimal}}.</p>"
            },
            "TemplateData": "{ \"name\":\"friend\", \"favoriteanimal\":\"unknown
\" }"
        }
    },
    "BulkEmailEntries": [
        {
            "Destination": {
                "ToAddresses": [
                    "anaya.iyengar@example.com"
                ]
            },
            "ReplacementEmailContent": {
                "ReplacementTemplate": {
                    "ReplacementTemplateData": "{ \"name\":\"Anaya\",
 \"favoriteanimal\":\"angelfish\" }"
                }
            }
        },
        {
            "Destination": {
                "ToAddresses": [
                    "liu.jie@example.com"
                ]
            },
            "ReplacementEmailContent": {
                "ReplacementTemplate": {
                    "ReplacementTemplateData": "{ \"name\":\"Liu\",
 \"favoriteanimal\":\"lion\" }"
                }
            }
        },
        {
```

```
            "Destination": {
                "ToAddresses": [
                    "shirley.rodriguez@example.com"
                ]
            },
            "ReplacementEmailContent": {
                "ReplacementTemplate": {
                    "ReplacementTemplateData": "{ \"name\":\"Shirley\",
 \"favoriteanimal\":\"shark\" }"
                }
            }
        },
        {
            "Destination": {
                "ToAddresses": [
                    "richard.roe@example.com"
                ]
            },
            "ReplacementEmailContent": {
                "ReplacementTemplate": {
                    "ReplacementTemplateData": "{}"
                }
            }
        }
    ],
    "ConfigurationSetName": "ConfigSet"
}
```

이 코드는 다음 속성을 포함하고 있습니다.

- FromEmailAddress - 발신자의 이메일 주소입니다.
- DefaultContent - `TemplateContent` 및 객체를 포함하는 JSON `TemplateData` 객체입니다.
- TemplateContent - 다음 속성을 위한 컨테이너입니다.
  - SubjectPart – 이메일의 제목 줄입니다. 이 속성에는 대체 태그가 포함될 수 있습니다. 이 태그는 `{{tagname}}` 형식을 사용합니다. 이메일을 보낼 때 각 대상에 대해 `tagname`의 값을 지정할 수 있습니다.
  - HtmlPart – 이메일의 HTML 본문입니다. 이 속성에는 대체 태그가 포함될 수 있습니다. 이전 예에는 `{{name}}` 및 `{{favoriteanimal}}`이라는 두 가지 태그가 포함됩니다.

  - TextPart – 이메일의 텍스트 본문입니다. 이메일 클라이언트에 HTML 콘텐츠가 표시되지 않는 수신자는이 버전의 이메일을 볼 수 있습니다. 이 속성에는 대체 태그도 포함될 수 있습니다.
- TemplateData - `ReplacementEmailContent` 객체에 `ReplacementTemplateData` 속성`{}`에 빈 JSON 객체가 포함된 경우 사용할 키-값 페어를 포함합니다.
- BulkEmailEntries - 하나 이상의 `Destination` 객체를 포함하는 배열입니다.
- 대상 - ToAddresses, CcAddresses 및 BccAddresses 속성에 정의된 이메일 수신자가 포함된 객체입니다. 이러한 주소는 어떤 조합으로든 사용할 수 있으며 동일한 이메일을 수신할 하나 이상의 이메일 주소를 포함할 수 있습니다.
- ReplacementTemplateData – 키-값 페어가 포함된 이스케이프 처리된 JSON 문자열입니다. 키는와 같이이 파일의 `TemplateContent` 속성에 정의된 변수에 해당합니다`{{name}}`. 값은 이메일의 변수를 대체하는 콘텐츠를 나타냅니다. (여기서 JSON 문자열이 로 표시된 비어 있는 경우 `DefaultContent` 객체 내의 `TemplateData` 속성에 정의된 `{}`키-값 페어가 사용됩니다.)
- ConfigurationSetName – 이메일 전송 시 사용할 구성 세트의 이름입니다.

> **Note**
>
> 렌더링 오류 이벤트를 Amazon SNS에 게시하도록 구성된 구성 세트를 사용하는 것이 좋습니다. 자세한 내용은 the section called “(선택 사항) 1부: 알림 설정” 단원을 참조하십시오.

2. 필요에 맞게 이전 단계의 코드 값을 변경한 다음 파일을 *`mybulkemail.json`*으로 저장합니다.
3. 명령줄에 다음 v2 API 명령을 입력하여 대량 이메일을 보냅니다.

```
aws sesv2 send-bulk-email --cli-input-json file://mybulkemail.json
```

고급 이메일 맞춤 설정

저장된 템플릿을 사용하는 경우, 즉 SES v2 API와 함께 작업을 사용하여 Amazon SES에서 `Template` 리소스를 생성한 경우, 핸들바 시스템을 활용하여 중첩 속성, 배열 반복, 기본 조건문, 인라인 부분 생성과 같은 고급 기능을 포함하는 템플릿을 생성할 수 있습니다. `CreateEmailTemplate` 이 섹션에서는 이러한 기능의 예를 제공합니다.

Handlebars에는 이 섹션에 설명되어 있는 기능 외의 추가 기능이 포함됩니다. 자세한 내용은 handlebarsjs.com에서 Built-In Helpers를 참조하세요.

> **Note**
>
> SES는 메시지의 HTML 템플릿을 렌더링할 때 HTML 콘텐츠를 이스케이프하지 않습니다. 즉, 연락처 양식과 같이 사용자 입력 데이터를 포함하는 경우 클라이언트 측에서 이스케이프해야 합니다.

주제

- 중첩 속성 분석
- 목록 살펴보기
- 기본 조건문 사용
- 인라인 부분 생성

중첩 속성 분석

Handlebars에는 복잡한 고객 데이터를 쉽게 구성할 수 있는 중첩 경로 지원이 포함되고 이메일 템플릿의 그 데이터를 참조합니다.

예를 들어 수신자 데이터를 여러 일반 범주로 조직할 수 있습니다. 각 범주 내에 세부 정보를 포함할 수 있습니다. 다음 코드 예에는 단일 수신자에 대한 이 구조의 예가 나옵니다.

```
{
  "meta":{
    "userId":"51806220607"
  },
  "contact":{
    "firstName":"Anaya",
    "lastName":"Iyengar",
    "city":"Bengaluru",
    "country":"India",
    "postalCode":"560052"
  },
  "subscription":[
    {
      "interest":"Sports"
    },
    {
      "interest":"Travel"
    },
```

```
    {
      "interest":"Cooking"
    }
  ]
}
```

이메일 템플릿에서 상위 속성의 이름 뒤에 마침표(.)와 값을 포함할 속성의 이름을 제공하여 중첩 속성을 참조할 수 있습니다. 예를 들어 앞의 예에 표시된 데이터 구조를 사용하고 이메일 템플릿에 각 수신자의 이름을 포함하려는 경우 이메일 템플릿에 `Hello {{contact.firstName}}!` 텍스트를 포함합니다.

Handlebars는 여러 수준 깊이로 중첩된 경로를 구문 분석할 수 있기 때문에 유연하게 템플릿 데이터를 구성할 수 있습니다.

목록 살펴보기

`each` 보조 도구 함수는 어레이로 항목을 살펴봅니다. 다음 코드는 `each` 헬퍼 함수를 사용하여 항목화된 각 수신자의 관심사 목록을 생성하는 이메일 템플릿의 예입니다.

```
{
  "Template": {
    "TemplateName": "Preferences",
    "SubjectPart": "Subscription Preferences for {{contact.firstName}}
 {{contact.lastName}}",
    "HtmlPart": "<h1>Your Preferences</h1>
                 <p>You have indicated that you are interested in receiving
                   information about the following subjects:</p>
                 <ul>
                   {{#each subscription}}
                     <li>{{interest}}</li>
                   {{/each}}
                 </ul>
                 <p>You can change these settings at any time by visiting
                    the <a href=https://www.example.com/prefererences/i.aspx?
id={{meta.userId}}>
                    Preference Center</a>.</p>",
    "TextPart": "Your Preferences\n\nYou have indicated that you are interested in
                 receiving information about the following subjects:\n
                 {{#each subscription}}
                   - {{interest}}\n
                 {{/each}}
                 \nYou can change these settings at any time by
                 visiting the Preference Center at
```

```
                    https://www.example.com/prefererences/i.aspx?id={{meta.userId}}"
  }
}
```

> **Important**
>
> 앞의 코드 예에는 예를 이해하기 쉽도록 `HtmlPart` 및 `TextPart` 속성의 값에 줄 바꿈이 포함되어 있습니다. 템플릿의 JSON 파일에는 이러한 값 내에 줄바꿈이 포함될 수 없습니다. 이 예를 복사하여 자체 JSON 파일에 붙여 넣으면 진행하기 전에 `HtmlPart` 및 `TextPart` 섹션에서 줄 바꿈 및 추가 공백을 제거합니다.

템플릿을 생성한 후 `SendEmail` 또는 `SendBulkEmail` 작업을 사용하여 수신자에게 이 템플릿으로 이메일을 보낼 수 있습니다. 각 수신자가 `Interests` 객체에 최소 하나의 값이 있는 한 항목화된 관심사 목록이 포함되는 이메일을 수신합니다. 다음 예는 앞의 템플릿을 사용하여 여러 수신자에게 이메일을 보내는 데 사용할 수 있는 JSON 템플릿을 보여 줍니다.

```
{
  "Source":"Sender Name <sender@example.com>",
  "Template":"Preferences",
  "Destinations":[
    {
      "Destination":{
        "ToAddresses":[
          "anaya.iyengar@example.com"
        ]
      },
      "ReplacementTemplateData":"{\"meta\":{\"userId\":\"51806220607\"},\"contact\":
{\"firstName\":\"Anaya\",\"lastName\":\"Iyengar\"},\"subscription\":[{\"interest\":
\"Sports\"},{\"interest\":\"Travel\"},{\"interest\":\"Cooking\"}]}"
      },
    {
      "Destination":{
        "ToAddresses":[
          "shirley.rodriguez@example.com"
        ]
      },
      "ReplacementTemplateData":"{\"meta\":{\"userId\":\"1981624758263\"},\"contact\":
{\"firstName\":\"Shirley\",\"lastName\":\"Rodriguez\"},\"subscription\":[{\"interest\":
\"Technology\"},{\"interest\":\"Politics\"}]}"
    }
```

```
  ],
  "DefaultTemplateData":"{\"meta\":{\"userId\":\"\"},\"contact\":{\"firstName\":
\"Friend\",\"lastName\":\"\"},\"subscription\":[]}"
}
```

SendBulkEmail 작업을 사용하여 앞의 예에 나열된 수신자에게 이메일을 보내면 다음 이미지에 표시된 예와 같은 메시지를 수신합니다.

Your Preferences

Dear Anaya,

You have indicated that you are interested in receiving information about the following subjects:

- Sports
- Travel
- Cooking

You can change these settings at any time by visiting the Preference Center.

기본 조건문 사용

이 섹션은 이전 섹션에서 설명한 예를 기반으로 합니다. 이전 섹션의 예는 each 보조 도구를 사용하여 관심사 목록을 살펴봅니다. 하지만 관심사가 지정되지 않은 수신자는 빈 목록이 포함된 이메일을 수신합니다. {{if}} 보조 도구를 사용하면 특정 속성이 템플릿 데이터에 있는 경우 이메일의 형식을 다르게 할 수 있습니다. 다음 코드는 Subscription 어레이에 값이 포함되는 경우 {{if}} 보조 도구를 사용하여 앞 섹션의 글머리표 목록을 표시합니다. 어레이가 비어 있으면 다른 텍스트 블록이 표시됩니다.

```
{
  "Template": {
    "TemplateName": "Preferences2",
    "SubjectPart": "Subscription Preferences for {{contact.firstName}}
 {{contact.lastName}}",
    "HtmlPart": "<h1>Your Preferences</h1>
                 <p>Dear {{contact.firstName}},</p>
                 {{#if subscription}}
                   <p>You have indicated that you are interested in receiving
                     information about the following subjects:</p>
                     <ul>
                     {{#each subscription}}
                       <li>{{interest}}</li>
```

```
                      {{/each}}
                      </ul>
                      <p>You can change these settings at any time by visiting
                        the <a href=https://www.example.com/prefererences/i.aspx?
id={{meta.userId}}>
                        Preference Center</a>.</p>
                  {{else}}
                    <p>Please update your subscription preferences by visiting
                      the <a href=https://www.example.com/prefererences/i.aspx?
id={{meta.userId}}>
                      Preference Center</a>.
                  {{/if}}",
    "TextPart": "Your Preferences\n\nDear {{contact.firstName}},\n\n
                  {{#if subscription}}
                    You have indicated that you are interested in receiving
                    information about the following subjects:\n
                    {{#each subscription}}
                      - {{interest}}\n
                    {{/each}}
                    \nYou can change these settings at any time by visiting the
                    Preference Center at https://www.example.com/prefererences/i.aspx?
id={{meta.userId}}.
                  {{else}}
                    Please update your subscription preferences by visiting the
                    Preference Center at https://www.example.com/prefererences/i.aspx?
id={{meta.userId}}.
                  {{/if}}"
  }
}
```

> **Important**
>
> 앞의 코드 예에는 예를 이해하기 쉽도록 `HtmlPart` 및 `TextPart` 속성의 값에 줄 바꿈이 포함되어 있습니다. 템플릿의 JSON 파일에는 이러한 값 내에 줄바꿈이 포함될 수 없습니다. 이 예를 복사하여 자체 JSON 파일에 붙여 넣으면 진행하기 전에 `HtmlPart` 및 `TextPart` 섹션에서 줄 바꿈 및 추가 공백을 제거합니다.

다음 예는 앞의 템플릿을 사용하여 여러 수신자에게 이메일을 보내는 데 사용할 수 있는 JSON 템플릿을 보여 줍니다.

```
{
```

```
  "Source":"Sender Name <sender@example.com>",
  "Template":"Preferences2",
  "Destinations":[
    {
      "Destination":{
        "ToAddresses":[
          "anaya.iyengar@example.com"
        ]
      },
      "ReplacementTemplateData":"{\"meta\":{\"userId\":\"51806220607\"},\"contact\":
{\"firstName\":\"Anaya\",\"lastName\":\"Iyengar\"},\"subscription\":[{\"interest\":
\"Sports\"},{\"interest\":\"Cooking\"}]}"
      },
    {
      "Destination":{
        "ToAddresses":[
          "shirley.rodriguez@example.com"
        ]
      },
      "ReplacementTemplateData":"{\"meta\":{\"userId\":\"1981624758263\"},\"contact\":
{\"firstName\":\"Shirley\",\"lastName\":\"Rodriguez\"}}"
    }
  ],
  "DefaultTemplateData":"{\"meta\":{\"userId\":\"\"},\"contact\":{\"firstName\":
\"Friend\",\"lastName\":\"\"},\"subscription\":[]}"
}
```

이 예에서는 템플릿 데이터에 관심사 목록이 포함된 수신자가 이전 섹션에 표시된 예와 같은 이메일을 수신합니다. 하지만 템플릿 데이터에 관심사가 포함되지 않은 수신자는 다음 이미지에 표시된 예와 같은 이메일을 수신합니다.

인라인 부분 생성

인라인 부분을 사용하여 반복된 문자열이 포함된 템플릿을 간소화할 수 있습니다. 예를 들어 사용 가능한 경우 템플릿의 첫 부분에 다음 코드를 추가하여 수신자의 이름 및 사용 가능한 경우 성이 포함된 인라인 부분을 만들 수 있습니다.

```
{{#* inline \"fullName\"}}{{firstName}}{{#if lastName}} {{lastName}}{{/if}}{{/
inline}}\n
```

> **Note**
>
> 줄바꿈 문서(\n)는 템플릿의 콘텐츠에서 {{inline}} 블록을 분리하는 데 필요합니다. 줄바꿈은 최종 출력에 렌더링되지 않습니다.

fullName 부분을 생성한 후 {{> fullName}}의 예와 같이 부분의 이름 앞에 이상(>) 기호를 표시하고 공백을 표시하여 템플릿의 어느 부분에든 포함할 수 있습니다. 인라인 부분은 이메일 부분 사이에 전송되지 않습니다. 예를 들어 이메일의 HTML과 텍스트 버전 모두에 동일한 인라인 부분을 사용하려는 경우 HtmlPart 및 TextPart 섹션 모두에서 정의해야 합니다.

어레이를 통해 살펴보는 경우 인라인 부분을 사용할 수도 있습니다. 다음 코드를 사용하여 fullName 인라인 부분을 사용한 템플릿을 생성할 수 있습니다. 이 예에서는 인라인 부분이 수신자의 이름과 다른 이름의 어레이 모두에 적용됩니다.

```
{
  "Template": {
    "TemplateName": "Preferences3",
    "SubjectPart": "{{firstName}}'s Subscription Preferences",
    "HtmlPart": "{{#* inline \"fullName\"}}
                  {{firstName}}{{#if lastName}} {{lastName}}{{/if}}
                {{/inline~}}\n
                <h1>Hello {{> fullName}}!</h1>
                <p>You have listed the following people as your friends:</p>
                <ul>
                {{#each friends}}
                  <li>{{> fullName}}</li>
                {{/each}}</ul>",
    "TextPart": "{{#* inline \"fullName\"}}
                  {{firstName}}{{#if lastName}} {{lastName}}{{/if}}
                {{/inline~}}\n
                Hello {{> fullName}}! You have listed the following people
                as your friends:\n
                {{#each friends}}
                  - {{> fullName}}\n
                {{/each}}"
  }
```

```
}
```

> **Important**
>
> 앞의 코드 예에는 예를 이해하기 쉽도록 `HtmlPart` 및 `TextPart` 속성의 값에 줄 바꿈이 포함되어 있습니다. 템플릿의 JSON 파일에는 이러한 값 내에 줄바꿈이 포함될 수 없습니다. 이 예를 복사하여 자체 JSON 파일에 붙여 넣으면 진행하기 전에 이 섹션들에서 줄 바꿈 및 추가 공백을 제거합니다.

이메일 템플릿 관리

이메일 템플릿을 생성하는 것 외에도 Amazon SES v2 API를 사용하여 기존 템플릿을 업데이트 또는 삭제하거나, 기존 템플릿을 모두 나열하거나, 템플릿의 내용을 볼 수도 있습니다.

이 단원에는 AWS CLI 를 사용하여 SES 템플릿과 관련된 작업을 수행하는 절차가 포함되어 있습니다.

> **Note**
>
> 이 섹션의 절차는 또한 AWS CLI을(를) 이미 설치하여 구성한 상태를 전제로 설명합니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

이메일 템플릿 목록 보기

`ListEmailTemplate` SES v2 API 작업을 사용하여 기존의 모든 이메일 템플릿 목록을 볼 수 있습니다.

이메일 템플릿 목록을 보려면

- 명령줄에 다음 명령을 입력합니다.

  ```
  aws sesv2 list-email-templates
  ```

  현재 리전의 SES 계정에 기존 이메일 템플릿이 있는 경우이 명령은 다음 예제와 유사한 응답을 반환합니다.

  ```
  {
  ```

```
    "TemplatesMetadata": [
        {
            "Name": "SpecialOffers",
            "CreatedTimestamp": "2020-08-05T16:04:12.640Z"
        },
        {
            "Name": "NewsAndUpdates",
            "CreatedTimestamp": "2019-10-03T20:03:34.574Z"
        }
    ]
}
```

템플릿을 생성하지 않았으면 명령은 멤버 없이 `TemplatesMetadata` 객체를 반환합니다.

특정 이메일 템플릿의 내용 보기

`GetEmailTemplate` SES v2 API 작업을 사용하여 특정 이메일 템플릿의 내용을 볼 수 있습니다.

이메일 템플릿의 내용 보기

- 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 get-email-template --template-name MyTemplate
```

위의 명령에서 *`MyTemplate`*을 보려는 템플릿의 이름으로 바꿉니다.

제공한 템플릿 이름이 SES 계정에 있는 템플릿과 일치하는 경우이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Template": {
        "TemplateName": "TestMessage",
        "SubjectPart": "Amazon SES Test Message",
        "TextPart": "Hello! This is the text part of the message.",
        "HtmlPart": "<html>\n<body>\n<h2>Hello!</h2>\n<p>This is the HTML part of
 the message.</p></body>\n</html>"
    }
}
```

제공한 템플릿 이름이 SES 계정에 있는 템플릿과 일치하지 않는 경우 명령은 `NotFoundException` 오류를 반환합니다.

이메일 템플릿 삭제

DeleteEmailTemplate SES v2 API 작업을 사용하여 특정 이메일 템플릿을 삭제할 수 있습니다.

이메일 템플릿을 삭제하려면

- 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 delete-email-template --template-name MyTemplate
```

  위의 명령에서 *MyTemplate*을 삭제하려는 템플릿의 이름으로 바꿉니다.

  이 명령은 출력을 제공하지 않습니다. GetTemplate작업을 사용하여 템플릿이 삭제되었는지 확인할 수 있습니다.

이메일 템플릿 업데이트

UpdateEmailTemplate SES v2 API 작업을 사용하여 기존 이메일 템플릿을 업데이트할 수 있습니다. 예를 들어 이 작업은 이메일 서식 파일의 제목 줄을 변경하려는 경우 또는 메시지 본문을 수정해야 하는 경우에 유용합니다.

이메일 템플릿을 업데이트하려면

1. `GetEmailTemplate` 명령을 사용하여 명령줄에 다음 명령을 입력하여 기존 템플릿을 검색할 수 있습니다.

```
aws sesv2 get-email-template --template-name MyTemplate
```

   위의 명령에서 *MyTemplate*을 업데이트하려는 템플릿의 이름으로 바꿉니다.

   제공한 템플릿 이름이 SES 계정에 있는 템플릿과 일치하는 경우이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Template": {
        "TemplateName": "TestMessage",
        "SubjectPart": "Amazon SES Test Message",
        "TextPart": "Hello! This is the text part of the message.",
        "HtmlPart": "<html>\n<body>\n<h2>Hello!</h2>\n<p>This is the HTML part of
 the message.</p></body>\n</html>"
```

```
    }
}
```

2. 텍스트 편집기에서 새로운 파일을 생성합니다. 이전 명령의 출력을 파일에 붙여 넣습니다.
3. 필요에 따라 템플릿을 수정합니다. 생략하는 모든 줄은 템플릿에서 제거됩니다. 예를 들어, 단지 템플릿의 `SubjectPart`을(를) 변경하려는 경우 여전히 `TextPart` 및 `HtmlPart` 속성을 포함해야 합니다.

   작업을 마치면 파일 이름을 `update_template.json`(으)로 저장합니다.
4. 명령줄에 다음 명령을 입력합니다.

   ```
   aws sesv2 update-email-template --cli-input-json file://path/to/
   update_template.json
   ```

   위의 명령에서 *`path/to/update_template.json`*을 이전 단계에서 생성한 `update_template.json` 파일의 경로로 바꿉니다.

   템플릿이 성공적으로 업데이트되면 이 명령은 출력을 제공하지 않습니다. `GetEmailTemplate` 작업을 사용하여 템플릿이 업데이트되었는지 확인할 수 있습니다.

   지정한 템플릿이 존재하지 않는 경우 이 명령은 `TemplateDoesNotExist` 오류를 반환합니다. 템플릿이 `TextPart` 또는 `HtmlPart` 속성(또는 둘 다)를 포함하지 않는 경우 이 명령은 `InvalidParameterValue` 오류를 반환합니다.

AWS SDK를 사용하여 Amazon SES를 통해 이메일 전송

AWS SDK를 사용하여 Amazon SES를 통해 이메일을 보낼 수 있습니다. AWS SDKs를 사용할 수 있습니다. 자세한 내용은 Amazon Web Services용 도구를 참조하세요.

사전 조건

다음 섹션의 코드 샘플을 완료하려면 다음 필수 구성 요소를 완료해야 합니다.

- 아직 하지 않았다면, Amazon Simple Email Service 설정의 작업을 완료합니다.
- Amazon SES에서 이메일 주소 확인—Amazon SES에서 이메일을 보내기 전에 발신자 이메일 주소의 소유자인지 확인해야 합니다. 사용자 계정이 아직 Amazon SES 샌드박스 환경에 있는 경우 수신자 이메일 주소도 확인해야 합니다. Amazon SES 콘솔을 사용하여 이메일 주소를 확인하는 것이 좋습니다. 자세한 내용은 이메일 주소 자격 증명 생성 단원을 참조하십시오.

- 자격 AWS 증명 가져오기 - SDK를 사용하여 Amazon SES에 액세스하려면 AWS 액세스 키 ID와 AWS 보안 액세스 키가 필요합니다. AWS Management Console의 보안 자격 증명 페이지를 사용하여 자격 증명을 찾을 수 있습니다. 자격 증명에 대한 자세한 내용은 Amazon SES 자격 증명 유형 단원을 참조하세요.
- 공유 자격 증명 파일 생성—이 단원의 샘플 코드가 올바로 실행되기 위해서는 공유 자격 증명 파일을 생성해야 합니다. 자세한 내용은 AWS SDK를 사용하여 Amazon SES를 통해 이메일을 전송할 때 사용할 공유 자격 증명 파일 생성 단원을 참조하십시오.

코드 예제

> **Important**
>
> 다음 자습서에서는 수신 여부를 확인할 수 있도록 자신에게 이메일을 발송합니다. 추가적인 실험 또는 로드 테스트는 Amazon SES 메일박스 시뮬레이터를 사용하세요. 메일박스 시뮬레이터로 전송되는 이메일은 전송 할당량이나 반송 메일 및 수신 거부 발생률에 포함되지 않습니다. 자세한 내용은 수동으로 메일박스 시뮬레이터 사용 단원을 참조하십시오.

.NET

다음 절차에서는 Visual Studio 및 AWS SDK for .NET를 사용하여 Amazon SES를 통해 이메일을 보내는 방법을 보여줍니다.

이 솔루션은 다음 구성 요소를 사용하여 테스트되었습니다.

- Microsoft Visual Studio Community 2017, 버전 15.4.0
- Microsoft .NET Framework 버전 4.6.1
- NuGet을 사용하여 설치된 AWSSDK.Core 패키지(버전 3.3.19)
- NuGet을 사용하여 설치된 AWSSDK.SimpleEmail 패키지(버전 3.3.6.1)

시작하기 전에 다음 작업을 수행하세요.

- Visual Studio 설치 - Visual Studio는 https://www.visualstudio.com/에서 사용할 수 있습니다.

를 사용하여 이메일을 보내려면 AWS SDK for .NET

1. 다음 단계에 따라 새 프로젝트를 만듭니다.

a. Visual Studio를 시작합니다.

b. File 메뉴에서 New와 Project를 차례대로 선택합니다.

c. [New Project] 창의 왼쪽 패널에서 [Installed]를 확장한 후 [Visual C#]을 확장합니다.

d. 오른쪽 패널에서 [Console App (.NET Framework)]을 선택합니다.

e. 이름에 **AmazonSESSample**을 입력한 다음 확인을 선택합니다.

2. NuGet을 사용해 다음 단계를 완료하여 솔루션에 Amazon SES 패키지를 포함시킵니다.

a. [Solution Explorer] 창에서 프로젝트를 마우스 오른쪽 버튼으로 클릭한 다음 [Manage NuGet Packages]를 선택합니다.

b. [NuGet: AmazonSESSample] 탭에서 [Browse]를 선택합니다.

c. 검색 상자에 **AWSSDK.SimpleEmail**를 입력합니다.

d. [AWSSDK.SimpleEmail] 패키지를 선택한 다음 [Install]을 선택합니다.

e. [Preview Changes] 창에서 [OK]를 선택합니다.

3. [Program.cs] 탭에서 다음 코드를 붙여넣습니다.

```
using Amazon;
using System;
using System.Collections.Generic;
using Amazon.SimpleEmail;
using Amazon.SimpleEmail.Model;

namespace AmazonSESSample
{
    class Program
    {
        // Replace sender@example.com with your "From" address.
        // This address must be verified with Amazon SES.
        static readonly string senderAddress = "sender@example.com";

        // Replace recipient@example.com with a "To" address. If your account
        // is still in the sandbox, this address must be verified.
        static readonly string receiverAddress = "recipient@example.com";

        // The configuration set to use for this email. If you do not want to
 use a
        // configuration set, comment out the following property and the
        // ConfigurationSetName = configSet argument below.
        static readonly string configSet = "ConfigSet";
```

```
        // The subject line for the email.
        static readonly string subject = "Amazon SES test (AWS SDK for .NET)";

        // The email body for recipients with non-HTML email clients.
        static readonly string textBody = "Amazon SES Test (.NET)\r\n"
                                        + "This email was sent through Amazon
 SES "
                                        + "using the AWS SDK for .NET.";

        // The HTML body of the email.
        static readonly string htmlBody = @"<html>
<head></head>
<body>
  <h1>Amazon SES Test (SDK for .NET)</h1>
  <p>This email was sent with
    <a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the
    <a href='https://aws.amazon.com/sdk-for-net/'> AWS SDK for .NET</a>.</p>
</body>
</html>";

        static void Main(string[] args)
        {
            // Replace USWest2 with the AWS Region you're using for Amazon SES.
            // Acceptable values are EUWest1, USEast1, and USWest2.
            using (var client = new
 AmazonSimpleEmailServiceClient(RegionEndpoint.USWest2))
            {
                var sendRequest = new SendEmailRequest
                {
                    Source = senderAddress,
                    Destination = new Destination
                    {
                        ToAddresses =
                        new List<string> { receiverAddress }
                    },
                    Message = new Message
                    {
                        Subject = new Content(subject),
                        Body = new Body
                        {
                            Html = new Content
                            {
                                Charset = "UTF-8",
```

```
                                Data = htmlBody
                            },
                            Text = new Content
                            {
                                Charset = "UTF-8",
                                Data = textBody
                            }
                        }
                    },
                    // If you are not using a configuration set, comment
                    // or remove the following line
                    ConfigurationSetName = configSet
                };
                try
                {
                    Console.WriteLine("Sending email using Amazon SES...");
                    var response = client.SendEmail(sendRequest);
                    Console.WriteLine("The email was sent successfully.");
                }
                catch (Exception ex)
                {
                    Console.WriteLine("The email was not sent.");
                    Console.WriteLine("Error message: " + ex.Message);

                }
            }

            Console.Write("Press any key to continue...");
            Console.ReadKey();
        }
    }
}
```

4. 코드 편집기에서 다음을 수행합니다.

   - *sender@example.com*을 "From:"의 이메일 주소로 바꿉니다. 이 주소가 확인되어야 합니다. 자세한 내용은 확인된 자격 증명 단원을 참조하세요.
   - *recipient@example.com*을 "To:"의 주소로 바꿉니다. 계정이 아직 샌드박스 환경에 있을 경우 이 주소도 확인해야 합니다.
   - *ConfigSet*을 이 이메일을 전송할 때 사용하는 구성 세트의 이름으로 바꿉니다.

- *USWest2*를 Amazon SES를 사용하여 이메일을 보내는 데 사용하는 AWS 리전 엔드포인트의 이름으로 바꿉니다. Amazon SES를 사용할 수 있는 리전 목록은 AWS 일반 참조의 Amazon Simple Email Service(Amazon SES)를 참조하세요.

완료되면 `Program.cs`를 저장합니다.

5. 다음 단계를 완료하여 애플리케이션을 빌드 및 실행합니다.

   a. [Build] 메뉴에서 [Build Solution]을 선택합니다.

   b. [Debug] 메뉴에서 [Start Debugging]을 선택합니다. 콘솔 창이 나타납니다.

6. 콘솔 출력을 확인합니다. 이메일이 성공적으로 전송되었으면 콘솔에 "`The email was sent successfully.`"가 표시됩니다.

7. 이메일이 성공적으로 전송되었으면 수신자 주소의 이메일 클라이언트에 로그인합니다. 보낸 메시지가 도착해 있을 것입니다.

Java

다음 절차에서는 Java EE Developers용 Eclipse IDE 및 AWS Toolkit for Eclipse를 사용하여 AWS SDK 프로젝트를 생성하고 Java 코드를 수정하여 Amazon SES를 통해 이메일을 보내는 방법을 보여줍니다.

시작하기 전에 다음 작업을 수행하세요.

- Eclipse 설치 - Eclipse는 https://www.eclipse.org/downloads에서 다운로드할 수 있습니다. 이 자습서의 코드는 Eclipse Neon.3(버전 4.6.3)에서 Java Runtime Environment 버전 1.8을 실행하여 테스트하였습니다.
- 설치 AWS Toolkit for Eclipse- Eclipse 설치에 AWS Toolkit for Eclipse 를 추가하는 지침은 https://aws.amazon.com/eclipse 확인할 수 있습니다. 이 자습서의 코드는 AWS Toolkit for Eclipse버전 2.3.1을 사용하여 테스트하였습니다.

를 사용하여 이메일을 보내려면 AWS SDK for Java

1. 다음 단계를 수행하여 Eclipse에서 AWS Java 프로젝트를 생성합니다.

   a. Eclipse를 시작합니다.

   b. [File] 메뉴에서 [New]와 [Other]를 차례대로 선택합니다. 새 창에서 AWS 폴더를 확장하여 AWS Java Project(Java 프로젝트)를 선택합니다.

c. 새 AWS Java 프로젝트 대화 상자에서 다음을 수행합니다.

   i. [Project name]에 프로젝트 이름을 입력합니다.

   ii. AWS SDK for Java 샘플 아래에서 Amazon Simple Email Service JavaMail 샘플을 선택합니다.

   iii. 마침을 클릭합니다.

2. Eclipse에서 [Package Explorer] 창에서 프로젝트를 확장합니다.
3. 프로젝트 아래에서 `src/main/java` 폴더와 `com.amazon.aws.samples` 폴더를 차례대로 확장한 후 `AmazonSESSample.java`를 두 번 클릭합니다.
4. `AmazonSESSample.java`의 전체 내용을 다음 코드로 바꿉니다.

```
package com.amazonaws.samples;

import java.io.IOException;

import com.amazonaws.regions.Regions;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailService;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClientBuilder;
import com.amazonaws.services.simpleemail.model.Body;
import com.amazonaws.services.simpleemail.model.Content;
import com.amazonaws.services.simpleemail.model.Destination;
import com.amazonaws.services.simpleemail.model.Message;
import com.amazonaws.services.simpleemail.model.SendEmailRequest;

public class AmazonSESSample {

  // Replace sender@example.com with your "From" address.
  // This address must be verified with Amazon SES.
  static final String FROM = "sender@example.com";

  // Replace recipient@example.com with a "To" address. If your account
  // is still in the sandbox, this address must be verified.
  static final String TO = "recipient@example.com";

  // The configuration set to use for this email. If you do not want to use a
  // configuration set, comment the following variable and the
  // .withConfigurationSetName(CONFIGSET); argument below.
  static final String CONFIGSET = "ConfigSet";

  // The subject line for the email.
  static final String SUBJECT = "Amazon SES test (AWS SDK for Java)";
```

```
  // The HTML body for the email.
  static final String HTMLBODY = "<h1>Amazon SES test (AWS SDK for Java)</h1>"
      + "<p>This email was sent with <a href='https://aws.amazon.com/ses/'>"
      + "Amazon SES</a> using the <a href='https://aws.amazon.com/sdk-for-
java/'>"
      + "AWS SDK for Java</a>";

  // The email body for recipients with non-HTML email clients.
  static final String TEXTBODY = "This email was sent through Amazon SES "
      + "using the AWS SDK for Java.";

  public static void main(String[] args) throws IOException {

    try {
      AmazonSimpleEmailService client =
          AmazonSimpleEmailServiceClientBuilder.standard()
          // Replace US_WEST_2 with the AWS Region you're using for
          // Amazon SES.
            .withRegion(Regions.US_WEST_2).build();
      SendEmailRequest request = new SendEmailRequest()
          .withDestination(
              new Destination().withToAddresses(TO))
          .withMessage(new Message()
              .withBody(new Body()
                  .withHtml(new Content()
                      .withCharset("UTF-8").withData(HTMLBODY))
                  .withText(new Content()
                      .withCharset("UTF-8").withData(TEXTBODY)))
              .withSubject(new Content()
                  .withCharset("UTF-8").withData(SUBJECT)))
          .withSource(FROM)
          // Comment or remove the next line if you are not using a
          // configuration set
          .withConfigurationSetName(CONFIGSET);
      client.sendEmail(request);
      System.out.println("Email sent!");
    } catch (Exception ex) {
      System.out.println("The email was not sent. Error message: "
          + ex.getMessage());
    }
  }
}
```

5. `AmazonSESSample.java`에서 다음을 자체 값으로 대체합니다.

> **Important**
> 이메일 주소는 대/소문자를 구분합니다. 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.

   - `SENDER@EXAMPLE.COM` - "From" 이메일 주소로 대체합니다. 이 프로그램을 실행하기 전에 이 주소를 확인해야 합니다. 자세한 내용은 Amazon SES에서 확인된 자격 증명 단원을 참조하세요.
   - `RECIPIENT@EXAMPLE.COM` - "To" 이메일 주소로 대체합니다. 계정이 아직 샌드박스에 있는 경우, 이 주소를 확인해야 계정을 사용할 수 있습니다. 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하세요.
   - (선택 사항) **`us-west-2`**—미국 서부(오레곤) 이외의 리전에서 Amazon SES를 사용하려면 이것을 사용할 리전으로 바꿉니다. Amazon SES를 사용할 수 있는 리전 목록은 AWS 일반 참조의 Amazon Simple Email Service(Amazon SES)를 참조하세요.
6. `AmazonSESSample.java`를 저장합니다.
7. 프로젝트를 빌드하려면 [Project]를 선택한 후 [Build Project]를 선택합니다.

> **Note**
> 이 옵션이 비활성화되어 있는 경우 자동 빌드가 활성화될 수도 있습니다. 그렇다면 이 단계를 건너뜁니다.

8. 프로그램을 시작하고 이메일을 전송하려면 [Run]을 선택하고 다시 [Run]을 선택합니다.
9. Eclipse의 콘솔 창에서 출력을 확인합니다. 이메일이 성공적으로 전송되었으면 콘솔에 "`Email sent!`"가 표시됩니다. 그렇지 않으면 오류 메시지가 표시됩니다.
10. 이메일이 성공적으로 전송되었으면 수신자 주소의 이메일 클라이언트에 로그인합니다. 보낸 메시지가 도착해 있을 것입니다.

PHP

이 주제에서는 AWS SDK for PHP을(를) 사용하여 Amazon SES를 통해 이메일을 전송하는 방법을 보여줍니다.

시작하기 전에 다음 작업을 수행하세요.

- PHP 설치 - PHP는 http://php.net/downloads.php에서 다운로드할 수 있습니다. 본 자습서에서는 PHP 버전 5.5 이상 사용이 필수적입니다. PHP를 설치한 후, 원하는 모든 명령 프롬프트에서 PHP를 실행할 수 있도록 PHP에 대한 경로를 환경 변수에 추가합니다. 이 자습서의 코드는 PHP 7.2.7을 사용하여 테스트하였습니다.
- AWS SDK for PHP 버전 3 설치 - 다운로드 및 설치 지침은 AWS SDK for PHP 설명서를 참조하세요. 이 자습서의 코드는 SDK 버전 3.64.13을 사용하여 테스트하였습니다.

를 사용하여 Amazon SES를 통해 이메일을 보내려면 AWS SDK for PHP

1. 텍스트 편집기에서 `amazon-ses-sample.php`이라는 파일을 만듭니다. 다음 코드를 붙여넣습니다.

```
<?php

// If necessary, modify the path in the require statement below to refer to the
// location of your Composer autoload.php file.
require 'vendor/autoload.php';

use Aws\Ses\SesClient;
use Aws\Exception\AwsException;

// Create an SesClient. Change the value of the region parameter if you're
// using an AWS Region other than US West (Oregon). Change the value of the
// profile parameter if you want to use a profile in your credentials file
// other than the default.
$SesClient = new SesClient([
    'profile' => 'default',
    'version' => '2010-12-01',
    'region'  => 'us-west-2'
]);

// Replace sender@example.com with your "From" address.
// This address must be verified with Amazon SES.
$sender_email = 'sender@example.com';

// Replace these sample addresses with the addresses of your recipients. If
// your account is still in the sandbox, these addresses must be verified.
$recipient_emails = ['recipient1@example.com','recipient2@example.com'];
```

```
// Specify a configuration set. If you do not want to use a configuration
// set, comment the following variable, and the
// 'ConfigurationSetName' => $configuration_set argument below.
$configuration_set = 'ConfigSet';

$subject = 'Amazon SES test (AWS SDK for PHP)';
$plaintext_body = 'This email was sent with Amazon SES using the AWS SDK for
 PHP.' ;
$html_body =  '<h1>AWS Amazon Simple Email Service Test Email</h1>'.
              '<p>This email was sent with <a href="https://aws.amazon.com/
ses/">'.
              'Amazon SES</a> using the <a href="https://aws.amazon.com/sdk-for-
php/">'.
              'AWS SDK for PHP</a>.</p>';
$char_set = 'UTF-8';

try {
    $result = $SesClient->sendEmail([
        'Destination' => [
            'ToAddresses' => $recipient_emails,
        ],
        'ReplyToAddresses' => [$sender_email],
        'Source' => $sender_email,
        'Message' => [
          'Body' => [
              'Html' => [
                  'Charset' => $char_set,
                  'Data' => $html_body,
              ],
              'Text' => [
                  'Charset' => $char_set,
                  'Data' => $plaintext_body,
              ],
          ],
          'Subject' => [
              'Charset' => $char_set,
              'Data' => $subject,
          ],
        ],
        // If you aren't using a configuration set, comment or delete the
        // following line
        'ConfigurationSetName' => $configuration_set,
    ]);
    $messageId = $result['MessageId'];
```

```
    echo("Email sent! Message ID: $messageId"."\n");
} catch (AwsException $e) {
    // output error message if fails
    echo $e->getMessage();
    echo("The email was not sent. Error message: ".$e-
>getAwsErrorMessage()."\n");
    echo "\n";
}
```

2. `amazon-ses-sample.php`에서 다음을 자체 값으로 대체합니다.

   - **`path_to_sdk_inclusion`**- 프로그램에를 포함하는 데 필요한 경로로 바꿉 AWS SDK for PHP 니다. 자세한 내용은 AWS SDK for PHP 설명서를 참조하세요.
   - **`sender@example.com`**—Amazon SES에서 확인한 이메일 주소로 바꿉니다. 자세한 내용은 확인된 자격 증명 단원을 참조하세요. Amazon SES에서 이메일 주소는 대/소문자를 구분합니다. 입력하는 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.
   - **`recipient1@example.com`**, **`recipient2@example.com`** - 수신자의 주소로 바꿉니다. 계정이 아직 샌드박스 환경에 있을 경우 수신자의 주소도 확인해야 합니다. 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하세요. 입력하는 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.
   - (선택 사항) **`ConfigSet`** - 이 이메일을 전송할 때 구성 세트를 사용하려면 이 값을 구성 세트의 이름으로 바꿉니다. 구성 세트에 대한 자세한 내용은 Amazon SES에서 구성 세트 사용 단원을 참조하세요.
   - (선택 사항) **`us-west-2`**—미국 서부(오레곤) 이외의 리전에서 Amazon SES를 사용하려면 이것을 사용할 리전으로 바꿉니다. Amazon SES를 사용할 수 있는 리전 목록은 AWS 일반 참조의 Amazon Simple Email Service(Amazon SES)를 참조하세요.
3. `amazon-ses-sample.php`를 저장합니다.
4. 프로그램을 실행하려면 `amazon-ses-sample.php`와 동일한 디렉터리에서 명령 프롬프트를 열고 다음 명령을 입력합니다.

```
$ php amazon-ses-sample.php
```

5. 출력 결과를 검토합니다. 이메일이 성공적으로 전송되었으면 콘솔에 `"Email sent!"`가 표시됩니다. 그렇지 않으면 오류 메시지가 표시됩니다.

> **Note**
>
> 프로그램을 실행할 때 "cURL error 60: SSL certificate problem" 오류가 발생하면 AWS SDK for PHP 설명서의 설명에 따라 최신 CA 번들을 다운로드합니다. 그런 다음 `amazon-ses-sample.php`에서 `SesClient::factory` 어레이에 다음 줄을 추가하고, `path_of_certs`를 CA 번들 다운로드 경로로 바꾼 후 프로그램을 다시 실행합니다.
>
> ```
> 'http' => [
>  'verify' => 'path_of_certs\ca-bundle.crt'
> ]
> ```

6. 수신자 주소의 이메일 클라이언트에 로그인합니다. 보낸 메시지가 도착해 있을 것입니다.

Ruby

이 주제에서는 AWS SDK for Ruby을(를) 사용하여 Amazon SES를 통해 이메일을 전송하는 방법을 보여줍니다.

시작하기 전에 다음 작업을 수행하세요.

- Ruby 설치 - Ruby는 https://www.ruby-lang.org/en/downloads/에서 다운로드할 수 있습니다. 이 자습서의 코드는 Ruby 1.9.3을 사용하여 테스트하였습니다. Ruby를 설치한 후 원하는 모든 명령 프롬프트에서 Ruby를 실행할 수 있도록 Ruby에 대한 경로를 환경 변수에 추가합니다.
- 설치 AWS SDK for Ruby - 다운로드 및 설치 지침은 AWS SDK for Ruby 개발자 안내서의 설치를 AWS SDK for Ruby 참조하세요. 이 자습서의 샘플 코드는 AWS SDK for Ruby버전 2.9.36을 사용하여 테스트하였습니다.
- 공유 자격 증명 파일 생성—이 단원의 샘플 코드가 올바로 실행되기 위해서는 공유 자격 증명 파일을 생성해야 합니다. 자세한 내용은 AWS SDK를 사용하여 Amazon SES를 통해 이메일을 전송할 때 사용할 공유 자격 증명 파일 생성 단원을 참조하십시오.

를 사용하여 Amazon SES를 통해 이메일을 보내려면 AWS SDK for Ruby

1. 텍스트 편집기에서 `amazon-ses-sample.rb`이라는 파일을 만듭니다. 다음 코드를 파일에 붙여넣습니다.

```
require 'aws-sdk'
```

```
# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
sender = "sender@example.com"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
recipient = "recipient@example.com"

# Specify a configuration set. If you do not want to use a configuration
# set, comment the following variable and the
# configuration_set_name: configsetname argument below.
configsetname = "ConfigSet"

# Replace us-west-2 with the AWS Region you're using for Amazon SES.
awsregion = "us-west-2"

# The subject line for the email.
subject = "Amazon SES test (AWS SDK for Ruby)"

# The HTML body of the email.
htmlbody =
  '<h1>Amazon SES test (AWS SDK for Ruby)</h1>'\
  '<p>This email was sent with <a href="https://aws.amazon.com/ses/">'\
  'Amazon SES</a> using the <a href="https://aws.amazon.com/sdk-for-ruby/">'\
  'AWS SDK for Ruby</a>.'

# The email body for recipients with non-HTML email clients.
textbody = "This email was sent with Amazon SES using the AWS SDK for Ruby."

# Specify the text encoding scheme.
encoding = "UTF-8"

# Create a new SES resource and specify a region
ses = Aws::SES::Client.new(region: awsregion)

# Try to send the email.
begin

  # Provide the contents of the email.
  resp = ses.send_email({
    destination: {
      to_addresses: [
        recipient,
```

```
      ],
    },
    message: {
      body: {
        html: {
          charset: encoding,
          data: htmlbody,
        },
        text: {
          charset: encoding,
          data: textbody,
        },
      },
      subject: {
        charset: encoding,
        data: subject,
      },
    },
  source: sender,
  # Comment or remove the following line if you are not using
  # a configuration set
  configuration_set_name: configsetname,
  })
  puts "Email sent!"

# If something goes wrong, display an error message.
rescue Aws::SES::Errors::ServiceError => error
  puts "Email not sent. Error message: #{error}"

end
```

2. `amazon-ses-sample.rb`에서 다음을 자체 값으로 대체합니다.

   - **`sender@example.com`**—Amazon SES에서 확인한 이메일 주소로 바꿉니다. 자세한 내용은 확인된 자격 증명 단원을 참조하세요. Amazon SES에서 이메일 주소는 대/소문자를 구분합니다. 입력하는 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.
   - **`recipient@example.com`**—수신자의 주소로 바꿉니다. 계정이 아직 샌드박스에 있는 경우, 이 주소를 확인해야 계정을 사용할 수 있습니다. 자세한 내용은 프로덕션 액세스 요청 (Amazon SES 샌드박스에서 이동) 단원을 참조하세요. 입력하는 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.

- (선택 사항) **`us-west-2`**—미국 서부(오레곤) 이외의 리전에서 Amazon SES를 사용하려면 이것을 사용할 리전으로 바꿉니다. Amazon SES를 사용할 수 있는 리전 목록은 AWS 일반 참조의 Amazon Simple Email Service(Amazon SES)를 참조하세요.

3. `amazon-ses-sample.rb`를 저장합니다.
4. 프로그램을 실행하려면 `amazon-ses-sample.rb`와 동일한 디렉터리에서 명령 프롬프트를 열고 ruby amazon-ses-sample.rb를 입력합니다.
5. 출력 결과를 검토합니다. 이메일이 성공적으로 전송되었으면 콘솔에 `"Email sent!"`가 표시됩니다. 그렇지 않으면 오류 메시지가 표시됩니다.
6. 수신자 주소의 이메일 클라이언트에 로그인합니다. 보낸 메시지가 도착해 있을 것입니다.

Python

이 주제에서는 AWS SDK for Python (Boto)을(를) 사용하여 Amazon SES를 통해 이메일을 전송하는 방법을 보여줍니다.

시작하기 전에 다음 작업을 수행하세요.

- Amazon SES에서 이메일 주소 확인—Amazon SES에서 이메일을 보내기 전에 발신자 이메일 주소의 소유자인지 확인해야 합니다. 사용자 계정이 아직 Amazon SES 샌드박스 환경에 있는 경우 수신자 이메일 주소도 확인해야 합니다. Amazon SES 콘솔을 사용하여 이메일 주소를 확인하는 것이 좋습니다. 자세한 내용은 이메일 주소 자격 증명 생성 단원을 참조하십시오.
- 자격 증명 가져오기 AWS - SDK를 사용하여 Amazon SES에 액세스하려면 AWS 액세스 키 ID와 AWS 보안 액세스 키가 필요합니다. AWS Management Console의 보안 자격 증명 페이지를 사용하여 자격 증명을 찾을 수 있습니다. 자격 증명에 대한 자세한 내용은 Amazon SES 자격 증명 유형 단원을 참조하세요.
- Python 설치 - Python은 https://www.python.org/downloads/에서 다운로드할 수 있습니다. 이 자습서의 코드는 Python 2.7.6 및 Python 3.6.1을 사용하여 테스트하였습니다. Python을 설치한 후 원하는 모든 명령 프롬프트에서 Python을 실행할 수 있도록 Python에 대한 경로를 환경 변수에 추가합니다.
- 설치 AWS SDK for Python (Boto) - 다운로드 및 설치 지침은 AWS SDK for Python (Boto) 설명서를 참조하세요. 이 자습서의 샘플 코드는 SDK for Python 버전 1.4.4를 사용하여 테스트하였습니다.

SDK for Python을 사용하여 Amazon SES를 통해 이메일을 전송하려면

1. 텍스트 편집기에서 `amazon-ses-sample.py`이라는 파일을 만듭니다. 다음 코드를 파일에 붙여넣습니다.

```
import boto3
from botocore.exceptions import ClientError

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
SENDER = "Sender Name <sender@example.com>"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
RECIPIENT = "recipient@example.com"

# Specify a configuration set. If you do not want to use a configuration
# set, comment the following variable, and the
# ConfigurationSetName=CONFIGURATION_SET argument below.
CONFIGURATION_SET = "ConfigSet"

# If necessary, replace us-west-2 with the AWS Region you're using for Amazon
 SES.
AWS_REGION = "us-west-2"

# The subject line for the email.
SUBJECT = "Amazon SES Test (SDK for Python)"

# The email body for recipients with non-HTML email clients.
BODY_TEXT = ("Amazon SES Test (Python)\r\n"
             "This email was sent with Amazon SES using the "
             "AWS SDK for Python (Boto)."
            )

# The HTML body of the email.
BODY_HTML = """<html>
<head></head>
<body>
  <h1>Amazon SES Test (SDK for Python)</h1>
  <p>This email was sent with
    <a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the
    <a href='https://aws.amazon.com/sdk-for-python/'> AWS SDK for Python
 (Boto)</a>.</p>
```

```
</body>
</html>
            """

# The character encoding for the email.
CHARSET = "UTF-8"

# Create a new SES resource and specify a region.
client = boto3.client('ses',region_name=AWS_REGION)

# Try to send the email.
try:
    #Provide the contents of the email.
    response = client.send_email(
        Destination={
            'ToAddresses': [
                RECIPIENT,
            ],
        },
        Message={
            'Body': {
                'Html': {
                    'Charset': CHARSET,
                    'Data': BODY_HTML,
                },
                'Text': {
                    'Charset': CHARSET,
                    'Data': BODY_TEXT,
                },
            },
            'Subject': {
                'Charset': CHARSET,
                'Data': SUBJECT,
            },
        },
        Source=SENDER,
        # If you are not using a configuration set, comment or delete the
        # following line
        ConfigurationSetName=CONFIGURATION_SET,
    )
# Display an error if something goes wrong.
except ClientError as e:
    print(e.response['Error']['Message'])
else:
```

```
    print("Email sent! Message ID:"),
    print(response['MessageId'])
```

2. `amazon-ses-sample.py`에서 다음을 자체 값으로 대체합니다.

   - **`sender@example.com`**—Amazon SES에서 확인한 이메일 주소로 바꿉니다. 자세한 내용은 확인된 자격 증명 단원을 참조하십시오. Amazon SES에서 이메일 주소는 대/소문자를 구분합니다. 입력하는 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.
   - **`recipient@example.com`**—수신자의 주소로 바꿉니다. 계정이 아직 샌드박스에 있는 경우, 이 주소를 확인해야 계정을 사용할 수 있습니다. 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하세요. 입력하는 주소는 사용자가 확인한 것과 정확하게 동일해야 합니다.
   - (선택 사항) **`us-west-2`**—미국 서부(오레곤) 이외의 리전에서 Amazon SES를 사용하려면 이것을 사용할 리전으로 바꿉니다. Amazon SES를 사용할 수 있는 리전 목록은 AWS 일반 참조의 Amazon Simple Email Service(Amazon SES)를 참조하세요.
3. `amazon-ses-sample.py`를 저장합니다.
4. 프로그램을 실행하려면 `amazon-ses-sample.py`와 동일한 디렉터리에서 명령 프롬프트를 열고 python amazon-ses-sample.py를 입력합니다.
5. 출력 결과를 검토합니다. 이메일이 성공적으로 전송되었으면 콘솔에 `"Email sent!"`가 표시됩니다. 그렇지 않으면 오류 메시지가 표시됩니다.
6. 수신자 주소의 이메일 클라이언트에 로그인합니다. 보낸 메시지가 도착해 있을 것입니다.

AWS SDK를 사용하여 Amazon SES를 통해 이메일을 전송할 때 사용할 공유 자격 증명 파일 생성

다음은 홈 디렉터리에서 공유 자격 증명 파일을 생성하는 방법에 대한 절차입니다. SDK 샘플 코드가 올바로 실행되려면 이 파일을 반드시 생성해야 합니다.

1. 텍스트 편집기에서 새로운 파일을 생성합니다. 생성된 파일에 다음 코드를 붙여넣습니다.

   ```
   [default]
   aws_access_key_id = YOUR_AWS_ACCESS_KEY_ID
   aws_secret_access_key = YOUR_AWS_SECRET_ACCESS_KEY
   ```

2. 방금 생성한 텍스트 파일에서를 고유한 AWS 액세스 키 IDYOUR_AWS_ACCESS_KEY로 바꾸고를 고유한 AWS 보안 액세스 키YOUR_AWS_SECRET_ACCESS_KEY로 바꿉니다.
3. 파일을 저장합니다. 다음 표는 운영 체제에 따라 올바른 파일 위치 및 이름을 나타낸 것입니다.

운영 체제	저장되는 파일 이름
Windows	`C:\Users\<yourUserName>\.aws\credentials`
Linux, macOS 또는 Unix	`~/.aws/credentials`

> **Important**
> 자격 증명 파일을 저장할 때 파일 확장명을 포함시키지 마세요.

Amazon SES에서 지원하는 콘텐츠 인코딩

다음은 참조를 위해 제공됩니다.

Amazon SES는 다음 콘텐츠 인코딩을 지원합니다.

- `deflate`
- `gzip`
- `identity`

RFC 7231 사양에 따르면 Amazon SES는 다음의 Accept-Encoding 헤더 형식도 지원합니다.

- `Accept-Encoding:deflate,gzip`
- `Accept-Encoding:`
- `Accept-Encoding:*`
- `Accept-Encoding:deflate;q=0.5,gzip;q=1.0`
- `Accept-Encoding:gzip;q=1.0,identity;q=0.5,*;q=0`

Amazon SES 및 보안 프로토콜

이 주제에서는 Amazon SES와 연결할 때, 그리고 Amazon SES가 이메일을 받는 사람에게 전송할 때 사용할 수 있는 보안 프로토콜을 설명합니다.

이메일 발신자와 Amazon SES 연결

Amazon SES와 연결할 때 사용하는 보안 프로토콜은 다음에서 설명하듯이 Amazon SES API 또는 Amazon SES SMTP 인터페이스를 사용하는가에 따라 달라집니다.

HTTPS

Amazon SES API(직접 또는 AWS SDK를 통해)를 사용하는 경우 모든 통신은 Amazon SES HTTPS 엔드포인트를 통해 TLS에 의해 암호화됩니다. Amazon SES HTTPS 엔드포인트는 TLS 1.2 및 TLS 1.3을 지원합니다.

SMTP 인터페이스

SMTP 인터페이스를 통해 Amazon SES에 액세스하는 경우 TLS(전송 계층 보안)를 사용하여 연결을 암호화해야 합니다. TLS는 종종 이전 프로토콜 이름 Secure Sockets Layer(SSL)로 불립니다.

Amazon SES는 TLS로 암호화된 연결 설정을 위한 두 가지 메커니즘인 STARTTLS 및 TLS 래퍼를 지원합니다.

- STARTTLS - STARTTLS는 암호화되지 않은 연결을 암호화된 연결로 업그레이드하는 방법입니다. 다양한 프로토콜을 위한 여러 버전의 STARTTLS이 있으며, SMTP 버전은 RFC 3207에서 정의됩니다. STARTTLS 연결의 경우 Amazon SES는 TLS 1.2, TLS 1.3을 지원합니다.
- TLS 래퍼 - TLS 래퍼(SMTPS 또는 핸드셰이크 프로토콜이라고도 함)는 먼저 암호화되지 않은 연결을 설정하지 않고 암호화된 연결을 시작하는 방법입니다. TLS 래퍼에서는 Amazon SES SMTP 엔드포인트가 TLS 협상을 수행하지 않습니다. TLS를 사용하여 엔드포인트와 연결한 후 전체 대화에 걸쳐 TLS를 계속 사용하는 것이 클라이언트의 책임입니다. TLS 래퍼는 더 오래된 프로토콜이지만 많은 클라이언트가 여전히 이 프로토콜을 지원합니다. TLS 래퍼 연결의 경우 Amazon SES는 TLS 1.2 및 TLS 1.3을 지원합니다.

이러한 방법을 사용한 Amazon SES SMTP 인터페이스 연결에 대한 자세한 내용은 Amazon SES SMTP 엔드포인트에 연결 섹션을 참조하세요.

Amazon SES와 수신자 연결

TLS 1.3은 기본 전송 방법이지만 SES는 이전 버전의 TLS를 사용하여 메일 서버에 이메일을 전송할 수 있습니다.

기본적으로 Amazon SES는 opportunistic TLS를 사용합니다. 즉, Amazon SES가 수신 메일 서버에 대한 보안 연결을 항상 시도한다는 것입니다. 만약 Amazon SES가 보안 연결을 설정할 수 없는 경우 암호화하지 않고 해당 메시지를 전송합니다.

이 동작은 구성 세트를 사용하여 변경할 수 있습니다. PutConfigurationSetDeliveryOptions API 작업을 사용하여 구성 세트에 대한 `TlsPolicy` 속성을 `Require`에 설정합니다. AWS CLI를 사용하여 이렇게 변경할 수 있습니다.

구성 세트에 대한 TLS 연결을 요청하여 Amazon SES를 구성하려면

- 명령줄에 다음 명령을 입력합니다.

  ```
  aws sesv2 put-configuration-set-delivery-options --configuration-set-
  name MyConfigurationSet --tls-policy REQUIRE
  ```

  이전 예제에서 *`MyConfigurationSet`*를 구성 세트의 이름으로 대체합니다.

  구성 세트를 사용하여 이메일을 전송할 경우, Amazon SES가 보안 연결을 설정할 수 있다면 수신 이메일 서버에 해당 메시지만을 전송합니다. Amazon SES가 수신 이메일 서버에 보안 연결을 할 수 없는 경우 해당 메시지를 삭제합니다.

종단 간 암호화

Amazon SES를 사용하여 S/MIME 또는 PGP를 통해 암호화된 메시지를 전송할 수 있습니다. 이러한 프로토콜을 사용하는 메시지는 발신자에 의해 암호화됩니다. 해당 콘텐츠는 메시지를 암호 해독하는 데 필요한 프라이빗 키를 소유하는 수신자만 볼 수 있습니다.

Amazon SES는 S/MIME으로 암호화된 이메일을 전송하는 데 사용할 수 있는 다음 MIME 유형을 지원합니다.

- `application/pkcs7-mime`
- `application/pkcs7-signature`
- `application/x-pkcs7-mime`
- `application/x-pkcs7-signature`

Amazon SES는 PGP로 암호화된 이메일을 전송하는 데 사용할 수 있는 다음 MIME 유형도 지원합니다.

- `application/pgp-encrypted`
- `application/pgp-keys`
- `application/pgp-signature`

Amazon SES 헤더 필드

Amazon SES는 RFC 822에 설명된 형식을 따르는 모든 이메일 헤더를 허용할 수 있습니다.

메시지 헤더 섹션에 다음 필드를 두 번 이상 표시할 수 없습니다.

- `Accept-Language`
- `acceptLanguage`
- `Archived-At`
- `Auto-Submitted`
- `Bounces-to`
- `Comments`
- `Content-Alternative`
- `Content-Base`
- `Content-Class`
- `Content-Description`
- `Content-Disposition`
- `Content-Duration`
- `Content-ID`
- `Content-Language`
- `Content-Length`
- `Content-Location`
- `Content-MD5`
- `Content-Transfer-Encoding`
- `Content-Type`
- `Date`
- `Delivered-To`

- Disposition-Notification-Options
- Disposition-Notification-To
- DKIM-Signature
- DomainKey-Signature
- Errors-To
- From
- Importance
- In-Reply-To
- Keywords
- List-Archive
- List-Help
- List-Id
- List-Owner
- List-Post
- List-Subscribe
- List-Unsubscribe
- List-Unsubscribe-Post
- Message-Context
- Message-ID
- MIME-Version
- Organization
- Original-From
- Original-Message-ID
- Original-Recipient
- Original-Subject
- Precedence
- Priority
- References
- Reply-To

- `Return-Path`
- `Return-Receipt-To`
- `Sender`
- `Solicitation`
- `Sensitivity`
- `Subject`
- `Thread-Index`
- `Thread-Topic`
- `User-Agent`
- `VBR-Info`

고려 사항

- `acceptLanguage` 필드는 표준이 아닙니다. 가능한 경우 `Accept-Language` 헤더를 대신 사용해야 합니다.
- `Date` 헤더를 지정하면 Amazon SES에서 메시지를 수락한 UTC 시간대의 날짜와 시간에 해당하는 타임스탬프로 Amazon SES가 헤더를 재정의합니다.
- `Message-ID` 헤더를 제공하면 Amazon SES가 자체 값으로 헤더를 재정의합니다.
- `Return-Path` 헤더를 지정하면 Amazon SES가 지정된 주소로 반송 메일과 수신 거부 알림을 보냅니다. 하지만 수신자가 받은 메시지에서는 `Return-Path` 헤더의 값이 다릅니다.
- Amazon SES API v2 `SendEmail` 작업을 단순 또는 템플릿 형식 콘텐츠와 함께 사용하거나 `SendBulkEmail` 작업을 사용하는 경우 SES에서 설정한 헤더에 대해 사용자 지정 헤더 콘텐츠를 설정할 수 없습니다. 따라서 다음 헤더는 사용자 지정 헤더로 허용되지 않습니다.
  - `BCC`, `CC`, `Content-Disposition`, `Content-Type`, `Date`, `From`, `Message-ID`, `MIME-Version`, `Reply-To`, `Return-Path`, `Subject`, `To`

SES에서 이메일 첨부 파일 작업

SES의 이메일 첨부 파일은 SES API v2 `SendEmail` 및 `SendBulkEmail` 작업을 사용할 때 이메일 메시지에 포함할 수 있는 파일입니다. 이 기능을 사용하면 PDFs, Word 파일, 이미지 또는 SES 지원 MIME 유형을 준수하는 기타 파일 유형과 같은 문서를 포함하여 이메일 콘텐츠를 보강할 수 있습니다. 수신자가 별도로 다운로드할 필요 없이 이메일 콘텐츠에 직접 렌더링되는 인라인 이미지를 포함할 수도 있습니다. 최대 40MB의 총 메시지 크기 한도까지 이메일당 여러 첨부 파일을 포함할 수 있습니다.

> **Note**
>
> `SendEmail` Raw 콘텐츠 유형이 있는 SES API v2, SMTP 인터페이스 및 SES API v1은 원시 이메일 MIME 메시지 구성을 통해 첨부 파일을 계속 처리합니다.

SES에서 첨부 파일이 작동하는 방식

첨부 파일이 포함된 이메일을 보낼 때 서로 다른 단계에서 발생하는 두 가지 유형의 인코딩이 있습니다.

1단계 - SES로 데이터 전송:

- SES에 첨부 파일을 보내려면 이진 데이터(예: PDF 또는 이미지)를 안전하게 전송할 수 있는 형식으로 변환해야 합니다.
- 여기서 base64 인코딩이 사용됩니다. JSON 요청에서 원시 바이너리 데이터를 전송할 수 없기 때문에 필요합니다.
- AWS SDK를 사용하는 경우이 인코딩을 자동으로 처리합니다.
- 를 사용하는 경우 첨부 파일을 보내기 전에 직접 base64 인코딩 AWS CLI해야 합니다.

2단계 - 이메일을 생성하는 SES:

- SES가 데이터를 수신하면 첨부 파일이 포함된 실제 이메일을 생성해야 합니다.
- 여기에서 ContentTransferEncoding 설정이 실행됩니다.
- SES는 ContentTransferEncoding에서 지정한 인코딩 방법을 사용하여 최종 이메일에서 첨부 파일의 형식을 자동으로 지정합니다.

이렇게 생각하세요. 메일을 통해 패키지를 보내는 것과 비슷합니다. 먼저 패키지를 우체국으로 가져와야 합니다(1단계 Base64-encoding 필요). 그러면 우체국에서 최종 배송에 적합하게 패키징합니다(2단계 - ContentTransferEncoding).

첨부 파일 객체 구조

SES를 통해 첨부 파일이 포함된 이메일을 보내면 서비스가 복잡한 MIME 메시지 구성을 자동으로 처리합니다. 다음 SES API v2 `Attachment` 객체 구조를 통해 첨부 파일 콘텐츠와 메타데이터를 제공하면 됩니다.

- `FileName` (필수) - 수신자에게 표시되는 파일 이름입니다(파일 확장명을 포함해야 함). 제공되지 않으면 SES는의 확장`ContentType`에서를 파생합니다`FileName`.
- `ContentType` (선택 사항) - IANA 호환 미디어 유형 식별자입니다.
- `ContentDisposition` (선택 사항) - (`ATTACHMENT`기본값) 또는와 같이 첨부 파일을 렌더링하는 방법을 지정합니다`INLINE`.
- `ContentDescription` (선택 사항) - 콘텐츠에 대한 간략한 설명입니다.
- `RawContent` (필수) - 첨부 파일의 실제 콘텐츠입니다.
- `ContentTransferEncoding` (선택 사항) - 첨부 파일 페이로드가 이메일의 mime 메시지로 어셈블될 때 인코딩되는 방법을 지정합니다. `SEVEN_BIT` (기본값) `BASE64` 또는 `QUOTED_PRINTABLE`.

연결된 모든 콘텐츠는 전송을 위해 SES 엔드포인트로 전송하기 전에 base64로 인코딩되어야 합니다. AWS SDK 클라이언트를 사용하여 API를 호출하는 경우 자동으로 처리됩니다. 를 사용 AWS CLI중이거나 자체 클라이언트를 구현한 경우 다음과 같이 인코딩을 직접 수행해야 합니다.

- 일반 텍스트 콘텐츠: `Text attachment sample content.`
- Base64 인코딩: `VGV4dCBhdHRhY2htZW50IHNhbXBsZSBjb250ZW50Lg==`

다음 예제에서는 SES API v2로 첨부 파일을 지정할 때 첨부 파일 객체 구조를 사용하는 방법과 첨부 파일 객체 요소가 포함된 JSON 파일을 AWS CLI 참조하는 `SendEmail` `SendBulkEmail` 작업을 보여줍니다.

Example - 간단한 콘텐츠가 포함된 SendEmail

```
aws sesv2 send-email --cli-input-json file://request-send-email-simple.json
```

request-send-email-simple.json

```
{
    "FromEmailAddress": "sender@example.com",
    "Destination": {
        "ToAddresses": [
            "recipient@example.com"
        ]
    },
    "Content": {
        "Simple": {
            "Subject": {
```

```
                "Data": "Email with attachment"
            },
            "Body": {
                "Text": {
                    "Data": "Please see attached document."
                },
                "Html": {
                    "Data": "Please see attached <b>document</b>."
                }
            },
            "Attachments": [
                {
                    "RawContent": "<base64-encoded-content>",
                    "ContentDisposition": "ATTACHMENT",
                    "FileName": "document.pdf",
                    "ContentDescription": "PDF Document Attachment",
                    "ContentTransferEncoding": "BASE64"
                }
            ]
        }
    }
}
```

Example - 간단한 콘텐츠와 인라인 연결이 포함된 SendEmail

```
aws sesv2 send-email --cli-input-json file://request-send-email-simple-inline-
attachment.json
```

request-send-email-simple-inline-attachment.json

```
{
    "FromEmailAddress": "sender@example.com",
    "Destination": {
        "ToAddresses": [
            "recipient@example.com"
        ]
    },
    "Content": {
        "Simple": {
            "Subject": {
                "Data": "Email with attachment"
            },
            "Body": {
```

```
                "Html": {
                    "Data": "<html><body>Our logo:<br><img src=\"cid:logo123\" alt=
\"Company Logo\"></body></html>"
                }
            },
            "Attachments": [
                {
                    "RawContent": "<base64-encoded-content>",
                    "ContentDisposition": "INLINE",
                    "FileName": "logo.png",
                    "ContentId": "logo123"
                }
            ]
        }
    }
}
```

Example - 템플릿 콘텐츠가 포함된 SendEmail

```
aws sesv2 send-email --cli-input-json file://request-send-email-template.json
```

request-send-email-template.json

```
{
    "FromEmailAddress": "sender@example.com",
    "Destination": {
        "ToAddresses": [
            "recipient@example.com"
        ]
    },
    "Content": {
        "Template": {
            "TemplateName": "MyTemplate",
            "TemplateData": "{\"name\":\"John\"}",
            "Attachments": [
                {
                    "RawContent": "<base64-encoded-content>",
                    "ContentDisposition": "ATTACHMENT",
                    "FileName": "document.pdf",
                    "ContentDescription": "PDF Document Attachment",
                    "ContentTransferEncoding": "BASE64"
                }
            ]
```

```
        }
    }
}
```

Example - 첨부 파일 콘텐츠가 포함된 SendBulkEmail

```
aws sesv2 send-bulk-email --cli-input-json file://request-send-bulk-email.json
```

request-send-bulk-email.json

```
{
    "FromEmailAddress": "sender@example.com",
    "DefaultContent": {
        "Template": {
            "TemplateName": "MyTemplate",
            "TemplateData": "{}",
            "Attachments": [
                {
                    "RawContent": "<base64-encoded-content>",
                    "ContentDisposition": "ATTACHMENT",
                    "FileName": "document.pdf",
                    "ContentDescription": "PDF Document Attachment",
                    "ContentTransferEncoding": "BASE64"
                }
            ]
        }
    },
    "BulkEmailEntries": [
        {
            "Destination": {
                "ToAddresses": [
                    "recipient@example.com"
                ]
            },
            "ReplacementEmailContent": {
                "ReplacementTemplate": {
                    "ReplacementTemplateData": "{\"name\":\"John\"}"
                }
            }
        }
    ]
}
```

모범 사례

- 총 메시지 크기(첨부 파일 포함)를 40MB 미만으로 유지합니다.
- 가능한 경우 SES가 파일 확장명을 기반으로 콘텐츠 유형을 자동으로 감지하도록 합니다.
- 콘텐츠 유형이 일반적인 MIME 유형을 벗어나는 경우에만 콘텐츠 유형을 명시적으로 지정합니다.
- 더 나은 이메일 렌더링을 위해 인라인 이미지를 사용하는 것이 좋습니다.
- SES는에 나열된 유형을 제외하고 첨부 파일에 대해 다양한 MIME 유형을 지원합니다지원되지 않는 연결 형식.

지원되지 않는 SES 연결 유형

Multipurpose Internet Mail Extensions(MIME) 표준을 사용하여 Amazon SES를 통해 첨부 파일이 포함된 메시지를 전송할 수 있습니다. Amazon SES는 모든 첨부 파일 유형을 허용합니다(다음 목록의 확장명을 사용하는 첨부파일 제외).

.ade	.hta	.mau	.mst	.psc1
.adp	.inf	.mav	.ops	.psc2
.app	.ins	.maw	.pcd	.tmp
.asp	.isp	.mda	.pif	.url
.bas	.its	.mdb	.plg	.vb
.bat	.js	.mde	.prf	.vbe
.cer	.jse	.mdt	.prg	.vbs
.chm	.ksh	.mdw	.reg	.vps
.cmd	.lib	.mdz	.scf	.vsmacros
.com	.lnk	.msc	.scr	.vss
.cpl	.mad	.msh	.sct	.vst
.crt	.maf	.msh1	.shb	.vsw

.csh	.mag	.msh2	.shs	.vxd
.der	.mam	.mshxml	.sys	.ws
.exe	.maq	.msh1xml	.ps1	.wsc
.fxp	.mar	.msh2xml	.ps1xml	.wsf
.gadget	.mas	.msi	.ps2	.wsh
.hlp	.mat	.msp	.ps2xml	.xnk

일부 ISP는 제한 사항(예: 보관된 첨부 파일 관련 제한 사항)을 추가로 두고 있으므로 프로덕션 이메일을 보내기 전에 주요 ISP를 통해 이메일 전송을 테스트하는 것이 좋습니다.

Amazon SES로 이메일 수신

Amazon SES를 사용하여 이메일 전송을 관리하는 것 외에도 하나 이상의 도메인을 대신하여 이메일을 수신하도록 SES를 구성할 수도 있습니다. 이메일 수신자인 SES는 다른 메일 서버와의 통신, 스팸 및 바이러스 검사, 신뢰할 수 없는 출처의 메일(Spamhaus 또는 SES의 차단 목록에 있는 주소) 차단, 도메인 내의 수신자에 대한 메일 수락 등의 기본 메일 수신 작업을 처리합니다.

수신한 이메일의 처리 범위는 지정한 사용자 지정 지침에 따라 결정됩니다. 이 지침은 두 가지 형태로 제공됩니다.

- 수신 규칙(받는 사람 기반 제어)은 수신 이메일에 대한 세분화된 제어 기능을 제공합니다. 수신 규칙은 수신 메일을 Amazon S3 버킷으로 전송하거나, Amazon SNS 주제에 게시하거나, Amazon WorkMail 로 전송하거나, 메시지가 특정 이메일 주소로 전송되면 자동으로 반송 메시지를 전송하는 작업과 같이 고급 처리를 수행할 수 있습니다.
- IP 주소 필터(IP 기반 제어)는 광범위한 제어 수준을 제공하며 설치가 간단합니다. 이러한 필터를 사용하면 특정 IP 주소 또는 IP 주소 범위의 모든 메시지를 명시적으로 차단하거나 허용할 수 있습니다.

수신 규칙 또는 IP 주소 필터를 사용한 이메일 수신, 설정 및 구현에 대한 학습을 시작하려면 먼저 이메일 수신 개념 및 사용 사례를 읽어서 작동 방식 및 사용할 수 있는 다양한 방법에 대한 개요를 파악하세요. 다음으로 이메일 수신 설정하기 섹션은 이메일 수신 설정 필수 조건을 설명합니다. 그런 다음 이메일 수신 콘솔 연습 섹션은 수신 규칙 및 IP 주소 필터 구성에 사용되는 마법사에 대해 설명합니다.

> **Note**
>
> 이메일 수신은 계정이 SES가 이메일 수신을 지원하는에 AWS 리전 있는 경우에만 사용할 수 있습니다. 의 이메일 수신 엔드포인트 테이블에는 SES AWS 리전 가 이메일 수신을 지원하는 모든이 AWS 일반 참조 나열됩니다.

이 섹션의 주제:

- Amazon SES 이메일 수신 개념 및 사용 사례
- Amazon SES 이메일 수신 설정
- Amazon SES 이메일 수신 콘솔 연습
- Amazon SES 이메일 수신의 지표 보기

Amazon SES 이메일 수신 개념 및 사용 사례

Amazon SES를 이메일 수신기로 사용하는 경우 서비스에 메일로 수행할 작업을 알려야 합니다. 기본 방법인 수신 규칙을 사용하면 수신자 기반 제어를 통해 수신자를 기반으로 수행할 작업 세트를 지정함으로써 이메일 수신을 세밀하게 제어할 수 있습니다. 다른 방법인 IP 주소 필터는 광범위한 IP 기반 제어를 사용하여 원래 IP 주소 또는 주소 범위에 따라 메일을 차단하거나 허용할 수 있습니다.

Amazon SES가 수신된 이메일을 처리하는 방법에 대한 개요와 규칙 및 필터를 설정할 때 이메일을 수신, 필터링, 처리하는 방법을 고려하는 데 도움이 되는 사용 사례에서 두 가지 방법에 대해 모두 설명합니다.

이 섹션의 주제:

- 수신 규칙을 사용한 수신자 기반 제어
- IP 주소 필터를 사용한 IP 기반 제어
- 이메일 수신 프로세스
- Amazon SES 이메일 수신에 대한 사용 사례 및 제한
- 이메일 수신 인증 및 맬웨어 검사

수신 규칙을 사용한 수신자 기반 제어

수신 메일을 제어하는 기본 방법은 도메인, 하위 도메인 또는 이메일 주소(확인된 도메인 ID 중 하나에 속해야 함)를 포함하는 확인된 ID에 대해 순서가 지정된 작업 목록을 통해 메일이 처리되는 방식을 지정하는 것입니다. 이러한 작업은 규칙 세트 내에서 생성하는 수신 규칙에 정의되고 정렬됩니다.

옵션으로 발송된 메일을 받는 수신자가 조건에 지정된 수신자 자격 증명과 일치하는 경우에만 작업을 수행하도록 지정하는 방법으로 수신자 조건을 추가할 수도 있습니다. 예를 들어 example.com을 소유하는 경우, user@example.com에 대한 메일이 반송되도록 지정하고 example.com 및 해당 하위 도메인에 대한 다른 모든 메일이 전달되도록 지정할 수 있습니다.

또는, 수신자 조건을 추가하지 않으면 확인된 도메인에 속한 모든 이메일 주소, 도메인 및 하위 도메인 등 모든 항목에 작업이 적용됩니다. 수신 규칙에 적용할 수 있는 조치는 다음과 같습니다.

- 헤더 추가 작업 - 수신 이메일에 헤더를 추가합니다. 이 작업은 보통 다른 작업과 연계해서 사용합니다.
- 반송 응답 반환 작업 - 발신자에게 반송 응답을 돌려보내 이메일 수신을 거부하고 설정에 따라 Amazon SNS를 통해 사용자에게 알립니다.

- AWS Lambda 함수 호출 작업 - Lambda 함수를 통해 코드를 호출하고 선택적으로 Amazon SNS를 통해 알립니다.
- S3 버킷에 전송 작업 - 메일을 Amazon S3 버킷으로 전송하고 설정에 따라 Amazon SNS를 통해 알립니다.
- Amazon SNS 주제에 게시 작업 - Amazon SNS 주제에 전체 이메일을 게시합니다.

> **Note**
>
> SNS 작업은 Amazon SNS 알림에 이메일 콘텐츠 사본을 첨부합니다. 여기에 언급된 다른 Amazon SNS 알림 옵션은 이메일 전송에 대한 알림만을 제공하며, 여기에는 이메일 내용 자체가 아니라 이메일에 대한 정보가 포함되어 있습니다.

- 규칙 세트 중지 작업 - 해당 수신 규칙 세트의 평가를 중단하고, 설정에 따라 Amazon SNS를 통해 사용자에게 알립니다.
- Amazon WorkMail 통합 작업 - Amazon WorkMail로 메일을 처리합니다. Amazon WorkMail이 설정을 처리하므로 일반적으로 이 작업을 직접 사용할 필요는 없습니다.

수신 규칙이 규칙 세트로 그룹화됩니다. 기존 규칙 세트가 없는 경우 수신 규칙 생성을 시작하기 전에 먼저 규칙 세트를 생성해야 합니다. AWS 계정에 대해 여러 규칙 세트를 정의할 수 있지만 한 번에 하나의 규칙 세트만 활성화됩니다. 다음 그림에서는 수신 규칙, 규칙 세트 및 작업이 서로 어떻게 관련되어 있는지 보여줍니다.

IP 주소 필터를 사용한 IP 기반 제어

IP 주소 필터를 설정하여 메일 흐름을 제어할 수 있습니다. IP 주소 필터는 선택사항이며 특정 IP 주소 또는 특정 범위의 IP 주소에서 발신한 메일의 수락 또는 차단을 지정할 수 있게 합니다. IP 주소 필터에 차단 목록(수신 메일을 차단하려는 IP 주소) 및 허용 목록(항상 메일을 수락하려는 IP 주소)이 포함될 수 있습니다.

IP 주소 필터는 스팸을 차단하는 데 유용합니다. Amazon SES는 Spamhaus에 나열된 것을 포함하여 스팸을 보내는 것으로 알려진 IP 주소의 차단 목록을 자체적으로 유지합니다. 하지만 허용 목록에 추가하여 해당 IP 주소에서 메일을 받도록 선택할 수 있습니다. 차단된 IP 주소를 보여주는 로그가 없으므로 차단된 발신자가 관리자에게 관련 정보를 알려야 합니다. 또한 이를 통해 발신자는 자신의 IP 주소가 Spamhaus와 같은 차단 목록에 있는지 확인하고 목록에서 제외하도록 요청할 수 있습니다. 이렇게 하면 IP 주소 필터를 유지 관리할 필요가 없으며 이메일 전송 가능성이 향상된다는 점에서 관리자와 발신자 모두에게 도움이 됩니다.

> **Note**
>
> - IP 주소 필터 구성과 관계없이 Amazon EC2는 허용 목록에 없는 한 포트 25(메일 전송)의 아웃바운드 트래픽을 차단합니다. 자세한 내용은 AWS re:Post 문서를 참조하세요.

- 정해진 일부 IP 주소에서만 메일을 받고 싶다면 차단 목록에는 `0.0.0.0/0`를 입력하고, 허용 목록에는 신뢰하는 IP 주소를 입력하세요. 이 구성은 기본적으로 모든 IP 주소를 차단하고, 명시적으로 지정하는 IP 주소에서 발신하는 메일만 허용합니다.

이메일 수신 프로세스

Amazon SES가 도메인에 대한 이메일을 수신하면 다음과 같은 이벤트가 발생합니다.

1. Amazon SES는 먼저 발신자의 IP 주소를 확인합니다. Amazon SES는 다음과 같은 경우를 제외하고 메일이 이 단계를 통과하도록 허용합니다.
   - IP 주소가 차단 목록에 존재합니다.
   - IP 주소가 Amazon SES 차단 목록에 있지만 허용 목록에는 없습니다.
2. Amazon SES는 활성 규칙 세트를 검사하여 수신 규칙이 수신자 조건을 포함하고 있는지 확인합니다.
   - 수신자 조건이 있고 수신 이메일의 수신자 중 하나와 일치하는 경우 Amazon SES는 이메일을 수락합니다. 일치하는 항목이 없으면 Amazon SES는 이메일을 차단합니다.
   - 수신 규칙이 수신자 조건을 포함하지 않는 경우 Amazon SES는 메일을 수락합니다. 규칙의 모든 작업은 사용자가 소유한 확인된 모든 자격 증명에 적용됩니다.
3. Amazon SES는 이메일을 인증하고 해당 콘텐츠에서 스팸 및 맬웨어를 검사합니다.
   - Amazon SES로 이메일을 전송한 원격 호스트의 IP 주소는 SMTP 트랜잭션 중에 사용된 MAIL FROM의 도메인에 지정된 SPF 정책에 따라 검사됩니다.
   - 이메일의 헤더 섹션에 있는 DKIM 서명이 검사됩니다.
   - 콘텐츠 검사가 활성화된 경우 이메일 콘텐츠에서 스팸 및 맬웨어가 있는지 검사합니다.
   - 수신 규칙 평가 중에 이메일 인증 및 콘텐츠 검색 결과를 사용할 수 있습니다.

   자세한 정보는 이메일 인증 및 맬웨어 탐지을 참조하세요.
4. Amazon SES가 수락하는 이메일의 경우 활성 규칙 세트 내의 모든 수신 규칙이 사용자가 정의한 순서대로 적용됩니다. 각 수신 규칙 내에서 작업은 정의한 순서대로 실행됩니다.

Amazon SES 이메일 수신에 대한 사용 사례 및 제한

이 단원에서는 Amazon SES 이메일 수신에 대한 몇 가지 일반적인 고려 사항 및 사용 사례에 대해 설명합니다. 질문 및 답변 형식으로 제공되며, 이는 Amazon SES를 사용하여 사용자가 소유한 하나 이상

의 확인된 도메인을 대신하여 이메일을 수신하고 관리하는 데 도움이 되는지 판단할 수 있는 자주 묻는 질문과 사실입니다.

리전별 가용성

Amazon SES가 해당 리전에서 이메일 수신을 지원합니까?

Amazon SES는 특정 AWS 리전에서만 이메일 수신을 지원합니다. 이메일 수신이 지원되는 리전의 전체 목록은 AWS 일반 참조의 Amazon Simple Email Service 엔드포인트 및 할당량을 참조하세요.

POP 또는 IMAP 기반 이메일 클라이언트

수신 전자 메일을 받는 데 Microsoft Outlook을 사용할 수 있습니까?

Amazon SES에는 들어오는 이메일을 수신하기 위한 POP 또는 IMAP 서버가 포함되어 있지 않습니다. 즉, Microsoft Outlook과 같은 이메일 클라이언트를 사용하여 들어오는 이메일을 수신할 수 없습니다. 이메일 클라이언트를 사용하여 이메일을 전송하고 수신할 수 있는 해결 방법이 필요한 경우 Amazon WorkMail 사용을 고려해 보십시오.

다른 AWS 서비스 사용

권한을 적절히 설정하셨습니까?

메일을 S3 버킷으로 전송하거나 본인 소유가 아닌 Amazon SNS 주제로 게시하거나 Lambda 함수를 트리거하거나 사용자 지정 관리형 키를 사용하고자 하는 경우 Amazon SES에 해당 리소스에 대한 액세스 권한을 주어야 합니다. Amazon SES에 액세스 권한을 부여하려면 콘솔 또는 해당 AWS 서비스의 APIs에서 리소스에 대한 정책을 생성합니다. 자세한 정보는 권한 부여하기 단원을 참조하십시오.

이메일 콘텐츠

Amazon SES가 이메일 콘텐츠를 어떻게 전달하기를 바라십니까?

Amazon SES는 이메일 콘텐츠를 두 가지 방식으로 전달합니다. 즉, 이메일을 사용자가 지정하는 S3 버킷에 저장하거나 이메일 사본이 포함된 Amazon SNS 알림을 전송합니다. Amazon SES는 수정되지 않은 원시 이메일을 다목적 인터넷 전자 우편(MIME) 형식으로 전송합니다. MIME 형식에 대한 자세한 내용은 RFC 2045를 참조하세요.

수신하게 될 이메일의 크기는 얼마나 됩니까?

S3 버킷에 이메일을 저장하는 경우 이메일(헤더 포함)의 최대 용량은 40MB입니다. Amazon SNS 알림을 통해 이메일을 수신하는 경우 이메일(헤더 포함)의 최대 용량은 150KB입니다.

메일 처리 프로세스를 어떻게 트리거하고자 하십니까?

메일을 수신한 후에는 자체 코드로 처리하고 싶을 것입니다. 예를 들어 사용자 애플리케이션이 Base64 인코딩의 이메일을 표시할 수 있는 형식으로 변환해서 최종 사용자가 이메일 클라이언트로 확인할 수 있도록 할 것입니다. 이 프로세스를 시작하는 두 가지 방법이 있습니다.

- 이메일을 Amazon S3 버킷으로 전송하는 경우 애플리케이션이 S3 활동으로 발생하는 Amazon SNS 알림을 청취하고, 알림에서 이메일의 메시지 ID를 추출하고 이를 통해 Amazon S3에서 이메일을 가져오면 됩니다.

  또는 Lambda 함수를 작성하여 이메일 처리 프로세스를 수신 규칙에 포함시켜도 됩니다. 이 경우에는 수신 규칙이 우선 이메일을 Amazon S3에 쓴 다음 Lambda 함수를 트리거해야 합니다. Lambda 함수가 다른 작업의 실행에 영향을 미치는 결과물을 반환해야 하는지 여부에 따라서 수신 규칙 내에서 Lambda 작업을 동기식으로 또는 비동기식으로 실행할 수 있습니다. 해당 사용 사례에 반드시 동기식 실행이 필요한 경우가 아니라면 비동기식 실행 사용이 좋습니다. 에 대한 자세한 내용은 AWS Lambda 개발자 안내서를 AWS Lambda참조하세요.
- 이메일을 SNS 작업을 사용해 Amazon SNS 알림으로 전송하는 경우 애플리케이션이 Amazon SNS 알림을 청취하고 알림에서 이메일 메시지를 추출하면 됩니다.

이메일을 암호화하고자 하십니까?

Amazon SES는 AWS Key Management Service (AWS KMS)와 통합되어 선택적으로 S3 버킷에 쓰는 메일을 암호화합니다. Amazon SES는 Amazon S3 작성하기 전에 이메일을 암호화하는 데 사용합니다. 따라서 Amazon S3에서 메일을 가져온 다음 사용자 측에서 콘텐츠를 해독해야 합니다. 이 AWS SDK for Java 및 AWS SDK for Ruby은(는) 암호 해독을 처리할 수 있는 클라이언트를 제공합니다. Amazon SES는 이메일을 S3 버킷으로 전송하는 경우에만 이메일을 암호화할 수 있습니다.

원치 않는 메일

이메일 수신 프로세스의 어느 시점에서 원치 않는 메일을 차단하고자 하십니까?

발신자가 수신자에게 이메일을 전송하려고 시도할 때 발신자의 이메일 서버는 수신자의 서버와 명령 시퀀스를 교환합니다. 이러한 시퀀스를 SMTP 대화라고 합니다.

이메일 수신 프로세스의 두 시점에서 수신되는 이메일을 차단할 수 있는데 SMTP 대화 중과 SMTP 대화 이후, 이렇게 두 시점입니다. SMTP 대화 중에는 IP 주소 필터를 사용하여 메시지를 차단하고, SMTP 대화 후에는 거부 규칙을 사용하여 이메일을 차단합니다.

IP 주소 필터를 사용하면 특정 IP 주소에서 발신된 이메일을 차단할 수 있습니다. IP 주소 필터를 사용하여 원치 않는 메일을 차단하면 SMTP 대화 중 차단된 메시지에는 비용이 청구되지 않는다는 이점이

있습니다. IP 주소 필터 사용의 단점은 메시지의 실제 내용을 분석하지 않고 지정한 IP 주소에서 발신된 이메일을 차단한다는 점입니다. IP 주소 필터에 대한 자세한 내용은 IP 주소 필터 생성 콘솔 연습 단원을 참조하세요.

거부 규칙을 사용하면 메시지가 전송된 주소(혹은 도메인 또는 하위 도메인)를 기반으로 이메일 발신자에게 반송 메일 알림을 보낼 수 있습니다. 거부 규칙을 사용하면 발신자에게 반송 메일 알림을 보내기 전에 수신되는 메시지에 대해 추가 분석을 수행할 수 있다는 이점이 있습니다. 예를 들어 AWS Lambda 를 사용하여 메시지가 DKIM 인증에 실패하거나 스팸으로 식별된 경우에만 반송 메일 알림을 보낼 수 있습니다. 하지만 거부 규칙을 사용하면 SMTP 대화 이후에 거부 규칙이 처리되기 때문에 수신한 각 메시지에 대해 비용이 청구된다는 단점이 있습니다. Lambda를 사용하여 수신되는 메시지의 내용을 분석하는 경우에도 비용이 청구될 수 있습니다. 수신 규칙에 대한 자세한 내용은 수신 규칙 만들기 콘솔 연습 단원을 참조하세요. Lambda를 사용하여 수신되는 이메일 분석에 대한 자세한 내용은 Lambda 함수 예제 단원을 참조하십시오.

메일 스트림

메일 스트림을 어떻게 나누고 싶으십니까?

도메인은 다양한 종류의 메일을 수신합니다. 예를 들어 도메인이 수신하는 메일 중에서 user@example.com으로 가는 메일은 개인 수신함으로 들어가야 합니다. 한편 unsubscribe@example.com으로 가는 메일은 자동화 시스템으로 처리하는 것이 좋습니다. 이처럼 이메일을 종류에 따라 다르게 처리할 수 있도록 수신 규칙을 사용해 수신 메일을 분류할 수 있습니다. 수신 규칙 설정 방법에 대한 자세한 내용은 다음(수신 규칙 만들기)을 참조하세요.

이메일 수신 인증 및 맬웨어 검사

Amazon SES는 수신된 각 이메일을 인증하고 해당 이메일 콘텐츠에서 스팸 및 맬웨어를 선택적으로 검사합니다. SES는 이메일 인증 또는 콘텐츠 검색 결과에 따라 수신된 이메일에 대해 어떠한 작업도 수행하지 않습니다. 그러나 이러한 작업의 결과는 Amazon SNS 알림 또는 Amazon S3로 전송된 메시지의 헤더와 같은 SES 수신 규칙 작업에서 사용할 수 있는 속성으로 제공됩니다.

이메일 인증

Amazon SES는 SPF, DKIM 및 DMARC를 사용하여 수신된 각 이메일을 인증합니다. 각 인증 메커니즘의 결과는 활성 수신 규칙 세트의 규칙을 평가하는 과정에서 SES가 발송하는 Amazon SNS 알림에 제공됩니다. 또한 Amazon S3의 이메일 사본을 수신하도록 선택한 경우 이메일 인증 결과는 SES가 이메일의 헤더 섹션에 추가하는 `Authentication-Results` 헤더에 캡처됩니다.

```
Authentication-Results: example.com;
```

```
spf=pass (spfCheck: 10.0.0.1 is permitted by domain of example.com) client-ip=10.0.0.1;
 envelope-from=example@example.com; helo=10.0.0.1;
dkim=pass header.i=example.com;
dkim=permerror header.i=some-example.com;
dmarc=pass header.from=example@example.com;
```

`Authentication-Results` 헤더는 RFC 8601에 설명되어 있습니다.

스팸 및 맬웨어 탐지를 위한 이메일 콘텐츠 검사

Amazon SES는 이메일과 일치하는 수신 규칙의 ScanEnabled(API) 또는스팸 및 바이러스 검사 속성 값에 따라 수신된 이메일 콘텐츠에서 맬웨어를 검사합니다. 기본적으로 SES는 수신된 이메일 콘텐츠에서 맬웨어를 검사합니다. 특정 수신 규칙과 일치하는 수신된 이메일에 대한 콘텐츠 검사를 비활성화하려면 API를 사용하는 경우 수신 규칙의 ScanEnabled 플래그를 false로 지정하거나 콘솔을 사용하는 경우 스팸 및 바이러스 검사 확인란을 선택 해제해야 합니다. 이메일과 일치하는 수신 규칙에 대한 검사를 활성화한 경우 콘텐츠 검색 결과는 활성 수신 규칙 세트의 규칙을 평가하는 과정에서 SES가 발송하는 Amazon SNS 알림에 제공됩니다. 또한 Amazon S3의 이메일 사본을 수신하도록 선택한 경우 콘텐츠 검사 결과는 SES가 이메일의 헤더 섹션에 추가하는 `X-SES-Spam-Verdict` 및 `X-SES-Virus-Verdict` 헤더에 캡처됩니다.

```
X-SES-Spam-Verdict: PASS
X-SES-Virus-Verdict: FAIL
```

위의 헤더에 사용할 수 있는 값은 다음에 나열되어 있습니다.

- 스팸
- 바이러스

이제 이메일 수신 개념, 작동 방식 및 사용 사례에 대해 이해했으므로 이메일 수신 설정하기로 이동하여 시작할 수 있습니다.

Amazon SES 이메일 수신 설정

이 섹션에서는 메일을 수신하도록 Amazon SES를 구성하기 전에 필요한 사전 요구 사항에 대해 설명합니다. Amazon SES 작동 방식에 대한 개념을 이해하고 이메일을 수신, 필터링 및 처리하는 방법을 고려하기 위하여 이메일 수신 개념 및 사용 사례을(를) 읽는 것이 중요합니다.

규칙 세트, 수신 규칙 및 IP 주소 필터를 생성하여 이메일 수신을 구성하기 전에 먼저 다음 설정 전제 조건을 완료해야 합니다.

- DNS 레코드를 게시하여 Amazon SES을 포함하는 도메인을 확인하여 소유하고 있는지 입증합니다.
- MX 레코드를 게시하여 Amazon SES가 도메인에 대한 이메일을 수신하도록 허용합니다.
- 수신 규칙 작업을 실행하려면 Amazon SES에 다른 AWS 리소스에 액세스할 수 있는 권한을 부여합니다.

도메인 자격 증명을 만들고 확인하면 확인 프로세스를 완료하기 위해 DNS 설정에 레코드를 게시하지만 이메일 수신을 사용하기에는 충분하지 않습니다. 이메일 수신과 관련하여 사용자 지정 발신 메일 도메인을 지정하기 위해 MX 레코드를 게시해야 합니다. 이 레코드는 도메인의 DNS 설정에서 SES가 도메인에 대한 이메일을 수신하도록 허용하는 데 사용됩니다. Amazon SES에 해당 작업에 필요한 각 AWS 서비스를 사용할 권한이 없는 한 수신 규칙에서 선택한 작업이 작동하지 않으므로 권한을 부여해야 합니다.

이메일 수신을 사용하기 위해 필요한 세 가지 사전 조건은 다음 주제에서 설명합니다.

- Amazon SES 이메일 수신을 위한 도메인 확인
- Amazon SES 이메일 수신을 위해 MX 레코드 게시하기
- Amazon SES에 이메일 수신을 위한 권한 부여

Amazon SES 이메일 수신을 위한 도메인 확인

도메인의 이메일을 전송 또는 수신할 때 Amazon SES를 사용할 경우 먼저 해당 도메인의 소유 사실을 증명해야 합니다. 확인 절차는 SES로 도메인 확인 절차를 시작한 다음, 사용하는 확인 방법에 따라 DNS 레코드(CNAME 또는 TXT)를 DNS 제공자에게 게시하는 것을 포함합니다.

콘솔을 통해 Easy DKIM 또는 자체 DKIM 사용(BYODKIM)으로 도메인을 확인하고, DNS 레코드를 쉽게 복사하여 DNS 공급자에게 게시할 수 있습니다. 이 작업을 수행하는 방법은 도메인 자격 증명 생성 섹션을 참조하세요. 필요에 따라 SES `VerifyDomainDkim` 또는 `VerifyDomainIdentity` API를 사용할 수 있습니다.

도메인 또는 이메일 주소는 SES 콘솔의 확인된 자격 증명 테이블에서 상태를 확인하거나, SES `GetIdentityVerificationAttributes` 또는 `GetEmailIdentity` API를 사용하여 쉽게 확인할 수 있습니다.

Amazon SES 이메일 수신을 위해 MX 레코드 게시하기

메일 교환기 레코드(MX 레코드)는 어떤 메일 서버가 사용자의 도메인에서 전송된 이메일을 허용할 수 있는지를 지정하는 구성합니다.

Amazon SES가 수신 이메일을 관리하도록 하려면 도메인의 DNS 구성에 MX 레코드를 추가해야 합니다. 생성한 MX 레코드는 Amazon SES를 사용하는 AWS 리전에 대한 이메일을 수신하는 엔드포인트를 나타냅니다. 예를 들어 미국 서부(오레곤) 리전에 대한 엔드포인트는 inbound-smtp.us-west-2.amazonaws.com입니다. 전체 엔드포인트 목록은 SES 리전 및 엔드포인트 단원을 참조하세요.

> **Note**
>
> Amazon SES에서 이메일을 수신하는 엔드포인트는 IMAP 또는 POP3 이메일 서버가 아닙니다. 이러한 URL은 이메일 클라이언트의 수신되는 메일 서버로 사용할 수 없습니다. 이메일 클라이언트를 사용하여 이메일을 전송하고 수신할 수 있는 해결 방법이 필요한 경우 Amazon WorkMail 사용을 고려해 보십시오.

다음 절차에는 MX 레코드를 생성하기 위한 일반적인 단계가 포함되어 있습니다. MX 레코드를 생성하는 특정 절차는 DNS 또는 호스팅 공급자에 따라 다릅니다. 도메인에 대한 DNS 구성에 MX 레코드를 추가하는 것에 관한 정보는 공급자의 설명서를 참조하세요.

> **Note**

도메인의 DNS 구성에 MX 레코드를 추가하려면

1. (사전 조건) 이러한 절차를 완료하려면 도메인의 DNS 레코드를 수정해야 합니다. DNS 레코드에 액세스할 수 없거나 그렇게 하는 것이 불편한 경우 시스템 관리자에게 문의하여 도움을 받으세요.
2. DNS 공급자의 관리 콘솔에 로그인합니다.
3. 새 MX 레코드를 생성합니다.
4. MX 레코드 이름으로 도메인을 입력합니다. 예를 들어, Amazon SES에서 example.com 도메인으로 보낸 이메일을 관리하도록 하려면 다음을 입력합니다.

   ```
   example.com.
   ```

   > **Note**
   >
   > DNS 공급자에 따라: 1) 도메인 확장의 끝에 `.` 있는 후행이 필요하지 않을 수 있습니다. 2) 이름 필드는 호스트, 도메인 또는 메일 도메인이라고 할 수 있습니다.

5. Type(유형)으로 MX를 선택합니다.

> **Note**
>
> 일부 DNS 공급자는 Type(유형) 필드를 Record Type(레코드 유형) 또는 유사한 이름으로 지칭합니다.

6. Value(값)로 다음을 입력합니다.

```
10 inbound-smtp.region.amazonaws.com
```

앞의 예에서 *region*을 Amazon SES에서 사용하는 AWS 리전에 대한 이메일을 수신하는 엔드포인트의 주소로 바꿉니다. 예를 들어 미국 동부(버지니아 북부) 리전을 사용하는 경우 *##*을 `us-east-1`로 바꿉니다. 이메일 수신 엔드포인트의 전체 목록은 SES 리전 및 엔드포인트 단원을 참조하세요.

> **Note**
>
> 일부 DNS 공급자의 관리 콘솔에는 별도의 레코드 Value(값) 및 레코드 Priority(우선 순위) 필드가 있습니다. 이 경우에 해당하면, DNS 공급자의 경우 Priority(우선 순위) 값에 `10`을 입력하고 Value(값)에 수신 메일 엔드포인트 URL을 입력합니다.

> **Important**
>
> MX 레코드 생성하는 구체적인 절차는 DNS 또는 호스팅 공급자에 따라 다릅니다. 도메인의 DNS 구성에 MX 레코드를 추가하는 방법에 대한 자세한 내용은 공급자 설명서를 참조하거나 공급자에게 문의하십시오.

다양한 공급자를 위한 MX 레코드 생성 지침

도메인의 MX 레코드를 생성하는 절차는 이용하는 DNS 공급자에 따라 다릅니다. 이 단원에는 몇몇 공통 DNS 공급자의 설명서에 대한 링크가 포함되어 있습니다. 이것이 공급자의 전체 목록은 아닙니다. 공급자가 아래 목록에 없더라도 Amazon SES에서 여전히 이를 사용할 수 있습니다. 이 목록에 포함되어 있다고 해서 어떤 회사의 제품 또는 서비스를 승인 또는 추천하는 것은 아닙니다.

DNS/호스팅 공급자 이름	설명서 링크
Amazon Route 53	Amazon Route 53 콘솔을 사용하여 레코드 생성
GoDaddy	MX 레코드 추가(외부 링크)
DreamHost	MX 레코드를 변경하는 방법(외부 링크)
Cloudflare	이메일 레코드 설정(외부 링크)
HostGator	MX 레코드 변경 - Windows(외부 링크)
Namecheap	메일 서비스에 필요한 MX 레코드를 설정하는 방법(외부 링크)
Names.co.uk	도메인의 DNS 설정 변경(외부 링크)
Wix	Wix 계정에서 MX 레코드 추가 또는 업데이트(외부 링크)

Amazon SES에 이메일 수신을 위한 권한 부여

Amazon Simple Storage Service(Amazon S3) 버킷으로 이메일을 보내거나 AWS Lambda 함수를 호출하는 등 SES에서 이메일을 수신할 때 수행할 수 있는 일부 작업에는 특별한 권한이 필요합니다. 이 섹션에서는 일반 사용 사례에 대한 정책 예제를 보여줍니다.

이 섹션의 주제:

- S3 버킷으로 전송 작업을 위한 IAM 역할 권한 설정
- SES에 S3 버킷에 작성할 수 있는 권한 부여
- SES에 AWS KMS 키 사용 권한 부여
- SES에 AWS Lambda 함수를 호출할 수 있는 권한 부여
- SES에 다른 AWS 계정에 속한 Amazon SNS 주제에 게시할 수 있는 권한 부여

S3 버킷으로 전송 작업을 위한 IAM 역할 권한 설정

이 IAM 역할에는 다음 사항이 적용됩니다.

- S3 버킷으로 전송 작업에만 사용할 수 있습니다.
- SES the section called “이메일 수신”을 사용할 수 없는 리전에 있는 S3 버킷에 쓰려는 경우 이 역할을 사용해야 합니다.

S3 버킷에 쓰려는 경우 S3 버킷으로 전송 작업의 관련 리소스에 액세스할 수 있는 권한을 IAM 역할에 제공할 수 있습니다. 또한 다음 섹션에 설명된 대로 IAM 신뢰 정책을 통해 해당 역할을 맡아 작업을 수행할 수 있는 권한을 SES에 부여해야 합니다.

이 권한 정책은 IAM 역할의 인라인 정책 편집기에 붙여넣어야 합니다. S3 버킷으로 전송 작업의 내용을 참조하고 IAM 역할 항목에 제공된 단계를 따르세요. (다음 예에는 SNS 주제 알림 또는 S3 작업에서 고객 관리형 키를 사용하려는 경우의 선택적 권한도 포함되어 있습니다.)

> **Note**
>
> - 와 같이 S3 버킷 정책에서 SES 서비스만 허용하여 IAM 역할을 지정하지 않고 S3 작업을 설정할 수 있습니다the section called “SES에 S3 버킷에 작성할 수 있는 권한 부여”. 이는 교차 계정 시나리오에서도 작동합니다.
> - S3 작업에 대한 IAM 역할을 지정하는 경우 SES는 'PutObject' 작업에 대해 해당 역할을 수임하며, 여기에 지정된 IAM 권한은 동일한 계정 사용에 충분합니다. 그러나 교차 계정 사용의 경우 IAM 역할이 버킷에서 'PutObject'가 되도록 허용하는 추가 버킷 정책이 필요합니다. 이는 버킷 소유자가 교차 계정 버킷 권한을 부여하는 것에 설명된 대로 교차 계정 버킷 권한을 부여하는 버킷 소유자가 지정합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3Access",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        },
        {
            "Sid": "SNSAccess",
```

```
            "Effect": "Allow",
            "Action": "sns:Publish",
            "Resource": "arn:aws:sns:us-east-1:111122223333:my-topic"
        },
        {
            "Sid": "KMSAccess",
            "Effect": "Allow",
            "Action": "kms:GenerateDataKey*",
            "Resource": "arn:aws:kms:us-east-1::111122223333:key/key-id"
        }
    ]
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- *amzn-s3-demo-bucket*을 작성하려는 S3 버킷의 이름으로 바꿉니다.
- *region*을 수신 규칙을 생성한 AWS 리전 로 바꿉니다.
- *111122223333*을 AWS 계정 ID로 바꿉니다.
- *my-topic*을 알림을 게시할 SNS 주제의 이름으로 바꿉니다.
- *key-id*를 KMS 키의 ID로 바꿉니다.

S3 작업 IAM 역할에 대한 신뢰 정책

SES가 해당 역할을 맡을 수 있도록 IAM 역할의 신뢰 관계에 다음 신뢰 정책을 추가해야 합니다.

> **Note**
>
> 이 신뢰 정책의 수동 추가는 S3 버킷으로 전송 작업 워크플로의 IAM 역할 항목에 제공된 단계를 사용하여 SES 콘솔에서 IAM 역할을 생성하지 않은 경우에만 필요합니다. 콘솔에서 IAM 역할을 생성하면 이 신뢰 정책이 자동으로 생성되어 적용되므로 이 단계가 필요하지 않게 됩니다.

JSON

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

```
    "Statement": [
        {
            "Sid": "AllowSESAssume",
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                  "AWS:SourceAccount":"111122223333",
                  "AWS:SourceArn": "arn:aws:ses:region:111122223333:receipt-rule-
set/rule_set_name:receipt-rule/receipt_rule_name"
                }
            }
        }
    ]
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- *region*을 수신 규칙을 생성한 AWS 리전 로 바꿉니다.
- *111122223333*을 AWS 계정 ID로 바꿉니다.
- *rule_set_name*을 Amazon S3 버킷 작업에 대한 전송을 포함하는 수신 규칙이 포함된 규칙 세트의 이름으로 바꿉니다.
- *receipt_rule_name*을 Amazon S3 버킷 작업에 대한 전송을 포함하는 수신 규칙 이름으로 바꿉니다.

SES에 S3 버킷에 작성할 수 있는 권한 부여

S3 버킷에 다음 정책을 적용하면 SES 이메일 수신을 사용할 수 있는 리전에 있는 한 해당 버킷에 쓸 수 있는 권한을 SES에 부여합니다. 이메일 수신 리전 외부의 버킷에 작성하려는 경우에는 S3 버킷으로 전송 작업을 위한 IAM 역할 권한 설정의 내용을 참조하세요. Amazon S3로 수신 이메일을 전송하는 수신 규칙을 만드는 방법에 대한 자세한 내용은 S3 버킷으로 전송 작업 섹션을 참조하세요.

S3의 버킷 정책에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 정책 및 사용자 정책 사용을 참조하세요.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AllowSESPuts",
      "Effect":"Allow",
      "Principal":{
        "Service":"ses.amazonaws.com"
      },
      "Action":"s3:PutObject",
      "Resource":"arn:aws:s3:::amzn-s3-demo-bucket/*",
      "Condition":{
        "StringEquals":{
          "AWS:SourceAccount":"111122223333",
          "AWS:SourceArn": "arn:aws:ses:region:111122223333:receipt-rule-
set/rule_set_name:receipt-rule/receipt_rule_name"
        }
      }
    }
  ]
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- *amzn-s3-demo-bucket*을 작성하려는 S3 버킷의 이름으로 바꿉니다.
- *region*을 수신 규칙을 생성한 AWS 리전으로 바꿉니다.
- *111122223333*을 AWS 계정 ID로 바꿉니다.
- *rule_set_name*을 Amazon S3 버킷 작업에 대한 전송을 포함하는 수신 규칙이 포함된 규칙 세트의 이름으로 바꿉니다.
- *receipt_rule_name*을 Amazon S3 버킷 작업에 대한 전송을 포함하는 수신 규칙 이름으로 바꿉니다.

SES에 AWS KMS 키 사용 권한 부여

SES가 이메일을 암호화하기 위해서는 수신 규칙을 설정할 때 지정한 AWS KMS 키에 대한 사용 권한이 있어야 합니다. 계정의 기본 KMS 키(aws/ses)를 사용하거나, 직접 만드는 사용자 지정 관리형 키를

사용할 수 있습니다. 기본 KMS 키를 사용하는 경우 SES에 사용 권한을 부여하는 추가 절차가 필요하지 않습니다. 사용자 지정 관리형 키를 사용하는 경우 키의 정책에 설명을 추가하여 SES에 사용 권한을 부여해야 합니다.

다음 정책 설명을 키 정책에 사용하여 도메인에서 이메일을 수신할 때 SES가 사용자 지정 관리형 키를 사용할 수 있도록 허용합니다.

```
{
  "Sid": "AllowSESToEncryptMessagesBelongingToThisAccount",
  "Effect": "Allow",
  "Principal": {
    "Service":"ses.amazonaws.com"
  },
  "Action": [
    "kms:GenerateDataKey*"
  ],
  "Resource": "*",
  "Condition":{
        "StringEquals":{
          "AWS:SourceAccount":"111122223333",
          "AWS:SourceArn": "arn:aws:ses:region:111122223333:receipt-rule-
set/rule_set_name:receipt-rule/receipt_rule_name"
        }
      }
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- *region*을 수신 규칙을 생성한 AWS 리전으로 바꿉니다.
- *111122223333*을 AWS 계정 ID로 바꿉니다.
- *rule_set_name*을 이메일 수신과 연결한 수신 규칙이 포함된 규칙 세트의 이름으로 바꿉니다.
- *receive_rule_name*을 이메일 수신과 연결한 수신 규칙의 이름으로 바꿉니다.

AWS KMS 를 사용하여 서버 측 암호화가 활성화된 S3 버킷으로 암호화된 메시지를 전송하는 경우 정책 작업을 추가해야 합니다`"kms:Decrypt"`. 앞의 예를 다시 사용하여 이 작업을 정책에 추가하면 다음과 같이 표시됩니다.

```
{
  "Sid": "AllowSESToEncryptMessagesBelongingToThisAccount",
```

```
  "Effect": "Allow",
  "Principal": {
    "Service":"ses.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey*"
  ],
  "Resource": "*",
  "Condition":{
        "StringEquals":{
          "AWS:SourceAccount":"111122223333",
          "AWS:SourceArn": "arn:aws:ses:region:111122223333:receipt-rule-
set/rule_set_name:receipt-rule/receipt_rule_name"
        }
      }
}
```

AWS KMS 키에 정책을 연결하는 방법에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 에서 키 정책 사용을 AWS KMS 참조하세요.

SES에 AWS Lambda 함수를 호출할 수 있는 권한 부여

SES가 AWS Lambda 함수를 호출하도록 하려면 SES 콘솔에서 수신 규칙을 생성할 때 함수를 선택할 수 있습니다. 이렇게 하면 SES가 자동으로 함수에 필요한 권한을 추가합니다.

또는 AWS Lambda API에서 `AddPermission` 작업을 사용하여 정책을 함수에 연결할 수 있습니다. 다음 `AddPermission` API 호출은 SES에 Lambda 함수 호출 권한을 부여합니다. Lambda 함수에 정책을 연결하는 방법에 대한 자세한 내용은 AWS Lambda 개발자 가이드의 AWS Lambda 권한을 참조하세요.

```
{
  "Action": "lambda:InvokeFunction",
  "Principal": "ses.amazonaws.com",
  "SourceAccount": "111122223333",
  "SourceArn": "arn:aws:ses:region:111122223333:receipt-rule-set/rule_set_name:receipt-
rule/receipt_rule_name",
  "StatementId": "GiveSESPermissionToInvokeFunction"
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- *region*을 수신 규칙을 생성한 AWS 리전으로 바꿉니다.
- *111122223333*을 AWS 계정 ID로 바꿉니다.
- *## ## ##*을 Lambda 함수를 만든 수신 규칙을 포함하는 규칙 세트 이름으로 바꿉니다.
- *receipt_rule ##*을 Lambda 함수를 포함하는 수신 규칙 이름으로 바꿉니다.

SES에 다른 AWS 계정에 속한 Amazon SNS 주제에 게시할 수 있는 권한 부여

별도의 AWS 계정에 있는 주제에 알림을 게시하려면 Amazon SNS 주제에 정책을 연결해야 합니다. SNS 주제는 도메인 및 수신 규칙 세트와 같은 리전에 있어야 합니다.

다음 정책은 SES에 별도의 AWS 계정의 Amazon SNS 주제에 게시할 수 있는 권한을 부여합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": "SNS:Publish",
            "Resource": "arn:aws:sns:us-east-1:111122223333:topic_name",
            "Condition": {
                "StringEquals": {
                "AWS:SourceAccount": "444455556666",
                "AWS:SourceArn": "arn:aws:ses:us-east-1:777788889999:receipt-
rule-set/rule_set_name:receipt-rule/rule_name"
                }
            }
        }
    ]
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- *topic_region*을 Amazon SNS 주제 AWS 리전 가 생성된 로 바꿉니다.
- *sns_topic_account_id*를 Amazon SNS 주제를 소유한 AWS 계정의 ID로 바꿉니다.

- *topic_name*을 알림을 게시할 Amazon SNS 주제의 이름으로 바꿉니다.
- *aws_account_id*를 이메일을 수신하도록 구성된 AWS 계정의 ID로 바꿉니다.
- *receipt_region*을 수신 규칙을 생성한 AWS 리전 로 바꿉니다.
- *rule_set_name*을 Amazon SNS 주제 작업에 대한 게시를 만든 수신 규칙을 포함하는 규칙 세트 이름으로 바꿉니다.
- *receipt_rule_name*을 Amazon SNS 주제 작업에 대한 게시를 포함하는 수신 규칙 이름으로 바꿉니다.

Amazon SNS 주제가 서버 측 암호화 AWS KMS 에를 사용하는 경우 AWS KMS 키 정책에 권한을 추가해야 합니다. AWS KMS 키 정책에 다음 정책을 연결하여 권한을 추가할 수 있습니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSESToUseKMSKey",
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon SES 이메일 수신 콘솔 연습

이 단원에서는 이메일 수신을 관리하기 위해 수신 규칙및 IP 주소 필터를 구성하는 데 사용되는 이메일 수신 콘솔 마법사에 대해 설명합니다. 콘솔 마법사를 사용하기 전에 이메일 수신의 작동 방식에 대한 개념을 이해하기 위해 이메일 수신 개념 및 사용 사례을 읽고 설정 사전 요구 사항을 완료하기 위해 이메일 수신 설정하기를 모두 읽어야 합니다.

수신 규칙 및 IP 주소 필터를 구성하기 위한 콘솔 마법사는 다음과 같이 설명됩니다.

- 수신 규칙 만들기 콘솔 연습
- IP 주소 필터 생성 콘솔 연습

수신 규칙 만들기 콘솔 연습

이 섹션에서는 Amazon SES 콘솔을 사용하여 수신 규칙을 만들고 정의하는 방법을 설명합니다. 수신 규칙 작동 방식을 이해하기 위한 핵심 사항은 다음과 같습니다.

- 규칙 세트에는 정렬된 수신 규칙 세트가 포함됩니다. 수신 규칙에는 정렬된 일련의 작업이 포함됩니다.
- 수신 규칙은 사용자가 지정한 일련의 작업 목록을 실행하여 수신 메일을 처리하는 방법을 Amazon SES에 알려줍니다.
- 이 정렬된 작업 목록은 먼저 수신자 조건과 일치하는지 여부에 따라 선택적으로 만들 수 있습니다. 지정하지 않으면 확인된 도메인에 속하는 모든 자격 증명에 작업이 적용됩니다.
- 수신 규칙은 규칙 세트라는 컨테이너에서 만들어지고 정의됩니다. 여러 개의 규칙 세트를 만들 수 있지만 한 번에 하나만 활성화할 수 있습니다.
- 활성 규칙 세트 내의 수신 규칙은 지정한 순서대로 실행됩니다.
- 수신 규칙을 만들기 전에 먼저 규칙 세트를 만들어 수신 규칙을 포함해야 합니다.

Amazon Simple Email Service API 참조에 설명된 대로 `CreateReceiptRuleSet` API를 사용하여 빈 수신 규칙 세트를 선택적으로 만들 수 있습니다. 이후 Amazon SES 콘솔 또는 `CreateReceiptRule` API를 사용해 세트에 수신 규칙을 추가할 수 있습니다.

연습을 진행하기 전에 수신자 기반 이메일 수신을 사용하는 데 필요한 모든 필수 구성 요소를 충족했는지 확인하세요. 또한

사전 조건

수신 규칙을 사용하여 수신자 기반 이메일 제어를 설정하려면 다음 사전 요구 사항을 충족해야 합니다.

1. Amazon SES AWS 리전 가 이메일 수신을 지원하는에 엔드포인트가 있는지 확인합니다. 의 이메일 수신 엔드포인트 테이블에는 SES가 이메일 수신을 지원하는 모든에 대한 이메일 수신 엔드포인트 AWS 리전 가 AWS 일반 참조 나열됩니다.
2. Amazon SES에서 먼저 도메인 ID를 만들고 확인해야 합니다.

3. 다음으로 도메인의 DNS 설정에 MX 레코드를 게시하여 도메인에 대한 메일을 수락할 수 있는 메일 서버를 지정해야 합니다 (MX 레코드는 Amazon SES를 사용하는 AWS 리전의 메일을 수신하는 Amazon SES 엔드포인트를 참조해야 합니다.)
4. 마지막으로 수신 규칙 작업을 실행하려면 Amazon SES에 다른 리소스에 액세스할 수 있는 권한을 부여해야 합니다. AWS

규칙 세트 및 수신 규칙 만들기

이 연습은 먼저 규칙을 포함하는 규칙 세트를 만들고 규칙 만들기 마법사를 사용하여 수신 규칙을 만들고 정의하고 정렬합니다. 마법사에는 규칙 설정을 정의하고, 수신자 조건을 추가하고, 작업을 추가하고, 모든 설정을 검토할 수 있는 네 개의 화면이 있습니다.

콘솔을 사용하여 규칙 세트 및 수신 규칙을 만들려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 이메일 수신에서 Email Receiving(이메일 수신)을 선택합니다.

> **Note**
>
> 계정이 SES에서 이메일 수신을 지원하지 않는 AWS 리전 에 있는 경우 SES 콘솔의 왼쪽 탐색 창에 이메일 수신이 표시되지 않습니다. the section called “사전 조건”에 나열된 첫 번째 항목을 참조하세요.

3. Email receiving(이메일 수신) 창의 Receipt rule sets(수신 규칙 세트) 탭에서 Create rule set(규칙 세트 만들기)를 선택합니다.
4. 규칙 세트에 고유한 이름을 입력하고 Create rule set(규칙 세트 만들기)를 선택합니다.
5. Create rule(규칙 만들기)을 선택하면 규칙 만들기 마법사가 열립니다.
6. 규칙 설정 정의 페이지의 수신 규칙 세부 정보에서 Rule name(규칙 이름)을 입력합니다.
7. 만든 후 Amazon SES에서 이 규칙을 실행하지 않도록 하려면 상태에서 Enabled(활성화됨) 확인란을 선택 취소합니다. 또는 이 옵션을 선택한 상태로 둡니다.
8. (선택 사항) Amazon SES가 보안 연결을 통하지 않고 전송되는 수신 메시지를 거부할 수 있도록 하려면 보안 및 보호 옵션의 Transport Layer Security(TLS)에서 Required(필수)를 선택합니다.
9. (선택 사항) Amazon SES에서 수신 이메일에 대한 스팸 및 바이러스 검사를 수행하려면 스팸 및 바이러스 검색에 대해 Enabled(활성화)를 선택합니다.
10. 다음 단계로 진행하려면 Next(다음)를 선택합니다.

11. (선택 사항) 수신자 조건 추가 페이지에서 다음 절차에 따라 하나 이상의 수신자 조건을 지정합니다. 수신 규칙당 최대 100명의 수신자 조건을 지정할 수 있습니다.

    a. 수신자 조건에서 Add new recipient condition(새로운 수신자 조건 추가)를 선택하여 수신 규칙을 적용할 수신 이메일주소나 도메인을 지정합니다. 다음 표는 주소 user@example.com을 사용하여 수신자 조건을 지정하는 방식을 나타낸 것입니다.

다음을 수행하려는 경우	지정할 수신자	Notes
특정 이메일 주소와 일치시키는 경우	user@example.com을 선택합니다.	여러 레이블이 포함된 다양한 형식의 주소(user+123@example.com, user+xyz@example.com 등)와도 일치시킵니다. 하지만 단일 레이블이 포함된 주소 하나를 지정하는 경우에는 해당하는 특정 주소만 일치시킵니다.
하위 도메인의 주소를 제외한 도메인 내 모든 주소와 일치시키는 경우	example.com	
상위 도메인의 주소를 제외한 특정 하위 도메인 내 모든 주소와 일치시키는 경우	subdomain.example.com	
상위 도메인의 주소를 제외한 모든 하위 도메인 내 모든 주소와 일치시키는 경우	.example.com을 선택합니다	도메인 이름 앞에 마침표(.)를 입력합니다.
도메인 내 모든 주소와 모든 하위 도메인 내 모든 주소를 일치시키는 경우	example.com .example.com을 선택합니다	하나는 도메인 이름으로, 나머지 하나는 도메인 이름 앞에 마침표(.)를 입력하여 수신자 2개를 따로 만듭니다.
확인된 모든 도메인 내 모든 수신자를 일치시키는 경우	[없음]	수신자 필드를 비워둡니다.

다음을 수행하려는 경우	지정할 수신자	Notes

> **Important**
>
> 다수의 Amazon SES 계정이 공통 도메인을 통해 이메일을 수신하는 경우(예: 한 회사의 여러 팀들이 각각 별도의 Amazon SES 계정을 가지고 있는 경우) Amazon SES는 각 계정마다 일치하는 수신 규칙을 모두 동시에 처리합니다. 이때 하나의 계정이 반송 메일을 생성하는 반면 다른 계정이 이메일을 허용하는 상황이 발생할 수도 있습니다. 따라서 조직 내에서 Amazon SES를 사용하는 다른 팀과 협의하여 각 계정마다 고유한 수신 규칙을 사용하고 이러한 규칙이 서로 겹치지 않도록 하는 것이 바람직합니다. 이러한 상황에서는 그룹 또는 팀에 고유한 이메일 주소나 하위 도메인만 사용하도록 수신 규칙을 구성하는 것이 가장 좋습니다.

b. 추가할 각 수신자 조건에 대해 이 단계를 반복합니다. 수신자 조건의 추가를 마치면 Next(다음)를 선택합니다.

12. 작업 추가 페이지에서 다음 절차를 따라 수신 규칙에 작업을 하나 이상 추가합니다.

a. 새 작업 추가 메뉴를 열고 다음 작업 유형 중 하나를 선택합니다.

- 헤더 추가 - 이 작업은 수신 이메일에 사용자 지정 헤더를 추가합니다.
- 반송 응답 반환 - 이 작업은 발신자에게 반송 응답을 반환하여 이메일 수신을 거부합니다.
- Lambda 함수 호출 -이 작업은 AWS Lambda 함수를 통해 코드를 호출합니다.
- S3 버킷으로 전송 - 수신한 이메일을 Amazon Simple Storage Service(S3) 버킷에 저장합니다.
- Amazon SNS 주제에 게시하기 - 이 작업은 Amazon Simple Notification Service(SNS) 주제로 이메일 전체를 게시합니다.
- 규칙 세트 중지 - 수신 규칙 세트의 평가가 종료됩니다.
- Amazon WorkMail과 통합 - 이 작업은 Amazon WorkMail과 통합합니다.

이러한 작업에 대한 자세한 내용은 작업 옵션 섹션을 참조하세요.

b. 정의할 각 작업에 대해 이 단계를 반복합니다. 여러 작업이 정의되어 있는 경우 작업 컨테이너 내에서 위쪽/아래쪽 화살표를 사용하여 작업을 재정렬할 수 있습니다. Next(다음)를 선택하여 검토 페이지로 이동합니다.

13. 검토 페이지에서 규칙의 설정과 작업을 검토합니다. 변경을 해야 하는 경우 창의 왼쪽에 있는 Edit(편집) 옵션을 선택하여 편집하려는 내용이 포함된 단계로 바로 이동합니다. 재정렬 열의 위쪽/아래쪽 화살표를 사용하여 검토 페이지의 작업 테이블에 나열된 작업 순서를 선택적으로 변경할 수 있습니다.
14. 계속 진행할 준비가 되었으면 Create rule(규칙 만들기)을 선택합니다.
15. 규칙 세트를 즉시 적용하려면 해당 규칙 세트의 확인 페이지에서 Set as active(활성으로 설정)를 선택합니다.

만든 후 규칙 수정

규칙 세트를 만든 후에는 규칙 세트와 규칙 세트에 포함된 수신 규칙을 모두 편집할 수 있습니다. 편집이 가능할 뿐만 아니라 규칙 세트나 규칙을 복제하여 새 규칙을 신속하게 만들 수 있는 옵션도 있습니다. 다음 목록에는 규칙 세트 및 수신 규칙에 대해 사용 가능한 수정 사항이 나와 있습니다.

- 규칙 세트는 이름, 상태 및 만든 날짜와 함께 나열됩니다. 규칙 세트에 대한 수정 옵션은 다음과 같습니다.
  - Set as active/inactive(활성/비활성으로 설정) 토글 버튼은 상태 설정 사이를 전환합니다.
  - Duplicate(복제) 버튼을 클릭하면 규칙 세트가 복사됩니다. 고유한 이름을 입력하라는 메시지가 표시됩니다.
  - Delete(삭제) 버튼을 클릭하면 규칙 세트가 삭제됩니다. 이러한 되돌릴 수 없는 작업을 확인하라는 메시지가 표시됩니다.
- 수신 규칙은 이름, 상태, 보안 및 순서와 함께 나열됩니다. 수신 규칙의 수정 옵션은 다음과 같습니다.
  - 위쪽/아래쪽 화살표를 사용하여 규칙 세트 내에서 규칙 실행 순서를 재정렬할 수 있습니다.
  - Duplicate(복제) 버튼을 클릭하면 선택한 규칙의 복사본을 만듭니다. 고유한 이름을 입력하라는 메시지가 표시됩니다.
  - Edit(편집) 버튼을 클릭하면 규칙 설정, 수신자 조건 및 작업과 같은 파라미터를 편집할 수 있도록 선택한 규칙이 열립니다.
  - Delete(삭제) 버튼을 클릭하면 선택한 규칙이 삭제됩니다. 이러한 되돌릴 수 없는 작업을 확인하라는 메시지가 표시됩니다.
  - Create rule(규칙 만들기) 버튼을 클릭하면 새 규칙을 만들고 현재 규칙 세트에 추가할 수 있습니다.

작업 옵션

Amazon SES 이메일 수신을 위한 수신 규칙은 순서를 매긴 작업 목록을 포함합니다. 이 섹션에서는 작업 유형에 따른 구체적인 옵션에 대해 설명합니다.

작업의 유형은 다음과 같습니다.

- 헤더 추가 작업
- 반송 메일 응답 반환 작업
- Lambda 함수 호출 작업
- S3 버킷으로 전송 작업
- Amazon SNS 주제에 게시하는 작업
- 규칙 세트 중지 작업
- Amazon WorkMail과 통합 작업

헤더 추가 작업

[Add Header] 작업은 수신 이메일에 사용자 지정 헤더를 추가합니다. 이 작업은 보통 다른 작업과 연계해서 사용합니다. 이 작업의 옵션은 다음과 같습니다.

- 헤더 이름 - 추가할 헤더의 이름입니다. 영숫자(a~z, A~Z, 0~9)와 대시만 사용해 1~50자 사이로 작성해야 합니다.
- 헤더 값 - 추가할 헤더의 값입니다. 2048자 미만이어야 하며 줄바꿈 문자(\r 또는 \n)를 포함하지 않아야 합니다.

반송 메일 응답 반환 작업

반송 메일 작업은 발신자에게 반송 응답을 돌려보내 이메일 수신을 거부하고 설정에 따라 Amazon SNS를 통해 사용자에게 알립니다. 이 작업의 옵션은 다음과 같습니다.

- SMTP 응답 코드 - RFC 5321의 정의에 따르는 SMTP 응답 코드입니다.
- SMTP 상태 코드 - RFC 3463의 정의에 따르는 향상된 SMTP 상태 코드입니다.
- 메시지 - 사람이 읽을 수 있는 텍스트로 반송 이메일에 포함됩니다.
- 회신 발신자 - 반송 이메일 발신자의 이메일 주소입니다. 반송 이메일을 발신할 때 사용할 주소입니다. Amazon SES의 확인 절차를 거쳐야 합니다.

> **Note**
>
> 반송 메일 메시지는 사용자 지정 MAIL FROM 도메인을 통해 전송되지 않지만 SES에 의해 내부적으로 생성되고 `amazonses.com` DKIM 서명으로만 서명됩니다. 해결 방법으로 회신 발신자 옵션을 사용하여 반송 메시지의 이메일 주소를 설정합니다. 자세한 내용은이 AWS re:Post 문서를 참조하세요.

- SNS 주제 - 설정에 따라 반송 이메일 전송 사실을 통지할 Amazon SNS 주제의 이름 또는 ARN입니다. Amazon SNS 주제 ARN의 한 가지 예시는 arn:aws:sns:us-east-1:123456789012:MyTopic입니다. 작업을 설정할 때 SNS 주제 만들기를 선택하여 Amazon SNS 주제를 만들 수도 있습니다. Amazon SNS 주제에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 가이드를 참조하세요.

> **Note**
>
> 선택한 Amazon SNS 주제는 이메일을 수신하는 데 사용하는 Amazon SES 엔드포인트와 동일한 AWS 리전에 있어야 합니다.

이 필드에는 직접 값을 입력할 수도 있고, 반송 이유에 따라 SMTP 응답 코드, SMTP 상태 코드, 메시지를 자동으로 채워 넣는 템플릿을 선택할 수도 있습니다. 다음의 템플릿을 사용할 수 있습니다:

- 메일박스가 존재하지 않음 - SMTP 응답 코드 = 550, SMTP 상태 코드 = 5.1.1
- 메시지가 너무 큼 - SMTP 응답 코드 = 552, SMTP 상태 코드 = 5.3.4
- 메일박스가 가득 참 - SMTP 응답 코드 = 552, SMTP 상태 코드 = 5.2.2
- 메시지 콘텐츠가 거부됨 - SMTP 응답 코드 = 500, SMTP 상태 코드 = 5.6.1
- 알 수 없는 오류 - SMTP 응답 코드 = 554, SMTP 상태 코드 = 5.0.0
- 일시적인 오류 - SMTP 응답 코드 = 450, SMTP 상태 코드 = 4.0.0

필드에 사용자 지정 값을 입력하여 사용할 수 있는 추가 반송 코드는 RFC 3463을 참조하세요.

Lambda 함수 호출 작업

Lambda 작업은 Lambda 함수를 통해 코드를 호출하고, 설정에 따라 Amazon SNS를 통해 알립니다. 이 작업에는 다음 옵션 및 요구 사항이 있습니다.

옵션

- Lambda 함수 - Lambda 함수의 ARN입니다. Lambda 함수 ARN의 한 가지 예는 arn:aws:lambda:us-east-1:account-id:function:MyFunction입니다.
- 호출 유형 - Lambda 함수의 호출 유형입니다. RequestResponse 호출 유형은 함수를 실행할 경우 즉각적인 응답이 발생한다는 것을 의미합니다. Event 호출 유형은 함수가 비동기식으로 호출된다는 것을 의미합니다. 해당 사용 사례에 반드시 동기식 실행이 필요한 경우가 아니라면 이벤트(Event) 호출 유형을 사용할 것을 권장합니다.

  [RequestResponse] 호출에는 30초의 요청 제한 시간이 있습니다.

  자세한 내용은 AWS Lambda 개발자 가이드의 Lambda 함수 호출을 참조하세요.
- SNS 주제 - 지정된 Lambda 함수를 트리거한 후 알림을 전송할 Amazon SNS 주제의 이름 또는 ARN입니다. Amazon SNS 주제 ARN의 한 가지 예시는 arn:aws:sns:us-east-1:123456789012:MyTopic입니다. 자세한 설명은 Amazon Simple Notification Service 개발자 안내서에서 Amazon SNS 주제 생성을 참조하세요.

요구 사항

- 선택한 Lambda 함수는 이메일 수신에 사용하는 Amazon SES 엔드포인트와 동일한 AWS 리전에 있어야 합니다.
- 선택한 Amazon SNS 주제는 이메일 수신에 사용하는 Amazon SES 엔드포인트와 동일한 AWS 리전에 있어야 합니다.

Lambda 함수 작성하기

이메일을 처리할 때(즉, `Event` 호출 유형을 사용) Lambda 함수를 비동기식으로 호출할 수 있습니다. Lambda 함수에 전달하는 이벤트 객체는 인바운드 이메일 이벤트에 대한 메타데이터를 담고 있습니다. 이 메타데이터를 사용해 Amazon S3 버킷의 메시지 콘텐츠에 액세스할 수도 있습니다.

메일 흐름을 실제로 제어하고 싶다면 Lambda 함수를 동기식으로 호출해야 합니다(즉, `RequestResponse` 호출 유형 사용). 그러면 Lambda 함수가 두 개의 인수가 있는 `callback` 메서드를 호출합니다. 첫 번째 인수는 `null`이고 두 번째 인수는 `STOP_RULE`, `STOP_RULE_SET` 또는 `CONTINUE`로 설정된 `disposition` 속성입니다. 두 번째 인수가 `null`이거나 유효한 `disposition` 속성이 없는 경우, `CONTINUE` 실행 시와 마찬가지로 메일 흐름이 계속되어 작업과 규칙이 순서대로 처리됩니다.

예를 들어 Lambda 함수 코드의 끝에 다음 줄을 추가하면 수신 규칙 세트 처리를 중단시킬 수 있습니다.

```
callback( null, { "disposition" : "STOP_RULE_SET" });
```

AWS Lambda 코드 샘플은 섹션을 참조하세요Lambda 함수 예제. 고급 사용 사례 예시는 다음(사용 사례)을 참조하세요.

입력 형식

Amazon SES는 JSON 형식으로 Lambda 함수에 정보를 전달합니다. 이 최고 레벨 객체는 `Records` 배열을 포함하며, 이 배열에는 `eventSource`, `eventVersion`, `ses` 속성이 들어 있습니다. `ses` 객체에는 `receipt`와 `mail` 객체가 있고, 알림 내용에서 설명한 Amazon SNS 알림과 형식이 동일합니다.

Amazon SES가 Lambda로 전달하는 데이터에는 메시지에 대한 메타데이터와 여러 이메일 헤더가 포함됩니다. 그러나 메시지의 본문은 포함되어 있지 않습니다.

다음은 Amazon SES가 Lambda 함수에 제공하는 입력 데이터의 구조를 세부적으로 나타낸 것입니다.

```
{
   "Records": [
      {
        "eventSource": "aws:ses",
        "eventVersion": "1.0",
        "ses": {
           "receipt": {
               <same contents as SNS notification>
            },
           "mail": {
               <same contents as SNS notification>
           }
         }
     }
   ]
}
```

반환 값

Lambda 함수는 다음 값 중 하나를 반환하여 메일 흐름을 제어합니다.

- `STOP_RULE` - 현재 수신 규칙의 작업 수행은 중단하되 다음 수신 규칙을 처리할 수 있습니다.
- `STOP_RULE_SET` - 작업 또는 수신 규칙 처리를 완전히 중단합니다.

- CONTINUE 또는 기타 무효 값 - 다음 작업과 수신 규칙을 처리할 수 있습니다.

다음 주제에서는 수신 메일 이벤트의 샘플, 상위 수준 사용 사례의 예제 및 AWS Lambda 코드 예제를 다룹니다.

- 사용 사례
- Lambda 함수 예제

사용 사례

다음은 Lambda 함수의 처리 결과로 메일 흐름을 제어하기 위해 설정할 수 있는 규칙의 예시입니다. 설명을 위해서 이 예시에서는 대부분 S3 작업을 처리 결과로 사용합니다.

사용 사례 1: 모든 도메인의 스팸 거부하기

이 예시는 사용자의 모든 도메인에서 스팸을 거부하는 전역 규칙입니다. 규칙 2와 규칙 3은 모든 도메인에서 스팸을 거부한 후 도메인별 규칙을 적용할 수 있음을 보여 주기 위해 포함했습니다.

규칙 1

수신자 목록: Empty. 따라서 이 규칙은 확인 절차를 마친 모든 도메인의 모든 수신자에게 적용됩니다.

작업

1. 이메일이 스팸인 경우 STOP_RULE_SET를 반환하는 Lambda 작업(동기식). 그렇지 않은 경우에는 CONTINUE를 반환합니다. Lambda 함수 예제에서 스팸 거부에 대해서는 Lambda 함수 예제를 참조하십시오.

규칙 2

수신자 목록: example1.com

작업

1. 아무 작업이나.

규칙 3

수신자 목록: example2.com

작업

1. 아무 작업이나.

사용 사례 2: 모든 도메인의 스팸 반송하기

이 예시는 모든 도메인에서 스팸을 반송하는 전역 규칙을 보여 줍니다. 규칙 2와 규칙 3은 모든 도메인에서 스팸을 반송한 후 도메인별 규칙을 적용할 수 있음을 보여 주기 위해 포함했습니다.

규칙 1

수신자 목록: Empty. 따라서 이 규칙은 확인 절차를 마친 모든 도메인의 모든 수신자에게 적용됩니다.

작업

1. 이메일이 스팸인 경우 `CONTINUE`를 반환하는 Lambda 작업(동기식). 그렇지 않은 경우에는 `STOP_RULE`를 반환합니다.
2. 반송 작업("500 5.6.1. 메시지 콘텐츠 거부됨").
3. 중지 작업.

규칙 2

수신자 목록: example1.com

작업

1. 아무 작업이나.

규칙 3

수신자 목록: example2.com

작업

1. 아무 작업이나.

사용 사례 3: 가장 구체적인 규칙 적용하기

이 예시에서는 중지 작업을 사용해 이메일이 다수의 규칙으로 처리되는 것을 방지합니다. 이 예시에서는 특정 주소에 대한 규칙이 하나 있고, 도메인 내의 모든 이메일 주소에 대한 규칙도 하나 있습니다. 중지 작업을 사용하면 메시지가 특정 이메일 주소에 대한 규칙과 일치할 경우 도메인 전체에 대한 전반적인 규칙으로는 처리하지 않습니다.

규칙 1

수신자 목록: user@example.com

작업

1. Lambda 작업(비동기식).
2. 중지 작업.

규칙 2

수신자 목록: example.com

작업

1. 아무 작업이나.

사용 사례 4: CloudWatch에 메일 이벤트 기록

이 예시에서는 이메일 수신 시 메일을 Amazon SES에 저장하기에 앞서, 시스템을 통과하는 모든 메일에 대해 감사 로그를 작성하는 방법을 보여 줍니다.

규칙 1

수신자 목록: example.com

작업

1. 이벤트 객체를 CloudWatch 로그에 기록하는 Lambda 작업(비동기식). Lambda 함수 예제의 예제 Lambda 함수는 CloudWatch에 기록합니다.
2. S3 작업.

사용 사례 5: DKIM 인증을 통과하지 못한 메일 거부하기

이 예시는 수신 이메일을 모두 Amazon S3 버킷에 저장하되, 특정 이메일 주소로 가며 DKIM 인증을 통과하는 메일만 사용자의 자동화 이메일 애플리케이션으로 전송하는 방법을 보여 줍니다.

규칙 1

수신자 목록: example.com

작업

1. S3 작업.
2. 메시지가 DKIM 인증을 통과하지 못한 경우 `STOP_RULE_SET`를 반환하는 Lambda 작업(동기식). 그렇지 않은 경우에는 `CONTINUE`를 반환합니다.

규칙 2

수신자 목록: support@example.com

작업

1. 자동화 애플리케이션을 트리거하는 Lambda 작업(비동기식).

사용 사례 6: 제목에 따라 메일 필터링하기

이 예시에서는 도메인의 수신 메일 중 제목에 '할인'이 들어간 메일을 모두 거부한 다음, 자동화 시스템으로 가야 할 메일을 따로 처리하고, 도메인의 나머지 수신자에게 가야 할 메일을 처리합니다.

규칙 1

수신자 목록: example.com

작업

1. 제목에 '할인'이라는 단어가 들어간 경우 `STOP_RULE_SET`를 반환하는 Lambda 작업(동기식). 그렇지 않은 경우에는 `CONTINUE`를 반환합니다.

규칙 2

수신자 목록: support@example.com

작업

1. 버킷 1에 대한 S3 작업.
2. 자동화 애플리케이션을 트리거하는 Lambda 작업(비동기식).
3. 중지 작업.

규칙 3

수신자 목록: example.com

작업

1. 버킷 2에 대한 S3 작업.
2. 도메인의 나머지 수신자에게 보낼 이메일을 처리하는 Lambda 작업(비동기식).

Lambda 함수 예제

이 주제는 메일 흐름을 제어하는 Lambda 함수의 예시를 소개합니다.

예제 1: 스팸 거부

이 예제는 스팸 지표를 하나 이상 가진 메시지에 대한 처리를 중단합니다.

```
export const handler = async (event, context, callback) => {
    console.log('Spam filter');

    const sesNotification = event.Records[0].ses;
    console.log("SES Notification:\n", JSON.stringify(sesNotification, null, 2));

    // Check if any spam check failed
    if (sesNotification.receipt.spfVerdict.status === 'FAIL'
            || sesNotification.receipt.dkimVerdict.status === 'FAIL'
            || sesNotification.receipt.spamVerdict.status === 'FAIL'
            || sesNotification.receipt.virusVerdict.status === 'FAIL') {

        console.log('Dropping spam');

        // Stop processing rule set, dropping message
        callback(null, {'disposition':'STOP_RULE_SET'});
    } else {
        callback(null, {'disposition':'CONTINUE'});
    }
};
```

예제 2: 특정 헤더가 발견되면 계속

이 예제는 이메일이 특정 헤더 값을 포함하는 경우에만 현재 규칙을 계속 처리합니다.

```
export const handler = async (event, context, callback) => {
    console.log('Header matcher');

    const sesNotification = event.Records[0].ses;
    console.log("SES Notification:\n", JSON.stringify(sesNotification, null, 2));
```

```
    // Iterate over the headers
    for (let index in sesNotification.mail.headers) {
        const header = sesNotification.mail.headers[index];

        // Examine the header values
        if (header.name === 'X-Header' && header.value === 'X-Value') {
            console.log('Found header with value.');
            callback(null, {'disposition':'CONTINUE'});
            return;
        }
    }

    // Stop processing the rule if the header value wasn't found
    callback(null, {'disposition':'STOP_RULE'});
};
```

예제 3: Amazon S3에서 이메일 검색

이 예제는 Amazon S3에서 이메일 원본을 가져와서 처리합니다.

> **Note**
>
> - 그러려면 우선 S3 작업을 사용하는 Amazon S3에 이메일을 써야 합니다.
> - Lambda 함수에 S3 버킷에서 객체를 가져올 수 있는 IAM 권한이 있는지 확인합니다. 자세한 내용은이 AWS re:Post 문서를 참조하세요.
> - 기본 Lambda 실행 제한 시간이 워크플로에 너무 짧을 수 있습니다. 시간을 늘리는 것이 좋습니다.

```
import { S3Client, GetObjectCommand } from "@aws-sdk/client-s3";
const bucketName = '<Your Bucket Name>';

export const handler = async (event, context, callback) => {
    const client = new S3Client();
    console.log('Process email');

    var sesNotification = event.Records[0].ses;
    console.log("SES Notification:\n", JSON.stringify(sesNotification, null, 2));
    console.log("MessageId: " + sesNotification.mail.messageId)
```

```
    const getObjectCommand = new GetObjectCommand({
        Bucket: bucketName,
        Key: sesNotification.mail.messageId
    });

    try {
        const response = await client.send(getObjectCommand);
        const receivedMail = await response.Body.transformToString();
        console.log(receivedMail);
        callback(null, {'disposition':'CONTINUE'})
    } catch (e) {
        // Perform error handling here
        console.log("Encountered S3 client error: "+ e, e.stack);
        callback(null, {'disposition':'STOP_RULE_SET'})
    }
};
```

예제 4: DMARC 인증에 실패한 반송 메일 메시지

이 예제에서는 수신 이메일이 DMARC 인증에 실패한 경우 반송 메일 메시지를 보냅니다.

> **Note**
>
> - 이 예제를 사용하는 경우 `emailDomain` 환경 변수의 값을 이메일 수신 도메인으로 설정합니다.
> - Lambda 함수에 반송 메시지를 보내는 SES 자격 증명에 대한 `ses:SendBounce` 권한이 있는지 확인합니다.

```
import { SESClient, SendBounceCommand } from "@aws-sdk/client-ses";
const sesClient = new SESClient();
// Assign the emailDomain environment variable to a constant.
const emailDomain = process.env.emailDomain;

export const handler = async (event, context, callback) => {
    console.log('Spam filter starting');

    const sesNotification = event.Records[0].ses;
    const messageId = sesNotification.mail.messageId;
    const receipt = sesNotification.receipt;
```

```
    console.log('Processing message:', messageId);

    // If DMARC verdict is FAIL and the sending domain's policy is REJECT
    // (p=reject), bounce the email.
    if (receipt.dmarcVerdict.status === 'FAIL'
        && receipt.dmarcPolicy.status === 'REJECT') {
        // The values that make up the body of the bounce message.
        const sendBounceParams = {
            BounceSender: `mailer-daemon@${emailDomain}`,
            OriginalMessageId: messageId,
            MessageDsn: {
                ReportingMta: `dns; ${emailDomain}`,
                ArrivalDate: new Date(),
                ExtensionFields: [],
            },
            // Include custom text explaining why the email was bounced.
            Explanation: "Unauthenticated email is not accepted due to the sending
domain's DMARC policy.",
            BouncedRecipientInfoList: receipt.recipients.map((recipient) => ({
                Recipient: recipient,
                // Bounce with 550 5.6.1 Message content rejected
                BounceType: 'ContentRejected',
            })),
        };

        console.log('Bouncing message with parameters:');
        console.log(JSON.stringify(sendBounceParams, null, 2));

        const sendBounceCommand = new SendBounceCommand(sendBounceParams);

        // Try to send the bounce.
        try {
          const response = await sesClient.send(sendBounceCommand);
          console.log(response);
          console.log(`Bounce for message ${messageId} sent, bounce message ID:
${response.MessageId}`);
          // Stop processing additional receipt rules in the rule set.
          callback(null, {disposition: 'STOP_RULE_SET'});
        } catch (e) {
          // If something goes wrong, log the issue.
          console.log(`An error occurred while sending bounce for message:
${messageId}`, e);
          // Perform any additional error handling here
          callback(e)
```

```
        }

    // If the DMARC verdict is anything else (PASS, QUARANTINE or GRAY), accept
    // the message and process remaining receipt rules in the rule set.
    } else {
        console.log('Accepting message:', messageId);
        callback(null, {disposition: 'CONTINUE'});
    }
};
```

S3 버킷으로 전송 작업

S3 버킷에 전송 작업은 S3 버킷으로 메일을 전송하며, 선택적으로 SNS 등을 통해 알림을 보낼 수 있습니다. 이 작업의 옵션은 다음과 같습니다.

- S3 버킷 - 수신 이메일을 저장할 S3 버킷의 이름입니다. S3 버킷 생성을 선택하여 작업을 설정할 때 새 S3 버킷을 생성할 수도 있습니다. Amazon SES는 수정되지 않은 원시 이메일을 제공하며, 이는 주로 다목적 인터넷 전자 우편(MIME) 형식입니다. MIME 형식에 대한 자세한 내용은 RFC 2045를 참조하세요.

> **Important**
>
> - Amazon S3 버킷은 SES the section called “이메일 수신”을 사용할 수 있는 리전에 있어야 합니다. 그렇지 않으면 아래 설명된 IAM 역할 옵션을 사용해야 합니다.
> - S3 버킷에 이메일을 저장할 때 이메일(헤더 포함)의 최대 기본 용량은 40MB입니다.
> - SES는 기본 보존 기간으로 구성된 객체 잠금을 사용하여 사용 설정된 S3 버킷에 업로드하는 수신 규칙을 지원하지 않습니다.
> - 자체 KMS 키를 지정하여 S3 버킷에 암호화를 적용하는 경우 KMS 키 별칭이 아닌 정규화된 KMS 키 ARN을 사용해야 합니다. 별칭을 사용하면 버킷 관리자가 아닌 요청자에 속한 KMS로 데이터가 암호화될 수 있습니다. 교차 계정 작업에 암호화 사용을 참조하세요.

- 객체 키 접두사 - S3 버킷에서 사용할 선택적 키 이름 접두사입니다. 키 이름 접두사를 사용하여 S3 버킷을 폴더 구조로 구성할 수 있습니다. 예를 들어 이메일을 객체 키 접두사로 사용하면 S3 버킷의 이메일이라는 폴더에 이메일이 표시됩니다.
- 메시지 암호화 - S3 버킷에 전송하기 전에 수신 이메일 메시지를 암호화하는 옵션입니다.
- KMS 암호화 키 – (메시지 암호화를 선택한 경우 사용할 수 있습니다.) SES가 S3 버킷에 저장하기 전에 이메일을 암호화하는 데 사용해야 하는 AWS KMS 키입니다. 기본 KMS 키를 사용하거나 KMS에 생성한 고객 관리형 키를 사용할 수 있습니다.

> **Note**
> 선택한 KMS 키는 이메일을 수신하는 데 사용하는 SES 엔드포인트와 동일한 AWS 리전에 있어야 합니다.

- 기본 KMS 키를 사용하려면 SES 콘솔에서 수신 규칙을 설정할 때 aws/ses를 선택합니다. SES API를 사용할 경우, ARN을 `arn:aws:kms:REGION:AWSACCOUNTID:alias/aws/ses`의 형태로 입력하면 기본 KMS 키를 지정할 수 있습니다. 예를 들어 AWS 계정 ID가 123456789012이고 us-east-1 리전에서 기본 KMS 키를 사용하려는 경우 기본 KMS 키의 ARN은 입니다`arn:aws:kms:us-east-1:123456789012:alias/aws/ses`. 기본 KMS 키를 사용하는 경우, SES에 키 사용 권한을 부여하는 절차가 필요하지 않습니다.
- KMS에 생성한 고객 관리형 키를 사용하려면, KMS 키의 ARN을 입력하고 키의 정책에 설명을 추가하여 SES에 사용 권한을 부여해야 합니다. 권한 설정에 대한 자세한 내용은 다음(Amazon SES에 이메일 수신을 위한 권한 부여)을 참조하세요.

SES를 포함한 KMS 사용에 대한 자세한 내용은 AWS Key Management Service 개발자 가이드를 참조하세요. 콘솔 또는 API에서 KMS 키를 지정하지 않으면 SES가 이메일을 암호화하지 않습니다.

> **Important**
> 저장을 위해 메일이 S3에 제출되기 전에 S3 암호화 클라이언트를 사용하여 SES에서 메일을 암호화합니다. 메일을 암호화할 때 S3 서버 측 암호화를 사용하지 않습니다. 따라서, S3에서 이메일을 가져온 후에는 S3 암호화 클라이언트를 사용해 이메일을 해독해야 합니다. 서비스에 해독할 때 KMS 키를 사용할 권한이 없기 때문입니다. 이 암호화 클라이언트는 AWS SDK for Java 및 AWS SDK for Ruby에서 사용할 수 있습니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서를 참조하세요.

- IAM 역할 - SES가 S3에 전송 작업(Amazon S3 버킷, SNS 주제 및 KMS 키)의 리소스에 액세스하는 데 사용하는 IAM 역할입니다. 제공되지 않은 경우 각 리소스에 개별적으로 액세스할 수 있는 권한을 SES에 명시적으로 부여해야 합니다. Amazon SES에 이메일 수신을 위한 권한 부여의 내용을 참조하세요.

  SES 이메일 수신을 사용할 수 없는 리전에 속해 있는 S3 버킷에 쓰려면 S3에 대한 쓰기 권한 정책이 있는 IAM 역할을 역할의 인라인 정책으로 사용해야 합니다. 콘솔에서 직접 이 작업에 대한 권한 정책을 적용할 수 있습니다.

1. IAM 역할 필드에서 새 역할 생성을 선택하고 이름을 입력한 다음 역할 생성을 선택합니다. (이 역할에 대한 IAM 신뢰 정책은 백그라운드에서 자동으로 생성됩니다.)
2. IAM 신뢰 정책이 자동으로 생성되었으므로 역할에 작업의 권한 정책만 추가하면 됩니다. IAM 역할 필드에서 역할 보기를 선택하여 IAM 콘솔을 엽니다.
3. 권한 탭에서 권한 추가를 선택하고 인라인 정책 만들기를 선택합니다.
4. 권한 지정 페이지의 정책 편집기에서 JSON을 선택합니다.
5. S3 작업을 위한 IAM 역할 권한에서 권한 정책을 복사하여 정책 편집기에 붙여넣고 빨간색 텍스트의 데이터를 자체 데이터로 바꿉니다. (편집기에서 예제 코드를 모두 삭제해야 합니다.)
6. 다음을 선택합니다.
7. 정책 생성을 선택하여 IAM 역할에 대한 권한 정책을 검토하고 생성합니다.
8. SES 규칙 생성 - 작업 추가 페이지가 열려 있는 브라우저의 탭을 선택하고 규칙 생성을 위한 나머지 단계를 계속 진행합니다.

- SNS 주제 - 이메일을 S3 버킷에 저장한 후 알림을 전송할 Amazon SNS 주제의 이름 또는 ARN입니다. SNS 주제 ARN은 arn:aws:sns:us-east-1:123456789012:MyTopic과 같은 형식일 수 있습니다. 작업을 설정할 때 SNS 주제 생성을 선택하여 SNS 주제를 생성할 수도 있습니다. SNS 주제에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 가이드를 참조하세요.

> **Note**
>
> - 선택한 SNS 주제는 이메일을 수신하는 데 사용하는 SES 엔드포인트와 동일한 AWS 리전에 있어야 합니다.
> - SES가 SNS에 게시할 수 있도록 KMS 키 정책을 편집해야 하므로 SES 수신 규칙과 연관된 SNS 주제에서만 고객 관리형 KMS 키 암호화를 사용합니다. 이는 계획적으로 편집할 수 없는 AWS 관리형 KMS 키 정책과 대조됩니다.

Amazon SNS 주제에 게시하는 작업

SNS 작업은 Amazon SNS 알림을 사용해 메일을 게시합니다. 이때 알림은 이메일 콘텐츠 전체를 포함합니다. 이 작업의 옵션은 다음과 같습니다.

- SNS 주제 - 이메일을 게시할 Amazon SNS 주제의 이름 또는 ARN입니다. Amazon SNS 알림은 보통 MIME(Multipurpose Internet Mail Extensions) 형식으로 작성된 이메일의 사본을 그대로 포함합니다. MIME 형식에 대한 자세한 내용은 RFC 2045를 참조하세요.

> **Important**
>
> Amazon SNS 알림을 통해 이메일을 수신하고자 하는 경우 이메일(헤더 포함)의 최대 용량은 150KB입니다. 용량이 더 큰 이메일은 반송됩니다. 이보다 용량이 큰 이메일을 수신할 것으로 예상한다면, 이메일을 Amazon S3 버킷에 저장하세요.

Amazon SNS 주제 ARN의 한 가지 예시는 arn:aws:sns:us-east-1:123456789012:MyTopic입니다. 작업을 설정할 때 SNS 주제 만들기를 선택하여 Amazon SNS 주제를 만들 수도 있습니다. Amazon SNS 주제에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 가이드를 참조하세요.

> **Note**
>
> - 선택한 Amazon SNS 주제는 이메일을 수신하는 데 사용하는 Amazon SES 엔드포인트와 동일한 AWS 리전에 있어야 합니다.
> - SES가 SNS에 게시할 수 있도록 KMS 키 정책을 편집해야 하므로 SES 수신 규칙과 연관된 SNS 주제에서만 고객 관리형 KMS 키 암호화를 사용합니다. 이는 계획적으로 편집할 수 없는 AWS 관리형 KMS 키 정책과 대조됩니다.

- 인코딩 - Amazon SNS 알림 내에서 이메일에 사용할 인코딩. UTF-8은 사용하기 쉽지만 메시지가 여러 가지 인코딩 형식으로 인코딩된 경우 일부 특수 문자를 보존하지 못할 수 있습니다. Base64는 모든 특수 문자를 보존합니다. UTF-8 및 Base64에 관한 정보는 각각 RFC 3629 및 RFC 4648을 참조하세요.

이메일을 받으면 Amazon SES가 활성 수신 규칙 세트의 규칙을 실행합니다. Amazon SNS을 사용하여 알림을 전송하도록 수신 규칙을 구성할 수 있습니다. 수신 규칙에서는 다음과 같은 두 가지 유형의 알림을 보낼 수 있습니다.

- SNS 작업에서 전송된 알림 – SNS 작업을 수신 규칙에 추가하면 이메일 및 이메일 콘텐츠에 대한 정보를 전송합니다. 메시지가 150KB 이하인 경우 이 알림 유형에는 이메일의 전체 MIME 본문도 포함됩니다.
- 다른 작업 유형에서 전송된 알림 – 다른 작업 유형(예: 반송 메일, Lambda, 규칙 세트 중지 또는 WorkMail 작업)을 수신 규칙에 추가하는 경우 필요에 따라 Amazon SNS 주제를 지정할 수 있습니다. 이 경우 이러한 작업이 수행되면 알림을 받게 됩니다. 이러한 알림에는 이메일에 대한 정보가 포함되지만 이메일의 내용은 포함되지 않습니다.

이 섹션에서는 이러한 알림의 내용에 대해 설명하고 각 알림 유형의 예를 제공합니다.

- Amazon SES 이메일 수신에 대한 알림 내용
- Amazon SES 이메일 수신에 대한 알림의 예

Amazon SES 이메일 수신에 대한 알림 내용

이메일 수신에 대한 모든 알림은 Amazon Simple Notification Service(Amazon SNS) 주제에 JavaScript Object Notation(JSON) 형식으로 게시됩니다.

알림 예는 알림 예제 단원을 참조하세요.

목차

- 최상위 JSON 객체
- 수신 객체
  - 작업 객체
  - dkimVerdict 객체
  - dmarcVerdict 객체
  - spamVerdict 객체
  - spfVerdict 객체
  - virusVerdict 객체
- Mail 객체
  - commonHeaders 객체

최상위 JSON 객체

최상위 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`notificationType`	알림 유형입니다. 이 유형의 알림의 경우 값은 항상 `Received`입니다.
`receipt`	이메일 전송 관련 정보가 포함된 객체
`mail`	알림과 연결된 이메일 관련 정보를 포함하는 객체입니다.

필드 이름	설명
`content`	일반적으로 다목적 인터넷 전자 우편(MIME) 형식으로 작성된 수정되지 않은 원본 이메일이 포함된 문자열입니다. MIME 형식에 대한 자세한 내용은 RFC 2045를 참조하세요. **Note** 이 필드는 SNS 작업에 의해 알림이 트리거된 경우에만 표시됩니다. 다른 모든 작업에 의해 트리거된 알림에는 이 필드가 포함되지 않습니다.

수신 객체

`receipt` 객체에는 다음 필드가 있습니다.

필드 이름	설명
`action`	실행된 작업에 대한 정보를 캡슐화하는 객체입니다. 가능한 값 목록은 작업 객체 단원을 참조하세요.
`dkimVerdict`	DomainKeys Identified Mail(DKIM) 검사를 통과했는지 여부를 나타내는 객체입니다. 가능한 값 목록은 dkimVerdict 객체 단원을 참조하세요.
`dmarcPolicy`	발신 도메인에 대한 Domain-based Message Authentication, Reporting & Conformance(DMARC) 설정을 나타냅니다. 이 필드는 메시지가 DMARC 인증에 실패한 경우에만 나타납니다. 이 필드에 가능한 값은 다음과 같습니다.

필드 이름	설명
	• none: 발신 도메인의 소유자가 DMARC 인증에 실패한 메시지에 대해 특정 작업을 수행하도록 요청합니다. • quarantine : 발신 도메인의 소유자가 DMARC 인증에 실패한 메시지를 수신자가 의심스러운 것으로 처리하도록 요청합니다. • reject: 발신 도메인의 소유자가 DMARC 인증에 실패한 메시지를 거부하도록 요청합니다.
dmarcVerdict	Domain-based Message Authentication, Reporting & Conformance(DMARC) 검사를 통과했는지 여부를 나타내는 객체입니다. 가능한 값 목록은 dmarcVerdict 객체 단원을 참조하세요.
processingTimeMillis	Amazon SES가 메시지를 수신한 시간부터 작업을 트리거한 시간까지의 기간(밀리초)을 지정하는 문자열입니다.
recipients	활성 수신 규칙과 일치하는 수신자(특히 엔벌로프 RCPT TO 주소)입니다. 여기에 나열된 주소는 the section called “Mail 객체”의 destination 필드에 나열된 주소와 다를 수 있습니다.
spamVerdict	메시지가 스팸인지 여부를 나타내는 객체입니다. 가능한 값 목록은 spamVerdict 객체 단원을 참조하세요.
spfVerdict	Sender Policy Framework(SPF) 검사를 통과했는지 여부를 나타내는 객체입니다. 가능한 값 목록은 spfVerdict 객체 단원을 참조하세요.
timestamp	작업이 트리거된 날짜와 시간을 ISO 8601 형식으로 지정하는 문자열입니다.

필드 이름	설명
virusVerdict	메시지에 바이러스가 포함되어 있는지 여부를 나타내는 객체입니다. 가능한 값 목록은 virusVerdict 객체 단원을 참조하세요.

작업 객체

`action` 객체에는 다음 필드가 있습니다.

필드 이름	설명
`type`	실행된 작업의 유형을 나타내는 문자열입니다. 가능한 값은 `S3`, `SNS`, `Bounce`, `Lambda`, `Stop`, `WorkMail`입니다.
`topicArn`	알림이 게시된 Amazon SNS 주제의 Amazon 리소스 이름(ARN)이 포함된 문자열입니다.
`bucketName`	메시지가 게시된 Amazon S3 버킷의 이름이 포함된 문자열입니다. S3 작업 유형에 대해서만 표시됩니다.
`objectKey`	Amazon S3 버킷의 이메일을 고유하게 식별하는 이름을 포함하는 문자열로, the section called "Mail 객체"의 `messageId` 와(과) 동일합니다. S3 작업 유형에 대해서만 표시됩니다.
`smtpReplyCode`	RFC 5321의 정의에 따르는 SMTP 응답 코드를 포함하는 문자열입니다. 반송 작업 유형에 대해서만 표시됩니다.
`statusCode`	RFC 3463의 정의에 따르는 향상된 SMTP 상태 코드를 포함하는 문자열입니다. 반송 작업 유형에 대해서만 표시됩니다.

필드 이름	설명
`message`	반송 메시지에 포함할 사람이 읽을 수 있는 텍스트가 포함된 문자열입니다. 반송 작업 유형에 대해서만 표시됩니다.
`sender`	반송된 이메일 발신자의 이메일 주소가 포함된 문자열입니다. 반송 메시지를 보낼 때 사용한 주소입니다. 반송 작업 유형에 대해서만 표시됩니다.
`functionArn`	트리거된 Lambda 함수의 ARN을 포함하는 문자열입니다. Lambda 작업 유형에 대해서만 표시됩니다.
`invocationType`	Lambda 함수의 호출 유형을 포함하는 문자열입니다. 가능한 값은 `RequestResponse` 및 `Event`입니다. Lambda 작업 유형에 대해서만 표시됩니다.
`organizationArn`	Amazon WorkMail 조직의 ARN을 포함하는 문자열입니다. WorkMail 작업 유형에 대해서만 표시됩니다.

dkimVerdict 객체

`dkimVerdict` 객체에는 다음 필드가 있습니다.

필드 이름	설명
`status`	DKIM 판정이 포함된 문자열입니다. 가능한 값은 다음과 같습니다. • `PASS`: 메시지가 DKIM 인증을 통과했습니다. • `FAIL`: 메시지가 DKIM 인증에 실패했습니다.

필드 이름	설명
	• GRAY: 메시지에 DKIM 서명이 없거나 발신 도메인과 DKIM 서명 도메인이 일치하지 않습니다. • PROCESSING_FAILED : Amazon SES가 DKIM 서명을 확인하지 못하는 문제가 있습니다. 예를 들어 DNS 쿼리가 실패하거나 DKIM 서명 헤더가 제대로 포맷되지 않았습니다.

dmarcVerdict 객체

`dmarcVerdict` 객체에는 다음 필드가 있습니다.

필드 이름	설명
`status`	DMARC 판정이 포함된 문자열입니다. 가능한 값은 다음과 같습니다. • PASS: 메시지가 DMARC 인증을 통과했습니다. • FAIL: 메시지가 DMARC 인증에 실패했습니다. • GRAY: SPF 또는 DKIM 중 하나 이상이 인증을 통과했지만 전송 도메인에 DMARC 정책이 없거나 p=none 정책을 사용합니다. • PROCESSING_FAILED : Amazon SES가 DMARC 판정을 제공하지 못하는 문제가 있습니다.

spamVerdict 객체

`spamVerdict` 객체에는 다음 필드가 있습니다.

필드 이름	설명
`status`	스팸 검사 결과가 포함된 문자열입니다. 가능한 값은 다음과 같습니다. • `PASS`: 스팸 검사에서 메시지에 스팸이 포함될 가능성이 낮은 것으로 확인되었습니다. • `FAIL`: 스팸 검사에서 메시지에 스팸이 포함될 가능성이 높은 것으로 확인되었습니다. • `GRAY`: Amazon SES가 이메일을 스캔했지만 스팸인지 여부를 확인할 수 없습니다. • `PROCESSING_FAILED` : Amazon SES가 이메일을 스캔할 수 없습니다. 예를 들어, 이메일은 유효한 MIME 메시지가 아닙니다.

spfVerdict 객체

`spfVerdict` 객체에는 다음 필드가 있습니다.

필드 이름	설명
`status`	SPF 판정이 포함된 문자열입니다. 가능한 값은 다음과 같습니다. • `PASS`: 메시지가 SPF 인증을 통과했습니다. • `FAIL`: 메시지가 SPF 인증에 실패했습니다. • `GRAY`: SPF 결과는`none`, `softfail`, 또는 `neutral`입니다. • `PROCESSING_FAILED` : Amazon SES가 SPF 서명을 확인하지 못하는 문제가 있습니다. 예를 들어 DNS 쿼리가 실패합니다.

virusVerdict 객체

`virusVerdict` 객체에는 다음 필드가 있습니다.

필드 이름	설명
`status`	바이러스 검사 결과가 포함된 문자열입니다. 가능한 값은 다음과 같습니다. • `PASS`: 메시지에 바이러스가 포함되어 있지 않습니다. • `FAIL`: 메시지에 바이러스가 포함되어 있습니다. • `GRAY`: Amazon SES가 이메일을 스캔했지만 바이러스 포함 여부를 확인할 수 없습니다. • `PROCESSING_FAILED` : Amazon SES가 이메일의 내용을 스캔할 수 없습니다. 예를 들어, 이메일은 유효한 MIME 메시지가 아닙니다.

Mail 객체

`mail` 객체에는 다음 필드가 있습니다.

필드 이름	설명
`destination`	수신 이메일의 MIME 헤더의 모든 수신자 주소(To: 및 CC: 수신자 포함)의 전체 목록입니다.
`messageId`	Amazon SES에서 이메일에 할당한 고유 ID가 포함된 문자열입니다. 이메일이 Amazon S3로 전송된 경우 메시지 ID는 Amazon S3 버킷에 메시지를 쓰는 데 사용된 Amazon S3 객체 키이기도 합니다.
`source`	이메일을 보낸 주소(특히 엔벌로프 MAIL FROM 주소)가 포함된 문자열입니다.
`timestamp`	ISO8601 포멧으로 이메일 수신 시간이 포함된 문자열입니다.

필드 이름	설명
headers	Amazon SES 헤더 및 사용자 지정 헤더입니다. 각 헤더에는 name 및 value 필드가 있습니다.
commonHeaders	모든 이메일에 공통적인 헤더입니다. 각 헤더에는 name 및 value 필드가 있습니다.
headersTruncated	알림에서 헤더가 잘렸는지 여부를 나타내며 헤더의 용량이 10KB를 초과할 경우 헤더가 잘립니다. 가능한 값은 true 및 false입니다.

commonHeaders 객체

commonHeaders 객체는 다음 표에 있는 필드를 가질 수 있습니다. 이 객체에 있는 필드는 수신 이메일에 있는 필드에 따라 다릅니다.

필드 이름	설명
messageId	원본 메시지의 ID입니다.
date	Amazon SES가 메시지를 수신한 날짜와 시간입니다.
to	이메일의 To 헤더입니다.
cc	이메일의 CC 헤더입니다.
bcc	이메일의 BCC 헤더입니다.
from	이메일의 From 헤더입니다.
sender	이메일의 Sender 헤더입니다.
returnPath	이메일의 Return-Path 헤더입니다.
replyTo	이메일의 Reply-To 헤더입니다.
subject	이메일의 Subject 헤더입니다.

Amazon SES 이메일 수신에 대한 알림의 예

이 단원에는 다음 유형의 알림에 대한 예제가 나와 있습니다.

- SNS 작업의 결과로 전송된 알림입니다.
- 다른 유형의 작업의 결과로 전송된 알림(경고 알림).

SNS 작업 알림

이 단원에서 SNS 작업 알림의 예가 나와 있습니다. 이전에 표시된 경고 알림과 달리 `content` 단원에는 일반적으로 다목적 인터넷 전자 우편(MIME) 형식으로 작성된 이메일이 포함되어 있습니다.

```
{
  "notificationType":"Received",
  "receipt":{
    "timestamp":"2015-09-11T20:32:33.936Z",
    "processingTimeMillis":222,
    "recipients":[
      "recipient@example.com"
    ],
    "spamVerdict":{
      "status":"PASS"
    },
    "virusVerdict":{
      "status":"PASS"
    },
    "spfVerdict":{
      "status":"PASS"
    },
    "dkimVerdict":{
      "status":"PASS"
    },
    "action":{
      "type":"SNS",
      "topicArn":"arn:aws:sns:us-east-1:012345678912:example-topic"
    }
  },
  "mail":{
    "timestamp":"2015-09-11T20:32:33.936Z",
    "source":"61967230-7A45-4A9D-BEC9-87CBCF2211C9@example.com",
    "messageId":"d6iitobk75ur44p8kdnnp7g2n800",
    "destination":[
```

```
      "recipient@example.com"
    ],
    "headersTruncated":false,
    "headers":[
      {
        "name":"Return-Path",

 "value":"<0000014fbe1c09cf-7cb9f704-7531-4e53-89a1-5fa9744f5eb6-000000@amazonses.com>"
      },
      {
        "name":"Received",
        "value":"from a9-183.smtp-out.amazonses.com (a9-183.smtp-out.amazonses.com
 [54.240.9.183]) by inbound-smtp.us-east-1.amazonaws.com with SMTP id
 d6iitobk75ur44p8kdnnp7g2n800 for recipient@example.com; Fri, 11 Sep 2015 20:32:33
 +0000 (UTC)"
      },
      {
        "name":"DKIM-Signature",
        "value":"v=1; a=rsa-sha256; q=dns/txt; c=relaxed/simple;
 s=ug7nbtf4gccmlpwj322ax3p6ow6yfsug; d=amazonses.com; t=1442003552;
 h=From:To:Subject:MIME-Version:Content-Type:Content-Transfer-Encoding:Date:Message-
ID:Feedback-ID; bh=DWr3IOmYWoXCA9ARqGC/UaODfghffiwFNRIb2Mckyt4=;
 b=p4ukUDSFqhqiub+zPR0DW1kp7oJZakrzupr6LBe6sUuvqpBkig56UzUwc29rFbJF
 hlX3Ov7DeYVNoN38stqwsF8ivcajXpQsXRC1cW9z8x875J041rClAjV7EGbLmudVpPX
 4hHst1XPyX5wmgdHIhmUuh8oZKpVqGi6bHGzzf7g="
      },
      {
        "name":"From",
        "value":"sender@example.com"
      },
      {
        "name":"To",
        "value":"recipient@example.com"
      },
      {
        "name":"Subject",
        "value":"Example subject"
      },
      {
        "name":"MIME-Version",
        "value":"1.0"
      },
      {
        "name":"Content-Type",
```

```
        "value":"text/plain; charset=UTF-8"
      },
      {
        "name":"Content-Transfer-Encoding",
        "value":"7bit"
      },
      {
        "name":"Date",
        "value":"Fri, 11 Sep 2015 20:32:32 +0000"
      },
      {
        "name":"Message-ID",
        "value":"<61967230-7A45-4A9D-BEC9-87CBCF2211C9@example.com>"
      },
      {
        "name":"X-SES-Outgoing",
        "value":"2015.09.11-54.240.9.183"
      },
      {
        "name":"Feedback-ID",
        "value":"1.us-east-1.Krv2FKpFdWV+KUYw3Qd6wcpPJ4Sv/pOPpEPSHn2u2o4=:AmazonSES"
      }
    ],
    "commonHeaders":{

 "returnPath":"0000014fbe1c09cf-7cb9f704-7531-4e53-89a1-5fa9744f5eb6-000000@amazonses.com",
      "from":[
        "sender@example.com"
      ],
      "date":"Fri, 11 Sep 2015 20:32:32 +0000",
      "to":[
        "recipient@example.com"
      ],
      "messageId":"<61967230-7A45-4A9D-BEC9-87CBCF2211C9@example.com>",
      "subject":"Example subject"
    }
  },
  "content":"Return-Path: <61967230-7A45-4A9D-BEC9-87CBCF2211C9@example.com>\r
\nReceived: from a9-183.smtp-out.amazonses.com (a9-183.smtp-out.amazonses.com
 [54.240.9.183])\r\n by inbound-smtp.us-east-1.amazonaws.com with SMTP id
 d6iitobk75ur44p8kdnnp7g2n800\r\n for recipient@example.com;\r\n Fri, 11 Sep 2015
 20:32:33 +0000 (UTC)\r\nDKIM-Signature: v=1; a=rsa-sha256; q=dns/txt; c=relaxed/
simple;\r\n\ts=ug7nbtf4gccmlpwj322ax3p6ow6yfsug; d=amazonses.com; t=1442003552;\r\n
\th=From:To:Subject:MIME-Version:Content-Type:Content-Transfer-Encoding:Date:Message-
```

```
ID:Feedback-ID;\r\n\tbh=DWr3IOmYWoXCA9ARqGC/UaODfghffiwFNRIb2Mckyt4=;\r\n
\tb=p4ukUDSFqhqiub+zPR0DW1kp7oJZakrzupr6LBe6sUuvqpBkig56UzUwc29rFbJF\r\n
\thlX3Ov7DeYVNoN38stqwsF8ivcajXpQsXRC1cW9z8x875J041rClAjV7EGbLmudVpPX\r\n
\t4hHst1XPyX5wmgdHIhmUuh8oZKpVqGi6bHGzzf7g=\r\nFrom: sender@example.com\r\nTo:
 recipient@example.com\r\nSubject: Example subject\r\nMIME-Version: 1.0\r\nContent-
Type: text/plain; charset=UTF-8\r\nContent-Transfer-Encoding: 7bit\r\nDate: Fri, 11 Sep
 2015 20:32:32 +0000\r\nMessage-ID: <61967230-7A45-4A9D-BEC9-87CBCF2211C9@example.com>
\r\nX-SES-Outgoing: 2015.09.11-54.240.9.183\r\nFeedback-ID: 1.us-east-1.Krv2FKpFdWV
+KUYw3Qd6wcpPJ4Sv/pOPpEPSHn2u2o4=:AmazonSES\r\n\r\nExample content\r\n"
}
```


경고 알림

이 단원에는 S3 작업에 의해 트리거될 수 있는 Amazon SNS 알림의 예가 포함되어 있습니다. Lambda 작업, 반송 작업, 중지 작업 및 WorkMail 작업에 의해 트리거되는 알림은 비슷합니다. 이러한 알림에는 이메일에 대한 정보가 포함되지만 이메일의 내용은 포함되지 않습니다.


```
{
   "notificationType": "Received",
    "receipt": {
      "timestamp": "2015-09-11T20:32:33.936Z",
      "processingTimeMillis": 406,
      "recipients": [
        "recipient@example.com"
      ],
      "spamVerdict": {
        "status": "PASS"
      },
      "virusVerdict": {
        "status": "PASS"
      },
      "spfVerdict": {
        "status": "PASS"
      },
      "dkimVerdict": {
        "status": "PASS"
      },
      "action": {
        "type": "S3",
        "topicArn": "arn:aws:sns:us-east-1:012345678912:example-topic",
        "bucketName": "amzn-s3-demo-bucket",
        "objectKey": "\email"
      }
```

```
    },
    "mail": {
      "timestamp": "2015-09-11T20:32:33.936Z",
      "source":
 "0000014fbe1c09cf-7cb9f704-7531-4e53-89a1-5fa9744f5eb6-000000@amazonses.com",
      "messageId": "d6iitobk75ur44p8kdnnp7g2n800",
      "destination": [
        "recipient@example.com"
      ],
      "headersTruncated": false,
      "headers": [
        {
          "name": "Return-Path",
          "value":
 "<0000014fbe1c09cf-7cb9f704-7531-4e53-89a1-5fa9744f5eb6-000000@amazonses.com>"
        },
        {
          "name": "Received",
          "value": "from a9-183.smtp-out.amazonses.com (a9-183.smtp-out.amazonses.com
 [54.240.9.183]) by inbound-smtp.us-east-1.amazonaws.com with SMTP id
 d6iitobk75ur44p8kdnnp7g2n800 for recipient@example.com; Fri, 11 Sep 2015 20:32:33
 +0000 (UTC)"
        },
        {
          "name": "DKIM-Signature",
          "value": "v=1; a=rsa-sha256; q=dns/txt; c=relaxed/simple;
 s=ug7nbtf4gccmlpwj322ax3p6ow6yfsug; d=amazonses.com; t=1442003552;
 h=From:To:Subject:MIME-Version:Content-Type:Content-Transfer-Encoding:Date:Message-
ID:Feedback-ID; bh=DWr3IOmYWoXCA9ARqGC/UaODfghffiwFNRIb2Mckyt4=;
 b=p4ukUDSFqhqiub+zPR0DW1kp7oJZakrzupr6LBe6sUuvqpBkig56UzUwc29rFbJF
 hlX3Ov7DeYVNoN38stqwsF8ivcajXpQsXRC1cW9z8x875J041rClAjV7EGbLmudVpPX
 4hHst1XPyX5wmgdHIhmUuh8oZKpVqGi6bHGzzf7g="
        },
        {
          "name": "From",
          "value": "sender@example.com"
        },
        {
          "name": "To",
          "value": "recipient@example.com"
        },
        {
          "name": "Subject",
          "value": "Example subject"
```

```
        },
        {
          "name": "MIME-Version",
          "value": "1.0"
        },
        {
          "name": "Content-Type",
          "value": "text/plain; charset=UTF-8"
        },
        {
          "name": "Content-Transfer-Encoding",
          "value": "7bit"
        },
        {
          "name": "Date",
          "value": "Fri, 11 Sep 2015 20:32:32 +0000"
        },
        {
          "name": "Message-ID",
          "value": "<61967230-7A45-4A9D-BEC9-87CBCF2211C9@example.com>"
        },
        {
          "name": "X-SES-Outgoing",
          "value": "2015.09.11-54.240.9.183"
        },
        {
          "name": "Feedback-ID",
          "value": "1.us-east-1.Krv2FKpFdWV+KUYw3Qd6wcpPJ4Sv/pOPpEPSHn2u2o4=:AmazonSES"
        }
      ],
      "commonHeaders": {
        "returnPath":
 "0000014fbe1c09cf-7cb9f704-7531-4e53-89a1-5fa9744f5eb6-000000@amazonses.com",
        "from": [
          "sender@example.com"
        ],
        "date": "Fri, 11 Sep 2015 20:32:32 +0000",
        "to": [
          "recipient@example.com"
        ],
        "messageId": "<61967230-7A45-4A9D-BEC9-87CBCF2211C9@example.com>",
        "subject": "Example subject"
      }
    }
```

```
}
```

규칙 세트 중지 작업

중지 작업은 해당 수신 규칙 세트의 평가를 중단하고, 설정에 따라 Amazon SNS를 통해 사용자에게 알림을 전송합니다. 이 작업의 옵션은 다음과 같습니다.

- SNS 주제 - 중지 작업을 수행한 후 알림을 전송할 Amazon SNS 주제의 이름 또는 ARN입니다. Amazon SNS 주제 ARN의 한 가지 예시는 arn:aws:sns:us-east-1:123456789012:MyTopic입니다. 작업을 설정할 때 SNS 주제 만들기를 선택하여 Amazon SNS 주제를 만들 수도 있습니다. Amazon SNS 주제에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 가이드를 참조하세요.

> **Note**
>
> 선택한 Amazon SNS 주제는 이메일을 수신하는 데 사용하는 Amazon SES 엔드포인트와 동일한 AWS 리전에 있어야 합니다.

Amazon WorkMail과 통합 작업

WorkMail 작업은 Amazon WorkMail과 통합됩니다. Amazon WorkMail이 모든 이메일 처리를 수행하는 경우 Amazon WorkMail에서 설정이 자동으로 처리되므로 일반적으로 이 작업을 직접 사용할 필요는 없습니다. 이 작업의 옵션은 다음과 같습니다.

- 조직 ARN - Amazon WorkMail 조직의 ARN입니다. Amazon WorkMail 조직의 ARN 형식은 `arn:aws:workmail:`*`region`*`:`*`account_ID`*`:organization/`*`organization_ID`*와(과) 같습니다. 여기서,
  - `region`은(는) Amazon SES 및 Amazon WorkMail을 사용 중인 리전입니다. (동일한 리전의 항목을 사용해야 합니다.) 예: us-east-1.
  - `account_ID`는 AWS 계정 ID입니다. AWS 관리 콘솔의 계정 페이지에서 계정 ID를 찾을 수 있습니다 AWS .
  - `organization_ID`은(는) 사용자가 조직을 생성할 때 Amazon WorkMail이 생성하는 고유한 식별자입니다. 조직 ID는 Amazon WorkMail 콘솔에서 소속 조직의 Organization Settings 페이지를 보면 확인할 수 있습니다.

  전체 Amazon WorkMail 조직 ARN의 예제는 arn:aws:workmail:us-east-1:123456789012:organization/m-68755160c4cb4e29a2b2f8fb58f359d7입니다. Amazon WorkMail 조직에 대한 자세한 내용은 Amazon WorkMail 관리자 안내서를 참조하세요.

- SNS 주제 - Amazon WorkMail 작업을 수행한 후 알림을 전송할 Amazon SNS 주제의 이름 또는 ARN입니다. Amazon SNS 주제 ARN의 한 가지 예시는 arn:aws:sns:us-east-1:123456789012:MyTopic입니다. 작업을 설정할 때 SNS 주제 만들기를 선택하여 Amazon SNS 주제를 만들 수도 있습니다. Amazon SNS 주제에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 가이드를 참조하세요.

> **Note**
>
> 선택한 Amazon SNS 주제는 이메일을 수신하는 데 사용하는 Amazon SES 엔드포인트와 동일한 AWS 리전에 있어야 합니다.

> **Note**
>
> Amazon SES는 WorkMail을 사용할 수 있는 리전에서만 WorkMail 작업을 지원합니다. AWS 일반 참조의 Amazon WorkMail 엔드포인트 및 할당량을 참조하세요.

IP 주소 필터 생성 콘솔 연습

이 섹션에서는 Amazon SES 콘솔을 사용하여 IP 주소 필터를 설정하는 방법을 안내합니다. IP 주소 필터링을 사용하면 광범위한 제어 수준을 제공할 수 있습니다. 이러한 IP 필터를 사용하면 특정 IP 주소 또는 IP 주소 범위의 모든 메시지를 명시적으로 차단하거나 허용할 수 있습니다.

Amazon Simple Email Service API 참조에 설명된 대로 `CreateReceiptFilter` API를 사용하여 IP 주소 필터를 선택적으로 생성할 수 있습니다.

> **Note**
>
> 정해진 일부 IP 주소에서만 메일을 받고 싶다면 차단 목록에는 `0.0.0.0/0`를 입력하고, 허용 목록에는 신뢰하는 IP 주소를 입력하세요. 이 구성은 기본적으로 모든 IP 주소를 차단하고, 명시적으로 지정하는 IP 주소에서 발신하는 메일만 허용합니다.

사전 조건

IP 주소 필터를 사용하여 수신자 기반 이메일 제어를 설정하기 전에 다음 필수 구성 요소를 충족해야 합니다.

1. Amazon SES에서 먼저 도메인 자격 증명을 생성하고 확인해야 합니다.
2. 다음으로 도메인의 DNS 설정에 MX 레코드를 게시하여 도메인에 대한 메일을 수락할 수 있는 메일 서버를 지정해야 합니다 (MX 레코드는 Amazon SES를 사용하는 AWS 리전의 메일을 수신하는 Amazon SES 엔드포인트를 참조해야 합니다.)

IP 주소 필터 생성

콘솔을 사용하여 IP 주소 필터를 생성하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Email receiving(이메일 수신)을 선택합니다.
3. IP address filters(IP 주소 필터) 탭을 선택합니다.
4. Create Filter(필터 생성)를 선택합니다.
5. 필터의 고유 이름을 입력합니다. 필드의 범례는 구문 요구 사항을 나타냅니다. (이름은 64자 미만의 영숫자, 하이픈(-), 밑줄(_), 마침표(.) 문자를 포함해야 합니다. 이름은 문자 또는 숫자로 끝나야 합니다.)
6. IP 주소 또는 IP 주소 범위를 입력합니다. 필드의 범례는 Classless Inter-Domain Routing(CIDR) 구문에 지정된 예제를 제공합니다. (단일 IP 주소의 예는 10.0.0.1입니다. IP 주소 범위의 예는 10.0.0.1/24입니다. CIDR 표기법에 대한 자세한 정보는 RFC 2317를 참조하십시오.)
7. Block(블록) 또는 Allow(허용) 라디오 버튼 중 하나를 선택하여 Policy type(정책 유형)을 선택합니다.
8. Create filter(필터 생성)를 선택합니다.
9. 다른 IP 필터를 추가하려는 경우 Create filter(필터 생성)를 클릭하고 추가하려는 각 추가 필터에 대해 이전 단계를 반복합니다.
10. IP 주소 필터를 제거하려면 해당 필터를 선택하고Delete(삭제) 버튼을 선택합니다.

Amazon SES 이메일 수신의 지표 보기

Amazon SES에서 이메일 수신을 활성화하고 이메일에 대한 수신 규칙을 만든 경우 Amazon CloudWatch를 사용하여 해당 수신 규칙 세트 및 규칙에 대한 지표를 볼 수 있습니다.

CloudWatch 콘솔에서 지표 > 모든 지표 > SES > 수신 규칙 세트 지표 및 수신 규칙 지표에서 지표를 찾을 수 있습니다.

> **Note**
>
> 아직 확인하지 않은 경우 수신 규칙 세트 지표 및 수신 규칙 지표는 SES에 표시되지 않습니다.
>
> - 이메일 수신 활성화
> - 만들어진 모든 수신 규칙
> - 귀하의 규칙과 일치하는 메일을 모두 수신했습니다.

다음과 같은 메시지 지표를 사용할 수 있습니다.

- 메시지 수신

범위	지표	설명	차원
수신 규칙 세트 지표	수신됨	SES는 적용되는 규칙이 하나 이상 있는 메시지를 성공적으로 받았습니다. 이 지표는 1 값만 가집니다.	RuleSetName
수신 규칙 지표	수신됨	SES가 메시지를 성공적으로 수신했으며 적용된 규칙을 처리하려고 시도합니다. 이 지표는 1 값만 가집니다.	RuleName

- 메시지 게시

범위	지표	설명	차원
수신 규칙 세트 지표	PublishSuccess	SES는 규칙 세트 내에 적용되는 모든 규칙을 성공적으로 실행했습니다.	RuleSetName
수신 규칙 지표	PublishSuccess	SES는 수신 메시지에 적용되는 규칙을 성공적으로 실행했습니다.	RuleName
수신 규칙 세트 지표	PublishFailure	SES가 규칙 세트 내에서 규칙을 실행하려고 할 때 오류가 발생하여 실행이 재시도됩니다.	RuleSetName

범위	지표	설명	차원
수신 규칙 지표	PublishFailure	SES에서 규칙의 작업을 실행하려고 시도할 때 오류가 발생하여 오류에 따라 실행이 재시도될 수 있습니다.	RuleName
수신 규칙 세트 지표	PublishExpired	SES가 36시간 이내에 성공하지 못했거나 재시도할 수 없는 오류가 발생했으므로 더 이상 규칙 실행을 재시도하지 않습니다.	RuleSetName
수신 규칙 지표	PublishExpired	SES가 36시간 이내에 성공하지 못해 더 이상 규칙 작업을 재시도하지 않습니다.	RuleName

Note

- 위 표에서 적용이라는 용어는 발신자가 IP 필터 또는 SES의 내부 차단 목록으로 차단되지 않았으며, 규칙에 일치하는 수신자 조건과 일치하는 TLS 정책이 있음을 의미합니다.
- 예를 들어 수신 규칙 중 하나의 작업이 사용하도록 구성된 Amazon S3 버킷, Amazon SNS 주제 또는 Lambda 함수에 대한 권한을 삭제하거나 취소한 경우 게시 실패 오류가 발생할 수 있습니다.
- 한 번에 하나의 규칙 세트만 활성화할 수 있으므로 SES는 CloudWatch에서 선택한 시간 범위 동안 활성화된 모든 규칙 세트에 대해 RuleSetName:[Active] 로 표시된 집계 지표를 게시합니다. 이렇게 하면 알람 설정을 변경하지 않고도 규칙 세트를 자유롭게 변경할 수 있다는 장점이 있습니다.

Important

수신 규칙 세트를 수정하기 위해 변경한 사항은 업데이트 후 Amazon SES가 수신하는 이메일에만 적용됩니다. 이메일은 항상 이메일을 수신할 당시의 수신 규칙 세트를 기준으로 평가됩니다.

CloudWatch 콘솔에 표시된 SES 수신 규칙 세트에 대한 지표입니다.

CloudWatch 콘솔에 표시된 SES 수신 규칙의 지표입니다.

테넌트

이 장에서는 Amazon SES 테넌트 관리를 사용하여 SES 계정 내의 여러 테넌트에서 이메일 전송을 격리, 모니터링 및 관리하는 방법을 설명합니다. 이 기능은 여러 다운스트림 엔터티를 대신하여 이메일을 보내는 독립 소프트웨어 공급업체(ISVs), 엔터프라이즈 및 조직이 별도의 평판 프로파일을 유지하고 한 테넌트의 문제가 다른 테넌트에 영향을 미치지 않도록 하는 데 도움이 됩니다.

테넌트 관리란 무엇입니까?

테넌트 관리는 SES 계정 내에서 "테넌트"라는 격리된 컨테이너를 생성할 수 있는 기능입니다. 각 테넌트는 고유한 이메일 자격 증명, 구성 세트, 템플릿 및 평판 지표를 보유하여 이메일 활동이 서로 다른 고객 또는 사업부 간에 완전히 분리되도록 할 수 있습니다.

테넌트 관리는 한 고객의 잘못된 이메일 관행이 이전에 전체 SES 계정을 일시 중지하여 다른 모든 고객에게 영향을 미칠 수 있는 문제를 해결합니다. 테넌트 격리를 사용하면 중앙 집중식 감독 및 제어를 유지하면서 여러 이메일 스트림을 독립적으로 관리할 수 있습니다.

테넌트는 관련 SES 리소스를 그룹화하는 논리적 컨테이너 역할을 합니다. 특정 테넌트를 대신하여 이메일을 보내면 SES는 평판 지표를 추적하고 테넌트 수준에서 정책을 적용합니다. 이 격리는 한 테넌트의 높은 반송률 또는 수신 거부율이 다른 테넌트가 보낸 이메일의 배달 가능성에 영향을 미치지 않도록 합니다.

테넌트 관리는 다음과 같은 경우에 특히 유용합니다.

- 여러 고객을 대신하여 이메일을 보내는 독립 소프트웨어 공급업체(ISVs)
- 기업은 다양한 사업부에서 이메일 통신을 관리합니다.
- 클라이언트 또는 애플리케이션별로 이메일 평판을 격리해야 하는 서비스 공급자.
- 테넌트별로 다양한 규제 요구 사항을 준수해야 하는 조직입니다.

테넌트 관리 작동 방식

테넌트 평판 관리는 이메일 전송 리소스에 대한 논리적 컨테이너를 생성하여 작동합니다. 확인된 자격 증명(도메인 및 이메일 주소), 구성 세트 및 템플릿을 포함하여 각 테넌트에 특정 리소스를 할당합니다. 테넌트를 대신하여 이메일을 전송할 때 API 호출 또는 SMTP 헤더에 테넌트를 지정하면 SES는 사용 중인 리소스가 해당 테넌트와 올바르게 연결되어 있는지 확인합니다.

리소스 연결

SES 계정의 리소스는 두 가지 방법으로 테넌트와 연결할 수 있습니다.

- 전용 할당 - 특정 테넌트만 사용하는 리소스입니다.
- 공유 할당 - 여러 테넌트가 사용할 수 있는 리소스입니다.

리소스를 테넌트와 연결하면 해당 테넌트는 해당 리소스를 이메일 전송에 사용할 수 있는 권한을 얻습니다. 각 전송 요청에서 SES는 지정된 테넌트가 요청에서 자격 증명, 구성 세트 및 템플릿을 사용할 권한이 있는지 확인합니다. 리소스가 제대로 연결되지 않으면 전송 요청이 실패합니다.

평판 모니터링 및 시행

SES는 반송률, 수신 거부율(사서함 공급자 피드백 루프(FBL) 시스템의 지표 포함) 및 타사 피드백 신호를 포함하여 각 테넌트의 주요 평판 지표를 지속적으로 모니터링합니다. 이러한 지표가 정의된 임계값을 초과하면 SES는 다음과 같이 분류된 "평판 조사 결과"를 생성합니다.

- 낮은 심각도 경고 - 해결하지 않으면 전달 가능성에 영향을 미칠 수 있는 사소한 문제입니다.
- 심각도가 높은 경고 - 적용이 트리거될 수 있는 전달 가능성에 영향을 미칠 수 있는 심각한 문제입니다.

이러한 결과를 기반으로 SES는 구성한 평판 정책을 통해 문제가 있는 테넌트를 자동으로 일시 중지할 수 있습니다. 세 가지 적용 수준을 사용할 수 있습니다.

- 표준(권장) - 심각도가 높은 평판 조사 결과가 감지되면 테넌트 전송을 자동으로 일시 중지합니다. 이를 통해 중단을 최소화하면서 균형 잡힌 보호 기능을 제공합니다.
- 엄격한 - 심각도가 낮은 문제를 포함하여 평판 조사 결과가 감지되면 테넌트 전송을 자동으로 일시 중지합니다. 최대 보호를 제공하지만 일시 중지가 더 자주 발생할 수 있습니다.
- 없음 - 테넌트에 대한 자동 일시 중지를 비활성화합니다. 모든 평판 조사 결과는 여전히 기록되고 표시되지만 자동 적용 작업은 수행되지 않습니다.

테넌트의 지표가 선택한 정책에 따른 적용 임계값을 충족하는 평판 조사 결과를 트리거하면 시스템은 다른 테넌트의 전송 기능에 영향을 주지 않고 테넌트의 전송 상태를 "일시 중지"로 자동 업데이트합니다. 테넌트가 일시 중지된 동안 문제를 검토하고 전송을 수동으로 다시 활성화할 때까지 해당 테넌트를 사용하여 이메일을 보내려는 시도는 실패합니다. 또한 필요한 경우 테넌트의 전송 기능을 수동으로 일시 중지하고 전송을 재개할 준비가 되면 일시 중지를 해제할 수 있습니다.

테넌트 설정

주제

- 테넌트 생성
- 테넌트에 리소스 할당
- 평판 정책 구성

테넌트 생성

콘솔 사용:

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/ses/ SES 콘솔을 엽니다.
2. 탐색 창에서 테넌트를 선택합니다.
3. 테넌트 생성을 선택합니다.
4. 테넌트 이름에 테넌트의 고유한 이름을 입력합니다.
5. 테넌트 생성을 선택합니다.

사용 AWS CLI:

```
aws sesv2 create-tenant \
    --tenant-name "MyTenant" \
    --region us-east-1
```

테넌트에 리소스 할당

테넌트를 생성한 후 테넌트가 이메일을 보내기 전에 확인된 자격 증명과 구성 세트를 하나 이상 할당해야 합니다.

콘솔 사용:

1. SES 콘솔에서 테넌트 페이지로 이동합니다.
2. 구성할 테넌트를 선택합니다.
3. 테넌트 설정 섹션에서 자격 증명 및 구성 세트 카드를 사용하여 이러한 리소스를 할당할 수 있습니다. 선택적으로 탭 섹션까지 아래로 스크롤하고 자격 증명 및 구성 세트 탭을 사용하여 동일한 작업을 수행할 수 있습니다.

> **Note**
> 테넌트와 연결된 자격 증명 또는 구성 세트를 삭제하려는 시도는 실패합니다. 연결된 리소스를 삭제하려면 먼저 테넌트에서 이러한 연결을 제거해야 합니다.

4. (선택 사항) 태그 탭을 선택하여 테넌트에 하나 이상의 태그를 할당할 수 있습니다.

사용 AWS CLI:

```
# Assign an identity to a tenant
aws sesv2 create-tenant-resource-association \
    --tenant-name "MyTenant" \
    --resource-arn "arn:aws:ses:us-east-1:123456789012:identity/example.com" \
    --region us-east-1

# Assign a configuration set to a tenant
aws sesv2 create-tenant-resource-association \
    --tenant-name "MyTenant" \
    --resource-arn "arn:aws:ses:us-east-1:123456789012:configuration-set/MyConfigSet" \
    --region us-east-1
```

평판 정책 구성

평판 정책은 평판 지표를 기반으로 테넌트의 이메일 전송이 자동으로 일시 중지되는 시기를 결정합니다. 테넌트를 생성하면 SES가 표준 평판 정책을 자동으로 할당합니다. 다른 정책으로 변경하려면 다음 단계를 따르세요.

콘솔 사용:

1. SES 콘솔에서 테넌트 페이지로 이동합니다.
2. 구성할 테넌트를 선택합니다.
3. 테넌트 설정 섹션에서 평판 정책 카드를 사용하여 정책을 할당할 수 있습니다. 선택적으로 탭 섹션까지 아래로 스크롤하고 평판 정책 탭을 사용하여 동일한 작업을 수행할 수 있습니다.
4. 다음 정책 중 하나를 선택합니다.
   - 표준(권장) - 심각도가 높은 결과가 감지되면 전송을 일시 중지합니다.
   - 엄격한 - 결과(낮은 심각도 포함)가 감지되면 전송을 일시 중지합니다.

- 없음 - 결과에 관계없이 전송을 자동으로 일시 중지하지 않습니다. 에 설명된 대로이 수준과 관련된 위험이 있습니다신뢰 및 안전.

사용 AWS CLI:

```
update-reputation-entity-policy \
    --reputation-entity-type "RESOURCE" \
    --reputation-entity-reference "arn:aws:ses:us-east-1:123456789012:tenant/tenantId"
 \
    --reputation-entity-policy "arn:aws:ses:us-east-1:aws:reputation-policy/standard"
```

테넌트와 이메일 전송

테넌트를 통해 이메일을 전송할 때 API 호출 또는 SMTP 헤더에 테넌트를 지정하고 사용된 모든 리소스가 해당 테넌트와 연결되어 있는지 확인해야 합니다.

주제

- 테넌트와 함께 SendEmail API 사용
- 테넌트와 함께 SMTP 사용

테넌트와 함께 SendEmail API 사용

AWS CLI 예:

```
aws sesv2 send-email \
  --tenant-name "MyTenant" \
  --from-email-address "sender@example.com" \
  --destination "ToAddresses=recipient@example.com" \
  --content "Simple={Subject={Data='Test Subject',Charset=utf-8},Body={Text={Data='Test
 email body',Charset=utf-8}}}" \
  --configuration-set-name "MyConfigSet"
```

AWS SDK for Python 예제:

```
import boto3

client = boto3.client('sesv2')
```

```
response = client.send_email(
    FromEmailAddress='sender@example.com',
    Destination={
        'ToAddresses': ['recipient@example.com']
    },
    Content={
        'Simple': {
            'Subject': {
                'Data': 'Test email'
            },
            'Body': {
                'Text': {
                    'Data': 'This is a test email sent using a tenant.'
                }
            }
        }
    },
    ConfigurationSetName='MyConfigurationSet',
     TenantName='MyTenant'
)
```

테넌트와 함께 SMTP 사용

SMTP를 사용하여 테넌트를 통해 이메일을 전송하는 경우 이메일 헤더에 테넌트 정보를 포함합니다.

```
X-SES-TENANT: MyTenant
```

이 헤더는 SES가 적절한 리소스 검증 및 평판 추적을 적용할 수 있도록 이메일 전송 작업에 사용할 테넌트를 SES에 알려줍니다.

테넌트 상태 관리

주제

- 테넌트 상태 및 지표 보기
- 테넌트 일시 중지 및 일시 중지 해제

테넌트 상태 및 지표 보기

콘솔 사용:

1. SES 콘솔에서 테넌트 페이지로 이동합니다.
2. 테넌트를 선택하여 세부 정보를 봅니다.
3. 테넌트 상태 상자에 다음이 표시됩니다.

   전송 상태:.

   - 활성화됨 - 테넌트가 이메일을 보낼 수 있습니다.
   - 일시 중지됨 - 사용자 또는 SES 자동 정책이이 테넌트에 대한 전송을 일시 중지했습니다.
   - 적용됨 - 심각한 평판 문제로 인해 SES가 전송을 일시 중지했습니다.
   - 복원됨 - 일시 중지된 후 전송이 다시 활성화되었습니다.

   평판 상태:

   - 탐지된 결과 없음 - 전달 가능성에 영향을 미치는 문제가 없습니다.
   - 낮은 심각도 경고 - 해결하지 않으면 전달 가능성에 영향을 미칠 수 있는 사소한 문제입니다.
   - 심각도가 높은 경고 - 전달 가능성에 영향을 미칠 수 있는 심각한 문제입니다.
4. 선택한 날짜 범위의 전송, 반송 메일 및 수신 거부 지표를 보려면 리소스 탭에서 통계 전송까지 아래로 스크롤합니다.

사용 AWS CLI:

```
# Get tenant details
aws sesv2 get-tenant --tenant-name "MyTenant"
```

테넌트 일시 중지 및 일시 중지 해제

필요한 경우 테넌트의 전송 기능을 수동으로 일시 중지하고 전송을 재개할 준비가 되면 일시 중지를 해제할 수 있습니다.

콘솔 사용:

1. SES 콘솔에서 테넌트 페이지로 이동합니다.
2. 일시 중지하거나 일시 중지 해제하려는 테넌트 옆의 확인란을 선택합니다.
3. 전송 일시 중지 또는 전송 재개를 선택합니다.
4. 작업을 확인합니다.

사용 AWS CLI:

테넌트를 일시 중지하려면:

```
# Pause a tenant
aws sesv2 update-reputation-entity-customer-managed-status \
  --reputation-entity-type RESOURCE
  --reputation-entity-reference "arn:aws:ses:us-east-1:593442965613:tenant/tenantId"
  --sending-status DISABLED
```

테넌트 일시 중지를 해제하려면:

```
# Unpause a tenant
aws sesv2 update-reputation-entity-customer-managed-status \
  --reputation-entity-type RESOURCE
  --reputation-entity-reference "arn:aws:ses:us-east-1:593442965613:tenant/tenantId"
  --sending-status ENABLED
```

평판 조사 결과 작업

주제

- 평판 조사 결과 보기
- 평판 조사 결과 이해
- 평판 문제 해결

평판 조사 결과 보기

콘솔 사용:

1. SES 콘솔에서 테넌트 페이지로 이동합니다.
2. 검사할 테넌트를 선택합니다.
3. 평판 조사 결과 테이블로 이동합니다.
4. 유형, 심각도, 탐지 날짜 및 문제 해결을 위한 지침을 포함하여 활성 조사 결과를 검토합니다.

사용 AWS CLI:

```
aws sesv2 list-recommendations \
```

```
    --filter='{ "RESOURCE_ARN":"arn:aws:ses:us-east-1:012345678901:tenant/tenantId"}'
```

응답에는 모든 활성 결과에 대한 세부 정보가 포함됩니다.

```
{
    "Recommendations": [
        {
            "ResourceArn": "arn:aws:ses:us-east-1:012345678901:tenant/{tenant-name}/
{tenant-id}",
            "Type": "BOUNCE",
            "Description": "The bounce rate exceeded 15.0% based on a representative
 volume of 664 emails from July 11, 2025 at 14:41 (UTC) to July 11, 2025 at 16:26
 (UTC).",
            "Status": "OPEN",
            "CreatedTimestamp": "2025-07-11T16:16:14.029000+00:00",
            "LastUpdatedTimestamp": "2025-07-11T16:37:14.145000+00:00",
            "Impact": "HIGH"
        }
    ]
}
```

평판 조사 결과 이해

평판 조사 결과는 테넌트의 이메일 전송 관행과 관련된 잠재적 문제에 대한 인사이트를 제공합니다. 각 결과에는 다음이 포함됩니다.

- 영향 - 심각도 수준(높음 또는 낮음).
- 조사 결과 유형 - 반송 메일 발생률, 수신 거부 발생률, 메일박스 공급자의 타사 피드백, IP 차단 목록 목록.
- 연령 - 처음 감지된 날짜 이후의 시간입니다.
- 설명 - 문제에 대한 컨텍스트입니다(예: 결과를 트리거한 특정 속도).
- 마지막 확인됨 - 가장 최근 상태 업데이트 날짜입니다.
- 문제 해결 - 문제 해결에 도움이 되는 지침이 포함된 SES 개발자 안내서의 관련 섹션으로 연결되는 링크입니다.

일반적인 결과는 다음과 같습니다.

- 높은 반송률 - 반송률이 구성된 임계값을 초과하는 경우.

- 수신 거부 활동 - 임곗값을 초과하는 사서함 공급자로부터 스팸 보고서가 수신되는 경우.
- 타사 피드백 - 사서함 공급자의 부정적인 신호입니다.
- 차단 목록 표시 - 전송 IPs 평판 차단 목록에 표시되는 경우.

평판 문제 해결

평판 조사 결과를 받으면 근본 원인을 조사하고 즉시 시정 조치를 취하는 것이 중요합니다.

1. 근본 원인 해결 - 테넌트에게 목록 위생을 개선하거나, 이메일 콘텐츠를 업데이트하거나, 기술적 문제를 해결하도록 조언합니다.
2. 전송 사례 검토 - 테넌트가 이메일 모범 사례를 준수하는지 확인합니다.
3. 지표 모니터링 - 반송 메일 및 수신 거부 발생률이 개선되었는지 확인합니다.
4. 영향을 받는 당사자와 소통 - 관련 이해관계자에게 문제 및 해결 단계에 대해 알립니다.

모니터링 및 분석

주제

- CloudWatch 지표 설정
- EventBridge 알림 설정

CloudWatch 지표 설정

SES는 테넌트별 지표를 Amazon CloudWatch에 게시합니다. 각 테넌트에 대해 다음 지표를 사용할 수 있습니다.

- 전송 - 테넌트가 보낸 총 이메일 수입니다.
- 반송 메일 - 테넌트에 대해 반송된 이메일 수입니다.
- 수신 거부 - 테넌트에 대해 반송된 이메일 수입니다.

CloudWatch에서 테넌트 지표에 액세스:

1. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 지표를 선택합니다.
3. AWS/SES 네임스페이스를 선택합니다.

4. 테넌트별 지표를 보려면 테넌트 TenantId 및 TenantName별을 선택합니다.

EventBridge 알림 설정

기본적으로 SES는 테넌트 평판 조사 결과가 감지되거나 테넌트 상태 변경이 발생할 때 EventBridge 기본 이벤트 버스로 이벤트를 전송합니다.

기본 이벤트 버스에 규칙을 만들어 EventBridge가 하나 이상의 지정된 대상으로 보낼 특정 이벤트를 식별할 수 있습니다.

다음과 같은 세부 유형을 사용할 수 있습니다.

테넌트 전송 상태 변경의 경우:

- `Sending Status Enabled` - 테넌트가 활성화되어 이메일을 보낼 수 있습니다.
- `Sending Status Disabled` - 테넌트가 일시 중지되어 이메일을 보낼 수 없습니다.

평판 조사 결과의 경우:

- `Advisor Recommendation Status Open`
- `Advisor Recommendation Status Closed`

EventBridge 규칙 설정

1. https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. Create rule을 선택합니다.
3. 규칙의 이름과 설명을 입력합니다.
4. 패턴 정의(Define pattern)에서 이벤트 패턴(Event pattern)을 선택합니다.
5. 서비스별 사전 정의된 패턴(Pre-defined pattern by service)을 선택합니다.
6. 서비스 이름으로 SES를 선택합니다.
7. 이벤트 유형에서 전송 상태 활성화됨 또는 전송 상태 비활성화됨을 선택합니다.
8. 관심 있는 특정 이벤트와 일치하도록 이벤트 패턴을 구성합니다.
9. 대상(예: AWS Lambda 함수, Amazon SNS 주제 또는 Amazon SQS 대기열)을 선택합니다.
10. 필요에 따라 추가 설정을 구성합니다.
11. 생성(Create)을 선택합니다.

평판 조사 결과에 대한 EventBridge 규칙 패턴의 예:

```
{
  "detail-type": "Advisor Recommendation Status Open",
  "source": "aws.ses",
  "account": "012345678901",
  "time": "2023-11-15T17:00:59Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ses:us-east-1:012345678901:tenant/{tenant-name}/{tenant-id}"
  ],
  "detail": {
       "version": "1.0.0",
       "data": "The bounce rate exceeded 15.0% based on a representative volume of 197
 emails from July 11, 2025 at 14:43 (UTC) to July 11, 2025 at 16:13 (UTC).",
       "metadata":{"impact":"HIGH","type":"BOUNCE"}

    }
}
```

상태 변경 전송을 위한 EventBridge 규칙 패턴의 예:

```
{
        "detail-type": "Sending Status Disabled",
        "source": "aws.ses",
        "account": "012345678901",
        "time": "2025-07-24T12:44:28Z",
        "region": "us-east-1",
        "resources": [
            "arn:aws:ses:us-east-1:012345678901:tenant/{tenant-name}/{tenant-id}"
        ],
        "detail": {
            "version": "1.0.0",
            "data": {
                "origin": "CUSTOMER_MANAGED",
                "record": {
                    "status": "DISABLED",
                    "cause": "Status manually updated.",
                    "lastUpdatedTimestamp": [2025, 7, 24, 12, 44, 28, 995000000]
                }
            }
        }
```

```
    }
```

필드 설명:

- source - 이벤트를 생성한 서비스를 식별합니다. SES 이벤트의 경우 이 값은 `aws.ses`입니다.
- detail-type - 상태 변경 이벤트의 유형입니다(위의 이벤트 유형 참조).
- 리소스 - 영향을 받는 테넌트의 ARN을 포함하는 배열입니다.
- detail.data.origin – 상태 변경의 소스(예: "CUSTOMER_MANAGED" 또는 "AWS_MANAGED").
- detail.data.record.status - 테넌트의 새 상태(ENABLED, DISABLED 또는 REINSTATED).
- detail.data.record.cause – 상태가 변경된 이유에 대한 설명입니다.
- detail.data.record.lastUpdatedTimestamp – 상태가 업데이트된 시점의 타임스탬프입니다.

이러한 이벤트를 사용하여 테넌트 상태 변경을 모니터링하고 애플리케이션에서 응답을 자동화할 수 있습니다. 예를 들어 테넌트가 비활성화될 때 알림을 트리거하거나 시간 경과에 따른 테넌트 상태 지표를 추적할 수 있습니다.

AWS 신뢰 및 안전과의 통합

AWS 신뢰 및 안전이 일반적으로 계정 수준 적용을 초래하는 문제를 감지하면 테넌트 시스템은 더 많은 대상 응답을 활성화합니다. Trust & Safety는 전체 계정을 일시 중지하는 대신 문제가 있는 테넌트만 일시 중지하는 동시에 규정 준수 테넌트가 계속 전송하도록 허용할 수 있습니다.

이 테넌트 수준 적용은 평판 문제의 영향을 줄이고 다른 이메일 스트림의 비즈니스 연속성을 유지하는 데 도움이 됩니다.

SES가 테넌트 평판에 영향을 미치는 심각도가 높은 문제로 인해 전송을 비활성화하고 계정을 검토 중으로 설정하면 알림을 받게 됩니다. Trust & Safety와 협력하여 문제를 해결하고 영향을 받는 테넌트에 대한 전송을 복원할 수 있도록 AWS 지원 센터에서 사례가 열립니다.

> **Note**
>
> 계정 소유자는 모든 테넌트의 평판 지표를 모니터링할 책임이 있습니다. 테넌트 기능은 테넌트 수준에서 평판 추적을 격리하지만 결합된 전송 활동은 여전히 전체 계정 평판에 영향을 미칩니다. 잘못된 전송 관행이 발생하는 테넌트는 전체 계정을 위험에 빠뜨릴 수 있습니다. 따라서 모든 테넌트가 계정의 상태를 보호하기 위해 올바른 이메일 전송 관행을 유지하는 것이 중요합니다.

모범 사례

Amazon SES에서 테넌트 평판을 효과적으로 관리하려면 다음 모범 사례를 따르세요.

- 표준 정책으로 시작 - 대부분의 테넌트에 대해 표준 평판 정책은 보호와 운영 안정성 간에 좋은 균형을 제공합니다.
- 적용 전 모니터링 - 새 테넌트를 온보딩할 때 자동 적용을 활성화하기 전에 전송 패턴(예: EventBridge)을 모니터링하는 동안 "없음" 정책을 일시적으로 사용하는 것이 좋습니다.
- EventBridge 알림 설정 - 평판 조사 결과에 대한 알림을 생성하여 자동 일시 중지가 발생하기 전에 사전 조치를 취합니다.
- 테넌트 지표 정기 검토 - 평판 조사 결과가 없는 경우에도 테넌트 수준 전송 통계를 모니터링하여 새로운 패턴을 식별합니다.
- 테넌트 교육 - 테넌트가 양호한 전송 평판을 유지하는 이메일 모범 사례 데 도움이 되는 지침을 제공합니다.
- 적절한 정책 적용 - 고위험 테넌트 또는 평판 문제의 이력이 있는 테넌트에 엄격한 정책을 적용합니다.
- 결과에 신속하게 대응 - 조사 결과가 감지되면 즉시 근본 원인을 조사하고 적절한 수정 조치를 취합니다.
- 리소스 계획 - 비즈니스 요구 사항에 맞게 테넌트 구조를 설계합니다. 테넌트는 여러 고객(ISVs), 사업부, 클라이언트/애플리케이션 유형 및 규제 요구 사항을 나타낼 수 있습니다.

제한 사항

테넌트 관리를 사용할 때는 다음 제한 사항에 유의하세요.

- 리전 범위 - 테넌트는 리전별로 다르며 자동으로 복제되지 않습니다 AWS 리전. 여러 리전에서 이메일을 보내는 경우 각 리전에서 테넌트 평판을 별도로 구성하고 모니터링해야 합니다.
- 할당량 제한 - 기본적으로 계정은 최대 10,000개의 테넌트를 생성할 수 있습니다. Service Quota 콘솔을 AWS 통해 증가를 요청할 수 있으며, 적격 계정에 대해 최대 300,000개의 테넌트를 자동 승인할 수 있습니다.
- 구성 세트 요구 사항 - 테넌트를 대신하여 전송할 때는 해당 테넌트와 연결된 구성 세트를 지정하거나 테넌트와 연결된 기본 구성 세트가 있는 자격 증명을 사용해야 합니다.
- 지표 계산 기간 - 평판 지표는 지표 유형에 따라 일반적으로 24시간~7일 동안 최근 전송 활동을 기준으로 계산됩니다.

- 최소 전송 볼륨 - 일부 평판 조사 결과를 트리거하려면 최소 대표 이메일 볼륨이 필요합니다.
- 유예 기간 다시 활성화 - 일시 중지된 테넌트를 다시 활성화한 후 활성 평판 조사 결과가 일시적으로 무시되어 테넌트가 복구할 수 있는 '복구됨' 상태가 됩니다. 테넌트는 모든 활성 결과가 해결될 때까지 이 상태를 유지합니다.
- 교차 계정 제한 - 테넌트는 여러 AWS 계정에 걸쳐 있을 수 없습니다. 각 계정은 자체 테넌트 세트를 독립적으로 관리합니다.
- 테넌트 중첩 없음 - 테넌트는 다른 테넌트를 포함할 수 없습니다. 테넌트는 플랫 구조입니다.

요금

이메일 수에 따라 매월 테넌트당 추가 요금이 부과됩니다. 자세한 요금 정보는 SES 요금 페이지를 참조하세요.

테넌트 지표와 함께 CloudWatch를 사용하는 경우 각 테넌트에 대한 표준 CloudWatch 지표는 기본 모니터링의 일부로 추가 비용 없이 제공됩니다. 사용자 지정 대시보드, 경보 또는 세부 모니터링과 같은 추가 CloudWatch 기능에는 표준 CloudWatch 요금이 발생할 수 있습니다.

Amazon SES에서 확인된 자격 증명

Amazon SES에서 확인된 자격 증명은 사용자가 이메일을 보내거나 받기 위해 사용하는 도메인 또는 이메일 주소입니다. Amazon SES를 사용하여 이메일을 보내려면 먼저 "From", "Source", "Sender" 또는 "Return-Path" 주소로 사용하려는 각 자격 증명을 생성하고 확인해야 합니다. Amazon SES를 통해 자격 증명 확인함으로써 사용자의 소유임을 확인하고 무단 사용을 방지할 수 있습니다.

계정이 여전히 Amazon SES 샌드박스 환경에 있을 경우, Amazon SES 메일박스 시뮬레이터에서 제공하는 수신함을 테스트하기 위해 이메일을 보내는 경우를 제외하고 이메일을 전송할 예정인 모든 이메일 주소도 확인해야 합니다. 자세한 내용은 the section called “수동으로 메일박스 시뮬레이터 사용” 단원을 참조하십시오.

Amazon SES 콘솔 또는 Amazon SES API를 사용하여 자격 증명을 생성할 수 있습니다. 자격 증명 확인 프로세스는 생성하도록 선택한 자격 증명 유형에 따라 다릅니다.

> **Tip**
>
> SES를 처음 사용하는 경우 시작하기 마법사를 사용하여 첫 번째 ID (이메일 주소 또는 도메인)를 생성하고 확인할 수 있습니다.

내용

- Amazon SES에서 자격 증명 생성 및 확인
- Amazon SES에서 ID 관리
- Amazon SES의 자격 증명 구성
- 시뮬레이터를 사용하여 Amazon SES에서 테스트 이메일 전송

Amazon SES에서 자격 증명 생성 및 확인

Amazon SES에서는 도메인 수준의 자격 증명을 생성하거나 이메일 주소 자격 증명을 만들 수 있습니다. 이러한 자격 증명은 유형은 상호 배타적이지 않습니다. 특정 이메일 메일 주소에 사용자 지정 구성을 적용하려는 경우가 아니라면 대부분의 경우 도메인 자격 증명을 만든 후 개별 이메일 주소 자격 증명을 만들고 확인할 필요가 없습니다. 도메인을 생성하고 도메인을 기반으로 이메일 주소를 활용하든 개별 이메일 주소를 생성하든 두 접근 방식 모두 이점이 있습니다. 선택하는 방법은 아래에 설명된 대로 특정 요구 사항에 따라 달라집니다.

이메일 주소 자격 증명을 생성하고 확인하는 것이 SES를 시작하는 가장 빠른 방법이지만 도메인 수준에서 자격 증명을 확인하는 데는 이점이 있습니다. 이메일 주소 자격 증명을 확인하면 해당 이메일 주소만 메일을 보내는 데 사용할 수 있지만, 도메인 자격 증명을 확인하면 각각을 개별적으로 확인하지 않고도 확인된 도메인의 모든 하위 도메인이나 이메일 주소에서 이메일을 보낼 수 있습니다. 예를 들어 example.com이라는 도메인 자격 증명을 생성하고 확인하는 경우 a.example.com, a.b.example.com에 대한 별도의 하위 도메인 자격 증명 또는 user@example.com, user@a.example.com 등에 대한 별도의 이메일 주소 자격 증명을 생성할 필요가 없습니다.

단, 도메인에서 상속된 확인을 사용하는 이메일 주소 자격 증명은 간단한 이메일 전송으로 제한됩니다. 그 이상의 고급 전송을 수행하려면 이메일 주소 자격 증명으로도 명시적으로 확인해야 합니다. 고급 보내기에는 구성 세트가 포함된 이메일 주소, 위임 전송에 대한 정책 권한 부여 및 도메인 설정을 재정의하는 구성 사용이 포함됩니다.

위에서 설명한 확인 상속 및 이메일 전송 기능을 명확히 이해하는 데 도움이 되도록 다음 표에 도메인/이메일 주소 확인의 각 조합이 분류되어 있으며 각각의 상속, 전송 수준 및 표시 상태가 나열되어 있습니다.

	도메인만 확인됨	이메일 주소만 확인됨	도메인과 이메일 주소 모두 확인됨
상속 수준	하위 도메인과 이메일 주소에서 상위 도메인의 확인을 상속합니다.	이메일 주소가 명시적으로 확인되었습니다.	• 하위 도메인에서 상위 도메인의 확인을 상속합니다. • 이메일 주소가 명시적으로 확인되었습니다.
전송 수준	이메일 주소가 간단한 이메일 전송으로 제한됩니다.	이메일 주소를 고급 전송*에 사용할 수 있습니다.	이메일 주소를 고급 전송*에 사용할 수 있습니다.
표시된 상태	콘솔/API 상태: • 도메인/하위 도메인 = 확인됨 • 이메일 주소 = 확인되지 않음.	콘솔/API 상태: • 이메일 주소 = 확인됨	콘솔/API 상태: • 도메인/하위 도메인 = 확인됨 • 이메일 주소 = 확인됨.

*고급 전송에는 구성 세트가 포함된 이메일 주소, 위임 전송에 대한 정책 권한 부여 및 도메인 설정을 재정의하는 구성 사용이 포함됩니다.

둘 이상의 동일한 도메인 또는 이메일 주소에서 이메일을 보내려면 각 리전에 대해 별도의 자격 증명을 생성하고 확인해야 AWS 리전합니다. 각 리전에서 최대 10,000개의 자격 증명을 확인할 수 있습니다.

이메일 주소 자격 증명을 생성하고 확인할 때는 다음을 고려하세요.

- 각각을 개별적으로 확인하지 않고도 확인된 도메인의 모든 하위 도메인이나 이메일 주소에서 이메일을 전송할 수 있습니다. 예를 들어 example.com에 대한 자격 증명을 만들고 확인하면 a.example.com, a.b.example.com, user@example.com, user@a.example.com 등에 대해 별도의 자격 증명을 만들 필요가 없습니다.
- RFC 1034에 지정된 바에 따라 각 DNS 레이블은 최대 63자를 사용할 수 있으며 전체 도메인 이름은 총 255자를 초과할 수 없습니다.
- 도메인, 하위 도메인 또는 루트 도메인을 공유하는 이메일 주소를 확인하면, 자격 증명 설정(예: 피드백 알림)이 사용자가 확인한 가장 세부적인 수준까지 적용됩니다.
  - 확인된 이메일 주소 자격 증명 설정은 확인된 도메인 자격 증명 설정보다 우선합니다.
  - 확인된 하위 도메인 자격 증명 설정은 확인된 도메인 자격 증명 설정보다 우선하며, 낮은 수준의 하위 도메인 설정은 높은 수준의 하위 도메인 설정보다 우선합니다.

    예를 들어, user@a.b.example.com, a.b.example.com, b.example.com 및 example.com을 확인한다고 가정해 보겠습니다. 이들은 다음 시나리오에서 사용될 확인된 자격 증명 설정입니다.

    - user@example.com(확인되지 않은 이메일 주소)에서 보낸 이메일은 example.com의 설정을 사용합니다.
    - user@a.b.example.com(확인된 이메일 주소)에서 보낸 이메일은 user@a.b.example.com의 설정을 사용합니다.
    - user@b.example.com(확인되지 않은 이메일 주소)에서 보낸 이메일은 b.example.com의 설정을 사용합니다.
- 추가 확인 단계를 수행하지 않고도 확인된 이메일 주소에는 레이블을 추가할 수 있습니다. 이메일 주소에 레이블을 추가하려면 계정 이름과 @ 기호 사이에 더하기 기호(+)를 추가한 다음, 텍스트 레이블을 입력합니다. 예를 들어, sender@example.com을 이미 확인한 경우, 이메일의 "From" 또는 "Return-Path" 주소로 sender+myLabel@example.com을 사용할 수 있습니다. 이 기능을 사용하면 VERP(Variable Envelope Return Path)를 구현할 수 있습니다. 그러면 VERP를 사용하여 메일 그룹에서 배달 불가 이메일 주소를 탐지 및 제거할 수 있습니다.
- 도메인 이름은 대/소문자를 구분하지 않습니다. example.com을 확인하면 EXAMPLE.com에서도 보낼 수 있습니다.

- 이메일 주소는 대/소문자를 구분합니다. sender@EXAMPLE.com을 확인한 경우 sender@example.com을 확인하지 않는 한sender@example.com에서 이메일을 전송할 수 없습니다.
- 각각에서 최대 10 AWS 리전,000개의 자격 증명(도메인 및 이메일 주소, 모든 조합)을 확인할 수 있습니다.

> **Tip**
>
> SES를 처음 사용하는 경우 시작하기 마법사를 사용하여 첫 번째 ID (이메일 주소 또는 도메인)를 생성하고 확인할 수 있습니다.

내용

- 도메인 자격 증명 생성
- DNS 공급자로 DKIM 도메인 자격 증명 확인
- 이메일 주소 자격 증명 생성
- 이메일 주소 자격 증명 확인
- 자격 증명을 생성 및 확인하는 동시에 기본 구성 세트 할당
- 사용자 지정 확인 이메일 템플릿 사용

도메인 자격 증명 생성

도메인 자격 증명 만들기의 일부는 DKIM 기반 확인을 구성하는 것입니다. DomainKeys Identified Mail(DKIM)은 Amazon SES가 도메인 소유권을 확인하는 데 사용하는 이메일 인증 방법이며 수신 메일 서버가 이메일 신뢰성을 검증하는 데 사용합니다. Easy DKIM 또는 자체 DKIM 가져오기(BYODKIM)를 사용하여 DKIM을 구성하도록 선택할 수 있으며 선택에 따라 다음과 같이 프라이빗 키의 서명 키 길이를 구성해야 합니다.

- Easy DKIM - Amazon SES 기본값인 2048비트를 수락하거나 1024비트를 선택하여 재정의합니다.
- BYODKIM - 프라이빗 키 길이는 최소 1024비트 이상에서 최대 2048비트여야 합니다.

DKIM 서명 키 길이 및 키 변경 방법에 대해 자세히 알아보려면 the section called “DKIM 서명 키 길이” 단원을 참조하십시오.

다음 절차에서는 Amazon SES 콘솔을 사용하여 도메인 자격 증명을 생성하는 방법을 보여줍니다.

- 이미 도메인을 만들었는데 확인이 필요한 경우 이 페이지의 the section called "도메인 자격 증명 확인" 절차로 건너뜁니다.

도메인 자격 증명을 생성하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 자격 증명을 선택합니다.
3. 자격 증명 생성(Create identity)을 선택합니다.
4. 자격 증명 세부 정보에서 생성하려는 자격 증명 유형으로 Domain(도메인)을 선택합니다. 도메인 확인 프로세스를 완료하려면 도메인의 DNS 설정에 액세스할 수 있어야 합니다.
5. 도메인 필드에 도메인 또는 하위 도메인의 이름을 입력합니다.

> **Tip**
> 도메인이 www.example.com인 경우 example.com을 도메인으로 입력합니다. "www." 부분을 포함해서는 안 되며 이는 도메인 확인 프로세스에 실패할 수 있기 때문입니다.

6. (선택) 기본 구성 세트를 할당하려면 확인란을 선택합니다.

   1. 기본 구성 세트에서 자격 증명에 할당할 기존 구성 세트를 선택합니다. 아직 구성 세트를 생성하지 않은 경우 구성 세트 단원을 참조하십시오.

   > **Note**
   > Amazon SES는 전송 시 다른 세트가 지정되지 않은 경우에만 할당된 구성 세트로 기본 설정됩니다. 구성 세트가 지정된 경우 Amazon SES는 기본 세트 대신 지정된 세트를 적용합니다.

7. (선택) 사용자 지정 MAIL FROM 도메인을 사용하려면 확인란을 선택하고 다음 단계를 완료합니다. 자세한 내용은 the section called "사용자 지정 MAIL FROM 도메인 사용" 단원을 참조하십시오.

   1. MAIL FROM 도메인에 MAIL FROM 도메인으로 사용할 하위 도메인을 입력합니다. 이 도메인은 확인 중인 도메인 ID의 하위 도메인이어야 합니다. MAIL FROM 도메인은 사용자가 이메일을 보내는 도메인이 아니어야 합니다.

2. MX 실패에 대한 조치는 전송 시 필요한 MX 레코드를 찾을 수 없는 경우 Amazon SES가 수행해야 하는 조치를 나타냅니다. 다음 옵션 중 하나를 선택하세요.
   - 기본 MAIL FROM 도메인 사용 - 사용자 지정 MAIL FROM 도메인의 MX 레코드가 올바르게 설정되지 않은 경우 Amazon SES는 amazonses.com의 하위 도메인을 사용합니다. 하위 도메인은 Amazon SES를 사용하는 AWS 리전 에 따라 달라집니다.
   - 메시지 거부 - 사용자 지정 MAIL FROM 도메인의 MX 레코드가 올바르게 설정되지 않은 경우 Amazon SES는 `MailFromDomainNotVerified` 오류를 반환합니다. 이 옵션을 선택한 경우, 이 도메인에서 보내려는 이메일이 자동으로 거부됩니다.
3. Route53에 DNS 레코드 게시(Publish DNS records to Route53)에서 사용(Enabled)을 선택하여 도메인이 Amazon Route 53를 통해 호스트되는 경우 생성 시에 SES가 연결된 TXT 및 MX 레코드를 게시하도록 할 수 있습니다. 이들 레코드를 나중에 게시하려면 사용(Enabled) 확인란을 선택 취소합니다. (나중에 다시 돌아와 자격 증명을 편집하여 Route 53에 레코드를 게시할 수 있습니다. the section called “콘솔을 사용하여 ID 편집” 섹션을 참조하세요.)

8. (선택 사항) Easy DKIM을 사용하는 SES 기본 설정을 제외하고 사용자 지정 DKIM 기반 확인을 구성하려면 도메인 확인에서 고급 DKIM 설정을 확장하고 구성하려는 DKIM 유형을 선택합니다.

   a. Easy DKIM:

      i. 자격 증명 유형에서 Easy DKIM을 선택합니다.

      ii. DKIM 서명 키 길이 필드에서 RSA_2048_BIT 또는 RSA_1024_BIT를 선택합니다.

      iii. Route53에 DNS 레코드 게시(Publish DNS records to Route53)에서 사용(Enabled)을 선택하여 도메인이 Amazon Route 53를 통해 호스트되는 경우 생성 시에 SES가 연결된 CNAME 레코드를 게시하도록 할 수 있습니다. 이들 레코드를 나중에 게시하려면 사용(Enabled) 확인란을 선택 취소합니다. (나중에 다시 돌아와 자격 증명을 편집하여 Route 53에 레코드를 게시할 수 있습니다. the section called “콘솔을 사용하여 ID 편집” 섹션을 참조하세요.)

   b. Deterministic Easy DKIM(DEED):

> **Tip**
>
> 글로벌(복제본) 자격 증명을 생성하는 경우이 형식의 DKIM을 사용해야 합니다. DEED는 상위 리전에서 동일한 이름의 기존 자격 증명에 대한 Easy DKIM 설정을 활용하고 추가 DNS 설정을 수행할 필요 없이 새 자격 증명에 서명합니다. 자세한 내용은 DEED를 참조하세요.

i. 자격 증명 유형에서 Deterministic Easy DKIM을 선택합니다.

ii. 상위 리전 드롭다운 메뉴에서 글로벌(복제본) 자격 증명에 입력한 것과 동일한 이름의 Easy DKIM 서명 자격 증명이 있는 상위 리전을 선택합니다. (복제 리전은 기본적으로 SES 콘솔에 로그인한 리전으로 설정됩니다.)

c. DKIM 인증 토큰 제공(BYODKIM):

i. 이미 퍼블릭-프라이빗 키 페어를 생성하고 DNS 호스트 공급자에 퍼블릭 키를 추가했는지 확인합니다. 자세한 내용은 the section called “BYODKIM - 자체 DKIM 사용” 단원을 참조하십시오.

ii. 자격 증명 유형에서 DKIM 인증 토큰 제공(BYODKIM)을 선택합니다.

iii. 프라이빗 키(Private key)에서 퍼블릭-프라이빗 키 페어로 생성한 프라이빗 키를 붙여 넣습니다. 프라이빗 키는 최소 1024비트 RSA 암호화와 최대 2048비트를 사용하고 base64(PEM) 인코딩을 사용하여 인코딩해야 합니다.

> **Note**
>
> 생성된 프라이빗 키의 첫 번째 줄(`-----BEGIN PRIVATE KEY-----`)과 마지막 줄(`-----END PRIVATE KEY-----`)을 삭제해야 합니다. 또한 생성된 프라이빗 키에서 줄 바꿈을 제거해야 합니다. 결과 값은 공백이나 줄 바꿈이 없는 문자열입니다.

iv. 선택기 이름(Selector name)의 경우, 도메인의 DNS 설정에서 지정된 선택기 이름을 입력합니다.

9. DKIM 서명 필드에서 Enabled(활성화됨) 상자를 확인합니다.
10. (선택) 태그 키와 키 값(선택)을 포함시켜 하나 이상의 태그를 도메인 자격 증명에 추가합니다.
    1. Add new tag(새 태그 추가)를 선택하고 키를 입력합니다. 또한, 태그에 값을 추가할 수 있습니다.
    2. 50개 이하의 추가 태그에 대해 반복하거나 Remove(제거)를 선택하여 태그를 제거합니다.
11. Create identity(자격 증명 생성)를 선택합니다.

DKIM을 사용하여 도메인 자격 증명을 생성하고 구성했으므로 DNS 공급자로 확인 프로세스를 완료해야 합니다. the section called “도메인 자격 증명 확인” 섹션으로 진행하여 자격 증명 구성에 사용한 DKIM 유형에 대한 DNS 인증 절차를 따르세요.

> **Note**

DNS 공급자로 DKIM 도메인 자격 증명 확인

DKIM으로 구성된 도메인 자격 증명을 만든 후에는 선택한 DKIM 유형에 대한 각 인증 절차에 따라 DNS 공급자로 확인 프로세스를 완료해야 합니다.

도메인 자격 증명을 생성하지 않은 경우 the section called "도메인 자격 증명 생성" 섹션을 참조하세요.

> **Note**
>
> - 도메인 자격 증명을 확인하려면 도메인의 DNS 설정에 액세스해야 합니다. 이러한 설정을 변경하려면 최대 72시간이 소요될 수 있습니다.
> - Deterministic Easy DKIM(DEED)을 사용하여 글로벌(복제본) 자격 증명을 생성한 경우 추가 DNS 설정이 필요하지 않습니다.이 단계를 건너뛸 수 있습니다. 자세한 내용은 DEED를 참조하세요.

DNS 공급자로 DKIM 도메인 자격 증명 확인

1. 로드된 자격 증명(Loaded identities)에서 확인할 도메인을 선택합니다.
2. 자격 증명 세부 정보 페이지의 인증(Authentication) 탭에서 DNS 레코드 게시(Publish DNS records)를 확장합니다.
3. 도메인을 구성한 DKIM의 종류가 Easy DKIM 또는 BYODKIM인지에 따라, 해당하는 지침을 따릅니다.

   Easy DKIM

   Easy DKIM을 사용하여 구성된 도메인 자격 증명을 확인하는 방법

   1. DNS 레코드 게시(Publish DNS records) 테이블에서 DNS 공급자에 게시(추가)되어 이 섹션에 표시한 CNAME 레코드 3개를 복사합니다. 또는 Download .csv record set(레코드 세트를 .csv로 다운로드)를 선택하여 레코드 사본을 컴퓨터에 저장할 수도 있습니다.

      다음 이미지는 DNS 공급자에 게시한 CNAME 레코드의 예를 보여줍니다.

▼ Publish DNS records

After you've created your domain identity with Easy DKIM, you must complete the verification process with DKIM authentication by copying the following generated CNAME records to publish to your domain's DNS provider. Detection of these records may take up to 72 hours. For more information, see Verifying a domain identity with DKIM and Easy DKIM.

Type	Name	Value
CNAME	a32gfwufpxmw36t5sf2owbszlde3sof7._domainkey.adzel.com	a32gfwufpxmw36t5sf2owbszlde3sof7.dkim.amazonses.com
CNAME	redmf6qg6wg3no6ulb6mrmwxjeypgpdh._domainkey.adzel.com	redmf6qg6wg3no6ulb6mrmwxjeypgpdh.dkim.amazonses.com
CNAME	6d5oug5am4wtxnkr4rdwluadqdd5l74l._domainkey.adzel.com	6d5oug5am4wtxnkr4rdwluadqdd5l74l.dkim.amazonses.com

Download .csv record set

2. 각 DNS 호스트 공급자의 도메인 DNS 설정에 CNAME 레코드를 추가합니다.

   - 모든 DNS 호스트 공급자(Route 53 제외) – 도메인 DNS 또는 웹 호스팅 공급자에 로그인한 후, 이전에 복사하거나 저장한 값이 포함된 CNAME 레코드를 추가합니다. 공급자마다 DNS 레코드를 업데이트하는 절차가 다릅니다. 이 절차에 따라 DNS/호스팅 공급자 테이블을 참조하세요.

     > **Note**
     >
     > 소수의 DNS 공급자는 레코드 이름에 밑줄(_)을 포함하는 것을 허용하지 않습니다. 그러나 DKIM 레코드 이름에서 밑줄은 필수입니다. DNS 공급자가 레코드 이름에 밑줄을 입력하는 것을 허용하지 않는 경우 해당 공급자의 고객 지원 팀에 문의하세요.

   - Route 53를 DNS 호스트 공급자로 사용하는 경우 – SES를 사용하여 이메일을 보낼 때 사용한 것과 동일한 계정에 있는 Route 53를 사용하고 도메인이 등록된 경우, 생성 시 SES가 게시하도록 사용 설정했다면 SES는 도메인에 대한 DNS 설정을 자동으로 업데이트합니다. 그렇지 않은 경우, 생성 후 버튼 클릭만으로 Route 53에 쉽게 게시할 수 있습니다. the section called "콘솔을 사용하여 ID 편집" 섹션을 참조하세요. DNS 설정이 자동으로 업데이트되지 않거나 SES를 사용하여 이메일을 보낼 때 사용하는 것과 동일한 계정에 있지 않은 CNAME 레코드를 Route 53에 추가하려는 경우 레코드 편집의 절차를 완료합니다.

   - DNS 공급자가 누군지 잘 모르는 경우 – 자세한 내용은 시스템 관리자에게 문의하십시오.

BYODKIM

BYODKIM을 사용하여 구성된 도메인을 확인하는 방법

1. 요약하면, BYODKIM을 사용하여 도메인을 만들거나 BYODKIM을 사용하여 기존 도메인을 구성했을 때 SES 콘솔의 고급 DKIM 설정 페이지에서 해당 필드에 프라이빗 키(자체 생성된 퍼블릭-프라이빗 키 페어의 프라이빗 키)와 선택기 이름 접두사를 추가했습니다. 이제 DNS 호스트 공급자에서 다음 레코드를 업데이트하여 확인 프로세스를 완료해야 합니다.
2. DNS 레코드 게시(Publish DNS records) 테이블에서 DNS 공급자에 게시(추가)되어 이름(Name) 열에 표시한 선택기 이름 레코드를 복사합니다. 또는 레코드 세트를 .csv로 다운로드(Download .csv record set)를 선택하여 컴퓨터에 저장할 수도 있습니다.

   다음 이미지는 DNS 공급자에 게시한 선택기 이름 레코드의 예를 보여줍니다.

3. 도메인 DNS 또는 웹 호스팅 공급자에 로그인한 후, 이전에 복사하거나 저장한 선택기 이름 레코드를 추가합니다. 공급자마다 DNS 레코드를 업데이트하는 절차가 다릅니다. 이 절차에 따라 DNS/호스팅 공급자 테이블을 참조하세요.

   > **Note**
   >
   > 소수의 DNS 공급자는 레코드 이름에 밑줄(_)을 포함하는 것을 허용하지 않습니다. 그러나 DKIM 레코드 이름에서 밑줄은 필수입니다. DNS 공급자가 레코드 이름에 밑줄을 입력하는 것을 허용하지 않는 경우 해당 공급자의 고객 지원 팀에 문의하세요.

4. 아직 설정하지 않은 경우, 자체 생성된 퍼블릭-프라이빗 키 페어의 퍼블릭 키를 도메인의 DNS 또는 웹 호스팅 공급자에 추가해야 합니다.

DNS 레코드 게시(Publish DNS records) 테이블의 값(Value) 열에 표시되는 퍼블릭 키 레코드만 표시됩니다. 'p=customerProvidedPublicKey'는 컴퓨터에 저장하거나 DNS 공급자에게 제공한 퍼블릭 키 값의 자리표시자입니다.

> **Note**
>
> DNS 공급자에 퍼블릭 키를 게시(추가)할 때는 다음과 같은 형식을 사용해야 합니다.
>
> - 생성된 퍼블릭 키의 첫 번째 줄(`-----BEGIN PUBLIC KEY-----`)과 마지막 줄(`-----END PUBLIC KEY-----`)을 삭제해야 합니다. 또한 생성된 퍼블릭 키에서 줄 바꿈을 제거해야 합니다. 결과 값은 공백이나 줄 바꿈이 없는 문자열입니다.
> - DNS 레코드 게시(Publish DNS records) 테이블의 값(Value) 열에 표시된 대로 p= 접두사를 포함해야 합니다.

4. DNS 설정에 대한 변경 사항이 배포되려면 최대 72시간이 소요될 수 있습니다. Amazon SES가 도메인 DNS 설정에서 이러한 DKIM 레코드를 모두 감지하면 확인 프로세스가 완료됩니다. 사용자 도메인의 DKIM 구성은 성공함으로 나타나고 자격 증명 상태는 확인됨으로 나타납니다.
5. 사용자 지정 MAIL FROM 도메인을 구성하고 확인하려면 사용자 지정 MAIL FROM 도메인 구성의 절차를 따릅니다.

다음 표에는 널리 사용되는 몇몇 DNS 공급자의 설명서에 대한 링크가 포함되어 있습니다. 이는 전체 목록이 아니며 목록에 포함되어 있다고 해서 해당 공급자를 승인하는 것은 아닙니다. 마찬가지로 DNS 공급자가 목록에 없는 경우에도 Amazon SES에서 도메인을 사용할 수 없다는 의미는 아닙니다.

DNS/호스팅 공급자	설명서 링크
GoDaddy	CNAME 레코드 추가(외부 링크)
DreamHost	사용자 지정 DNS 레코드를 추가하는 방법(외부 링크)
Cloudflare	Cloudflare에서 DNS 레코드 관리(외부 링크)
HostGator	HostGator/eNom을 통한 DNS 레코드 관리(외부 링크)

DNS/호스팅 공급자	설명서 링크
Namecheap	도메인에서 TXT/SPF/DKIM/DMARC 레코드를 추가하는 방법(외부 링크)
Names.co.uk	도메인 DNS 설정 변경(외부 링크)
Wix	Wix 계정에서 CNAME 레코드 추가 또는 업데이트(외부 링크)

도메인 확인과 관련된 문제 해결

상기 단계들을 완료했지만 72시간 후에 도메인이 확인되지 않는 경우에는 다음을 확인하십시오.

- DNS 레코드의 값을 올바른 필드에 입력했는지 확인합니다. 일부 DNS 공급자들은 이름/호스트 필드를 호스트 또는 호스트 이름으로 지칭합니다. 또한 일부 공급자는 레코드 값 필드를 가리키는 대상 또는 결과로 지칭합니다.
- 자신의 공급자가 DNS 레코드에 입력한 이름/호스트 값에 도메인 이름을 자동으로 추가하지 않았는지 확인합니다. 일부 공급자는 추가했다는 표시를 하지 않고 도메인 이름을 추가합니다. 공급자가 이름/호스트 값에 도메인 이름을 추가한 경우, 값 끝에서 도메인 이름을 제거합니다. 또한 DNS 레코드의 값 끝에 마침표를 추가할 수도 있습니다. 이 마침표는 그 도메인 이름이 완전한 이름임을 공급자에게 알려 줍니다.
- 각 DNS 레코드의 이름/호스트 값에는 밑줄 문자(_)가 필요합니다. 공급자가 DNS 레코드 이름에 밑줄을 포함하는 것을 허용하지 않는 경우, 해당 공급자의 고객 지원 팀에 문의해서 추가적인 도움을 받으세요.
- 도메인의 DNS 설정에 추가해야 하는 검증 레코드는 각각 다릅니다 AWS 리전. 도메인을 사용하여 여러에서 이메일을 보내려면 각 리전에 대해 별도의 도메인 자격 증명을 생성하고 확인해야 AWS 리전합니다.

이메일 주소 자격 증명 생성

다음 절차를 완료하면 Amazon SES 콘솔을 사용하여 이메일 주소 자격 증명을 생성할 수 있습니다.

이메일 주소 자격 증명을 생성하려면(콘솔)

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.

2. 탐색 창의 구성 아래에서 확인된 자격 증명(Verified identities)을 선택합니다.
3. Create identity(자격 증명 생성)를 선택합니다.
4. 자격 증명 세부 정보에서 생성하려는 자격 증명 유형으로 Email address(이메일 주소)를 선택합니다.
5. 이메일 주소에 사용하려는 이메일 주소를 입력합니다. 이메일 주소는 메일 수신이 가능하며 액세스 권한을 보유해야 합니다.
6. (선택) 기본 구성 세트를 할당하려면 확인란을 선택합니다.

   1. 기본 구성 세트에서 자격 증명에 할당할 기존 구성 세트를 선택합니다. 아직 구성 세트를 생성하지 않은 경우 구성 세트 단원을 참조하십시오.

   > **Note**
   >
   > Amazon SES는 전송 시 다른 세트가 지정되지 않은 경우에만 할당된 구성 세트로 기본 설정됩니다. 구성 세트가 지정된 경우 Amazon SES는 기본 세트 대신 지정된 세트를 적용합니다.

7. (선택) 태그 키와 키 값(선택)을 포함시켜 하나 이상의 태그를 도메인 자격 증명에 추가합니다.

   1. Add new tag(새 태그 추가)를 선택하고 키를 입력합니다. 또한, 태그에 값을 추가할 수 있습니다.
   2. 50개 이하의 추가 태그에 대해 반복하거나 Remove(제거)를 선택하여 태그를 제거합니다.

8. 이메일 주소 자격 증명을 생성하려면 자격 증명 생성(Create identity)을 선택합니다. 생성 후 5분 내에 확인 이메일을 수신해야 합니다. 다음 단계는 다음 섹션의 확인 절차에 따라 이메일 주소를 확인하는 것입니다.

   > **Note**
   >
   > 확인을 요청하는 이메일 주소로 전송된 메시지를 사용자 지정할 수 있습니다. 자세한 내용은 the section called “사용자 지정 확인 이메일 템플릿 사용” 단원을 참조하십시오.

이제 이메일 주소 자격 증명을 만들었으므로 확인 프로세스를 완료해야 합니다. the section called “이메일 주소 자격 증명 확인” 섹션으로 진행하세요.

이메일 주소 자격 증명 확인

이메일 주소 자격 증명을 만든 후에는 확인 프로세스를 완료해야 합니다.

이메일 주소 자격 증명을 생성하지 않은 경우 the section called “이메일 주소 자격 증명 생성” 섹션을 참조하세요.

이메일 주소 자격 증명을 확인하는 방법

1. 자격 증명을 생상하는 데 사용한 이메일 주소의 받은 편지함에서 no-reply-aws@amazon.com에서 보낸 이메일이 있는지 확인합니다.
2. 이메일을 열고 링크를 클릭하여 해당 이메일 주소에 대한 확인 프로세스를 완료합니다. 작업이 완료되면 자격 증명 상태가 확인됨으로 업데이트됩니다.

이메일 주소 확인 문제 해결

자격 증명을 생성한 후 5분 이내에 확인 이메일을 받지 못한 경우, 다음 문제 해결 단계를 수행해 보십시오.

- 이메일 주소를 정확하게 입력했는지 확인합니다.
- 확인하려는 이메일 주소가 이메일 수신이 가능한지 확인합니다. 다른 이메일 주소를 사용하여 확인하려는 주소에 테스트 이메일을 보내 이를 테스트할 수 있습니다.
- 정크 메일 폴더를 확인합니다.
- 확인 이메일의 링크는 24시간 후에 만료됩니다. 새 확인 이메일을 보내려면 자격 증명 세부 정보 페이지의 맨 위에 있는 Resend(재전송)을 선택하십시오.

자격 증명을 생성 및 확인하는 동시에 기본 구성 세트 할당

Amazon SES API v2에서 CreateEmailIdentity 작업을 사용하여 새 이메일 자격 증명을 생성하고 동시에 기본 구성 세트를 설정할 수 있습니다.

> **Note**
>
> 이 단원의 절차를 완료하기 전에 먼저 AWS CLI을(를) 설치하고 구성해야 합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

를 사용하여 기본 구성 세트를 설정하려면 AWS CLI

- CreateEmailIdentity 작업을 사용하려면 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 create-email-identity --email-identity ADDRESS-OR-DOMAIN --configuration-set-name CONFIG-SET
```

위의 명령에서 *ADDRESS-OR-DOMAIN*을 확인하려는 이메일 자격 증명으로 바꿉니다. *CONFIG-SET*를 자격 증명에 대한 기본 구성 세트로 설정하려는 구성 세트의 이름으로 바꿉니다.

명령이 성공적으로 실행되면 출력을 제공하지 않고 종료됩니다.

이메일 주소를 확인하는 방법

1. 확인하고 있는 이메일 주소의 받은 편지함을 선택합니다. "Amazon Web Services - *RegionName*의 이메일 주소 확인 요청"이라는 제목의 메시지가 표시됩니다. 여기서 *RegionName*은 이메일 주소를 확인 AWS 리전 하려고 시도한의 이름입니다.

   메시지를 연 다음 해당 메시지에서 링크를 클릭합니다.

   > **Note**
   > 확인 메시지의 링크는 메시지 전송 24시간 후에 만료됩니다. 확인 이메일을 받은 후 24시간이 지나면 1~5단계를 반복하여 유효한 링크가 있는 확인 이메일을 받을 수 있습니다.

2. Amazon SES 콘솔의 자격 증명 관리 아래에서 이메일 주소를 선택합니다. 이메일 주소 목록에서, 확인하고 있는 이메일 주소를 찾습니다. 이메일 주소가 확인된 경우 [Status] 열의 값이 "verified"입니다.

도메인을 확인하는 방법

위 명령줄 절차에서 `--email-identity` 파라미터의 도메인 이름을 입력한 경우 자세한 내용은 도메인 자격 증명 확인 섹션을 참조하세요.

사용자 지정 확인 이메일 템플릿 사용

이메일 주소 확인을 시도하면 Amazon SES가 다음 이미지에 표시된 예와 비슷한 해당 주소로 이메일을 전송합니다.

Dear Amazon Web Services Customer,

We have received a request to authorize this email address for use with Amazon SES and Amazon Pinpoint in region US West (Oregon). If you requested this verification, please go to the following URL to confirm that you are authorized to use this email address:

https://email-verification.us-west-2.amazonaws.com/?AWSAccessKeyId=AKIADQKE4EXAMPLE&Context=10987654321&Identity.IdentityName=recipient%40example.com&Identity.IdentityType=EmailAddress&Namespace=Bacon&Operation=ConfirmVerification&Signature=TJDufFhYYK1fSHCSBq4cjbodBQq%2FnyyZgzjqZ%2BXsDYEXAMPLE&SignatureMethod=HmacSHA256&SignatureVersion=2&Timestamp=2017-12-06T19%3A53%3A12.311Z

Your request will not be processed unless you confirm the address using this URL. This link expires 24 hours after your original verification request.

If you did NOT request to verify this email address, do not click on the link. Please note that many times, the situation isn't a phishing attempt, but either a misunderstanding of how to use our service, or someone setting up email-sending capabilities on your behalf as part of a legitimate service, but without having fully communicated the procedure first. If you are still concerned, please forward this notification to aws-email-domain-verification@amazon.com and let us know in the forward that you did not request the verification.

To learn more about sending email from Amazon Web Services, please refer to the Amazon SES Developer Guide at http://docs.aws.amazon.com/ses/latest/DeveloperGuide/Welcome.html and Amazon Pinpoint Developer Guide at http://docs.aws.amazon.com/pinpoint/latest/userguide/welcome.html.

Sincerely,

The Amazon Web Services Team.

많은 Amazon SES 고객은 자체 고객을 대신하여 Amazon SES를 통해 이메일을 보내는 애플리케이션(예: 이메일 마케팅 제품군 또는 티켓 시스템)을 빌드합니다. 이러한 애플리케이션의 최종 사용자의 경우 이메일 확인 프로세스가 혼동스러울 수 있습니다. 확인 이메일은 애플리케이션의 브랜드가 아니라 Amazon SES 브랜드를 사용하는데 그 최종 사용자가 Amazon SES를 사용하도록 직접 등록한 적이 없습니다.

Amazon SES 사용 사례에서 고객이 이메일 주소를 Amazon SES에 사용할 수 있는지 확인해야 하는 경우 사용자 지정 확인 이메일을 생성할 수 있습니다. 이 사용자 지정 이메일은 고객의 혼란을 줄이고 고객이 등록 프로세스를 완료하는 비율을 높이는 데 도움이 될 수 있습니다.

> **Note**
>
> 이 기능을 사용하려면 Amazon SES 계정이 샌드박스에서 나가야 합니다. 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하세요.

이 섹션의 주제:

- 사용자 지정 확인 이메일 템플릿 생성
- 사용자 지정 확인 이메일 템플릿 편집
- 사용자 지정 템플릿을 사용하여 확인 이메일 전송
- 사용자 지정 확인 이메일 FAQ

사용자 지정 확인 이메일 템플릿 생성

사용자 지정 확인 이메일을 생성하려면 `CreateCustomVerificationEmailTemplate` API 작업을 사용합니다. 이 작업은 다음 입력을 사용합니다.

속성	설명
`TemplateName`	템플릿의 이름입니다. 지정하는 이름은 고유해야 합니다.
`FromEmailAddress`	확인 이메일이 전송된 이메일 주소입니다. 지정하는 주소 또는 도메인은 Amazon SES 계정에 사용할 수 있는지 확인해야 합니다. **Note** `FromEmailAddress` 속성은 표시 이름("대화명"이라고도 함)을 지원하지 않습니다.
`TemplateSubject`	확인 이메일의 제목 줄입니다.
`TemplateContent`	이메일의 본문입니다. 이메일 본문에는 특정 제한 사항이 있는 HTML이 포함될 수 있습니다. 자세한 내용은 사용자 지정 확인 이메일 FAQ 단원을 참조하십시오.
`SuccessRedirectionURL`	이메일 주소가 성공적으로 확인된 경우 사용자에게 전송되는 URL입니다.
`FailureRedirectionURL`	이메일 주소가 성공적으로 확인되지 않은 경우 사용자에게 전송되는 URL입니다.

AWS SDKs 또는 AWS CLI 를 사용하여 `CreateCustomVerificationEmailTemplate` 작업으로 사용자 지정 확인 이메일 템플릿을 생성할 수 있습니다. AWS SDKs. https://aws.amazon.com/tools/#sdk 에 대한 자세한 내용은 명령줄 인터페이스를 AWS CLI참조하세요. AWS

다음 단원에는 AWS CLI를 사용하여 사용자 지정 확인 이메일을 생성하는 절차가 포함되어 있습니다. 이 절차에서는 AWS CLI를 설치하고 구성한 것으로 가정합니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

> **Note**
>
> 이 섹션의 절차를 완료하려면 AWS CLI의 버전 1.14.6 이상을 사용해야 합니다. 최상의 결과를 얻으려면 AWS CLI의 최신 버전으로 업그레이드하십시오. 업데이트에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 설치를 AWS Command Line Interface AWS CLI 참조하세요.

1. 텍스트 편집기에서 새로운 파일을 생성합니다. 다음 콘텐츠를 편집기에 붙여 넣습니다.

```
{
  "TemplateName": "SampleTemplate",
  "FromEmailAddress": "sender@example.com",
  "TemplateSubject": "Please confirm your email address",
  "TemplateContent": "<html>
                      <head></head>
                      <body style='font-family:sans-serif;'>
                        <h1 style='text-align:center'>Ready to start sending
                        email with ProductName?</h1>
                        <p>We here at Example Corp are happy to have you on
                          board! There's just one last step to complete before
                          you can start sending email. Just click the following
                          link to verify your email address. Once we confirm that
                          you're really you, we'll give you some additional
                          information to help you get started with ProductName.</p>
                      </body>
                      </html>",
  "SuccessRedirectionURL": "https://www.example.com/verifysuccess",
  "FailureRedirectionURL": "https://www.example.com/verifyfailure"
}
```

> **Important**
>
> 앞의 예를 이해하기 쉽도록 `TemplateContent` 속성에는 줄 바꿈이 포함되어 있습니다. 앞의 예를 텍스트 파일에 붙여 넣으면 진행하기 전에 줄 바꿈을 제거합니다.

`TemplateName`, `FromEmailAddress`, `TemplateSubject`, `TemplateContent`, `SuccessRedirectionURL` 및 `FailureRedirectionURL` 값을 사용자의 값으로 바꿉니다.

> **Note**
>
> `FromEmailAddress`매개 변수에 지정한 이메일 주소는 확인해야 하거나 확인된 도메인의 주소여야 합니다. 자세한 내용은 Amazon SES에서 확인된 자격 증명 단원을 참조하세요.

작업을 마치면 파일 이름을 `customverificationemail.json`(으)로 저장합니다.

2. 명령줄에 다음 명령을 입력하여 사용자 지정 확인 이메일 템플릿을 생성합니다.

```
aws sesv2 create-custom-verification-email-template --cli-input-json file:// customverificationemail.json
```

3. (선택 사항) 다음 명령을 입력하여 템플릿이 생성되었는지 확인할 수 있습니다.

```
aws sesv2 list-custom-verification-email-templates
```

사용자 지정 확인 이메일 템플릿 편집

`UpdateCustomVerificationEmailTemplate` 작업을 사용하여 사용자 지정 확인 이메일 템플릿을 편집할 수 있습니다. 이 작업은 `CreateCustomVerificationEmailTemplate` 작업과 같은 입력(즉 `TemplateName`, `FromEmailAddress`, `TemplateSubject`, `TemplateContent`, `SuccessRedirectionURL` 및 `FailureRedirectionURL` 속성)을 허용합니다. 하지만 `UpdateCustomVerificationEmailTemplate` 작업의 경우 이 속성 중 아무것도 필요하지 않습니다. 기존 사용자 지정 확인 이메일 템플릿의 이름과 같은 `TemplateName`의 값을 전달하면 지정하는 속성이 원래 템플릿에 있던 속성을 덮어씁니다.

사용자 지정 템플릿을 사용하여 확인 이메일 전송

최소 하나의 사용자 지정 확인 이메일 템플릿을 생성한 후 SendCustomVerificationEmail API 작업을 호출하여 고객에게 전송할 수 있습니다. AWS SDKs 또는를 사용하여 `SendCustomVerificationEmail` 작업을 호출할 수 있습니다 AWS CLI. `SendCustomVerificationEmail` 작업은 다음 입력을 사용합니다.

속성	설명
`EmailAddress`	확인 중인 이메일 주소입니다.
`TemplateName`	확인 중인 이메일 주소에 전송된 사용자 지정 확인 이메일 템플릿의 이름입니다.
`ConfigurationSetName`	(선택) 확인 이메일을 보낼 때 사용하도록 설정된 구성의 이름입니다.

예를 들어 고객이 애플리케이션의 양식을 사용하여 서비스에 등록한다고 가정해 보겠습니다. 고객이 양식을 작성하여 제출하면 애플리케이션이 `SendCustomVerificationEmail` 작업을 호출하여 고객의 이메일 주소와 사용하려는 템플릿의 이름을 전달합니다.

고객은 사용자가 생성한 사용자 지정 이메일 템플릿을 사용하는 이메일을 수신합니다. Amazon SES는 자동으로 수신자에 고유 링크와 간략한 면책 조항을 추가합니다. 다음 이미지에는 사용자 지정 확인 이메일 템플릿 생성에서 생성된 템플릿을 사용하는 샘플 확인 이메일이 표시되어 있습니다.

Ready to start sending email with ProductName?

We here at Example Corp are happy to have you on board! There's just one last step to complete before you can start sending email. Just click the following link to verify your email address. Once we confirm that you're really you, we'll give you some additional information to help you get started with ProductName.

https://email-verification.us-west-2.amazonaws.com/?AWSAccessKeyId=AKIADQKE4EXAMPLE&Context=10987654321&Identity.IdentityName=recipient%40example.com&Identity.IdentityType=EmailAddress&Namespace=Bacon&Operation=ConfirmVerification&Signature=TJDufFhYYK1fSHCSBq4cjbodBQq%2FnyyZgzjqZ%2BXsDYEXAMPLE&SignatureMethod=HmacSHA256&SignatureVersion=2&Timestamp=2017-12-06T19%3A53%3A12.311Z

If you did not request to verify this email address, please disregard this message. If you have any concerns, please forward this message to the following email address along with your questions or concerns.

사용자 지정 확인 이메일 FAQ

이 단원에는 사용자 지정 확인 이메일 템플릿 기능에 대한 FAQ 답변이 포함되어 있습니다.

Q1. 사용자 지정 확인 이메일 템플릿은 몇 개나 생성할 수 있나요?

Amazon SES 계정당 최대 50개의 사용자 지정 확인 이메일 템플릿을 생성할 수 있습니다.

Q2. 사용자 지정 확인 이메일은 수신자에게 어떻게 보이나요?

사용자 지정 확인 이메일에는 템플릿을 생성할 때 지정한 콘텐츠가 포함되어 있고 그 다음 수신자가 이메일 주소를 확인하기 위해 클릭해야 하는 링크가 있습니다.

Q3. 사용자 지정 확인 이메일을 미리 볼 수 있나요?

사용자 지정 확인 이메일을 미리 보려면 `SendCustomVerificationEmail` 작업을 사용하여 사용자가 소유하는 주소에 확인 이메일을 보냅니다. 확인 링크를 클릭하지 않으면 Amazon SES는 새 자격 증명을 생성하지 않습니다. 확인 링크를 클릭하면 선택적으로 `DeleteIdentity` 작업을 사용하여 새로 생성된 자격 증명을 삭제할 수 있습니다.

Q4. 사용자 지정 확인 이메일 템플릿에 이미지가 포함될 수 있나요?

Base64 인코딩을 사용하여 템플릿의 HTML에 이미지를 포함할 수 있습니다. 이런 방식으로 이미지를 포함하면 Amazon SES는 자동으로 첨부 문서로 변환합니다. 다음 명령 중 하나를 실행하여 명령줄에서 이미지를 인코딩할 수 있습니다.

Linux, macOS, or Unix

```
base64 -i imagefile.png | tr -d '\n' > output.txt
```

Windows

```
certutil -encodehex -f imagefile.png output.txt 0x40000001
```

*`imagefile.png`*를 인코딩할 파일의 이름으로 바꿉니다. 위 두 명령 모두 Base64 인코딩 이미지가 `output.txt`에 저장됩니다.

템플릿의 HTML에 `<img src="`*`base64EncodedImage`*`"/>`를 포함하여 Base64 인코딩 이미지를 포함할 수 있습니다.

이전 예에서 *`png`*를 인코딩 이미지의 파일 유형(예: jpg 또는 gif)으로 바꾸고 *`base64EncodedImage`*를 base64 인코딩 이미지(앞의 명령 중 하나 중 `output.txt`의 콘텐츠)로 바꿉니다.

Q5. 사용자 지정 확인 이메일 템플릿에 포함할 수 있는 콘텐츠에 제한이 있습니까?

사용자 지정 확인 이메일 템플릿은 크기가 10MB를 초과할 수 없습니다. 또한 침해를 방지할 수 있도록 HTML이 포함되는 사용자 지정 확인 이메일 템플릿은 다음 표에 나열된 태그와 속성만 사용할 수 있습니다.

HTML 태그	허용되는 속성
`abbr`	`class, id, style, title`

HTML 태그	허용되는 속성
acronym	class, id, style, title
address	class, id, style, title
area	class, id, style, title
b	class, id, style, title
bdo	class, id, style, title
big	class, id, style, title
blockquote	cite, class, id, style, title
body	class, id, style, title
br	class, id, style, title
button	class, id, style, title
caption	class, id, style, title
center	class, id, style, title
cite	class, id, style, title
code	class, id, style, title
col	class, id, span, style, title, width
colgroup	class, id, span, style, title, width
dd	class, id, style, title
del	class, id, style, title
dfn	class, id, style, title

HTML 태그	허용되는 속성
dir	class, id, style, title
div	class, id, style, title
dl	class, id, style, title
dt	class, id, style, title
em	class, id, style, title
fieldset	class, id, style, title
font	class, id, style, title
form	class, id, style, title
h1	class, id, style, title
h2	class, id, style, title
h3	class, id, style, title
h4	class, id, style, title
h5	class, id, style, title
h6	class, id, style, title
head	class, id, style, title
hr	class, id, style, title
html	class, id, style, title
i	class, id, style, title
img	align, alt, class, height, id, src, style, title, width
input	class, id, style, title

HTML 태그	허용되는 속성
ins	class, id, style, title
kbd	class, id, style, title
label	class, id, style, title
legend	class, id, style, title
li	class, id, style, title
map	class, id, style, title
menu	class, id, style, title
ol	class, id, start, style, title, type
optgroup	class, id, style, title
option	class, id, style, title
p	class, id, style, title
pre	class, id, style, title
q	cite, class, id, style, title
s	class, id, style, title
samp	class, id, style, title
select	class, id, style, title
small	class, id, style, title
span	class, id, style, title
strike	class, id, style, title
strong	class, id, style, title

HTML 태그	허용되는 속성
sub	class, id, style, title
sup	class, id, style, title
table	class, id, style, summary, title, width
tbody	class, id, style, title
td	abbr, axis, class, colspan, id, rowspan, style, title, width
textarea	class, id, style, title
tfoot	class, id, style, title
th	abbr, axis, class, colspan, id, rowspan, scope, style, title, width
thead	class, id, style, title
tr	class, id, style, title
tt	class, id, style, title
u	class, id, style, title
ul	class, id, style, title, type
var	class, id, style, title

> **Note**
>
> 사용자 지정 확인 이메일 템플릿에는 주석 태그를 포함할 수 없습니다.

Q6. 계정에 확인된 이메일 주소가 몇 개 존재할 수 있나요?

Amazon SES 계정은 각 AWS 리전에서 최대 10,000개의 확인된 자격 증명을 가질 수 있습니다. Amazon SES는 자격 증명에 확인된 도메인과 이메일 주소가 모두 포함됩니다.

Q7. Amazon SES 콘솔을 사용하여 사용자 지정 확인 이메일 템플릿을 생성할 수 있나요?

현재는 Amazon SES API를 사용해야만 사용자 지정 확인 이메일을 생성, 편집 및 삭제할 수 있습니다.

Q8. 고객이 사용자 지정 확인 이메일을 수신할 때 일어나는 열기 및 클릭 이벤트를 추적할 수 있나요?

사용자 지정 확인 이메일에는 열기 또는 클릭 추적이 포함될 수 없습니다.

Q9. 사용자 지정 확인 이메일에 사용자 지정 헤더가 포함될 수 있나요?

사용자 지정 확인 이메일에는 사용자 지정 헤더가 포함될 수 없습니다.

Q10. 사용자 지정 확인 이메일 하단에 표시되는 텍스트를 제거할 수 있나요?

다음 텍스트는 모든 사용자 지정 확인 이메일의 끝 부분에 자동으로 추가되고 제거할 수 없습니다.

이 이메일 주소의 확인을 요청하지 않은 경우 이 메시지를 무시하세요.

Q11. 사용자 지정 확인 이메일은 DKIM으로 서명하나요?

확인 이메일에 DKIM으로 서명하려면 확인 이메일 템플릿을 생성할 때 `FromEmailAddress` 속성에서 지정하는 이메일 주소가 DKIM 서명을 생성하도록 구성되어야 합니다. 도메인 및 이메일 주소에 대한 DKIM 설정에 대한 자세한 내용은 the section called “DKIM을 사용하여 이메일 인증”을(를) 참조하세요.

Q12. 사용자 지정 확인 이메일 템플릿 API 작업이 SDK 또는 CLI에 나타나지 않는 이유는 무엇입니까?

SDK 또는에서 사용자 지정 확인 이메일 템플릿 작업을 사용할 수 없는 경우 이전 버전의 SDK 또는 CLI를 사용하고 있을 AWS CLI수 있습니다. 사용자 지정 확인 이메일 템플릿 작업은 다음 SDK 및 CLI에서 사용할 수 있습니다.

- 의 버전 1.14.6 이상 AWS Command Line Interface
- 의 버전 3.3.205.0 이상 AWS SDK for .NET
- AWS SDK for C++ 버전 1.3.20170531.19 이상
- 의 버전 1.12.43 이상 AWS SDK for Go
- 의 버전 1.11.245 이상 AWS SDK for Java

- 의 버전 2.166.0 이상 AWS SDK for JavaScript
- 의 버전 3.45.2 이상 AWS SDK for PHP
- 의 버전 1.5.1 이상 AWS SDK for Python (Boto)
- AWS SDK for Ruby에서는 `aws-sdk-ses` Gem 버전 1.5.0 이상

Q13. 사용자 지정 확인 이메일을 보낼 때 **`ProductionAccessNotGranted`** 오류가 생기는 이유는 무엇입니까?

`ProductionAccessNotGranted` 오류는 계정이 아직 Amazon SES 샌드박스에 있음을 나타냅니다. 계정이 샌드박스에서 제거된 경우에만 사용자 지정 확인 이메일을 보낼 수 있습니다. 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하십시오.

Amazon SES에서 ID 관리

Amazon SES 콘솔에서 각에 대해 생성된 자격 증명을 보거나 AWS 리전, 자격 증명을 열어 세부 정보 설정을 보고 편집하거나, 기본 구성 세트를 연결하거나, 하나 이상의 자격 증명을 삭제할 수 있습니다.

> **Note**
>
> 이 섹션에 나와 있는 절차는 선택한 AWS 리전의 ID에만 적용됩니다. 둘 이상의 리전에서 생성된 자격 증명을 관리하려면 각각에 대해 절차를 반복합니다 AWS 리전.

내용

- SES 콘솔을 사용하여 자격 증명 보기
- SES 콘솔을 사용하여 ID 삭제
- SES 콘솔을 사용하여 ID 편집
- SES API를 사용하여 기본 구성 세트를 사용하도록 ID 편집
- SES API를 사용하여 ID에 이용되는 기본 구성 세트 검색
- SES API를 사용하여 ID에 이용되는 현재 기본 구성 세트 재정의

SES 콘솔을 사용하여 자격 증명 보기

Amazon SES 콘솔을 사용하여 확인된 또는 확인 보류 중인 도메인 및 이메일 주소 자격 증명을 볼 수 있습니다. 확인에 실패한 자격 증명도 볼 수 있습니다.

도메인 및 이메일 주소 자격 증명을 보려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 콘솔에서 리전 선택기를 사용하여 자격 증명 목록을 보 AWS 리전 려는를 선택합니다.

> **Note**
> 이 절차에서는 선택한 AWS 리전에 대한 자격 증명 목록만을 표시합니다.

3. 탐색 창의 구성 아래에서 Verified identities(확인된 자격 증명)를 선택합니다. 로드된 자격 증명(Loaded identities) 표에는 도메인 및 이메일 주소 자격 증명이 모두 표시됩니다. 상태 열에는 자격 증명이 확인되었는지, 확인 보류 중인지 또는 확인 프로세스에 실패했는지 여부가 표시됩니다. 가능한 모든 상태 값의 정의는 다음과 같습니다.
   - 확인됨 - SES에서의 전송을 위해 자격 증명이 확인되었습니다.
   - 실패 - SES가 해당 자격 증명을 확인할 수 없습니다. 도메인인 경우 SES가 72시간 이내에 DNS 레코드를 검색할 수 없음을 의미합니다. 이메일 주소인 경우 해당 이메일 주소로 전송된 확인 이메일이 24시간 이내에 확인되지 않았음을 의미합니다.
   - 대기 중 - SES가 자격 증명을 계속 확인하고 있습니다.
   - 일시적인 장애 - 이전에 확인된 도메인의 경우 SES는 확인에 필요한 DNS 레코드를 주기적으로 확인합니다. 특정 시점에서 SES가 레코드를 검색할 수 없는 경우 상태가 일시적인 장애로 변경됩니다. SES가 72시간 동안 DNS 레코드를 다시 확인하며, 레코드를 검색할 수 없는 경우 도메인 상태가 실패로 변경됩니다. 레코드를 검색할 수 있는 경우 도메인 상태가 확인됨으로 변경됩니다.
   - 시작 안 됨 - 아직 확인 프로세스를 시작하지 않았습니다.
4. 확인 상태별로 자격 증명을 정렬하려면 상태 열을 선택합니다.
5. 자격 증명의 세부 정보 페이지를 보려면 보려는 자격 증명을 선택합니다.

SES 콘솔을 사용하여 ID 삭제

Amazon SES 콘솔을 사용하여 선택한의 계정에서 도메인 또는 이메일 주소 자격 증명을 제거할 수 있습니다 AWS 리전.

도메인 또는 이메일 주소 ID를 제거하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 콘솔에서 리전 선택기를 사용하여 하나 이상의 자격 증명을 삭제할 AWS 리전 를 선택합니다.
3. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.

   로드된 ID(Loaded identities) 테이블에는 도메인 및 이메일 주소 ID 목록이 표시됩니다.
4. ID 열에서 삭제할 ID를 선택합니다. 삭제할 각 ID 옆의 확인란을 선택하여 여러 ID를 삭제할 수 있습니다.
5. 삭제를 선택합니다.

SES 콘솔을 사용하여 ID 편집

Amazon SES 콘솔을 사용하여 선택한의 계정에서 도메인 또는 이메일 주소 자격 증명을 편집할 수 있습니다 AWS 리전.

도메인 또는 이메일 주소 ID를 편집하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 콘솔에서 리전 선택기를 사용하여 하나 이상의 자격 증명을 편집할 AWS 리전 를 선택합니다.
3. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.

   로드된 ID(Loaded identities) 테이블에는 도메인 및 이메일 주소 ID 목록이 표시됩니다.
4. ID(Identity) 열에서 편집하려는 ID를 선택합니다(확인란을 선택하는 대신 ID 이름 직접 클릭).
5. ID의 세부 정보 페이지에서 편집하려는 범주가 포함된 탭을 선택합니다.
6. 선택한 탭의 범주형 컨테이너 중 하나에서 편집하려는 속성의 편집(Edit) 버튼을 클릭하고 변경한 다음 변경 내용 저장(Save changes)을 선택합니다.

   a. 인증(Authentication) 탭에서 속성을 편집하는 경우 도메인 ID가 Amazon Route 53에서 호스팅되고 DNS 레코드를 아직 게시하지 않았다면, 도메인키 식별 메일(DKIM)(DomainKeys Identified Mail(DKIM)) 또는 사용자 지정 MAIL FROM 도메인(Custom MAIL FROM domain) 컨테이너 중 하나 또는 둘 모두에 Route 53에 DNS 레코드 게시(Publish DNS records to Route 53) 버튼(편집(Edit) 버튼 옆)이 표시됩니다.

> **Note**
> 인증(Authentication) 탭은 계정에 확인된 도메인이 있거나, 계정에 확인된 도메인을 사용하는 이메일 주소가 있는 경우에만 표시됩니다.

b. Route53에 DNS 레코드 게시(Publish DNS records to Route53) 버튼을 통해 직접 DNS 레코드를 게시할 수 있습니다. 이 버튼을 클릭하면 확인 배너가 표시되고 Route53에 DNS 레코드 게시(Publish DNS records to Route53) 버튼이 해당 컨테이너에 대해 더 이상 표시되지 않습니다.

7. 편집하려는 ID의 각 속성에 대해 5단계와 6단계를 반복합니다.

SES API를 사용하여 기본 구성 세트를 사용하도록 ID 편집

PutEmailIdentityConfigurationSetAttributes 작업을 사용하여 기존 이메일 ID에 기본 구성 세트를 추가하거나 제거할 수 있습니다.

> **Note**
> 이 단원의 절차를 완료하기 전에 먼저 AWS CLI을(를) 설치하고 구성해야 합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

를 사용하여 기본 구성 세트를 추가하려면 AWS CLI

- PutEmailIdentityConfigurationSetAttributes 작업을 사용하려면 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 put-email-identity-configuration-set-attributes --email-identity ADDRESS-OR-DOMAIN --configuration-set-name CONFIG-SET
```

위의 명령에서 *ADDRESS-OR-DOMAIN*을 확인하려는 이메일 자격 증명으로 바꿉니다. *CONFIG-SET*를 ID의 기본 구성 세트로 설정하려는 구성 세트의 이름으로 바꿉니다.

명령이 성공적으로 실행되면 출력을 제공하지 않고 종료됩니다.

를 사용하여 기본 구성 세트를 제거하려면 AWS CLI

- PutEmailIdentityConfigurationSetAttributes 작업을 사용하려면 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 put-email-identity-configuration-set-attributes --email-identity ADDRESS-OR-DOMAIN
```

위의 명령에서 *ADDRESS-OR-DOMAIN*을 확인하려는 이메일 자격 증명으로 바꿉니다.

명령이 성공적으로 실행되면 출력을 제공하지 않고 종료됩니다.

SES API를 사용하여 ID에 이용되는 기본 구성 세트 검색

GetEmailIdentity 작업을 사용하여 이메일 ID에 대한 기본 구성 세트를 반환합니다(해당하는 경우).

> **Note**
> 이 단원의 절차를 완료하기 전에 먼저 AWS CLI을(를) 설치하고 구성해야 합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

를 사용하여 기본 구성 세트를 반환하려면 AWS CLI

- CreateEmailIdentity 작업을 사용하려면 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 get-email-identity --email-identity ADDRESS-OR-DOMAIN
```

위의 명령에서 *ADDRESS-OR-DOMAIN*을 기본 구성 세트(해당하는 경우)를 알고 싶은 이메일 ID로 바꿉니다.

명령이 성공적으로 실행되면 JSON 객체에 이메일 ID 세부 정보가 표시됩니다.

SES API를 사용하여 ID에 이용되는 현재 기본 구성 세트 재정의

SendEmail 작업을 사용하여 다른 구성 세트로 이메일을 보낼 수 있습니다. 이렇게 하면 지정한 구성 세트가 ID에 대한 기본 구성 세트를 재정의합니다.

> **Note**
> 이 단원의 절차를 완료하기 전에 먼저 AWS CLI을(를) 설치하고 구성해야 합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

를 사용하여 기본 구성 세트를 재정의하려면 AWS CLI

- SendEmail 작업을 사용하려면 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 send-email --destination file://DESTINATION-JSON --content file://CONTENT-
JSON --from-email-address ADDRESS-OR-DOMAIN --configuration-set-name CONFIG-SET
```

위의 명령에서 *DESTINATION-JSON*을 대상 JSON 파일로, *CONTENT-JSON*을 콘텐츠 JSON 파일로, *ADDRESS-OR-DOMAIN*을 FROM 이메일 주소로 바꾸고 *CONFIG-SET*를 ID에 대한 기본 구성 세트 대신 사용할 구성 세트의 이름으로 바꿉니다.

명령이 성공적으로 실행되면 `MessageId`를 출력합니다.

Amazon SES의 자격 증명 구성

Amazon Simple Email Service(Amazon SES)는 간이 전자 우편 전송 프로토콜(SMTP)을 사용하여 이메일을 보냅니다. SMTP는 자체적으로 인증을 제공하지 않으므로, 스패머가 실제 오리진을 숨기고 다른 사용자가 오리진인 것처럼 위장하여 이메일 메시지를 보낼 수 있습니다. 스패머는 이메일 헤더를 위조하고 IP 주소를 스푸핑하여 수신자가 수신하는 이메일 메시지가 진본이라고 믿도록 유도할 수 있습니다.

이메일 트래픽을 전달하는 ISP는 대부분 이메일이 합법적인지 여부를 평가하기 위한 조치를 취합니다. ISP가 취하는 이러한 조치 중 하나가 이메일 인증 여부를 확인하는 것입니다. 인증은 발신자가 이메일을 보내는 계정의 소유자가 본인임을 확인해야 하는 절차입니다. 경우에 따라 ISP는 인증되지 않은 이메일의 전달을 거부합니다. 최적의 발송률을 실현하려면 이메일을 인증할 것을 권장합니다.

다음 섹션에서는 ISP가 사용하는 두 가지 인증 메커니즘, 즉 Sender Policy Framework(SPF) 및 DomainKeys Identitified Mail(DKIM)을 설명하고 Amazon SES에서 이러한 표준을 사용하는 방법에 대해 설명합니다.

- 이메일 메시지를 발송 시스템까지 역추적하는 수단을 제공하는 SPF에 대해 자세히 알아보려면 Amazon SES에서 SPF를 사용하여 이메일 인증 단원을 참조하세요.
- 이메일 메시지를 서명하여 ISP에게 해당 메시지가 합법적이고 전송 과정에서 수정되지 않았음을 증명하는 표준인 DKIM에 대해 자세히 알아보려면 Amazon SES에서 DKIM을 사용하여 이메일 인증 단원을 참조하세요.
- SPF 및 DKIM을 기반으로 Domain-based Message Authentication, Reporting and Conformance(DMARC)를 준수하는 방법에 대해 알아보려면 Amazon SES의 DMARC 인증 프로토콜 준수 단원을 참조하세요.

이메일 인증 방법

Amazon Simple Email Service(Amazon SES)는 간이 전자 우편 전송 프로토콜(SMTP)을 사용하여 이메일을 보냅니다. SMTP는 자체적으로 인증을 제공하지 않으므로, 스패머가 실제 오리진을 숨기고 다른 사용자가 오리진인 것처럼 위장하여 이메일 메시지를 보낼 수 있습니다. 스패머는 이메일 헤더를 위조하고 IP 주소를 스푸핑하여 수신자가 수신하는 이메일 메시지가 진본이라고 믿도록 유도할 수 있습니다.

이메일 트래픽을 전달하는 ISP는 대부분 이메일이 합법적인지 여부를 평가하기 위한 조치를 취합니다. ISP가 취하는 이러한 조치 중 하나가 이메일 인증 여부를 확인하는 것입니다. 인증은 발신자가 이메일을 보내는 계정의 소유자가 본인임을 확인해야 하는 절차입니다. 경우에 따라 ISP는 인증되지 않은 이메일의 전달을 거부합니다. 최적의 발송률을 실현하려면 이메일을 인증할 것을 권장합니다.

내용

- Amazon SES에서 DKIM을 사용하여 이메일 인증
- Amazon SES에서 SPF를 사용하여 이메일 인증
- 사용자 지정 MAIL FROM 도메인 사용
- Amazon SES의 DMARC 인증 프로토콜 준수
- Amazon SES에서 BIMI 사용하기

Amazon SES에서 DKIM을 사용하여 이메일 인증

DomainKeys Identified Mail(DKIM)은 특정 도메인에서 전송했다고 주장하는 이메일이 해당 도메인의 소유자가 실제로 승인했는지 확인하기 위해 고안된 이메일 보안 표준입니다. 퍼블릭 키 암호화를 사용하여 프라이빗 키를 사용한 이메일에 서명합니다. 그러면 수신자 서버는 도메인의 DNS에 게시된 퍼블릭 키를 사용하여 전송 중에 이메일의 일부가 수정되지 않았는지 확인할 수 있습니다.

DKIM 서명은 선택 사항입니다. DKIM 준수 이메일 공급자에서 배달 가능성을 개선하기 위해 DKIM 서명을 사용하여 이메일을 서명할 수 있습니다. Amazon SES는 DKIM 서명을 사용하여 메시지에 서명할 수 있는 세 가지 옵션을 제공합니다.

- Easy DKIM: SES가 퍼블릭-프라이빗 키 페어를 생성하고 해당 자격 증명에서 보내는 모든 메시지에 DKIM 서명을 자동으로 추가합니다. Amazon SES에서 Easy DKIM 섹션을 참조하세요.
- DEED(Deterministic Easy DKIM): DKIM 서명 속성을 Easy DKIM을 사용하는 상위 자격 증명으로 자동으로 상속하는 복제본 자격 증명을 생성 AWS 리전 하여 여러에서 일관된 DKIM 서명을 유지할 수 있습니다. 섹션을 참조하세요Amazon SES에서 Deterministic Easy DKIM(DEED) 사용.

- BYODKIM(자체 DKIM 사용): 사용자가 자체 퍼블릭-프라이빗 키 페어를 제공하면 SES가 해당 자격 증명에서 보내는 모든 메시지에 DKIM 서명을 추가합니다. Amazon SES에 자체 DKIM 인증 토큰(BYODKIM) 제공 섹션을 참조하세요.
- DKIM 서명 수동 추가: 사용자가 `SendRawEmail` API를 사용하여 전송하는 이메일에 자체 DKIM 서명을 추가합니다. Amazon SES에서 수동 DKIM 서명 섹션을 참조하세요.

DKIM 서명 키 길이

현재 많은 DNS 공급자가 DKIM 2048비트 RSA 암호화를 완벽하게 지원하므로 Amazon SES는 또한 이메일 인증을 보다 안전하게 수행할 수 있도록 DKIM 2048을 지원함으로써 API 또는 콘솔에서 Easy DKIM을 구성할 때 이를 기본 키 길이로 사용합니다. 2048비트 키는 BYODKIM(자체 DKIM 사용)에서 설정 및 사용할 수도 있습니다. 이때 서명 키 길이가 1024비트 이상이어야 하며 2048비트를 넘지 않아야 합니다.

Easy DKIM으로 구성된 경우 보안 및 이메일 전송 가능성을 위해 1024 및 2048비트 키 길이를 사용하도록 선택할 수 있으며 여전히 2048비트를 지원하지 않는 DNS 공급자에 의해 발생하는 문제가 있는 경우 1024비트로 되돌릴 수 있는 유연성도 제공합니다. 새 ID를 만들 때 1024비트를 지정하지 않으면 기본적으로 DKIM 2048비트를 사용하여 만들어집니다.

전송 중인 이메일의 전송 가능성을 유지하기 위해 사용자의 DKIM 키 길이를 변경할 수 있는 빈도에는 제한이 있습니다. 제한 사항은 다음과 같습니다.

- 이미 구성된 것과 동일한 키 길이로 전환할 수 없습니다.
- 24시간 동안 두 번 이상 다른 키 길이로 전환할 수 없습니다(해당 기간 동안 첫 번째 다운그레이드가 1024비트로 이루어지지 않은 경우).

이메일이 전송 중일 때 DNS는 퍼블릭 키를 사용하여 이메일 인증합니다. 따라서 키를 너무 빠르게 또는 자주 변경하면 이전 키가 이미 무효화되어 DNS가 이메일을 DKIM 인증하지 못할 수 있으므로 이러한 제한 사항으로 인해 보호됩니다.

DKIM 고려 사항

DKIM을 사용하여 이메일을 인증할 때 다음 규칙이 적용됩니다.

- 'From' 주소에 사용하는 도메인에 대해서만 DKIM을 설정하면 됩니다. 'Return-Path' 또는 'Reply-to' 주소에 사용하는 도메인에 대해서는 DKIM을 사용하지 않아도 됩니다.

- Amazon SES는 여러 AWS 리전에서 사용할 수 있습니다. 둘 이상의 AWS 리전을 사용하여 이메일을 보내는 경우 모든 이메일이 DKIM 서명되도록 각 모든 리전에서 DKIM 설정 프로세스를 완료해야 합니다.
- DKIM 속성은 상위 도메인에서 상속되기 때문에 DKIM 인증을 사용하여 도메인을 확인할 때 다음을 수행합니다.
  - DKIM 인증은 해당 도메인의 모든 하위 도메인에도 적용됩니다.
    - 하위 도메인에 대한 DKIM 설정은 하위 도메인에서 DKIM 인증을 사용하고 싶지 않은 경우 상속을 사용 중지하여 상위 도메인 설정을 재정의할 수 있습니다. 상속 기능은 나중에 다시 사용 설정할 수 있습니다.
  - DKIM 인증은 주소에서 DKIM 확인 도메인을 참조하는 이메일 자격 증명에서 보낸 모든 이메일에도 적용됩니다.
    - 이메일 주소에 대한 DKIM 설정은 DKIM 인증 없이 메일을 보내려고 하는 경우 하위 도메인(해당하는 경우) 및 상위 도메인에 대한 설정을 상속 사용 중지하여 재정의할 수 있습니다. 상속 기능은 나중에 다시 사용 설정할 수 있습니다.

상속된 DKIM 서명 속성 이해

Easy DKIM 또는 BYODKIM이 사용되는지 여부에 관계없이 해당 도메인이 DKIM으로 구성된 경우 이메일 주소 자격 증명이 상위 도메인에서 DKIM 서명 속성을 상속한다는 점을 먼저 이해하는 것이 중요합니다. 따라서 이메일 주소 자격 증명에 대해 DKIM 서명을 사용 중지하거나 사용하면 다음과 같은 주요 요인에 따라 도메인의 DKIM 서명 속성이 재정의됩니다.

- 이메일 주소가 속하는 도메인에 대해 DKIM을 이미 설정한 경우 해당 이메일 주소 자격 증명에 대해 DKIM 서명을 사용할 필요가 없습니다.
  - 도메인에 대해 DKIM을 설정하면 Amazon SES가 상위 도메인에서 상속한 DKIM 속성을 통해 해당 도메인의 모든 주소에서 보내는 모든 이메일을 자동으로 인증합니다.
- 특정 이메일 주소 자격 증명의 DKIM 설정은 해당 이메일이 속한 상위 도메인 또는 하위 도메인(해당하는 경우)의 설정을 자동으로 재정의합니다.

이메일 주소 자격 증명의 DKIM 서명 속성은 상위 도메인에서 상속되므로 이러한 속성을 재정의하려는 경우 아래 표에서 설명하는 대로 재정의의 계층적 규칙을 고려해야 합니다.

상위 도메인에 DKIM 서명이 사용되지 않음	상위 도메인에 DKIM 서명이 사용됨
이메일 주소 자격 증명에 대해 DKIM 서명을 사용할 수 없음	이메일 주소 자격 증명에 대해 DKIM 서명을 사용 중지할 수 있음
	이메일 주소 자격 증명에 대해 DKIM 서명을 다시 사용할 수 있음

일반적으로 DKIM 서명을 사용 중지하는 것은 권장되지 않으며 보낸 메일이 정크 또는 스팸 폴더로 이동하거나 도메인이 스푸핑될 위험이 높아집니다.

그러나 DKIM 서명을 영구적으로 또는 일시적으로 사용 중지하거나 나중에 다시 사용해야 할 수 있는 특정 사용 사례 또는 예외적인 비즈니스 결정에 대해 이메일 주소 자격 증명에서 도메인이 상속한 DKIM 서명 속성을 재정의하는 기능이 있습니다. the section called “이메일 주소의 DKIM 서명 재정의”을(를) 참조하세요.

Amazon SES에서 Easy DKIM

도메인 자격 증명에 대해 Easy DKIM을 설정하면 Amazon SES는 사용자가 해당 자격 증명에서 보내는 모든 이메일에 2048비트 DKIM 키를 자동으로 추가합니다. Amazon SES 콘솔을 사용하거나 API를 사용하여 Easy DKIM을 구성할 수 있습니다.

> **Note**
>
> Easy DKIM을 설정하려면 도메인의 DNS 설정을 수정해야 합니다. Route 53을 DNS 공급자로 사용하는 경우 Amazon SES가 해당 레코드를 자동으로 생성할 수 있습니다. 다른 DNS 공급자를 사용하는 경우 해당 공급자의 설명서에서 도메인의 DNS 설정 변경에 대해 자세히 알아보세요.

> **Warning**
>
> 현재 BYODKIM을 사용하는데 Easy DKIM으로 전환하는 경우 Easy DKIM이 설정 중이고 DKIM 상태가 보류 중인 동안에는 Amazon SES가 BYODKIM을 사용하여 이메일에 서명하지 않습니다. API 또는 콘솔을 통해 Easy DKIM을 사용 설정하도록 호출하는 시점부터 SES가 DNS 구성을 확인할 수 있는 시점까지는 SES에서 DKIM 서명 없이 이메일이 전송될 수 있습니다. 따라서 중간 단계를 사용하여 한 DKIM 서명 방법에서 다른 DKIM 서명 방법으로 마이그레

이션하거나(예: BYODKIM이 사용 설정된 도메인의 하위 도메인을 사용한 다음 Easy DKIM 확인이 통과되면 삭제) 애플리케이션의 가동 중지 시간 동안(있는 경우) 이 작업을 수행하는 것이 좋습니다.

검증된 도메인 자격 증명에 대해 Easy DKIM 설정

이 섹션의 절차는 이미 만든 도메인 자격 증명에 Easy DKIM을 구성하는 데 필요한 단계를 보여주기 위해 간소화되었습니다. 도메인 자격 증명을 아직 생성하지 않았거나 기본 구성 집합, 사용자 지정 MAIL FROM 도메인 및 태그 사용과 같이 도메인 자격 증명을 사용자 지정하는 데 사용할 수 있는 모든 옵션을 보려면 the section called “도메인 자격 증명 생성” 단원을 참조하세요.

Easy DKIM 도메인 자격 증명을 생성하는 과정의 일부는 Amazon SES 기본값 2048비트를 허용하거나 1024비트를 선택하여 기본값을 무시하도록 선택할 수 있는 DKIM 기반 확인을 구성하는 것입니다. DKIM 서명 키 길이 및 키 변경 방법에 대해 자세히 알아보려면 the section called “DKIM 서명 키 길이”을 참조하십시오.

도메인에 대해 Easy DKIM을 설정하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 자격 증명을 선택합니다.
3. 자격 증명 목록에서 자격 증명 유형이 도메인인 자격 증명을 선택합니다.

   > **Note**
   > 도메인을 생성하거나 확인해야 하는 경우 도메인 자격 증명 생성 단원을 참조하십시오.

4. DKIM(DomainKeys Identified Mail) 컨테이너의 인증 탭 아래에서 Edit(편집)를 선택합니다.
5. 고급 DKIM 설정 컨테이너에서 자격 증명 유형 필드의 Easy DKIM 버튼을 선택합니다.
6. DKIM 서명 키 길이 필드에서 RSA_2048_BIT 또는 RSA_1024_BIT를 선택합니다.
7. DKIM 서명 필드에서 Enabled(활성화됨) 상자를 확인합니다.
8. Save changes(변경 사항 저장)를 선택합니다.
9. Easy DKIM을 사용하여 도메인 자격 증명을 구성했으므로 DNS 공급자로 확인 프로세스를 완료해야 합니다. the section called “도메인 자격 증명 확인” 섹션으로 진행하여 Easy DKIM에 대한 DNS 인증 절차를 따릅니다.

자격 증명에 대한 Easy DKIM 서명 키 길이 변경

이 섹션의 절차에서는 서명 알고리즘에 필요한 Easy DKIM 비트를 쉽게 변경하는 방법을 보여줍니다. 보안 강화를 위해 2048비트의 서명 길이가 항상 선호되지만 DKIM 1024만 지원하는 DNS 공급자를 사용해야 하는 경우와 같이 1024비트 길이를 사용해야 할 경우가 있습니다.

전송 중인 이메일의 전송 가능성을 유지하기 위해 DKIM 키 길이를 변경하거나 되돌릴 수 있는 빈도에 제한이 있습니다.

이메일이 전송 중일 때 DNS는 퍼블릭 키를 사용하여 이메일 인증합니다. 따라서 키를 너무 빠르게 또는 자주 변경하면 이전 키가 이미 무효화되어 DNS가 이메일을 DKIM 인증하지 못할 수 있으므로 다음과 같은 제한 사항으로 인해 보호됩니다.

- 이미 구성된 것과 동일한 키 길이로 전환할 수 없습니다.
- 24시간 동안 두 번 이상 다른 키 길이로 전환할 수 없습니다(해당 기간 동안 첫 번째 다운그레이드가 1024비트로 이루어지지 않은 경우).

다음 절차를 사용하여 키 길이를 변경할 때 이러한 제한 사항 중 하나를 위반하면 콘솔은 입력한 입력이 유효하지 않습니다(the input you provided is invalid)라는 오류 메시지와 유효하지 않은 이유를 표시합니다.

DKIM 서명 키 길이 비트를 변경하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 자격 증명)를 선택합니다.
3. 자격 증명 목록에서 DKIM 서명 키 길이를 변경하려는 자격 증명을 선택합니다.
4. DomainKeys Identified Mail(DKIM) 컨테이너의 인증 탭 아래에서 Edit(편집)를 선택합니다.
5. 고급 DKIM 설정(Advanced DKIM settings) 컨테이너의 DKIM 서명 키 길이(DKIM signing key length) 필드에서 RSA_2048_BIT 또는RSA_1024_BIT 중 하나를 선택합니다.
6. 변경 사항 저장을 선택합니다.

Amazon SES에서 Deterministic Easy DKIM(DEED) 사용

DEED(Deterministic Easy DKIM)는 여러에서 DKIM 구성을 관리하기 위한 솔루션을 제공합니다 AWS 리전. DNS 관리를 간소화하고 일관된 DKIM 서명을 보장함으로써 DEED는 강력한 이메일 인증 관행을 유지하면서 다중 리전 이메일 전송 작업을 간소화하는 데 도움이 됩니다.

Deterministic Easy DKIM(DEED)이란 무엇입니까?

DEED(Deterministic Easy DKIM)는 Easy DKIM으로 구성된 상위 도메인을 AWS 리전 기반으로 모든에서 일관된 DKIM 토큰을 생성하는 기능입니다. ??? 이를 통해 현재 Easy DKIM으로 구성된 상위 자격 증명과 동일한 DKIM 서명 구성을 AWS 리전 자동으로 상속하고 유지하는 다른에서 자격 증명을 복제할 수 있습니다. DEED를 사용하면 상위 자격 증명에 대해 DNS 레코드를 한 번만 게시하면 되며, 복제본 자격 증명은 동일한 DNS 레코드를 사용하여 도메인 소유권을 확인하고 DKIM 서명을 관리합니다.

DNS 관리를 간소화하고 일관된 DKIM 서명을 보장함으로써 DEED는 모범 이메일 인증 사례를 유지하면서 다중 리전 이메일 전송 작업을 간소화하는 데 도움이 됩니다.

DEED에 대해 말할 때 사용되는 용어:

- 상위 자격 증명 - 복제본 자격 증명에 대한 DKIM 구성의 소스 역할을 하는 Easy DKIM으로 구성된 확인된 자격 증명입니다.
- 복제본 자격 증명 - 동일한 DNS 설정 및 DKIM 서명 구성을 공유하는 상위 자격 증명의 사본입니다.
- 상위 리전 - 상위 자격 증명이 설정된 AWS 리전 입니다.
- 복제본 리전 - AWS 리전 복제본 자격 증명이 설정된 입니다.
- DEED 자격 증명 - 상위 자격 증명 또는 복제본 자격 증명으로 사용되는 모든 자격 증명입니다. (새 자격 증명이 생성되면 처음에는 일반(비 DEED) 자격 증명으로 처리됩니다. 그러나 복제본이 생성되면 자격 증명이 DEED 자격 증명으로 간주됩니다.)

DEED 사용의 주요 이점은 다음과 같습니다.

- 간소화된 DNS 관리 - 상위 자격 증명에 대해 DNS 레코드를 한 번만 게시합니다.
- 더 쉬운 다중 리전 작업 - 이메일 전송 작업을 새 리전으로 확장하는 프로세스를 간소화합니다.
- 관리 오버헤드 감소 - 상위 자격 증명에서 DKIM 구성을 중앙에서 관리합니다.

Deterministic Easy DKIM(DEED) 작동 방식

복제본 자격 증명을 생성하면 Amazon SES는 상위 자격 증명에서 복제본 자격 증명으로 DKIM 서명 키를 자동으로 복제합니다. 상위 자격 증명에 대한 후속 DKIM 키 교체 또는 키 길이 변경은 모든 복제본 자격 증명에 자동으로 전파됩니다.

이 프로세스에는 다음 워크플로가 포함됩니다.

1. Easy DKIM을 AWS 리전 사용하여에서 상위 자격 증명을 생성합니다.

2. 상위 자격 증명에 필요한 DNS 레코드를 설정합니다.
3. 다른에서 복제본 자격 증명을 생성하여 상위 자격 증명의 도메인 이름과 DKIM 서명 리전을 AWS 리전지정합니다.
4. Amazon SES는 상위의 DKIM 구성을 복제본 자격 증명에 자동으로 복제합니다.

중요 고려 사항:

- 이미 복제본인 자격 증명의 복제본은 생성할 수 없습니다.
- 상위 자격 증명에는 Easy DKIM이 활성화되어 있어야 합니다. BYODKIM 또는 수동으로 서명된 자격 증명의 복제본을 생성할 수 없습니다.
- 모든 복제본 자격 증명이 삭제될 때까지 상위 자격 증명을 삭제할 수 없습니다.

DEED를 사용하여 복제본 자격 증명 설정

이 섹션에서는 필요한 권한과 함께 DEED를 사용하여 복제본 자격 증명을 생성하고 확인하는 방법을 보여주는 예제를 제공합니다.

주제

- 복제본 자격 증명 생성
- 복제본 자격 증명 구성 확인
- DEED를 사용하는 데 필요한 권한

복제본 자격 증명 생성

복제본 자격 증명을 생성하려면:

1. 복제본 자격 증명을 생성하려는 AWS 리전 에서 https://console.aws.amazon.com/ses/ SES 콘솔을 엽니다.

   (SES 콘솔에서는 복제본 자격 증명을 글로벌 자격 증명이라고 합니다.)
2. 탐색 창에서 자격 증명을 선택합니다.
3. 자격 증명 생성(Create identity)을 선택합니다.
4. 자격 증명 유형에서 도메인을 선택하고 복제하여 상위로 사용할 Easy DKIM으로 구성된 기존 자격 증명의 도메인 이름을 입력합니다.
5. 고급 DKIM 설정을 확장하고 Deterministic Easy DKIM을 선택합니다.

6. 상위 리전 드롭다운 메뉴에서 글로벌(복제본) 자격 증명에 입력한 것과 동일한 이름의 Easy DKIM 서명 자격 증명이 있는 상위 리전을 선택합니다. (복제 리전은 기본적으로 SES 콘솔에 로그인한 리전으로 설정됩니다.)
7. DKIM 서명이 활성화되어 있는지 확인합니다.
8. (선택 사항) 도메인 자격 증명에 하나 이상의 태그를 추가합니다.
9. 구성을 검토하고 자격 증명 생성을 선택합니다.

사용 AWS CLI:

Easy DKIM으로 구성된 상위 자격 증명을 기반으로 복제본 자격 증명을 생성하려면 다음 예제와 같이 상위 도메인 이름, 복제본 자격 증명을 생성하려는 리전 및 상위 DKIM 서명 리전을 지정해야 합니다.

```
aws sesv2 create-email-identity --email-identity example.com --region us-west-2 --dkim-signing-attributes '{"DomainSigningAttributesOrigin": "AWS_SES_US_EAST_1"}'
```

이전 예제에서:

1. *example.com* 복제 중인 상위 도메인 자격 증명으로 바꿉니다.
2. *us-west-2*를 복제본 도메인 자격 증명이 생성될 리전으로 바꿉니다.
3. *AWS_SES_US_EAST_1*을 복제본 자격 증명에 복제될 Easy DKIM 서명 구성을 나타내는 상위의 DKIM 서명 리전으로 바꿉니다.

> **Note**
>
> AWS_SES_ 접두사는 DKIM이 Easy DKIM을 사용하여 상위 자격 증명에 대해 구성되었음을 나타내며 US_EAST_1가 생성된 AWS 리전 입니다.

복제본 자격 증명 구성 확인

복제본 자격 증명을 생성한 후 상위 자격 증명의 DKIM 서명 구성으로 올바르게 구성되었는지 확인할 수 있습니다.

복제본 자격 증명을 확인하려면:

1. 복제본 자격 증명을 생성한 AWS 리전 에서 https://console.aws.amazon.com/ses/ SES 콘솔을 엽니다.
2. 탐색 창에서 자격 증명을 선택하고 자격 증명 테이블에서 확인하려는 자격 증명을 선택합니다.

3. 인증 탭에서 DKIM 구성 필드는 상태를 나타내고 상위 리전 필드는 DEED를 사용하여 자격 증명의 DKIM 서명 구성에 사용 중인 리전을 나타냅니다.

사용 AWS CLI:

복제본의 도메인 이름과 리전을 지정하는 `get-email-identity` 명령을 사용합니다.

```
aws sesv2 get-email-identity --email-identity example.com --region us-west-2
```

응답에는 복제본 자격 증명이 상위 자격 증명의 DKIM 서명 구성으로 성공적으로 구성되었음을 나타내는 `SigningAttributesOrigin` 파라미터의 상위 리전 값이 포함됩니다.

```
{
  "DkimAttributes": {
    "SigningAttributesOrigin": "AWS_SES_US_EAST_1"
  }
}
```

DEED를 사용하는 데 필요한 권한

DEED를 사용하려면 다음이 필요합니다.

1. 복제본 리전에서 이메일 자격 증명을 생성하기 위한 표준 권한입니다.
2. 상위 리전에서 DKIM 서명 키를 복제할 수 있는 권한.

DKIM 복제를 위한 IAM 정책 예제

다음 정책은 상위 자격 증명에서 지정된 복제본 리전으로의 DKIM 서명 키 복제를 허용합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDKIMReplication",
      "Effect": "Allow",
      "Action": "ses:ReplicateEmailIdentityDKIMSigningKey",
      "Resource": "arn:aws:ses:us-east-1:123456789124:identity/example.com",
      "Condition": {
```

```
        "ForAllValues:StringEquals": {
           "ses:ReplicaRegion": ["us-west-2", "eu-west-1"]
        }
      }
    }
  ]
}
```

모범 사례

다음 모범 사례가 권장됩니다.

- 상위 및 복제본 리전 계획 - 복제본 리전에 사용되는 DKIM 구성의 출처가 되므로 선택한 상위 리전을 고려합니다.
- 일관된 IAM 정책 사용 - IAM 정책이 의도한 모든 리전에서 DKIM 복제를 허용하는지 확인합니다.
- 상위 자격 증명을 활성 상태로 유지 - 복제본 자격 증명은 상위 자격 증명의 DKIM 서명 구성을 상속합니다.이 종속성으로 인해 모든 복제본 자격 증명이 삭제될 때까지 상위 자격 증명을 삭제할 수 없습니다.

문제 해결

DEED에 문제가 발생하면 다음을 고려하세요.

- 확인 오류 - DKIM 복제에 필요한 권한이 있는지 확인합니다.
- 복제 지연 - 특히 새 복제본 자격 증명을 생성할 때 복제가 완료될 때까지 잠시 기다립니다.
- DNS 문제 - 상위 자격 증명에 대한 DNS 레코드가 올바르게 설정 및 전파되었는지 확인합니다.

Amazon SES에 자체 DKIM 인증 토큰(BYODKIM) 제공

Easy DKIM을 사용하는 대신 자체 퍼블릭-프라이빗 키 페어를 사용하여 DKIM 인증을 구성할 수 있습니다. 이 프로세스는 자체 DKIM 사용(BYODKIM)이라고 합니다.

BYODKIM을 사용하면 세 개의 개별 DNS 레코드를 게시해야 하는 Easy DKIM과 달리 단일 DNS 레코드를 사용하여 도메인에 대한 DKIM 인증을 구성할 수 있습니다. 또한 BYODKIM을 사용하면 도메인의 DKIM 키를 원하는 만큼 자주 교체할 수 있습니다.

이 섹션의 주제:

- 1단계: 키 페어 생성

- 2단계: DNS 공급자의 도메인 구성에 선택기 및 퍼블릭 키 추가
- 3단계: BYODKIM을 사용하도록 도메인 구성 및 확인

> **Warning**
>
> 현재 Easy DKIM을 사용하는데 BYODKIM으로 전환하는 경우 BYODKIM이 설정 중이고 DKIM 상태가 보류 중인 동안에는 Amazon SES가 Easy DKIM을 사용하여 이메일에 서명하지 않습니다. API 또는 콘솔을 통해 BYODKIM을 사용 설정하도록 호출하는 시점부터 SES가 DNS 구성을 확인할 수 있는 시점까지는 SES에서 DKIM 서명 없이 이메일이 전송될 수 있습니다. 따라서 중간 단계를 사용하여 한 DKIM 서명 방법에서 다른 DKIM 서명 방법으로 마이그레이션하거나 (예: Easy DKIM이 사용 설정된 도메인의 하위 도메인을 사용한 다음 BYODKIM 확인이 통과되면 삭제) 애플리케이션의 가동 중지 시간 동안(있는 경우) 이 작업을 수행하는 것이 좋습니다.

1단계: 키 페어 생성

BYODKIM 기능을 사용하려면 먼저 RSA 키 페어를 생성해야 합니다.

생성하는 프라이빗 키는 PKCS #1 또는 PKCS #8 형식이어야 하며 최소 1024비트 RSA 암호화와 최대 2048비트를 사용하고 base64(PEM) 인코딩을 사용하여 인코딩해야 합니다. DKIM 서명 키 길이 및 키 변경 방법에 대해 자세히 알아보려면 the section called “DKIM 서명 키 길이” 단원을 참조하십시오.

> **Note**
>
> 프라이빗 키가 최소 1024비트 RSA 암호화 및 최대 2048비트로 생성되고 base64(PEM) 인코딩을 사용하여 인코딩된 경우 서드 파티 애플리케이션 및 도구를 사용하여 RSA 키 페어를 생성할 수 있습니다.

다음 절차에서 대부분의 Linux, macOS 또는 Unix 운영 체제에 내장된 `openssl genrsa` 명령을 사용하여 키 페어를 생성하는 예제 코드는 자동으로 base64(PEM) 인코딩을 사용합니다.

Linux, macOS 또는 Unix 명령줄에서 키 페어를 생성하려면

1. 명령줄에 다음 명령을 입력하여 최대 1024비트에서 최대 2048비트를 지닌 *nnnn* 비트로 대체하는 프라이빗 키를 생성합니다.

```
openssl genrsa -f4 -out private.key nnnn
```

2. 명령줄에 다음 명령을 입력하여 퍼블릭 키를 생성합니다.

```
openssl rsa -in private.key -outform PEM -pubout -out public.key
```

2단계: DNS 공급자의 도메인 구성에 선택기 및 퍼블릭 키 추가

키 페어를 생성했으면 이제 도메인의 DNS 구성에 퍼블릭 키를 TXT 레코드로 추가해야 합니다.

도메인의 DNS 구성에 퍼블릭 키를 추가하려면

1. DNS 또는 호스팅 공급자의 관리 콘솔에 로그인합니다.
2. 도메인의 DNS 구성에 새로운 텍스트 레코드를 추가합니다. 레코드는 다음 형식을 사용해야 합니다.

명칭	유형	값
selector._domainkey.*example.com*	TXT	p=*yourPublicKey*

이전 예제에서 다음과 같이 변경합니다.

- *selector*를 키를 식별하는 고유 이름으로 바꿉니다.

> **Note**
> 소수의 DNS 공급자는 레코드 이름에 밑줄(_)을 포함하는 것을 허용하지 않습니다. 그러나 DKIM 레코드 이름에서 밑줄은 필수입니다. DNS 공급자가 레코드 이름에 밑줄을 입력하는 것을 허용하지 않는 경우 해당 공급자의 고객 지원 팀에 문의하세요.

- *example.com*을 도메인으로 바꿉니다.
- *yourPublicKey*를 이전에 생성한 퍼블릭 키로 바꾸고 위의 값(Value) 열에 표시된 대로 p= 접두사를 포함합니다.

> **Note**
> DNS 공급자에 퍼블릭 키를 게시(추가)할 때는 다음과 같은 형식을 사용해야 합니다.

- 생성된 퍼블릭 키의 첫 번째 줄(`-----BEGIN PUBLIC KEY-----`)과 마지막 줄(`-----END PUBLIC KEY-----`)을 삭제해야 합니다. 또한 생성된 퍼블릭 키에서 줄 바꿈을 제거해야 합니다. 결과 값은 공백이나 줄 바꿈이 없는 문자열입니다.
- 위 테이블의 값(Value) 열에 표시된 대로 p= 접두사를 포함해야 합니다.

공급자마다 DNS 레코드를 업데이트하는 절차가 다릅니다. 다음 표에는 널리 사용되는 몇몇 DNS 공급자의 설명서에 대한 링크가 포함되어 있습니다. 이는 전체 목록이 아니며 목록에 포함되어 있다고 해서 해당 공급자를 승인하는 것은 아닙니다. 마찬가지로 DNS 공급자가 목록에 없는 경우에도 Amazon SES에서 도메인을 사용할 수 없다는 의미는 아닙니다.

DNS/호스팅 공급자	설명서 링크
Amazon Route 53	Amazon Route 53 개발자 가이드의 레코드 편집
GoDaddy	Add a TXT record(외부 링크)
DreamHost	사용자 지정 DNS 레코드를 추가하는 방법(외부 링크)
Cloudflare	Cloudflare에서 DNS 레코드 관리(외부 링크)
HostGator	HostGator/eNom을 통한 DNS 레코드 관리(외부 링크)
Namecheap	도메인에서 TXT/SPF/DKIM/DMARC 레코드를 추가하는 방법(외부 링크)
Names.co.uk	도메인 DNS 설정 변경(외부 링크)
Wix	Adding or Updating TXT Records in Your Wix Account(외부 링크)

3단계: BYODKIM을 사용하도록 도메인 구성 및 확인

콘솔 또는 AWS CLI를 사용하여 새 도메인(즉, 현재 Amazon SES를 통해 이메일을 보내는 데 사용하지 않는 도메인)과 기존 도메인(즉, Amazon SES에서 사용하도록 이미 설정한 도메인) 모두에 대해

BYODKIM을 설정할 수 있습니다. 이 섹션의 AWS CLI 절차를 사용하기 전에 먼저를 설치하고 구성해야 합니다 AWS CLI. 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

옵션 1: BYODKIM을 사용하는 새 도메인 자격 증명 생성

이 단원에서는 BYODKIM을 사용하는 새 도메인 자격 증명을 생성하는 절차에 대해 설명합니다. 새 도메인 자격 증명은 이전에 Amazon SES를 사용하여 이메일을 보내도록 설정하지 않은 도메인입니다.

BYODKIM을 사용하도록 기존 도메인을 구성하려면 대신 옵션 2: 기존 도메인 자격 증명 구성 절차를 완료하세요.

콘솔에서 BYODKIM을 사용하여 자격 증명을 생성하려면

- 도메인 자격 증명 생성의 절차를 따르고 8단계에 도달하면 BYODKIM 관련 지침을 따릅니다.

에서 BYODKIM을 사용하여 자격 증명을 생성하려면 AWS CLI

새 도메인을 구성하려면 Amazon SES API의 `CreateEmailIdentity` 작업을 사용합니다.

1. 텍스트 편집기에서 다음 코드를 붙여 넣습니다.

```
{
    "EmailIdentity":"example.com",
    "DkimSigningAttributes":{
        "DomainSigningPrivateKey":"privateKey",
        "DomainSigningSelector":"selector"
    }
}
```

이전 예제에서 다음과 같이 변경합니다.

- *`example.com`*을 생성하려는 도메인으로 바꿉니다.
- *`privateKey`*를 해당 프라이빗 키로 바꿉니다.

> **Note**
> 생성된 프라이빗 키의 첫 번째 줄(`-----BEGIN PRIVATE KEY-----`)과 마지막 줄(`-----END PRIVATE KEY-----`)을 삭제해야 합니다. 또한 생성된 프라이빗 키에서 줄 바꿈을 제거해야 합니다. 결과 값은 공백이나 줄 바꿈이 없는 문자열입니다.

- *selector*를 도메인에 대한 DNS 구성에서 TXT 레코드를 생성할 때 지정한 고유 선택기로 바꿉니다.

작업을 마치면 파일 이름을 `create-identity.json`(으)로 저장합니다.

2. 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 create-email-identity --cli-input-json file://path/to/create-identity.json
```

앞의 명령에서 *path/to/create-identity.json*을 이전 단계에서 생성한 파일의 전체 경로로 바꿉니다.

옵션 2: 기존 도메인 자격 증명 구성

이 단원에서는 BYODKIM을 사용하도록 기존 도메인 자격 증명을 업데이트하는 절차에 대해 설명합니다. 기존 도메인 자격 증명은 Amazon SES를 사용하여 이메일을 보내도록 이미 설정한 도메인입니다.

콘솔에서 BYODKIM을 사용하여 도메인 ID를 업데이트하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 자격 증명)를 선택합니다.
3. 자격 증명 목록에서 자격 증명 유형이 도메인인 자격 증명을 선택합니다.

> **Note**
> 도메인을 생성하거나 확인해야 하는 경우 도메인 자격 증명 생성 단원을 참조하십시오.

4. 도메인키 식별 메일(DKIM)((DomainKeys Identified Mail(DKIM)) 창의 인증(Authentication) 탭 아래에서 편집(Edit)을 선택합니다.
5. 고급 DKIM 설정(Advanced DKIM settings) 창에서 자격 증명 유형(Identity type) 필드의 DKIM 인증 토큰 제공(Provide DKIM authentication token) 버튼을 선택합니다.
6. 프라이빗 키(Private key)에 이전에 생성한 프라이빗 키를 붙여 넣습니다.

> **Note**
>
> 생성된 프라이빗 키의 첫 번째 줄(`-----BEGIN PRIVATE KEY-----`)과 마지막 줄(`-----END PRIVATE KEY-----`)을 삭제해야 합니다. 또한 생성된 프라이빗 키에서 줄 바꿈을 제거해야 합니다. 결과 값은 공백이나 줄 바꿈이 없는 문자열입니다.

7. 선택기 이름의 경우, 도메인의 DNS 설정에서 지정한 선택기 이름을 입력합니다.
8. DKIM 서명 필드에서 Enabled(활성화됨) 상자를 확인합니다.
9. Save changes(변경 사항 저장)를 선택합니다.

에서 BYODKIM을 사용하여 도메인 자격 증명을 업데이트하려면 AWS CLI

기존 도메인을 구성하려면 Amazon SES API의 `PutEmailIdentityDkimSigningAttributes` 작업을 사용합니다.

1. 텍스트 편집기에서 다음 코드를 붙여 넣습니다.

```
{
    "SigningAttributes":{
        "DomainSigningPrivateKey":"privateKey",
        "DomainSigningSelector":"selector"
    },
    "SigningAttributesOrigin":"EXTERNAL"
}
```

이전 예제에서 다음과 같이 변경합니다.

- *`privateKey`*를 해당 프라이빗 키로 바꿉니다.

> **Note**
>
> 생성된 프라이빗 키의 첫 번째 줄(`-----BEGIN PRIVATE KEY-----`)과 마지막 줄(`-----END PRIVATE KEY-----`)을 삭제해야 합니다. 또한 생성된 프라이빗 키에서 줄 바꿈을 제거해야 합니다. 결과 값은 공백이나 줄 바꿈이 없는 문자열입니다.

- *`selector`*를 도메인에 대한 DNS 구성에서 TXT 레코드를 생성할 때 지정한 고유 선택기로 바꿉니다.

작업을 마치면 파일 이름을 `update-identity.json`(으)로 저장합니다.

2. 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 put-email-identity-dkim-signing-attributes --email-identity example.com
 --cli-input-json file://path/to/update-identity.json
```

위의 명령에서 다음과 같이 변경하세요.

- *`path/to/update-identity.json`*을 이전 단계에서 생성한 파일의 전체 경로로 바꿉니다.
- *`example.com`*을 업데이트하려는 도메인으로 바꿉니다.

BYODKIM을 사용하는 도메인의 DKIM 상태 확인

콘솔에서 도메인의 DKIM 상태를 확인하려면

BYODKIM을 사용하도록 도메인을 구성한 후 SES 콘솔을 사용하여 DKIM이 제대로 구성되었는지 확인할 수 있습니다.

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. 자격 증명 목록에서 DKIM 상태를 확인하려는 자격 증명을 선택합니다.
4. DNS 설정에 대한 변경 사항이 배포되려면 최대 72시간이 소요될 수 있습니다. Amazon SES가 도메인 DNS 설정에서 이러한 DKIM 레코드를 모두 감지하면 확인 프로세스가 완료됩니다. 모든 것이 올바르게 구성된 경우 도메인키 식별 메일(DKIM)(DomainKeys Identified Mail(DKIM)) 창에서 도메인의 DKIM 구성(DKIM configuration) 필드에 성공(Successful)이 표시되고 요약(Summary) 창에서 자격 증명 상태(Identity status) 필드에 확인됨(Verified)이 표시됩니다.

를 사용하여 도메인의 DKIM 상태를 확인하려면 AWS CLI

BYODKIM을 사용하도록 도메인을 구성한 후 GetEmailIdentity 작업을 사용하여 DKIM이 제대로 구성되었는지 확인할 수 있습니다.

- 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 get-email-identity --email-identity example.com
```

앞의 명령에서 *example.com*을 해당 도메인으로 바꿉니다.

이 명령은 다음 예제와 유사한 섹션이 포함된 JSON 객체를 반환합니다.

```
{
    ...
    "DkimAttributes": {
        "SigningAttributesOrigin": "EXTERNAL",
        "SigningEnabled": true,
        "Status": "SUCCESS",
        "Tokens": [ ]
    },
    ...
}
```

다음 사항이 모두 true일 경우 BYODKIM이 도메인에 대해 올바르게 구성된 것입니다.

- `SigningAttributesOrigin` 속성의 값이 `EXTERNAL`입니다.
- `SigningEnabled`의 값이 `true`입니다.
- `Status`의 값이 `SUCCESS`입니다.

Easy DKIM 및 BYODKIM 관리

자격 증명 인증을 위한 DKIM 설정은 웹 기반 Amazon SES 콘솔이나 Amazon SES API를 사용하여 Easy DKIM 또는 BYODKIM으로 관리할 수 있습니다. 이러한 방법 중 하나를 사용하여 자격 증명의 DKIM 레코드를 얻거나, 자격 증명에 대해 DKIM 서명을 사용하거나 사용 중지할 수 있습니다.

자격 증명의 DKIM 레코드 얻기

Amazon SES 콘솔을 사용하여 언제든 도메인 또는 이메일 주소의 DKIM 레코드를 얻을 수 있습니다.

콘솔을 사용하여 자격 증명의 DKIM 레코드 얻기

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. 자격 증명 목록에서 DKIM 레코드를 얻으려는 자격 증명을 선택합니다.
4. 자적 증명 세부 정보 페이지의 인증 탭에서 View DNS records(DNS 레코드 보기)를 확장합니다.

5. Easy DKIM을 사용한 경우 CNAME 레코드 3개를 복사하고, 이 섹션에 나타나는 BYODKIM을 사용한 경우 TXT 레코드를 복사합니다. 또는 Download .csv record set(레코드 세트를 .csv로 다운로드)를 선택하여 레코드 사본을 컴퓨터에 저장할 수도 있습니다.

   다음 이미지는 Easy DKIM과 연결된 CNAME 레코드를 표시하는 확장된 DNS 레코드 보기 섹션의 예제입니다.

Amazon SES API를 사용하여 자격 증명의 DKIM 레코드를 얻을 수도 있습니다. API와 상호 작용하는 일반적인 방법은 AWS CLI을(를) 사용하는 것입니다.

를 사용하여 자격 증명에 대한 DKIM 레코드를 가져오려면 AWS CLI

1. 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses get-identity-dkim-attributes --identities "example.com"
```

   앞의 예에서 *example.com*을 DKIM 레코드를 얻으려는 자격 증명으로 바꿉니다. 이메일 주소 또는 도메인을 지정할 수 있습니다.

2. 다음 예와 같이 이 명령의 출력에는 `DkimTokens` 섹션이 포함됩니다.

```
{
    "DkimAttributes": {
        "example.com": {
            "DkimEnabled": true,
            "DkimVerificationStatus": "Success",
```

```
            "DkimTokens": [
                "hirjd4exampled5477y22yd23ettobi",
                "v3rnz522czcl46quexamplek3efo5o6x",
                "y4examplexbhyhnsjcmtvzotfvqjmdqoj"
            ]
        }
    }
}
```

토큰을 사용하여 도메인의 DNS 설정에 추가하는 CNAME 레코드를 생성할 수 있습니다. CNAME 레코드를 생성하려면 다음 템플릿을 사용합니다.

```
token1._domainkey.example.com CNAME token1.dkim.amazonses.com
token2._domainkey.example.com CNAME token2.dkim.amazonses.com
token3._domainkey.example.com CNAME token3.dkim.amazonses.com
```

*token1*의 각 인스턴스를 `get-identity-dkim-attributes` 명령을 실행할 때 받은 목록의 첫 번째 토큰으로 바꾸고, *token2*의 모든 인스턴스를 목록의 두 번째 토큰으로 바꾸고, *token3*의 모든 인스턴스를 목록의 세 번째 토큰으로 바꿉니다.

예를 들어 앞의 예에 나온 토큰에 이 템플릿을 적용하면 다음 레코드가 생성됩니다.

```
hirjd4exampled5477y22yd23ettobi._domainkey.example.com CNAME
 hirjd4exampled5477y22yd23ettobi.dkim.amazonses.com
v3rnz522czcl46quexamplek3efo5o6x._domainkey.example.com CNAME
 v3rnz522czcl46quexamplek3efo5o6x.dkim.amazonses.com
y4examplexbhyhnsjcmtvzotfvqjmdqoj._domainkey.example.com CNAME
 y4examplexbhyhnsjcmtvzotfvqjmdqoj.dkim.amazonses.com
```

> **Note**
>
> 모든 사용자가 기본 SES DKIM 도메인을 AWS 리전 사용하는 것은 아닙니다. 리전에서 리전별 DKIM 도메인을 사용하는지 `dkim.amazonses.com`확인하려면에서 DKIM 도메인 테이블을 확인하세요AWS 일반 참조.

자격 증명에 대해 Easy DKIM 비활성화

Amazon SES 콘솔을 사용하여 자격 증명의 DKIM 인증을 빠르게 비활성화할 수 있습니다.

자격 증명에 대해 DKIM 사용 중지

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. 자격 증명 목록에서 DKIM을 사용 중지하려는 자격 증명을 선택합니다.
4. DKIM(DomainKeys Identified Mail) 컨테이너의 인증 탭 아래에서 Edit(편집)를 선택합니다.
5. 고급 DKIM 설정(Advanced DKIM settings)에서 DKIM 서명(DKIM signatures) 필드의 사용(Enabled) 상자를 선택 취소합니다.

Amazon SES API를 사용하여 자격 증명에 대해 DKIM을 사용 중지할 수도 있습니다. API와 상호 작용하는 일반적인 방법은 AWS CLI을(를) 사용하는 것입니다.

를 사용하여 자격 증명에 대해 DKIM을 비활성화하려면 AWS CLI

- 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses set-identity-dkim-enabled --identity example.com --no-dkim-enabled
```

  앞의 예에서 *example.com*을 DKIM을 사용 중지하려는 자격 증명으로 바꿉니다. 이메일 주소 또는 도메인을 지정할 수 있습니다.

자격 증명에 대해 Easy DKIM 활성화

이전에 자격 증명에 대해 DKIM을 사용 중지한 경우 Amazon SES 콘솔을 사용하여 다시 사용 설정할 수 있습니다.

자격 증명에 대해 DKIM 사용 설정

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. 자격 증명 목록에서 DKIM을 사용 설정하려는 자격 증명을 선택합니다.
4. DomainKeys Identified Mail(DKIM) 컨테이너의 인증 탭 아래에서 Edit(편집)를 선택합니다.
5. 고급 DKIM 설정에서 DKIM 서명 필드의 Enabled(활성화) 상자를 확인합니다.

Amazon SES API를 사용하여 자격 증명에 대해 DKIM을 사용 설정할 수도 있습니다. API와 상호 작용하는 일반적인 방법은 AWS CLI을(를) 사용하는 것입니다.

를 사용하여 자격 증명에 대해 DKIM을 활성화하려면 AWS CLI

- 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses set-identity-dkim-enabled --identity example.com --dkim-enabled
```

  앞의 예에서 *example.com*을 DKIM을 사용 설정하려는 자격 증명으로 바꿉니다. 이메일 주소 또는 도메인을 지정할 수 있습니다.

이메일 주소 자격 증명의 상속된 DKIM 서명 재정의

이 섹션에서는 Amazon SES로 이미 확인한 특정 이메일 주소 자격 증명에서 상위 도메인으로부터 상속된 DKIM 서명 속성을 재정의(사용 중지 또는 사용 설정)하는 방법을 확인합니다. DNS 설정이 도메인 수준에서 구성되어 있으므로 이미 소유한 도메인에 속한 이메일 주소 자격 증명에 대해서만 이 작업을 수행할 수 있습니다.

> **Important**
>
> 다음에 해당하는 이메일 주소 자격 증명에 대해 DKIM 서명을 사용/사용 중지할 수 없습니다.
>
> - 소유하지 않은 도메인의 이메일 주소 자격 증명 예를 들어 gmail.com 또는 hotmail.com 주소에 대해 DKIM 서명을 토글할 수 없습니다.
> - 소유하고 있지만 Amazon SES에서 아직 확인되지 않은 도메인의 이메일 주소 자격 증명
> - 소유하고 있지만 DKIM 서명을 사용하지 않은 도메인의 이메일 주소 자격 증명

이 섹션은 다음 주제를 포함합니다:

- 상속된 DKIM 서명 속성 이해
- 이메일 주소 자격 증명의 DKIM 서명 재정의(콘솔)
- 이메일 주소 자격 증명의 DKIM 서명 재정의(AWS CLI)

상속된 DKIM 서명 속성 이해

Easy DKIM 또는 BYODKIM이 사용되는지 여부에 관계없이 해당 도메인이 DKIM으로 구성된 경우 이메일 주소 자격 증명이 상위 도메인에서 DKIM 서명 속성을 상속한다는 점을 먼저 이해하는 것이 중요합니다. 따라서 이메일 주소 자격 증명에 대해 DKIM 서명을 사용 중지하거나 사용하면 다음과 같은 주요 요인에 따라 도메인의 DKIM 서명 속성이 재정의됩니다.

- 이메일 주소가 속하는 도메인에 대해 DKIM을 이미 설정한 경우 해당 이메일 주소 자격 증명에 대해 DKIM 서명을 사용할 필요가 없습니다.
  - 도메인에 대해 DKIM을 설정하면 Amazon SES가 상위 도메인에서 상속한 DKIM 속성을 통해 해당 도메인의 모든 주소에서 보내는 모든 이메일을 자동으로 인증합니다.
- 특정 이메일 주소 자격 증명의 DKIM 설정은 해당 이메일이 속한 상위 도메인 또는 하위 도메인(해당하는 경우)의 설정을 자동으로 재정의합니다.

이메일 주소 자격 증명의 DKIM 서명 속성은 상위 도메인에서 상속되므로 이러한 속성을 재정의하려는 경우 아래 표에서 설명하는 대로 재정의의 계층적 규칙을 고려해야 합니다.

상위 도메인에 DKIM 서명이 사용되지 않음	상위 도메인에 DKIM 서명이 사용됨
이메일 주소 자격 증명에 대해 DKIM 서명을 사용할 수 없음	이메일 주소 자격 증명에 대해 DKIM 서명을 사용 중지할 수 있음
	이메일 주소 자격 증명에 대해 DKIM 서명을 다시 사용할 수 있음

일반적으로 DKIM 서명을 사용 중지하는 것은 권장되지 않으며 보낸 메일이 정크 또는 스팸 폴더로 이동하거나 도메인이 스푸핑될 위험이 높아집니다.

그러나 DKIM 서명을 영구적으로 또는 일시적으로 사용 중지하거나 나중에 다시 사용해야 할 수 있는 특정 사용 사례 또는 예외적인 비즈니스 결정에 대해 이메일 주소 자격 증명에서 도메인이 상속한 DKIM 서명 속성을 재정의하는 기능이 있습니다.

이메일 주소 자격 증명의 DKIM 서명 재정의(콘솔)

다음의 SES 콘솔 절차에서는 Amazon SES로 이미 확인한 특정 이메일 주소 자격 증명에서 상위 도메인으로부터 상속된 DKIM 서명 속성을 재정의(사용 중지 또는 사용 설정)하는 방법을 보여줍니다.

콘솔을 통해 이메일 주소 자격 증명의 DKIM 서명 사용 중지/사용 설정

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. 자격 증명 목록에서 자격 증명 유형(Identity type)이 이메일 주소(Email address)이고 확인된 도메인 중 하나에 속하는 자격 증명을 선택합니다.
4. DKIM(DomainKeys Identified Mail) 컨테이너의 인증 탭 아래에서 Edit(편집)를 선택합니다.

> **Note**
>
> 이 인증 탭은 선택한 이메일 주소 ID가 SES에서 이미 확인된 도메인에 속하는 경우에만 표시됩니다. 도메인을 아직 확인하지 않은 경우 도메인 자격 증명 생성 섹션을 참조하세요.

5. 고급 DKIM 설정Advanced DKIM settings)의 DKIM 서명(DKIM signatures) 필드에서 사용 설정(Enabled) 확인란을 선택 취소하여 DKIM 서명을 비활성화하거나 해당 확인란을 선택하여 DKIM 서명을 다시 활성화합니다(이전에 재정의한 경우).
6. 변경 사항 저장을 선택합니다.

이메일 주소 자격 증명의 DKIM 서명 재정의(AWS CLI)

다음 예제에서는 AWS CLI SES API 명령 및 파라미터를 사용하여 SES로 이미 확인한 특정 이메일 주소 자격 증명의 상위 도메인에서 상속된 DKIM 서명 속성을 재정의(비활성화 또는 활성화)합니다.

AWS CLI를 통해 이메일 주소 자격 증명의 DKIM 서명 사용 중지/사용 하기

- example.com 도메인을 소유하고 있다고 가정할 때, 도메인의 이메일 주소 중 하나에 대해 DKIM 서명을 사용 중지하려면 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 put-email-identity-dkim-attributes --email-identity marketing@example.com
 --no-signing-enabled
```

  a. *marketing@example.com*을 DKIM 서명을 사용 중지하려는 이메일 주소 자격 증명으로 바꿉니다.

  b. `--no-signing-enabled`는 DKIM 서명을 사용 중지합니다. DKIM 서명을 다시 사용하려면 `--signing-enabled`를 사용합니다.

Amazon SES에서 수동 DKIM 서명

Easy DKIM을 사용하는 대신에 메시지에 DKIM 서명을 수동으로 추가한 후 Amazon SES.를 사용하여 해당 메시지를 보낼 수도 있습니다. 메시지에 수동으로 서명하기로 선택한 경우 먼저 DKIM 서명을 생성해야 합니다. 메시지와 DKIM 서명을 생성한 후 SendRawEmail API를 사용하여 보낼 수 있습니다.

이메일에 수동으로 서명하기로 결정한 경우 다음 요소를 고려하세요.

- Amazon SES.를 사용하여 보내는 모든 메시지에는 amazonses.com의 서명 도메인을 참조하는 DKIM 헤더가 포함됩니다(즉 `d=amazonses.com` 문자열 포함). 따라서, 메시지에 수동으로 서명하는 경우 메시지에 두 개의 DKIM 헤더(사용자 도메인 용으로 하나, Amazon SES가 amazonses.com에 대해 자동으로 생성된 것 하나)를 포함해야 합니다.
- Amazon SES는 메시지에 수동으로 추가하는 DKIM 서명의 유효성을 검사하지 않습니다. 메시지의 DKIM 서명에 오류가 있는 경우 이메일 공급자가 거부할 수 있습니다.
- 메시지에 서명할 때 1,024비트 이상의 비트 길이를 사용해야 합니다.
- Message-ID, Date, Return-Path, Bounces-To 필드에 서명하지 마세요.

> **Note**
>
> Amazon SES SMTP 인터페이스를 사용하여 이메일 클라이언트로 이메일을 보내는 경우 클라이언트가 메시지에 대한 DKIM 서명을 자동으로 수행할 수 있습니다. 일부 클라이언트는 이러한 필드 중 일부에 서명할 수 있습니다. 기본적으로 서명되는 필드에 대한 정보를 보려면 해당 이메일 클라이언트의 설명서를 참조하십시오.

Amazon SES에서 SPF를 사용하여 이메일 인증

Sender Policy Framework(SPF)는 이메일 스푸핑 방지를 위해 마련된 이메일 검증 표준입니다. 도메인 소유자는 SPF를 사용하여 도메인에서 이메일을 전송하는 데 허용된 서버를 이메일 공급자에게 알립니다. SPF는 RFC 7208에 정의되어 있습니다.

Amazon SES를 통해 보내는 메시지는 `amazonses.com`의 하위 도메인을 기존 MAIL FROM 도메인으로 자동 사용합니다. 기본 MAIL FROM 도메인은 이메일을 전송한 애플리케이션(이 경우 SES)과 일치하므로 SPF 인증은 이 메시지를 성공적으로 검증합니다. 따라서 SES에서는 SPF가 암시적으로 설정됩니다.

그러나 SES 기본 MAIL FROM 도메인을 사용하지 않고 소유한 도메인의 하위 도메인을 사용하려는 경우, SES에서는 이를 사용자 지정 MAIL FROM 도메인을 사용한다고 합니다. 이렇게 하려면 사용자

지정 MAIL FROM 도메인에 대한 자체 SPF 레코드를 게시해야 합니다. 또한 SES에서는 사용자 지정 MAIL FROM 도메인이 이메일 공급자가 보내는 반송 메일 및 수신 거부 알림을 받을 수 있도록 MX 레코드를 설정해야 합니다.

SPF 인증을 설정하는 방법 알아보기

SPF로 도메인을 구성하고 the section called "사용자 지정 MAIL FROM 도메인 사용"에 MX 및 SPF(유형 TXT) 레코드를 게시하는 방법에 대한 지침이 제공됩니다.

사용자 지정 MAIL FROM 도메인 사용

이메일을 전송하는 경우 해당 출처를 나타내는 두 개의 주소가 있습니다. 즉, 메시지 수신자에게 표시되는 From 주소와 메시지가 발생한 위치를 나타내는 MAIL FROM 주소입니다. MAIL FROM 주소를 envelope sender, envelope from, bounce address 또는 Return Path 주소라고 부르기도 합니다. 메일 서버는 MAIL FROM 주소를 사용하여 반송 메시지 및 기타 오류 알림을 반환합니다. MAIL FROM 주소는 일반적으로 수신자가 메시지의 소스 코드를 표시할 경우에만 볼 수 있습니다.

자체(사용자 지정) 도메인을 지정하지 않으면 Amazon SES는 기본값으로 보내는 메시지에 대해 MAIL FROM 도메인을 설정합니다. 이 단원에서는 사용자 지정 MAIL FROM 도메인 설정의 이점에 대해 설명하고 설정 절차를 안내합니다.

사용자 지정 MAIL FROM 도메인을 사용하는 이유

Amazon SES를 통해 보내는 메시지는 `amazonses.com`의 하위 도메인을 기존 MAIL FROM 도메인으로 자동 사용합니다. 기본 MAIL FROM 도메인은 이메일을 전송한 애플리케이션(이 경우 SES)과 일치하므로 발신자 정책 프레임워크(SPF) 인증은 이 메시지를 성공적으로 검증합니다.

SES 기본 MAIL FROM 도메인을 사용하지 않고 소유한 도메인의 하위 도메인을 사용하려는 경우, SES에서는 이를 사용자 지정 MAIL FROM 도메인을 사용한다고 합니다. 이렇게 하려면 사용자 지정 MAIL FROM 도메인에 대한 자체 SPF 레코드를 게시해야 합니다. 또한 SES에서는 사용자의 도메인이 이메일 공급자가 보내는 반송 메일 및 수신 거부 알림을 받을 수 있도록 MX 레코드를 설정해야 합니다.

사용자 지정 MAIL FROM 도메인을 사용하면 SPF, DKIM 또는 둘 다를 유연하게 사용하여 DMARC(Domain-based Message Authentication, Reporting and Conformance) 검증을 수행할 수 있습니다. DMARC를 통해 발신자의 도메인은 해당 도메인에서 발송된 이메일이 하나 이상의 인증 시스템에 의해 보호되고 있음을 나타낼 수 있습니다. DMARC 검증을 획득하는 방법은 the section called "SPF를 통해 DMARC 준수" 및 the section called "DKIM을 통해 DMARC 준수" 등 두 가지입니다.

사용자 지정 MAIL FROM 도메인 선택

다음에서 MAIL FROM 도메인이라는 용어는 항상 소유한 도메인의 하위 도메인을 나타냅니다. 사용자 지정 MAIL FROM 도메인에 사용하는이 하위 도메인은 다른 용도로 사용해서는 안 되며 다음 요구 사항을 충족해야 합니다.

- MAIL FROM 도메인은 확인된 자격 증명(이메일 주소 또는 도메인)에 대한 상위 도메인의 하위 도메인이어야 합니다.
- MAIL FROM 도메인은 사용자가 이메일을 보내는 데 사용하는 하위 도메인이 아니어야 합니다.
- MAIL FROM 도메인은 사용자가 이메일을 수신하는 데 사용하는 하위 도메인이 아니어야 합니다.

사용자 지정 MAIL FROM 도메인에서 SPF 사용

Sender Policy Framework(SPF)는 이메일 스푸핑 방지를 위해 마련된 이메일 검증 표준입니다. SPF를 사용해 사용자 지정 MAIL FROM 도메인을 구성하여 사용자 지정 MAIL FROM 도메인에서 이메일을 보낼 수 있는 서버를 이메일 공급자에게 알릴 수 있습니다. SPF는 RFC 7208에 정의되어 있습니다.

SPF를 설정하려면 TXT 레코드를 사용자 지정 MAIL FROM 도메인의 DNS 구성에 게시해야 합니다. 이 레코드에는 사용자 지정 MAIL FROM 도메인을 사용하여 이메일을 보낼 수 있는 권한을 부여하는 서버의 목록이 포함되어 있습니다. 이메일 공급자가 사용자 지정 MAIL FROM 도메인으로부터 메시지를 받으면 이메일이 인증된 서버에서 전송되었는지 확인하기 위해 해당 도메인의 DNS 레코드를 검사합니다.

이 SPF 레코드를 DMARC를 준수하는 방법으로 사용하려면 발신 주소의 도메인이 MAIL FROM 도메인과 일치해야 합니다. the section called “SPF를 통해 DMARC 준수”을(를) 참조하세요.

다음 섹션 the section called “사용자 지정 MAIL FROM 도메인 구성”에서는 사용자 지정 MAIL FROM 도메인에 SPF를 설정하는 방법을 설명합니다.

사용자 지정 MAIL FROM 도메인 구성

사용자 지정 MAIL FROM 도메인을 설정하는 과정에서 도메인의 DNS 구성에 레코드를 추가해야 합니다. SES에서는 이메일 공급자가 보내는 반송 메일 및 수신 거부 알림을 도메인에서 받을 수 있게 MX 레코드를 게시하도록 요구합니다. 또한 Amazon SES가 도메인에서 이메일을 보낼 권한이 부여되었음을 증명하기 위해 SPF(TXT 형식) 레코드를 게시해야 합니다.

전체 도메인이나 하위 도메인은 물론, 개별 이메일 주소에도 사용자 지정 MAIL FROM 도메인을 설정할 수 있습니다. 다음 절차는 Amazon SES 콘솔을 사용하여 사용자 지정 MAIL FROM 도메인을 구성하는 방법을 보여줍니다. 또한 SetIdentityMailFromDomain API 작업을 사용하여 사용자 지정 MAIL FROM 도메인을 구성할 수도 있습니다.

확인된 도메인에 사용자 지정 MAIL FROM 도메인 설정

이 절차에서는 전체 도메인 또는 하위 도메인에 대해 사용자 지정 MAIL FROM 도메인을 구성하여 해당 도메인의 주소에서 전송된 모든 메시지가 이 사용자 지정 MAIL FROM 도메인을 사용하도록 하는 방법을 보여줍니다.

확인된 도메인이 지정된 사용자 지정 MAIL FROM 도메인을 사용하도록 구성하려면

1. https://console.aws.amazon.com/ses/에서 Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 창의 구성 아래에서 자격 증명을 선택합니다.
3. 자격 증명 목록에서 자격 증명 유형(Identity type)이 도메인(Domain)이고 상태(Status)가 확인됨(Verified)인 경우에 구성하려는 자격 증명을 선택합니다.
   - 상태(Status)가 확인되지 않음(Unverified)인 경우 DNS 공급자로 DKIM 도메인 자격 증명 확인의 절차를 완료하여 이메일 주소의 도메인을 확인합니다.
4. 사용자 지정 MAIL FROM 도메인 창의 화면 하단에서 편집을 선택합니다.
5. 일반 세부 정보(General details) 창에서 다음을 수행합니다.
   a. 사용자 지정 MAIL FROM 도메인 사용(Use a custom MAIL FROM domain) 확인란을 선택합니다.
   b. MAIL FROM 도메인에 MAIL FROM 도메인으로 사용할 하위 도메인을 입력합니다.
   c. MX 장애 발생 시 동작(Behavior on MX failure)에서 다음 옵션 중 하나를 선택합니다.
      - 기본 MAIL FROM 도메인 사용 – 사용자 지정 MAIL FROM 도메인의 MX 레코드가 올바르게 설정되지 않은 경우 Amazon SES가 `amazonses.com`의 하위 도메인을 사용합니다. 하위 도메인은 Amazon SES AWS 리전 를 사용하는에 따라 달라집니다.
      - 메시지 거부 – 사용자 지정 MAIL FROM 도메인의 MX 레코드가 올바르게 설정되지 않은 경우 Amazon SES는 `MailFromDomainNotVerified` 오류를 반환합니다. 이 도메인에서 보내려는 이메일이 자동으로 거부됩니다.
   d. 변경 사항 저장(Save changes)을 선택합니다. 이전 화면으로 돌아갑니다.
6. MX 및 SPF(TXT 형식) 레코드를 사용자 지정 MAIL FROM 도메인의 DNS 서버에 게시합니다.

   사용자 지정 MAIL FROM 도메인(Custom MAIL FROM domain) 창에서 이제 DNS 레코드 게시(Publish DNS records) 테이블에 도메인의 DNS 구성에 게시(추가)해야 하는 MX 및 SPF(TXT 형식) 레코드가 표시됩니다. 이 레코드는 다음 표에 표시된 형식을 사용합니다.

명칭	유형	값
subdomain .*domain*.*com*	MX	10 feedback-smtp.*region*.amazonses.com
subdomain .*domain*.*com*	TXT	"v=spf1 include:amazonses.com ~all"

이전 레코드에서

- *subdomain*.*domain*.*com*은 MAIL FROM 하위 도메인으로 채워집니다.
- *region*은 MAIL FROM 도메인을 확인 AWS 리전 하려는의 이름으로 채워집니다(예: `us-west-2`, `us-east-1`또는 등`eu-west-1`).
- MX 값과 함께 나열된 숫자 10은 메일 서버의 기본 설정 순서이며 DNS 공급자의 GUI에서 지정한 대로 별도의 값 필드에 입력해야 합니다.
- SPF의 TXT 레코드 값에는 일반적으로 따옴표가 포함되어야 하지만 일부 DNS 공급자는 필요하지 않습니다.

DNS 레코드 게시(Publish DNS records) 테이블에서 각 값 옆에 있는 복사 아이콘을 선택하여 MX 및 SPF(TXT 형식) 레코드를 복사하고 DNS 공급자 GUI의 해당 필드에 붙여 넣습니다. 또는 Download .csv record set(레코드 세트를 .csv로 다운로드)를 선택하여 레코드 사본을 컴퓨터에 저장할 수도 있습니다.

> **Important**
>
> - MX 및 SPF(유형 TXT) 레코드를 게시하는 특정 절차는 DNS 또는 호스팅 공급자에 따라 다릅니다. 도메인의 DNS 구성에 이러한 레코드를 추가하는 방법에 대한 자세한 내용은 공급자 설명서를 참조하거나 공급자에게 문의하십시오.
> - Amazon SES를 이용해 사용자 지정 MAIL FROM 도메인을 설정하려면 MAIL FROM 도메인의 DNS 서버에 정확히 하나의 MX 레코드를 게시해야 합니다. MAIL FROM 도메인에 MX 레코드가 여러 개인 경우 Amazon SES를 사용한 사용자 지정 MAIL FROM 설정이 실패합니다.

Route 53가 MAIL FROM 도메인에 대한 DNS 서비스를 제공하고 Route 53에 사용하는 것과 AWS Management Console 동일한 계정으로에 로그인한 경우 Route 53을 사용하여 레코드 게시를 선택합니다. DNS 레코드가 도메인의 DNS 구성에 자동으로 적용됩니다.

다른 DNS 공급자를 사용하는 경우, MAIL FROM 도메인의 DNS 서버에 DNS 레코드를 수동으로 게시해야 합니다. 도메인의 DNS 서버에 DNS 레코드를 추가하는 절차는 웹 호스팅 서비스 또는 DNS 공급자에 따라 다릅니다.

도메인의 DNS 레코드를 게시하는 절차는 이용하는 DNS 공급자에 따라 다릅니다. 다음 표에는 널리 사용되는 몇몇 DNS 공급자의 설명서에 대한 링크가 포함되어 있습니다. 이는 전체 목록이 아니며 목록에 포함되어 있다고 해서 해당 공급자를 승인하는 것은 아닙니다. 마찬가지로 DNS 공급자가 목록에 없는 경우에도 MAIL FROM 도메인 구성을 지원하지 않는다는 의미는 아닙니다.

DNS/호스팅 공급자 이름	설명서 링크
GoDaddy	• MX: MX 레코드 추가(외부 링크) • TXT: TXT 레코드 추가(외부 링크)
DreamHost	• MX: MX 레코드를 변경하는 방법(외부 링크) • TXT: 사용자 지정 DNS 레코드를 추가하는 방법(외부 링크)
Cloudflare	• MX: 메일 또는 MX 레코드를 추가하거나 편집하는 방법(외부 링크) • TXT: CloudFlare에서 DNS 레코드 관리(외부 링크)
HostGator	• MX: MX 레코드 설정(외부 링크) • TXT: HostGator/eNom을 통한 DNS 레코드 관리(외부 링크)
Namecheap	• MX: 메일 서비스에 필요한 MX 레코드를 설정하는 방법(외부 링크) • TXT: 도메인에서 TXT/SPF/DKIM/DMARC 레코드를 추가하는 방법(외부 링크)

DNS/호스팅 공급자 이름	설명서 링크
Names.co.uk	• MX: 도메인의 DNS 설정 변경(외부 링크) • TXT: 도메인 DNS 설정 변경(외부 링크)
Wix	• MX: Wix 계정에서 MX 레코드 추가 또는 업데이트(외부 링크) • TXT: Wix 계정에서 TXT 레코드 추가 또는 업데이트(외부 링크)

레코드가 있음을 Amazon SES가 감지하면 사용자 지정 MAIL FROM 도메인이 성공적으로 설정되었음을 알리는 이메일을 받습니다. DNS 공급자에 따라 Amazon SES가 MX 레코드를 감지하기까지 최대 72시간이 지연될 수도 있습니다.

확인된 이메일 주소에 사용자 지정 MAIL FROM 도메인 설정

또한 특정 이메일 주소에 사용자 지정 MAIL FROM 도메인을 설정할 수도 있습니다. 이메일 주소에 사용자 지정 MAIL FROM 도메인을 설정하려면 이메일 주소가 연결된 도메인의 DNS 레코드를 수정해야 합니다.

> **Note**
>
> 사용자는 소유하지 않은 도메인의 주소에 사용자 지정 MAIL FROM 도메인을 설정할 수 없습니다(예를 들어 사용자는 필요한 DNS를 해당 도메인에 추가할 수 없기 때문에 `gmail.com` 도메인의 주소에 사용자 지정 MAIL FROM 도메인을 생성할 수 없습니다).

확인된 이메일 주소가 지정된 MAIL FROM 도메인을 사용하도록 구성하려면,

1. https://console.aws.amazon.com/ses/에서 Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 창의 구성 아래에서 자격 증명을 선택합니다.
3. 자격 증명 목록에서 자격 증명 유형(Identity type)이 이메일 주소(Email address)이고 상태(Status)가 확인됨(Verified)인 구성하려는 자격 증명을 선택합니다.
   - 상태(Status)가 확인되지 않음(Unverified)인 경우 이메일 주소 자격 증명 확인의 절차를 완료하여 이메일 주소의 도메인을 확인합니다.

4. MAIL FROM 도메인(MAIL FROM Domain) 탭의 사용자 지정 MAIL FROM 도메인(Custom MAIL FROM domain) 창에서 편집(Edit)을 선택합니다.
5. 일반 세부 정보(General details) 창에서 다음을 수행합니다.
   a. 사용자 지정 MAIL FROM 도메인 사용(Use a custom MAIL FROM domain) 확인란을 선택합니다.
   b. MAIL FROM 도메인에 MAIL FROM 도메인으로 사용할 하위 도메인을 입력합니다.
   c. MX 장애 발생 시 동작(Behavior on MX failure)에서 다음 옵션 중 하나를 선택합니다.
      - 기본 MAIL FROM 도메인 사용 – 사용자 지정 MAIL FROM 도메인의 MX 레코드가 올바르게 설정되지 않은 경우 Amazon SES가 `amazonses.com`의 하위 도메인을 사용합니다. 하위 도메인은 Amazon SES AWS 리전 를 사용하는에 따라 달라집니다.
      - 메시지 거부 – 사용자 지정 MAIL FROM 도메인의 MX 레코드가 올바르게 설정되지 않은 경우 Amazon SES는 `MailFromDomainNotVerified` 오류를 반환합니다. 이 이메일 주소에서 보내려는 이메일이 자동으로 거부됩니다.
   d. 변경 사항 저장(Save changes)을 선택합니다. 이전 화면으로 돌아갑니다.
6. MX 및 SPF(TXT 형식) 레코드를 사용자 지정 MAIL FROM 도메인의 DNS 서버에 게시합니다.

   사용자 지정 MAIL FROM 도메인(Custom MAIL FROM domain) 창에서 이제 DNS 레코드 게시(Publish DNS records) 테이블에 도메인의 DNS 구성에 게시(추가)해야 하는 MX 및 SPF(TXT 형식) 레코드가 표시됩니다. 이 레코드는 다음 표에 표시된 형식을 사용합니다.

명칭	유형	값
subdomain .*domain*.*com*	MX	10 feedback-smtp.*region*.amazonses.com
subdomain .*domain*.*com*	TXT	"v=spf1 include:amazonses.com ~all"

   이전 레코드에서

   - *subdomain*.*domain*.*com*은 MAIL FROM 하위 도메인으로 채워집니다.
   - *region*은 MAIL FROM 도메인을 확인 AWS 리전 하려는의 이름으로 채워집니다(예: `us-west-2`, `us-east-1`또는 등`eu-west-1`).

- MX 값과 함께 나열된 숫자 10은 메일 서버의 기본 설정 순서이며 DNS 공급자의 GUI에서 지정한 대로 별도의 값 필드에 입력해야 합니다.
- SPF의 TXT 레코드 값에는 인용 부호를 포함해야 합니다.

DNS 레코드 게시(Publish DNS records) 테이블에서 각 값 옆에 있는 복사 아이콘을 선택하여 MX 및 SPF(TXT 형식) 레코드를 복사하고 DNS 공급자 GUI의 해당 필드에 붙여 넣습니다. 또는 Download .csv record set(레코드 세트를 .csv로 다운로드)를 선택하여 레코드 사본을 컴퓨터에 저장할 수도 있습니다.

> **Important**
>
> Amazon SES를 이용해 사용자 지정 MAIL FROM 도메인을 설정하려면 MAIL FROM 도메인의 DNS 서버에 정확히 하나의 MX 레코드를 게시해야 합니다. MAIL FROM 도메인에 MX 레코드가 여러 개인 경우 Amazon SES를 사용한 사용자 지정 MAIL FROM 설정이 실패합니다.

Route 53가 MAIL FROM 도메인에 대한 DNS 서비스를 제공하고 Route 53에 사용하는 것과 AWS Management Console 동일한 계정으로에 로그인한 경우 Route 53을 사용하여 레코드 게시를 선택합니다. DNS 레코드가 도메인의 DNS 구성에 자동으로 적용됩니다.

다른 DNS 공급자를 사용하는 경우, MAIL FROM 도메인의 DNS 서버에 DNS 레코드를 수동으로 게시해야 합니다. 도메인의 DNS 서버에 DNS 레코드를 추가하는 절차는 웹 호스팅 서비스 또는 DNS 공급자에 따라 다릅니다.

도메인의 DNS 레코드를 게시하는 절차는 이용하는 DNS 공급자에 따라 다릅니다. 다음 표에는 널리 사용되는 몇몇 DNS 공급자의 설명서에 대한 링크가 포함되어 있습니다. 이는 전체 목록이 아니며 목록에 포함되어 있다고 해서 해당 공급자를 승인하는 것은 아닙니다. 마찬가지로 DNS 공급자가 목록에 없는 경우에도 MAIL FROM 도메인 구성을 지원하지 않는다는 의미는 아닙니다.

DNS/호스팅 공급자 이름	설명서 링크
GoDaddy	• MX: MX 레코드 추가(외부 링크) • TXT: TXT 레코드 추가(외부 링크)

DNS/호스팅 공급자 이름	설명서 링크
DreamHost	• MX: MX 레코드를 변경하는 방법(외부 링크) • TXT: 사용자 지정 DNS 레코드를 추가하는 방법(외부 링크)
Cloudflare	• MX: 메일 또는 MX 레코드를 추가하거나 편집하는 방법(외부 링크) • TXT: CloudFlare에서 DNS 레코드 관리(외부 링크)
HostGator	• MX: MX 레코드 변경 - Windows(외부 링크) • TXT: HostGator/eNom을 통한 DNS 레코드 관리(외부 링크)
Namecheap	• MX: 메일 서비스에 필요한 MX 레코드를 설정하는 방법(외부 링크) • TXT: 도메인에서 TXT/SPF/DKIM/DMARC 레코드를 추가하는 방법(외부 링크)
Names.co.uk	• MX: 도메인의 DNS 설정 변경(외부 링크) • TXT: 도메인 DNS 설정 변경(외부 링크)
Wix	• MX: Wix 계정에서 MX 레코드 추가 또는 업데이트(외부 링크) • TXT: Wix 계정에서 TXT 레코드 추가 또는 업데이트(외부 링크)

레코드가 있음을 Amazon SES가 감지하면 사용자 지정 MAIL FROM 도메인이 성공적으로 설정되었음을 알리는 이메일을 받습니다. DNS 공급자에 따라 Amazon SES가 MX 레코드를 감지하기까지 최대 72시간이 지연될 수도 있습니다.

Amazon SES를 사용한 사용자 지정 MAIL FROM 도메인 설정 상태

사용자 지정 MAIL FROM 도메인을 사용하도록 자격 증명을 구성한 후 Amazon SES가 DNS 설정에 필요한 MX 레코드 감지를 시도하는 동안 설정 상태는 "대기 중"입니다. Amazon SES가 MX 레코드를 감지하는지에 따라 상태가 변경됩니다. 다음 표에서는 이메일 전송 동작 및 각각의 상태에 연결된 Amazon SES 작업을 설명합니다. 상태가 변경될 때마다 Amazon SES는와 연결된 이메일 주소로 알림을 보냅니다 AWS 계정.

State	이메일 전송 동작	Amazon SES 작업
보류중	사용자 지정 MAIL FROM 대체 설정 사용	Amazon SES가 72시간 동안 필요한 MX 레코드 감지를 시도합니다. 찾지 못한 경우 상태가 "Failed"로 변경됩니다.
Success	사용자 지정 MAIL FROM 도메인 사용	필요한 MX 레코드가 있는지 Amazon SES가 지속적으로 검사합니다.
일시적인 오류	사용자 지정 MAIL FROM 대체 설정 사용	Amazon SES가 72시간 동안 필요한 MX 레코드 감지를 시도합니다. 찾기에 실패하면 상태가 "Failed"로 변경되고 찾기에 성공하면 "Success"로 변경됩니다.
Failed	사용자 지정 MAIL FROM 대체 설정 사용	Amazon SES가 더 이상 필요한

State	이메일 전송 동작	Amazon SES 작업
		MX 레코드 감지를 시도하지 않습니다. 사용자 지정 MAIL FROM 도메인을 사용하려면 사용자 지정 MAIL FROM 도메인 구성에서 설정 프로세스를 다시 시작해야 합니다.

Amazon SES의 DMARC 인증 프로토콜 준수

Domain-based Message Authentication, Reporting and Conformance(DMARC)는 Sender Policy Framework(SPF)와 도메인키 식별 메일(DKIM)을 사용하여 이메일 스푸핑 및 피싱을 찾아내는 이메일 인증 프로토콜입니다. DMARC를 준수하려면 SPF 또는 DKIM을 통해 메시지를 인증해야 하지만, 이상적으로는 둘 모두 DMARC와 함께 사용하면 이메일 전송에 대해 가능한 한 높은 수준의 보호를 보장할 수 있습니다.

각각 어떤 역할을 하는지, DMARC가 이들을 어떻게 연결하는지 간략하게 살펴보겠습니다.

- SPF - DNS에서 사용하는 DNS TXT 레코드를 통해 사용자 지정 MAIL FROM 도메인을 대신하여 메일을 보낼 수 있는 메일 서버를 식별합니다. 수신자 메일 시스템은 SPF TXT 레코드를 참조하여 사용자 지정 도메인의 메시지가 승인된 메시징 서버에서 오는지를 판단합니다. 기본적으로 SPF는 스푸핑을 방지하는 데 도움이 되도록 설계되었지만, SPF가 실제로 취약한 스푸핑 기법이 있어 DMARC와 함께 DKIM도 사용해야 합니다.
- DKIM - 이메일 헤더의 아웃바운드 메시지에 디지털 서명을 추가합니다. 수신 이메일 시스템은 이 디지털 서명을 사용하여 수신 이메일이 도메인 소유의 키로 서명되었는지 확인할 수 있습니다. 그러나 수신 이메일 시스템이 메시지를 전달하면 SPF 인증을 무효화하는 방식으로 메시지의 봉투가 변경됩니다. 디지털 서명은 이메일 헤더의 일부여서 이메일 메시지와 함께 유지되므로, DKIM은 메시지 콘텐츠가 수정되지 않은 한 메시지가 메일 서버 간에 전달된 경우에도 효과가 있습니다.
- DMARC - SPF 및 DKIM 중 하나 이상과 도메인이 정렬되었는지 확인합니다. SPF와 DKIM만 사용하면 From 주소(수신자가 이메일 클라이언트에서 보는 이메일 주소)가 인증되었는지 확인할 수 없습

니다. SPF는 MAIL FROM 주소에 지정된 도메인만 확인합니다(수신자가 볼 수 없음). DKIM은 DKIM 서명에 지정된 도메인만 확인합니다(마찬가지로 수신자가 볼 수 없음). DMARC는 SPF 또는 DKIM에서 도메인 정렬을 올바르게 설정하도록 요구하여 다음 두 문제를 해결합니다.

- SPF가 DMARC 정렬을 전달하려면 From 주소의 도메인이 MAIL FROM 주소(Return-Path 및 Envelope-from 주소라고도 함)의 도메인과 일치해야 합니다. 전달된 메일의 경우 이런 일이 거의 일어나지 않습니다. 전달된 메일이 삭제되거나, 타사 대량 이메일 공급자를 통해 메일을 보내는 경우 Return-Path(MAIL FROM)가 공급자(SES)가 소유한 주소를 사용하여 추적하는 반송 메일 및 수신 거부에 사용되기 때문입니다.
- DKIM이 DMARC 정렬을 전달하려면 DKIM 서명에 지정된 도메인이 From 주소의 도메인과 일치해야 합니다. 사용자를 대신하여 메일을 보내는 타사 발신자 또는 서비스를 사용하는 경우 타사 발신자가 DKIM 서명을 위해 올바르게 구성되었는지 확인하고 도메인 내에 적절한 DNS 레코드를 추가했는지 확인하여 이 작업을 수행할 수 있습니다. 메일 서버를 수신하면 도메인 내 주소를 사용할 권한이 있는 누군가가 보낸 이메일처럼 제3자가 보낸 이메일을 확인할 수 있습니다.

모든 것을 DMARC와 통합

위에서 설명한 DMARC 정렬 검사는 SPF, DKIM 및 DMARC가 모두 함께 작용하여 도메인에 대한 신뢰를 높이고 이메일을 받은 편지함으로 전달하는 방법을 보여줍니다. DMARC는 수신자가 보는 From 주소가 SPF 또는 DKIM에 의해 인증되도록 하여 이 작업을 수행합니다.

- 설명된 SPF 또는 DKIM 검사 중 하나 또는 둘 다 통과하면 메시지가 DMARC를 전달합니다.
- 설명된 SPF 또는 DKIM 검사가 모두 실패하면 메시지가 DMARC를 전달하지 않습니다.

따라서 DMARC가 전송된 이메일에 대한 인증을 획득할 수 있는 최적의 기회를 얻으려면 SPF와 DKIM이 모두 필요하며, 3가지를 모두 활용하면 완전한 보호 기능을 갖춘 전송 도메인을 확보하는 데 도움이 됩니다.

또한, DMARC를 사용하면 이메일 서버가 설정한 정책을 통해 DMARC 인증에 실패할 때 이메일을 처리하는 방법을 이메일 서버에 지시할 수 있습니다. 다음 섹션(the section called “도메인의 DMARC 정책 설정”)에서는 SPF와 DKIM을 모두 사용하여 DMARC 인증 프로토콜을 준수하는 이메일이 전송되도록 SES 도메인을 구성하는 방법에 대한 정보를 설명합니다.

도메인의 DMARC 정책 설정

DMARC를 설정하려면 도메인의 DNS 설정을 수정해야 합니다. 도메인의 DNS 설정에는 도메인의 DMARC 설정을 지정하는 TXT 레코드가 포함되어 있어야 합니다. DNS 구성에 TXT 레코드를 추가하는 절차는 사용하는 DNS 또는 호스팅 공급자에 따라 다릅니다. Route 53을 사용하는 경우 Amazon

Route 53 개발자 가이드의 레코드 작업을 참조하세요. 다른 공급자를 사용하는 경우 해당 공급자에 대한 DNS 구성 설명서를 참조하세요.

생성하는 TXT 레코드의 이름은 `_dmarc.`*`example.com`*이어야 합니다. 여기서 *`example.com`*은(는) 사용자 도메인입니다. TXT 레코드의 값에는 사용자 도메인에 적용되는 DMARC 정책이 포함됩니다. 다음은 DMARC 정책이 포함된 TXT 레코드의 예제입니다.

명칭	유형	값
`_dmarc.example.com`	`TXT`	`"v=DMARC1;p=quarantine;rua=mailto:my_dmarc_report@example.com"`

앞의 DMARC 정책 예제에서 이 정책은 이메일 공급자에 다음을 수행하도록 지시합니다.

- 인증에 실패한 모든 메시지의 경우 정책 파라미터인 `p=quarantine`에 지정된 대로 스팸 폴더로 보냅니다. 다른 옵션에는 `p=none`을 사용하여 아무 작업도 수행하지 않거나 `p=reject`를 사용하여 메시지를 완전히 거부하는 것이 포함됩니다.
  - 다음 섹션에서는 이러한 3가지 정책 설정을 사용하는 방법과 시기에 대해 설명합니다. 잘못된 시간에 잘못된 설정을 사용하면 이메일이 전달되지 않을 수 있습니다. the section called “DMARC 구현” 섹션을 참조하세요.
- 보고 파라미터 `rua=mailto:my_dmarc_report@example.com`(rua는 Reporting URI for Aggregate report의 약자)에서 지정한 대로 다이제스트(즉, 각 이벤트에 대한 개별 보고서를 보내지 않고 특정 기간 동안 데이터를 집계한 보고서)에서 인증에 실패한 모든 이메일에 대한 보고서를 보냅니다. 이메일 공급자는 일반적으로 하루에 한 번씩 이러한 집계 보고서를 전송하지만 이러한 정책은 공급자마다 다릅니다.

도메인에 대한 DMARC를 구성하는 방법에 대해 자세히 알아보려면 DMARC 웹 사이트의 개요를 참조하세요.

DMARC 시스템의 전체 사양은 Internet Engineering Task Force (IETF) DMARC Draft를 참조하세요.

DMARC 구현에 대한 모범 사례

메일 흐름의 나머지 부분을 방해하지 않도록 점진적이고 단계적인 접근 방식으로 DMARC 정책 적용을 구현하는 것이 가장 좋습니다. 다음 단계를 따르는 롤아웃 계획을 생성하고 구현합니다. 다음 단계

로 넘어가기 전에 먼저 각 하위 도메인과 조직의 최상위 도메인을 사용하여 이러한 각 단계를 수행합니다.

1. DMARC 구현의 영향을 모니터링합니다(p=none).
   - 메일 수신 조직이 해당 도메인을 사용하여 표시되는 메시지에 대한 통계를 보내도록 요청하는 하위 도메인 또는 도메인의 간단한 모니터링 모드 레코드부터 시작합니다. 모니터링 모드 레코드는 정책이 없음 p=none으로 설정된 DMARC TXT 레코드입니다.
   - DMARC를 통해 생성된 보고서는 이러한 검사를 통과하는 메시지의 수와 소스를 그렇지 않은 항목과 비교합니다. 합법적인 트래픽 중 얼마나 많은 트래픽이 포함되는지 또는 포함되지 않는지 쉽게 확인할 수 있습니다. 콘텐츠가 수정되면 전달된 메시지가 SPF 및 DKIM에 실패하므로, 전달 징후를 알 수 있습니다. 또한, 전송되는 사기성 메시지 수와 전송 위치를 확인할 수 있습니다.
   - 이 단계의 목표는 다음 두 단계 중 하나를 구현할 때 어떤 이메일이 영향을 받는지 파악하고, 타사 또는 승인된 발신자가 SPF 또는 DKIM 정책을 정렬하도록 하는 것입니다.
   - 기존 도메인에 가장 적합합니다.
2. 외부 메일 시스템에서 DMARC(p=quarantine)에 실패한 메일을 격리하도록 요청합니다.
   - 합법적인 트래픽의 전부 또는 대부분이 SPF 또는 DKIM과 도메인 정렬된 상태로 전송 중이라고 생각하고 DMARC 구현의 영향을 이해한 경우 격리 정책을 구현할 수 있습니다. 격리 정책은 정책이 격리 `p=quarantine`으로 설정된 DMARC TXT 레코드입니다. 이렇게 하면 DMARC 수신자가 DMARC에 실패한 도메인의 메시지를 고객의 받은 편지함 대신 스팸 폴더의 로컬 편지함에 넣도록 요청하게 됩니다.
   - 1단계에서 DMARC 보고서를 해석한 도메인을 전환하는 데 가장 적합합니다.
3. 외부 메일 시스템이 DMARC(p=reject)에 실패한 메시지를 수락하지 않도록 요청합니다.
   - 일반적으로 거부 정책을 구현하는 것이 마지막 단계입니다. 거부 정책은 정책을 거부 `p=reject`로 설정한 DMARC TXT 레코드입니다. 이렇게 하면 DMARC 수신자가 DMARC 검사에 실패한 메시지를 수락하지 않도록 요청하게 됩니다. 즉, 스팸 또는 정크 폴더에 격리되지는 않지만, 바로 거부됩니다.
   - 거부 정책을 사용하면 거부로 인해 SMTP 반송이 발생하므로, DMARC 정책을 준수하지 않는 메시지가 정확히 무엇인지 알 수 있습니다. 격리를 통해 집계 데이터는 SPF, DKIM 및 DMARC 검사를 통과하거나 통과하지 못하는 이메일의 비율에 대한 정보를 제공합니다.
   - 이전 두 단계를 거친 새 도메인 또는 기존 도메인에 가장 적합합니다.

SPF를 통해 DMARC 준수

이메일이 SPF를 기반으로 DMARC를 준수하도록 하려면 다음 두 조건을 충족해야 합니다.

- 메시지는 사용자 지정 MAIL FROM 도메인의 DNS 구성에 게시한 유효한 SPF(유형 TXT) 레코드를 기반으로 SPF 검사를 통과해야 합니다.
- 이메일 헤더의 From 주소에 있는 도메인은 MAIL FROM 주소에 지정된 도메인 또는 하위 도메인과 정렬(일치)되어야 합니다. SES와 SPF를 정렬하려면 도메인의 DMARC 정책에서 엄격한 SPF 정책(aspf=s)을 지정해서는 안 됩니다.

이러한 요구 사항을 준수하려면 다음 단계를 완료합니다.

- the section called "사용자 지정 MAIL FROM 도메인 사용"의 절차를 완료하여 사용자 지정 MAIL FROM 도메인을 설정합니다.
- 보내는 도메인이 SPF에 대해 relaxed 정책을 사용하는지 확인합니다. 도메인의 정책 정렬을 변경하지 않았다면 SES와 마찬가지로 기본적으로 완화된 정책이 사용됩니다.

> **Note**
>
> *example.com*을 해당 도메인으로 바꾸고 명령줄에 다음 명령을 입력하여 SPF에 대한 도메인의 DMARC 일치를 확인할 수 있습니다.
>
> ```
> dig TXT _dmarc.example.com
> ```
>
> 이 명령의 출력에 있는 Non-authoritative answer 아래에서 `v=DMARC1`로 시작하는 레코드를 찾습니다. 이 레코드에 문자열 `aspf=r`이 포함되었거나 `aspf` 문자열이 없는 경우, 도메인이 SPF에 대해 relaxed alignment를 사용하는 것입니다. 레코드에 문자열 `aspf=s`가 포함된 경우, 도메인이 SPF에 대해 strict alignment를 사용하는 것입니다. 시스템 관리자는 도메인의 DNS 구성에 있는 DMARC TXT 레코드에서 이 태그를 제거해야 합니다.
> 또한 dmarcian 웹 사이트의 DMARC Inspector 또는 MxToolBox 웹 사이트의 DMARC Check Tool 도구와 같은 웹 기반 DMARC 조회 도구를 사용하여 도메인의 SPF 정책 정렬을 확인할 수 있습니다.

DKIM을 통해 DMARC 준수

이메일이 DKIM을 기반으로 DMARC를 준수하도록 하려면 다음 두 조건을 충족해야 합니다.

- 메시지에 유효한 DKIM 서명이 있어야 하며 DKIM 검사를 통과해야 합니다.

- DKIM 서명에 지정된 도메인은 From 주소의 도메인과 정렬(일치)되어야 합니다. 도메인의 DMARC 정책이 DKIM에 대해 엄격한 정렬을 지정한 경우 이러한 도메인이 정확히 일치해야 합니다(SES는 기본적으로 엄격한 DKIM 정책 사용).

이러한 요구 사항을 준수하려면 다음 단계를 완료합니다.

- the section called "Easy DKIM"의 절차를 완료하여 Easy DKIM을 설정합니다. Easy DKIM을 사용하면 Amazon SES가 이메일에 자동으로 서명합니다.

> **Note**
>
> Easy DKIM을 사용하는 대신에 메시지에 수동으로 서명할 수도 있습니다. 그러나 이 경우 사용자가 작성한 DKIM 서명을 Amazon SES가 확인하지 않으므로 주의해야 합니다. 이러한 이유로 Easy DKIM을 사용하는 것이 좋습니다.

- DKIM 서명에 지정된 도메인이 From 주소의 도메인과 정렬되는지 확인합니다. 아니면 From 주소의 도메인 하위 도메인에서 보내는 경우 DMARC 정책이 완화된 정렬로 설정되어 있는지 확인합니다.

> **Note**
>
> *example.com*을 해당 도메인으로 바꾸고 명령줄에 다음 명령을 입력하여 DKIM에 대한 도메인의 DMARC 일치를 확인할 수 있습니다.
>
> ```
> dig TXT _dmarc.example.com
> ```
>
> 이 명령의 출력에 있는 Non-authoritative answer 아래에서 `v=DMARC1`로 시작하는 레코드를 찾습니다. 이 레코드에 문자열 `adkim=r`이 포함되었거나 `adkim` 문자열이 없는 경우, 도메인이 DKIM에 대해 relaxed alignment를 사용하는 것입니다. 레코드에 문자열 `adkim=s`가 포함된 경우, 도메인이 DKIM에 대해 strict alignment를 사용하는 것입니다. 시스템 관리자는 도메인의 DNS 구성에 있는 DMARC TXT 레코드에서 이 태그를 제거해야 합니다.
> 또한 dmarcian 웹 사이트의 DMARC Inspector 또는 MxToolBox 웹 사이트의 DMARC Check Tool 도구와 같은 웹 기반 DMARC 조회 도구를 사용하여 도메인의 DKIM 정책 정렬을 확인할 수 있습니다.

Amazon SES에서 BIMI 사용하기

메시지 식별을 위한 브랜드 지표(BIMI)는 이메일 수신함을 지원하는 이메일 클라이언트 내에서 브랜드의 인증된 이메일 메시지 옆에 브랜드 로고를 표시할 수 있는 이메일 사양입니다.

BIMI는 인증에 직접 연결되는 이메일 사양이지만 모든 이메일이 DMARC 인증을 준수해야 하므로 독립형 이메일 인증 프로토콜은 아닙니다.

BIMI에는 DMARC가 필요하지만 DMARC에서는 정렬을 위해 도메인에 SPF 또는 DKIM 레코드를 요구하며 보안을 강화하기 위해 SPF와 DKIM 레코드를 모두 포함하는 것이 가장 좋습니다. 또한 일부 이메일 서비스 공급자(ESP)에서는 BIMI를 사용할 때 둘 다 필요하기 때문입니다. 다음 섹션에서는 Amazon SES에서 BIMI를 구현하는 단계를 설명합니다.

SES에서 BIMI 설정하기

소유한 이메일 도메인(SES의 사용자 지정 MAIL FROM 도메인)에 대해 BIMI를 구성할 수 있습니다. 구성이 완료되면 해당 도메인에서 보내는 모든 메시지가 BIMI를 지원하는 이메일 클라이언트에 BIMI 로고를 표시합니다.

이메일에 BIMI 로고를 표시하려면 SES 내에서 몇 가지 전제 조건을 마련해야 합니다. 다음 절차에서는 이러한 사전 요구 사항이 일반화되며 이러한 주제를 자세히 다루는 전용 섹션을 참조할 것입니다. BIMI와 관련된 단계와 SES에서 BIMI를 구성하는 데 필요한 사항은 여기에 자세히 설명되어 있습니다.

사용자 지정 MAIL FROM 도메인에서 BIMI를 설정하려면

1. 해당 도메인에 대해 게시된 SPF(TXT 유형) 및 MX 레코드가 모두 포함된 사용자 지정 MAIL FROM 도메인이 SES에 구성되어 있어야 합니다. 사용자 지정 MAIL FROM 도메인이 없거나 BIMI 로고에 대한 새 사용자 지정 MAIL FROM 도메인을 만들려는 경우 the section called “사용자 지정 MAIL FROM 도메인 사용” 단원을 참조하세요.
2. Easy DKIM으로 도메인을 구성하세요. the section called “Easy DKIM”을(를) 참조하세요.
3. 2가지 예 중 하나와 유사하게 BIMI에 필요한 다음과 같은 구체적인 적용 정책과 함께 DNS 공급자에 TXT 레코드를 게시하여 DMARC로 도메인을 구성하세요.

명칭	유형	값
`_dmarc.example.com`	TXT	`v=DMARC1;p=quarantine;pct=100;rua=mailto:dmarcreports@example.com`

명칭	유형	값
`_dmarc.example.com`	`TXT`	`v=DMARC1;p=reject;rua=mailto:dmarcreports@example.com`

BIMI에 필요한 이전 DMARC 정책 예에서는 다음과 같습니다.

- *`example.com`*은 도메인 또는 하위 도메인 이름으로 대체해야 합니다.
- 이때 p= 값은 다음 중 하나가 가능합니다.
  - 다음과 같이 pct 값을 100으로 설정하여 검역소에 보관하거나
  - 다음과 같이 거부합니다.
- 하위 도메인에서 전송하는 경우 BIMI에서는 상위 도메인에도 이 적용 정책이 있어야 합니다. 하위 도메인은 상위 도메인의 정책에 속합니다. 하지만 상위 도메인에 게시된 내용 외에 하위 도메인에 대한 DMARC 레코드를 추가하는 경우 해당 하위 도메인에도 동일한 집행 정책이 있어야 BIMI를 받을 수 있습니다.
- 도메인에 대해 DMARC 정책을 설정한 적이 없다면 표시된 대로 BIMI와 관련된 DMARC 정책 값만 사용하는지 the section called “SPF를 사용하여 이메일 인증”을 참조하세요.

4. BIMI 로고를 확장 가능한 벡터 그래픽(SVG) `.svg` 파일로 제작하세요. BIMI에 필요한 특정 SVG 프로필은 SVG 휴대용/보안(SVG P/S)으로 정의됩니다. 이메일 클라이언트에 로고를 표시하려면 로고가 이러한 사양을 정확히 준수해야 합니다. SVG 로고 파일 생성 및 권장 SVG 변환 도구를 만드는 방법은 BIMI Group의 지침을 참조하세요.
5. (선택 사항) 검증된 마크 인증서(VMC)를 받으세요. Gmail 및 Apple과 같은 일부 ESP에서는 VMC에서 BIMI 로고의 상표 및 콘텐츠를 소유하고 있다는 증거를 제공해야 합니다. 도메인에 BIMI를 구현하기 위한 필수 사항은 아니지만 메일을 보내는 ESP에서 VMC 규정 준수를 시행하는 경우 BIMI 로고가 이메일 클라이언트에 표시되지 않습니다. 로고에 사용할 VMC를 받으려면 BIMI Group의 참여 인증 기관 레퍼런스를 참조하세요.
6. 액세스 권한이 있는 서버에 BIMI 로고의 SVG 파일을 호스팅하여 HTTPS를 통해 공개적으로 액세스할 수 있도록 하세요. 예를 들어 Amazon S3 버킷에 업로드할 수 있습니다.
7. 로고 URL이 포함된 BIMI DNS 레코드를 만들고 게시하세요. BIMI를 지원하는 ESP가 DMARC 레코드를 검사하면 로고 `.svg` 파일의 URL 및 구성된 경우 VMC `.pem` 파일의 URL이 포함된 BIMI 레코드도 찾습니다. 기록이 일치하면 BIMI 로고가 표시됩니다.

   다음과 같이 DNS 공급자와 함께 다음 값을 사용하여 TXT 레코드를 게시하여 BIMI로 도메인을 구성합니다. 도메인에서 보내는 것은 첫 번째 예에 표시되고 하위 도메인에서 보내는 것은 두 번째 예에 표시됩니다.

명칭	유형	값
default._bimi.example.com	TXT	v=BIMI1;l=https://myhostingserver.com/images/logo.svg;a=https://myhostingserver.com/certificate/vmc_2023-01-01.pem
default._bimi.marketing.example.com		

이전 BIMI 레코드 예시에서는

- 이름 값은 문자 그대로 *example.com* 또는 *marketing.example.com*의 하위 도메인인 `default._bimi.`로 지정되어야 하며, 이는 사용자의 도메인 또는 하위 도메인명으로 교체되어야 합니다.
- `v=` 값은 BIMI 레코드의 버전입니다.
- `l=` 값은 이미지 `.svg` 파일을 가리키는 URL을 나타내는 로고입니다.
- `a=` 값은 인증서 `.pem` 파일을 가리키는 URL을 나타내는 권한입니다.

BIMI 그룹의 BIMI 인스펙터와 같은 도구를 사용하여 BIMI 레코드를 검증할 수 있습니다.

이 프로세스의 마지막 단계는 BIMI 로고 배치를 지원하는 ESP에 정기적인 전송 패턴을 적용하는 것입니다. 도메인은 정기적으로 전송되는 주기가 있어야 하며 전송 대상 ESP에서 좋은 평판을 얻어야 합니다. 명성이나 발송 주기가 정해지지 않은 ESP에 BIMI 로고를 배치하는 데 시간이 걸릴 수 있습니다.

BIMI 그룹 조직을 통해 BIMI와 관련된 추가 정보 및 리소스를 찾을 수 있습니다.

Amazon SES에 대한 이벤트 알림 설정

Amazon SES를 사용하여 이메일을 보내려면 반송 메일 및 수신 거부를 관리할 수 있는 시스템을 준비해야 합니다. Amazon SES 알림 이메일을 전송하거나 Amazon SNS 주제에 알리거나 전송 이벤트를 게시하여 반송 메일 또는 수신 거부 이벤트를 알릴 수 있습니다. 이 섹션에는 이메일을 보내거나 Amazon SNS 주제에 알려 특정 종류의 알림을 보내도록 Amazon SES를 설정하는 방법에 대한 정보가 들어 있습니다. 전송 이벤트 게시에 대한 자세한 내용은 Amazon SES 이벤트 게시를 사용하여 이메일 전송 모니터링 섹션을 참조하세요.

Amazon SES 콘솔 또는 Amazon SES API를 사용하여 알림을 설정할 수 있습니다.

주제

- 중요 고려 사항
- 이메일을 통해 Amazon SES 알림 수신
- Amazon SNS를 사용하여 Amazon SES 알림 수신

중요 고려 사항

알림을 보내도록 Amazon SES를 설정할 때 고려해야 할 몇 가지 중요한 사항이 있습니다.

- 이메일과 Amazon SNS 알림은 개별 ID(이메일을 보내는 데 사용하는 도메인 또는 확인된 이메일 주소)에 적용됩니다. 자격 증명에 대한 알림을 활성화하면 Amazon SES는 해당 자격 증명에서 보낸 이메일에 대한 알림만 보내고 알림을 구성한 AWS 리전에서만 보냅니다.
- 반송 메일 또는 수신 거부 알림을 수신하는 방법 한 가지를 사용해야 합니다. 반송 메일 또는 수신 거부를 생성한 도메인이나 이메일 주소 또는 Amazon SNS 주제로 알림을 보낼 수 있습니다. 또한, 이벤트 게시를 사용하여 Amazon SNS 주제 또는 Firehose 스트림으로 여러 유형의 이벤트(예: 바운스, 수신 거부, 전달 등)에 대한 알림을 보낼 수도 있습니다.

  반송 메일 또는 수신 거부 알림을 수신하는 이러한 방법 중 어떤 것도 설정하지 않은 경우 이메일 피드백 전달을 비활성화했더라도 Amazon SES는 반송 메일 또는 수신 거부 이벤트를 발생시킨 이메일의 Return-Path 주소(또는 Return-Path 주소를 지정하지 않은 경우에는 Source 주소)로 반송 메일 및 수신 거부 알림을 자동으로 전달합니다.

  이메일 피드백 전달을 비활성화하고 이벤트 게시를 활성화한 경우, 이벤트 게시 규칙이 포함된 구성 세트를 전송하는 모든 이메일에 적용해야 합니다. 이 상황에서 구성 세트를 사용하지 않는 경우, Amazon SES는 반송 메일 또는 수신 거부 이벤트를 발생시킨 이메일의 Return-Path 주소 또는 Source 주소로 반송 메일 및 수신 거부 알림을 자동으로 전달합니다.
- 두 가지 이상의 방법(예: 이메일 알림 전송 및 전송 이벤트 사용)을 사용하여 반송 메일 및 수신 거부 이벤트를 보내도록 Amazon SES를 설정한 경우 동일한 이벤트에 대해 두 개 이상의 알림을 받을 수 있습니다.

이메일을 통해 Amazon SES 알림 수신

사용자가 반송 메일 및 수신 거부를 수신하면 Amazon SES는 이메일 피드백 전달이라는 프로세스를 사용하여 사용자에게 이메일을 보낼 수 있습니다.

Amazon SES를 사용하여 이메일을 보내려면 다음 방법 중 하나를 사용하여 반송 메일 및 수신 거부 알림을 보내도록 이를 구성해야 합니다.

- 이메일 피드백 전달 활성화. 이 알림 유형을 설정하는 절차는 이 섹션에 있습니다.
- Amazon SNS 주제로 알림 전송. 자세한 내용은 Amazon SNS를 사용하여 Amazon SES 알림 수신 단원을 참조하십시오.
- 이벤트 알림 게시. 자세한 내용은 Amazon SES 이벤트 게시를 사용하여 이메일 전송 모니터링 단원을 참조하십시오.

> **Important**
>
> 알림에 대한 몇 가지 중요 사항은 Amazon SES에 대한 이벤트 알림 설정 섹션을 참조하세요.

주제

- 이메일 피드백 전달 활성화
- 이메일 피드백 전달 비활성화
- 이메일 피드백 전달 대상

이메일 피드백 전달 활성화

이메일 피드백 전달은 기본적으로 활성화되어 있습니다. 이전에 비활성화한 경우 이 섹션의 절차에 따라 활성화할 수 있습니다.

Amazon SES 콘솔을 사용하여 이메일을 통한 반송 메일과 수신 거부 전달을 활성화하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. 확인된 이메일 주소 또는 도메인 목록에서 반송 메일 및 수신 거부 알림을 구성할 이메일 주소 또는 도메인을 선택합니다.
4. 세부 정보 창에서 [Notifications] 섹션을 확장합니다.
5. [Edit Configuration]을 선택합니다.
6. [Email Feedback Forwarding] 아래에서 [Enabled]를 선택합니다.

> **Note**
>
> 이 페이지에서 수행한 변경 사항이 적용되려면 몇 분이 걸릴 수 있습니다.

SetIdentityFeedbackForwardingEnabled API 작업을 사용하여 이메일을 통한 반송 메일과 수신 거부 알림을 활성화할 수도 있습니다.

이메일 피드백 전달 비활성화

반송 메일 및 수신 거부 알림을 제공하는 다른 방법을 설정한 경우, 반송 메일 또는 수신 거부 이벤트가 발생할 때 여러 알림을 수신하지 않도록 이메일 피드백 전달을 비활성화할 수 있습니다.

Amazon SES 콘솔을 사용하여 이메일을 통한 반송 메일과 수신 거부 전달을 비활성화하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. 확인된 이메일 주소 또는 도메인 목록에서 반송 메일 및 수신 거부 알림을 구성할 이메일 주소 또는 도메인을 선택합니다.
4. 세부 정보 창에서 [Notifications] 섹션을 확장합니다.
5. [Edit Configuration]을 선택합니다.
6. [Email Feedback Forwarding] 아래에서 [Disabled]를 선택합니다.

> **Note**
>
> Amazon SES를 통해 이메일을 보내려면 반송 메일 및 수신 거부 알림을 수신하는 방법 한 가지를 구성해야 합니다. 이메일 피드백 전달을 비활성화하는 경우 Amazon SNS에서 전송한 알림을 활성화하거나 이벤트 게시를 사용하여 Amazon SNS 주제 또는 Firehose 스트림에 바운스 및 수신 거부 이벤트를 게시해야 합니다. 이벤트 게시를 사용하는 경우, 이벤트 게시 규칙이 포함된 구성 세트를 전송하는 각 이메일에 적용해야 합니다. 반송 메일 및 수신 거부 알림을 수신하는 방법을 설정하지 않은 경우 Amazon SES는 반송 메일 또는 수신 거부 이벤트를 발생시킨 메시지의 Return-Path 필드(또는 Return-Path 주소를 지정하지 않은 경우 Source 필드)에 있는 주소로 이메일을 통해 피드백 알림을 자동으로 전달합니다. 이 상황에서는 이메일 피드백 알림을 비활성화했더라도 Amazon SES가 반송 메일 및 수신 거부 알림을 전달합니다.

7. Save Config(구성 저장)를 클릭하여 알림 구성을 저장합니다.

> **Note**
>
> 이 페이지에서 수행한 변경 사항이 적용되려면 몇 분이 걸릴 수 있습니다.

SetIdentityFeedbackForwardingEnabled API 작업을 사용하여 이메일을 통한 반송 메일과 수신 거부 알림을 비활성화할 수도 있습니다.

이메일 피드백 전달 대상

이메일로 알림을 받으면 Amazon SES에서 `From` 헤더를 다시 쓰고 알림을 보냅니다. Amazon SES에서 알림을 전달하는 주소는 사용자가 원본 메시지를 보낸 방식에 따라 다릅니다.

SMTP 인터페이스를 사용하여 메시지를 보낸 경우, 알림이 다음 규칙에 따라 전송됩니다.

- `SMTP DATA` 섹션에 `Return-Path` 헤더가 지정된 경우에는 해당 주소로 알림이 전송됩니다.
- 그 외의 경우에는 MAIL FROM 명령을 실행할 때 지정한 주소로 알림이 전송됩니다.

`SendEmail` API 작업을 사용하여 메시지를 보낸 경우, 알림이 다음 규칙에 따라 전송됩니다.

- `SendEmail` API에 대한 호출에 선택적 `ReturnPath` 파라미터를 지정한 경우, 알림이 해당 주소로 전송됩니다.
- 그렇지 않은 경우 `SendEmail`의 필수 `Source` 파라미터에 지정된 주소로 알림이 전송됩니다.

`SendRawEmail` API 작업을 사용하여 메시지를 보낸 경우, 알림이 다음 규칙에 따라 전송됩니다.

- 원시 메시지에 `Return-Path` 헤더가 지정된 경우에는 해당 주소로 알림이 전송됩니다.
- 그 이외의 경우 `SendRawEmail` API에 대한 호출에 `Source` 파라미터를 지정하면, 알림이 해당 주소로 전송됩니다.
- 그렇지 않은 경우에는 원시 메시지의 `From` 헤더에 있는 주소로 알림이 전송됩니다.

> **Note**
>
> `Return-Path` 주소를 이메일에 지정한 경우 해당 주소에서 알림을 받게 됩니다. 그러나 수신자가 수신하는 메시지의 버전에는 익명 이메일 주소(예:a0b1c2d3e4f5a6b7-c8d9e0f1-a2b3-c4d5-e6f7-a8b9c0d1e2f3-000000@amazonses.com)를 포함하는 `Return-Path` 헤더가 포함됩니다. 이 익명화는 이메일 발신 방법과 상관없이 발생합니다.

Amazon SNS를 사용하여 Amazon SES 알림 수신

반송 메일 또는 수신 거부를 수신하거나 이메일이 전송되면 Amazon SNS 주제에 알리도록 Amazon SES를 구성할 수 있습니다. Amazon SNS 알림은 JavaScript Object Notation(JSON) 형식을 사용하여 프로그래밍 방식으로 처리할 수 있습니다.

Amazon SES를 사용하여 이메일을 보내려면 다음 방법 중 하나를 사용하여 반송 메일 및 수신 거부 알림을 보내도록 이를 구성해야 합니다.

- Amazon SNS 주제로 알림 전송. 이 알림 유형을 설정하는 절차는 이 섹션에 있습니다.
- 이메일 피드백 전달 활성화. 자세한 내용은 이메일을 통해 Amazon SES 알림 수신 단원을 참조하십시오.
- 이벤트 알림 게시. 자세한 내용은 Amazon SES 이벤트 게시를 사용하여 이메일 전송 모니터링 단원을 참조하십시오.

> **Important**
>
> 알림에 대한 중요한 정보는 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하세요.

주제

- Amazon SES에 대한 Amazon SNS 알림 구성
- Amazon SES에 대한 Amazon SNS 알림 내용
- Amazon SES에 대한 Amazon SNS 알림 예제

Amazon SES에 대한 Amazon SNS 알림 구성

Amazon SES는 Amazon Simple Notification Service(Amazon SNS)를 통해 반송 메일, 수신 거부 및 전송에 대해 알릴 수 있습니다.

Amazon SES 콘솔 또는 Amazon SES API를 사용하여 알림을 구성할 수 있습니다.

이 섹션의 주제:

- 사전 조건
- Amazon SES 콘솔을 사용하는 알림 구성
- Amazon SES API를 사용하여 알림 구성
- 피드백 알림 문제 해결

사전 조건

Amazon SES에서 Amazon SNS 알림을 설정하기 전에 다음 단계를 완료해야 합니다.

1. Amazon SNS 주제 생성 자세한 내용은 Amazon Simple Notification Service 개발자 가이드의 주제 생성을 참조하세요.

> **Important**
>
> Amazon SNS를 사용하여 주제를 생성할 때 유형으로 표준만을 선택해야 합니다. (SES는 FIFO 유형 주제를 지원하지 않습니다.)

새 SNS 주제를 생성하든 기존 주제를 선택하든 관계없이 SES에 대한 액세스 권한을 부여하여 주제에 알림을 게시해야 합니다.

Amazon SES에 주제에 대한 알림을 게시할 수 있는 권한을 부여하려면 SNS 콘솔의 주제 편집 화면에서 액세스 정책을 확장하고 JSON 에디터에 다음 권한 정책을 추가합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Id": "notification-policy",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": "sns:Publish",
            "Resource": "arn:aws:sns:us-east-1:111122223333:topic_name",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "111122223333",
                    "AWS:SourceArn":
 "arn:aws:ses:topic_region:111122223333:identity/identity_name"
                }
            }
        }
    ]
```

```
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- *topic_region*을 SNS 주제를 생성한 AWS 리전으로 바꿉니다.
- *111122223333*을 AWS 계정 ID로 바꿉니다.
- *topic*을 SNS 주제의 이름으로 바꿉니다.
- *identity_name*을 SNS 주제에 가입하려는 확인된 자격 증명(이메일 주소 또는 도메인) 으로 바꿉니다.

2. 하나 이상의 엔드포인트를 주제에 구독시킵니다. 예를 들어 문자 메시지로 알림을 수신하고자 하는 경우 SMS 엔드포인트(예: 휴대폰 번호)를 주제에 구독시킵니다. 이메일로 알림을 수신하려면 이메일 엔드포인트(이메일 주소)를 주제에 구독시킵니다.

   자세한 내용은 Amazon Simple Notification Service 개발자 가이드의 시작하기를 참조하세요.

3. (선택 사항) Amazon SNS 주제가 서버 측 암호화에 AWS Key Management Service (AWS KMS)를 사용하는 경우 AWS KMS 키 정책에 권한을 추가해야 합니다. AWS KMS 키 정책에 다음 정책을 연결하여 권한을 추가할 수 있습니다.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSESToUseKMSKey",
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon SES 콘솔을 사용하는 알림 구성

Amazon SES 콘솔을 사용하는 알림을 구성하려면

1. https://console.aws.amazon.com/ses/에서 Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 자격 증명을 선택합니다.
3. 자격 증명(Identities) 컨테이너에서 이 자격 증명에서 보낸 메시지로 인해 반송 메일, 수신 거부 또는 배달이 발생할 때 피드백 알림을 받을 확인된 자격 증명을 선택합니다.

> **Important**
>
> 확인된 도메인 알림 설정은 역시 확인된 이메일 주소를 제외하고 해당 도메인의 이메일 주소에서 보낸 모든 메일에 적용됩니다.

4. 선택한 확인된 자격 증명의 세부 정보 화면에서 알림(Notifications) 탭을 선택하고 피드백 알림(Feedback notifications) 컨테이너에서 편집(Edit)을 선택합니다.
5. 알림을 수신할 각 피드백 유형의 SNS 주제 목록 상자를 확장하고 소유한 SNS 주제를 선택하거나 SNS 주제 없음(No SNS topic) 또는 소유하지 않은 SNS 주제(SNS topic you don't own)를 선택합니다.

   - 소유하지 않은 SNS 주제(SNS topic you don't own)를 선택할 경우, 위임 발신자가 공유한 SNS 주제 ARN을 입력해야 하는 SNS 주제 ARN(SNS topic ARN) 필드가 표시됩니다. (위임 발신자만 SNS 주제를 소유하고 있기 때문에 이러한 알림을 받습니다. 위임 전송에 대한 자세한 내용은 전송 권한 부여의 개요 단원을 참조하세요.)

> **Important**
>
> 반송 메일, 수신 거부 및 전송 알림에 사용하는 Amazon SNS 주제는 Amazon SES를 사용하는 AWS 리전 주제와 동일해야 합니다.
> 또한 알림을 받으려면 엔드포인트 하나 이상을 주제에 대해 구독하도록 해야 합니다. 예를 들어, 이메일 주소로 알림을 전송하고자 하는 경우 이메일 엔드포인트를 주제에 대해 구독하도록 해야 합니다. 자세한 내용은 Amazon Simple Notification Service Developer Guide의 Getting Started를 참조하세요.

6. (선택 사항) 주제 알림에 원본 이메일의 헤더를 포함하려면 각 피드백 유형의 SNS 주제 이름 바로 아래에 있는 원본 이메일 헤더 포함(Include original email headers) 상자를 선택합니다. 이 옵션은

연결된 알림 유형에 Amazon SNS 주제를 지정한 경우에만 사용할 수 있습니다. 원래 이메일 헤더의 내용에 대한 자세한 내용은 `mail`에서 알림 내용 객체를 참조하세요.

7. 변경 사항 저장을 선택합니다. 알림 설정의 변경 사항이 적용되려면 몇 분 정도 걸릴 수 있습니다.
8. (선택 사항) 반송 메일과 수신 거부 둘 모두에 대해 Amazon SNS 주제 알림을 선택한 경우 이메일과 SNS 알림을 통해 이중으로 알림을 받지 않도록 이메일 알림을 완전히 사용 중지할 수 있습니다. 반송 메일과 수신 거부에 대해 이메일 알림을 사용 중지하려면, 확인된 자격 증명의 세부 정보 화면에 있는 알림(Notifications) 탭의 이메일 피드백 전달(Email Feedback Forwarding) 컨테이너에서 편집(Edit)을 선택하고 사용(Enabled) 상자를 선택 취소한 다음 변경 사항 저장(Save changes)을 선택합니다.

설정 구성을 마치면 반송 메일, 수신 거부 및 전송 알림이 사용자의 Amazon SNS 주제로 수신되기 시작할 것입니다. 이 알림은 JSON(JavaScript Object Notation) 형식이며 알림 내용에서 설명하는 구조를 따릅니다.

반송 메일, 수신 거부 및 전송 알림에 대한 표준 Amazon SNS 요금이 청구됩니다. 자세한 내용은 Amazon SNS 요금 페이지를 참조하십시오.

> **Note**
>
> 주제가 삭제되었거나에 더 이상 게시할 권한이 AWS 계정 없어 Amazon SNS 주제에 게시하려는 시도가 실패하면 Amazon SES는 반송 메일 또는 수신 거부에 대해 구성된 경우 해당 주제에 대한 구성을 제거합니다(전송 아님 - 전송 알림의 경우 SES는 SNS 주제 구성 설정을 삭제하지 않음). 또한 Amazon SES는 자격 증명의 반송 메일 및 수신 거부 이메일 알림을 다시 활성화하고, 변경 알림이 이메일로 수신됩니다. 주제를 사용하도록 여러 자격 증명이 구성되면 각 자격 증명이 주제에 게시하는 데 실패할 경우 각 자격 증명의 주제 구성이 변경됩니다.

Amazon SES API를 사용하여 알림 구성

Amazon SES API를 사용하여 반송 메일, 수신 거부 및 전송 알림을 구성할 수도 있습니다. 다음 작업을 사용하여 알림을 구성합니다.

- SetIdentityNotificationTopic
- SetIdentityFeedbackForwardingEnabled
- GetIdentityNotificationAttributes
- SetIdentityHeadersInNotificationsEnabled

이러한 API 작업을 사용하여 알림을 위한 사용자 지정 프런트 엔드 애플리케이션을 작성할 수 있습니다. 알림과 관련된 API 작업에 대한 자세한 설명은 Amazon Simple Email Service API 참조를 참조하십시오.

피드백 알림 문제 해결

알림이 수신되지 않음

알림이 수신되지 않으면 알림이 전송되는 주제에 대해 엔드포인트를 구독하도록 했는지 확인하세요. 이메일 엔드포인트를 주제에 대해 신청하면 신청의 확인을 요청하는 이메일을 받게 됩니다. 이메일 알림 수신을 시작하기 전에 구독을 확인해야 합니다. 자세한 내용은 Amazon Simple Notification Service Developer Guide의 Getting Started를 참조하세요.

주제를 선택할 때 **`InvalidParameterValue`** 오류 발생

`InvalidParameterValue` 오류가 발생했다는 내용의 메시지를 받으면 Amazon SNS 주제를 확인하여 이 주제가 AWS KMS을(를) 사용하여 암호화되었는지 확인하세요. 그렇다면 AWS KMS 키에 대한 정책을 수정해야 합니다. 샘플 정책은 사전 조건 단원을 참조하세요.

Amazon SES에 대한 Amazon SNS 알림 내용

반송 메일, 수신 거부 및 전송 알림이 Amazon Simple Notification Service(Amazon SNS) 주제에 JavaScript Object Notation(JSON) 형식으로 게시됩니다. 최상위 JSON 객체는 `notificationType` 문자열, `mail` 객체 및 `bounce` 객체, `complaint` 객체 또는 `delivery` 객체를 포함합니다.

다양한 객체 유형에 대한 설명은 다음 단원을 참조하세요.

- 최상위 JSON 객체
- `mail` 객체
- `bounce` 객체
- `complaint` 객체
- `delivery` 객체

다음은 Amazon SES용 Amazon SNS 알림의 내용에 관한 몇 가지 중요한 정보입니다.

- 특정 알림 유형에서, 여러 수신자를 위한 단일 Amazon SNS 알림을 수신할 수도 있고 수신자별 Amazon SNS 알림을 수신할 수도 있습니다. 코드가 Amazon SNS 알림을 파싱할 수 있고 두 경우를 모두 처리할 수 있어야 합니다. Amazon SES는 Amazon SNS를 통해 전송된 알림에 대해 정렬 및 배칭을 보장하지 않습니다. 하지만 다른 Amazon SNS 알림 유형(예: 반송 메일과 수신 거부)을 단일 알림으로 결합하면 안 됩니다.

- 단일 수신자를 위한 여러 유형의 Amazon SNS 알림을 수신할 수 있습니다. 예를 들어 수신 메일 서버가 이메일을 수락하지만(전송 알림을 트리거) 이메일을 처리한 후 해당 이메일이 실제로는 반송된 것으로 판단할 수 있습니다(반송 메일 알림을 트리거). 그러나 이들은 알림 유형이 다르므로 언제나 별개의 알림입니다.
- Amazon SES는 알림에 추가 필드를 추가할 권리가 있습니다. 따라서 이러한 알림을 파싱하는 애플리케이션은 알 수 없는 필드를 처리할 수 있도록 유연해야 합니다.
- Amazon SES는 이메일을 보낼 때 메시지 헤더를 덮어씁니다. `mail` 객체의 `headers` 및 `commonHeaders` 필드에서 원래 메시지의 헤더를 가져올 수 있습니다.

최상위 JSON 객체

Amazon SES 알림 내 최상위 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`notificationType`	JSON 객체가 나타내는 알림 유형을 담고 있는 문자열. 가능한 값은 `Bounce`, `Complaint` 또는 `Delivery`입니다. 이벤트 게시를 설정한 경우 이 필드의 이름은 `eventType` 입니다.
`mail`	알림과 관련이 있는 원래 메일에 대한 정보를 포함하는 JSON 객체. 자세한 내용은 Mail 객체 단원을 참조하십시오.
`bounce`	이 필드는 `notificationType` 이 `Bounce`이고 반송 메일에 대한 정보를 담고 있는 JSON 객체를 포함하는 경우에만 존재합니다. 자세한 내용은 Bounce 객체 단원을 참조하십시오.
`complaint`	이 필드는 `notificationType` 이 `Complaint` 이고 수신 거부에 대한 정보를 담고 있는 JSON 객체를 포함하는 경우에만 존재합니다. 자세한 내용은 Complaint 객체 단원을 참조하십시오.

필드 이름	설명
`delivery`	이 필드는 `notificationType` 이 `Delivery`이고 전송에 대한 정보를 담고 있는 JSON 객체를 포함하는 경우에만 존재합니다. 자세한 내용은 전송 객체 단원을 참조하십시오.

Mail 객체

각 반송 메일, 수신 거부 또는 전송 알림은 `mail` 객체에 원래 메일에 대한 정보를 포함하고 있습니다. `mail` 객체 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`timestamp`	원래 메시지가 전송된 시간(ISO8601 형식).
`messageId`	Amazon SES가 메시지에 할당한 고유 ID입니다. 메시지를 보낼 때 Amazon SES에서 이 값을 반환했습니다. **Note** 이 메시지 ID는 Amazon SES에서 할당한 것입니다. `mail` 객체의 `headers` 필드에서 원래 이메일의 메시지 ID를 찾을 수 있습니다.
`source`	원래 메시지를 전송한 이메일 주소(엔벌로프 MAIL FROM 주소).
`sourceArn`	이메일을 전송하는 데 사용된 ID의 Amazon 리소스 이름(ARN). 권한 부여 전송의 경우 `sourceArn` 은 ID 소유자가 위임 발신자에게 메일 전송 시 사용하도록 권한을 부여한 ID의 ARN입니다. 권한 부여 전송에 대한 자세한 내용은 이메일 인증 방법 단원을 참조하세요.

필드 이름	설명
`sourceIp`	Amazon SES에 이메일 전송 요청을 한 클라이언트의 원본 퍼블릭 IP 주소.
`sendingAccountId`	이메일을 보내는 데 사용된 계정의 AWS 계정 ID입니다. 권한 부여 전송의 경우 `sendingAccountId` 는 위임 발신자의 계정 ID입니다.
`callerIdentity`	이메일을 전송한 Amazon SES 사용자의 IAM 자격 증명입니다.
`destination`	원래 메일의 수신자인 이메일 주소의 목록.
`headersTruncated`	이 객체는 원본 이메일의 헤더를 포함하도록 알림 설정을 구성한 경우에만 존재합니다. 알림에서 헤더가 잘렸는지 여부를 나타냅니다. Amazon SES는 원본 메시지의 헤더 크기가 10KB 이상인 경우 알림에서 헤더를 자릅니다. 가능한 값은 `true` 및 `false`입니다.
`headers`	이 객체는 원본 이메일의 헤더를 포함하도록 알림 설정을 구성한 경우에만 존재합니다. 이메일의 원래 헤더 목록입니다. 목록의 각 헤더에는 `name` 필드와 `value` 필드가 존재합니다. **Note** `headers` 객체 내의 모든 메시지 ID는 Amazon SES로 전달한 원본 메시지에서 가져온 것입니다. 이어서 Amazon SES가 메시지에 할당한 메시지 ID는 `mail` 객체의 `messageId` 필드에 들어 있습니다.

필드 이름	설명
commonHeaders	이 객체는 원본 이메일의 헤더를 포함하도록 알림 설정을 구성한 경우에만 존재합니다. 원본 이메일의 공통 이메일 헤더(예: 보내는 사람, 받는 사람, 제목 필드)에 대한 정보를 포함합니다. 이 객체 내에서 각 헤더는 키입니다. 보내는 사람 및 받는 사람 필드는 여러 값을 포함할 수 있는 어레이로 표현됩니다. **Note** 이벤트의 경우 commonHeaders 필드의 모든 메시지 ID는 Amazon SES가 메일 객체의 messageId 필드 내 메시지에 할당한 메시지 ID입니다. 알림에는 원래 이메일의 메시지 ID가 포함됩니다.

다음은 원래 이메일 헤더가 포함되어 있는 mail 객체의 예제입니다. 이 알림 유형에 원래 이메일 헤더가 포함되도록 구성하지 않은 경우, mail 객체에 headersTruncated, headers 및 commonHeaders 필드가 없습니다.

```
{
   "timestamp":"2018-10-08T14:05:45 +0000",
   "messageId":"000001378603177f-7a5433e7-8edb-42ae-af10-f0181f34d6ee-000000",
   "source":"sender@example.com",
   "sourceArn": "arn:aws:ses:us-east-1:888888888888:identity/example.com",
   "sourceIp": "127.0.3.0",
   "sendingAccountId":"123456789012",
   "destination":[
      "recipient@example.com"
   ],
   "headersTruncated":false,
   "headers":[
      {
         "name":"From",
         "value":"\"Sender Name\" <sender@example.com>"
```

```
      },
      {
         "name":"To",
         "value":"\"Recipient Name\" <recipient@example.com>"
      },
      {
         "name":"Message-ID",
         "value":"custom-message-ID"
      },
      {
         "name":"Subject",
         "value":"Hello"
      },
      {
         "name":"Content-Type",
         "value":"text/plain; charset=\"UTF-8\""
      },
      {
         "name":"Content-Transfer-Encoding",
         "value":"base64"
      },
      {
         "name":"Date",
         "value":"Mon, 08 Oct 2018 14:05:45 +0000"
      }
   ],
   "commonHeaders":{
      "from":[
         "Sender Name <sender@example.com>"
      ],
      "date":"Mon, 08 Oct 2018 14:05:45 +0000",
      "to":[
         "Recipient Name <recipient@example.com>"
      ],
      "messageId":" custom-message-ID",
      "subject":"Message sent using Amazon SES"
   }
}
```

Bounce 객체

반송 메일에 대한 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`bounceType`	Amazon SES가 결정한 반송 메일의 유형. 자세한 내용은 반송 메일 유형 단원을 참조하십시오.
`bounceSubType`	Amazon SES가 결정한 반송 메일의 하위 유형. 자세한 내용은 반송 메일 유형 단원을 참조하십시오.
`bouncedRecipients`	반송된 원래 메일의 수신자 정보를 포함하는 목록. 자세한 내용은 반송 수신자 단원을 참조하십시오.
`timestamp`	반송 메일이 전송된 날짜 및 시간(ISO8601 형식). 이 값은 Amazon SES가 알림을 수신한 시간이 아니라 ISP가 알림을 전송한 시간입니다.
`feedbackId`	반송 메일의 고유 ID.

Amazon SES에서 원격 메시지 전송 권한(MTA)에 연락할 수 있는 경우 다음 필드도 표시됩니다.

필드 이름	설명
`remoteMtaIp`	Amazon SES에서 이메일 전송을 시도한 MTA의 IP 주소입니다.

반송 메일에 전송 상태 알림(DSN)이 첨부된 경우 다음 필드도 존재합니다.

필드 이름	설명
`reportingMTA`	DSN의 `Reporting-MTA` 필드의 값. DSN에서 설명하는 전송, 중계 또는 게이트웨이 작업을 시도하는 MTA의 값입니다.

다음은 `bounce` 객체의 예입니다.

```
{
   "bounceType":"Permanent",
   "bounceSubType": "General",
   "bouncedRecipients":[
      {
         "status":"5.0.0",
         "action":"failed",
         "diagnosticCode":"smtp; 550 user unknown",
         "emailAddress":"recipient1@example.com"
      },
      {
         "status":"4.0.0",
         "action":"delayed",
         "emailAddress":"recipient2@example.com"
      }
   ],
   "reportingMTA": "example.com",
   "timestamp":"2012-05-25T14:59:38.605Z",
   "feedbackId":"000001378603176d-5a4b5ad9-6f30-4198-a8c3-b1eb0c270a1d-000000",
   "remoteMtaIp":"127.0.2.0"
}
```

반송 수신자

반송 메일 알림은 단일 수신자 또는 여러 수신자와 관련이 있을 수 있습니다. `bouncedRecipients` 필드는 객체(반송 메일 알림이 관련된 수신자당 1개)의 목록을 포함하고 있으며 항상 다음 필드로 구성됩니다.

필드 이름	설명
`emailAddress`	수신자의 이메일 주소. DSN이 사용 가능할 경우, DSN의 `Final-Recipient` 필드의 값입니다.

또는 반송 메일에 DSN이 첨부된 경우 다음 필드도 존재할 수 있습니다.

필드 이름	설명
action	DSN의 Action 필드의 값. 이 수신자에게 메시지를 전송하려는 시도의 결과로 보고-MTA가 수행하는 작업을 나타냅니다.
status	DSN의 Status 필드의 값. 메시지의 전송 상태를 나타내는 수신자별 전송 독립적 상태 코드입니다.
diagnosticCode	보고-MTA가 발행한 상태 코드. DSN의 Diagnostic-Code 필드의 값입니다. 이 필드가 DSN에는 없을 수 있습니다(따라서 JSON에도 없음).

다음은 bouncedRecipients 목록에 포함될 수 있는 객체의 예입니다.

```
{
    "emailAddress": "recipient@example.com",
    "action": "failed",
    "status": "5.0.0",
    "diagnosticCode": "X-Postfix; unknown user"
}
```

반송 메일 유형

반송 메일 객체에는 Undetermined, Permanent 또는 Transient 반송 메일 유형이 포함됩니다. Permanent 및 Transient 반송 메일 유형에는 여러 반송 메일 하위 유형 중 하나도 포함될 수 있습니다.

Transient 반송 메일 유형이 포함된 반송 메일 알림을 받을 경우, 나중에 메시지 반송의 원인이 해결될 경우 해당 수신자에게 이메일을 보낼 수 있습니다.

Permanent 반송 메일 유형이 포함된 반송 메일 알림을 받을 경우에는 나중에도 해당 수신자에게 이메일을 보내지 못할 가능성이 큽니다. 이러한 이유로 메일 발송 목록에서 반송 메일을 생성한 주소의 수신자를 즉시 제거해야 합니다.

> **Note**
>
> 소프트 바운스(수신자의 받은 편지함이 꽉 찬 상태와 같은 일시적인 문제와 관련된 반송 메일)가 발생할 경우 Amazon SES에서는 일정 기간 동안 이메일 재전송을 시도합니다. 이 기간이 끝나도 이메일을 전송할 수 없으면 Amazon SES는 시도를 중단합니다.
> Amazon SES는 하드 바운스와 소프트 바운스에 대해서 전송 시도를 중지했다는 알림을 보냅니다. 소프트 바운스가 발생할 때마다 알림을 받으려면 이벤트 게시를 활성화하고 전송 지연 이벤트가 발생할 때 알림을 보내도록 구성합니다.

bounceType	bounceSubType	설명
`Undetermined`	`Undetermined`	수신자의 이메일 공급자가 반송 메일 메시지를 보냈습니다. 반송 메일 메시지에 Amazon SES에서 반송의 원인을 파악하기에 충분한 정보가 포함되어 있지 않습니다. 반송 메일이 발생한 이메일의 반환 경로 헤더로 전송된 반송 이메일에 이메일 반송을 일으킨 문제에 대한 추가 정보가 들어 있을 수 있습니다.
`Permanent`	`General`	수신자의 이메일 공급자가 하드 바운스 메시지를 보냈습니다. **Important** 이러한 유형의 반송 메일 알림을 받은 경우, 메일 발송 목록에서 이 수신자의 이메일 주소를 즉시 제거해야 합니다. 하드 바운스를 생성하는 주소로 메시지를 보내면 발신자로서 평판에 부정적인 영향을 줄 수 있습니다. 하드 바운스를 생성하는 주소로 계속 이메일을 보내면 더 이상 이메일을 보내지 못하게 계정 사용을 일시 중지할 수 있습니다. the section called “계정 수준 금지 목록 사용”을(를) 참조하세요.

bounceType	bounceSubType	설명
Permanent	NoEmail	바운스 메시지에서 수신자 이메일 주소를 검색할 수 없습니다.
Permanent	Suppressed	수신자의 이메일 주소가 최근에 하드 바운스를 생성했다는 이력 때문에 Amazon SES 금지 목록에 있습니다. 전역 금지 목록을 재정의하려면 Amazon SES 계정 수준 금지 목록 사용 섹션을 참조하세요.
Permanent	OnAccountSuppressionList	계정 수준 금지 목록에 있으므로 Amazon SES가 이 주소로 보내는 것을 금지했습니다. 이는 반송률 지표에 반영되지 않습니다.
Transient	General	수신자의 이메일 공급자가 일반 반송 메일 메시지를 보냈습니다. 나중에 메시지 반송의 원인이 해결될 경우 같은 수신자에게 이메일을 보낼 수 있습니다. **Note** 자동 응답 규칙(예: "out of the office" 메시지)이 활성화된 수신자에게 이메일을 보낼 경우 이러한 유형의 알림을 받을 수 있습니다. 응답에 Bounce 유형의 알림이 있더라도 Amazon SES는 계정의 반송 메일 발생률을 계산할 때 자동 응답을 계산하지 않습니다.
Transient	MailboxFull	수신자의 받은 편지함이 꽉 찼기 때문에 수신자의 이메일 공급자가 하드 바운스 메시지를 보냈습니다. 나중에 받은 편지함이 비워지면 같은 수신자에게 이메일을 보낼 수 있습니다.

bounceType	bounceSubType	설명
`Transient`	`MessageTooLarge`	보낸 메시지의 크기가 너무 크기 때문에 수신자의 이메일 공급자가 하드 바운스 메시지를 보냈습니다. 메시지 크기를 줄일 경우 같은 수신자에게 메시지를 보낼 수 있습니다.
`Transient`	`ContentRejected`	보낸 메시지에 공급자가 허용하지 않는 콘텐츠가 포함되어 있기 때문에 수신자의 이메일 공급자가 반송 메일 메시지를 보냈습니다. 메시지의 콘텐츠를 변경할 경우 같은 수신자에게 메시지를 보낼 수 있습니다.
`Transient`	`AttachmentRejected`	메시지에 허용되지 않는 첨부 파일이 포함되어 있기 때문에 수신자의 이메일 공급자가 하드 바운스 메시지를 보냈습니다. 예를 들어 일부 이메일 공급자는 특정한 파일 형식으로 된 첨부 파일이 들어 있는 메시지나 첨부 파일 크기가 매우 큰 메시지를 거부할 수 있습니다. 첨부 파일을 제거하거나 첨부 파일의 콘텐츠를 변경할 경우 같은 수신자에게 메시지를 보낼 수 있습니다.

Complaint 객체

수신 거부에 대한 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`complainedRecipients`	수신 거부에 책임이 있을 수 있는 수신자에 대한 정보를 포함하는 목록. 자세한 내용은 수신 거부한 수신자 단원을 참조하십시오.
`timestamp`	ISP가 수신 거부 알림을 전송한 날짜와 시간으로, ISO 8601 형식으로 표시됩니다. 이 필드의 날짜 및 시간이 Amazon SES에서 알림을 받은 날짜 및 시간과 다를 수 있습니다.

필드 이름	설명
`feedbackId`	불만과 연결된 고유한 ID입니다.
`complaintSubType`	`complaintSubType` 필드의 값은 null 또는 `OnAccountSuppressionList` 일 수 있습니다. 값이 `OnAccountSuppressionList` 인 경우 Amazon SES는 메시지를 수락했지만 계정 수준 금지 목록에 있었기 때문에 메시지를 보내려고 시도하지 않았습니다.

또한, 수신 거부에 피드백 보고서가 첨부된 경우 다음 필드가 포함될 수 있습니다.

필드 이름	설명
`userAgent`	피드백 보고서의 `User-Agent` 필드의 값입니다. 보고서를 생성한 시스템의 이름 및 버전을 나타냅니다.
`complaintFeedbackType`	ISP로부터 수신된 피드백 보고서의 `Feedback-Type` 필드의 값. 이 값은 피드백의 유형을 포함합니다.
`arrivalDate`	피드백 보고서의 `Arrival-Date` 또는 `Received-Date` 필드의 값(ISO8601 형식). 이 필드가 보고서에는 없을 수 있습니다(따라서 JSON에도 없음).

다음은 `complaint` 객체의 예입니다.

```
{
   "userAgent":"ExampleCorp Feedback Loop (V0.01)",
   "complainedRecipients":[
      {
         "emailAddress":"recipient1@example.com"
      }
   ],
```

```
    "complaintFeedbackType":"abuse",
    "arrivalDate":"2009-12-03T04:24:21.000-05:00",
    "timestamp":"2012-05-25T14:59:38.623Z",
    "feedbackId":"000001378603177f-18c07c78-fa81-4a58-9dd1-fedc3cb8f49a-000000"
}
```

수신 거부한 수신자

`complainedRecipients` 필드는 수신 거부를 제출했을 수 있는 수신자의 목록을 포함합니다. 이 정보를 사용하여 수신 거부를 제출한 수신자를 확인한 다음 해당 수신자를 메일링 목록에서 즉시 제거해야 합니다.

> **Important**
>
> 대부분의 ISP는 수신 거부 알림에서 불만을 제출한 수신자의 이메일 주소를 제거합니다. 따라서 이 목록에는 원래 메시지의 수신자 그리고 수신 거부를 보낸 ISP를 기준으로 수신 거부를 제출했을 수 있는 수신자의 목록이 포함되어 있습니다. Amazon SES는 원래 메시지를 조회하여 이 수신자 목록을 결정합니다.

이 목록의 JSON 객체는 다음 필드를 포함합니다.

필드 이름	설명
`emailAddress`	수신자의 이메일 주소.

다음은 수신 거부를 제기한 수신자 객체의 예입니다.

```
{ "emailAddress": "recipient1@example.com" }
```

> **Note**
>
> 이 동작 때문에 (bcc 행에 나열된 30개 이메일 주소로 메시지 1개를 전송하는 대신) 수신자당 메시지 1개로 전송을 제한할 경우 어느 이메일 주소가 메시지에 대해 수신 거부를 제기했는지 더 확실히 알 수 있습니다.

수신 거부 유형

Internet Assigned Numbers Authority 웹사이트에 따라 `complaintFeedbackType` 필드에 보고 ISP가 할당한 다음과 같은 수신 거부 유형이 나타날 수 있습니다.

- `abuse` - 원치 않는 이메일 또는 기타 유형의 이메일 침해를 나타냅니다.
- `auth-failure` - 이메일 인증 실패 보고서.
- `fraud` - 일종의 사기 또는 피싱 활동을 나타냅니다.
- `not-spam` - 보고서를 제공하는 엔터티가 메시지를 스팸으로 간주하지 않음을 나타냅니다. 이는 스팸으로 잘못 태그 지정 또는 분류된 메시지를 교정하기 위해 사용될 수 있습니다.
- `other` - 다른 등록된 유형에 들어맞지 않는 기타 피드백을 나타냅니다.
- `virus` - 발원 메시지에서 바이러스가 발견되었다는 보고서.

전송 객체

전송에 대한 정보를 포함하는 JSON 객체에는 항상 다음 필드가 포함됩니다.

필드 이름	설명
`timestamp`	Amazon SES가 수신자의 메일 서버로 메일을 전송한 시간(ISO8601 형식).
`processingTimeMillis`	Amazon SES가 발신자로부터 요청을 수락한 때로부터 수신자의 메일 서버로 메시지를 전송한 때까지의 시간(단위: 밀리초).
`recipients`	전송 알림이 적용되는 이메일이 의도한 수신자의 목록.
`smtpResponse`	Amazon SES로부터 이메일을 수락한 원격 ISP의 SMTP 응답 메시지. 이 메시지는 이메일, 수신 메일 서버, 수신 ISP마다 다릅니다.
`reportingMTA`	메일을 전송한 Amazon SES 메일 서버의 호스트 이름.
`remoteMtaIp`	Amazon SES에서 이메일을 전송한 MTA의 IP 주소입니다.

다음은 `delivery` 객체의 예입니다.

```
{
   "timestamp":"2014-05-28T22:41:01.184Z",
   "processingTimeMillis":546,
   "recipients":["success@simulator.amazonses.com"],
   "smtpResponse":"250 ok:  Message 64111812 accepted",
   "reportingMTA":"a8-70.smtp-out.amazonses.com",
   "remoteMtaIp":"127.0.2.0"
}
```

Amazon SES에 대한 Amazon SNS 알림 예제

다음 단원에서는 다음과 같은 세 가지 유형의 알림 예를 다룹니다.

- 반송 메일 알림 예는 Amazon SNS 반송 메일 알림 예제 단원을 참조하세요.
- 수신 거부 알림 예는 Amazon SNS 수신 거부 알림 예제 단원을 참조하세요.
- 전송 알림 예는 Amazon SNS 전송 알림 예제 단원을 참조하세요.

Amazon SNS 반송 메일 알림 예제

이 섹션에서는 피드백을 보낸 이메일 수신자가 제공한 DSN(전송 상태 알림)이 포함되거나 포함되지 않은 반송 메일 알림의 예를 제시합니다.

DSN이 있는 반송 메일 알림

다음은 DSN과 원래 이메일 헤더가 포함되어 있는 반송 메일 알림의 예제입니다. 반송 메일 알림에 원래 이메일 헤더가 포함되도록 구성하지 않은 경우, 알림의 `mail` 객체에 `headersTruncated`, `headers` 및 `commonHeaders` 필드가 없습니다.

```
   {
       "notificationType":"Bounce",
       "bounce":{
          "bounceType":"Permanent",
          "reportingMTA":"dns; email.example.com",
          "bouncedRecipients":[
             {
                "emailAddress":"jane@example.com",
                "status":"5.1.1",
                "action":"failed",
                "diagnosticCode":"smtp; 550 5.1.1 <jane@example.com>... User"
```

```
            }
         ],
         "bounceSubType":"General",
         "timestamp":"2016-01-27T14:59:38.237Z",
         "feedbackId":"00000138111222aa-33322211-cccc-cccc-cccc-ddddaaaa068a-000000",
         "remoteMtaIp":"127.0.2.0"
      },
      "mail":{
         "timestamp":"2016-01-27T14:59:38.237Z",
         "source":"john@example.com",
         "sourceArn": "arn:aws:ses:us-east-1:888888888888:identity/example.com",
         "sourceIp": "127.0.3.0",
         "sendingAccountId":"123456789012",
         "callerIdentity": "IAM_user_or_role_name",
         "messageId":"00000138111222aa-33322211-cccc-cccc-cccc-ddddaaaa0680-000000",
         "destination":[
           "jane@example.com",
           "mary@example.com",
           "richard@example.com"],
         "headersTruncated":false,
         "headers":[
          {
            "name":"From",
            "value":"\"John Doe\" <john@example.com>"
          },
          {
            "name":"To",
            "value":"\"Jane Doe\" <jane@example.com>, \"Mary Doe\" <mary@example.com>,
\"Richard Doe\" <richard@example.com>"
          },
          {
            "name":"Message-ID",
            "value":"custom-message-ID"
          },
          {
            "name":"Subject",
            "value":"Hello"
          },
          {
            "name":"Content-Type",
            "value":"text/plain; charset=\"UTF-8\""
          },
          {
            "name":"Content-Transfer-Encoding",
```

```
            "value":"base64"
          },
          {
            "name":"Date",
            "value":"Wed, 27 Jan 2016 14:05:45 +0000"
          }
         ],
         "commonHeaders":{
            "from":[
               "John Doe <john@example.com>"
            ],
            "date":"Wed, 27 Jan 2016 14:05:45 +0000",
            "to":[
               "Jane Doe <jane@example.com>, Mary Doe <mary@example.com>, Richard Doe
 <richard@example.com>"
            ],
            "messageId":"custom-message-ID",
            "subject":"Hello"
          }
       }
   }
```

DSN이 없는 반송 메일 알림

다음은 원래 이메일 헤더가 포함되어 있지만 DSN은 없는 반송 메일 알림의 예제입니다. 반송 메일 알림에 원래 이메일 헤더가 포함되도록 구성하지 않은 경우, 알림의 `mail` 객체에 `headersTruncated`, `headers` 및 `commonHeaders` 필드가 없습니다.

```
   {
      "notificationType":"Bounce",
      "bounce":{
         "bounceType":"Permanent",
         "bounceSubType": "General",
         "bouncedRecipients":[
            {
               "emailAddress":"jane@example.com"
            },
            {
               "emailAddress":"richard@example.com"
            }
         ],
         "timestamp":"2016-01-27T14:59:38.237Z",
         "feedbackId":"00000137860315fd-869464a4-8680-4114-98d3-716fe35851f9-000000",
```

```
         "remoteMtaIp":"127.0.2.0"
      },
      "mail":{
         "timestamp":"2016-01-27T14:59:38.237Z",
         "messageId":"00000137860315fd-34208509-5b74-41f3-95c5-22c1edc3c924-000000",
         "source":"john@example.com",
         "sourceArn": "arn:aws:ses:us-east-1:888888888888:identity/example.com",
         "sourceIp": "127.0.3.0",
         "sendingAccountId":"123456789012",
         "callerIdentity": "IAM_user_or_role_name",
         "destination":[
            "jane@example.com",
            "mary@example.com",
            "richard@example.com"
         ],
        "headersTruncated":false,
        "headers":[
         {
            "name":"From",
            "value":"\"John Doe\" <john@example.com>"
         },
         {
            "name":"To",
            "value":"\"Jane Doe\" <jane@example.com>, \"Mary Doe\" <mary@example.com>,
 \"Richard Doe\" <richard@example.com>"
         },
         {
            "name":"Message-ID",
            "value":"custom-message-ID"
         },
         {
            "name":"Subject",
            "value":"Hello"
         },
         {
            "name":"Content-Type",
            "value":"text/plain; charset=\"UTF-8\""
         },
         {
            "name":"Content-Transfer-Encoding",
            "value":"base64"
         },
         {
            "name":"Date",
```

```
            "value":"Wed, 27 Jan 2016 14:05:45 +0000"
         }
        ],
        "commonHeaders":{
          "from":[
             "John Doe <john@example.com>"
          ],
          "date":"Wed, 27 Jan 2016 14:05:45 +0000",
          "to":[
             "Jane Doe <jane@example.com>, Mary Doe <mary@example.com>, Richard Doe
 <richard@example.com>"
          ],
          "messageId":"custom-message-ID",
          "subject":"Hello"
        }
     }
 }
```

Amazon SNS 수신 거부 알림 예제

이 섹션에서는 피드백을 보낸 이메일 수신자가 제공한 피드백 보고서가 포함되거나 포함되지 않은 수신 거부 알림의 예를 제시합니다.

피드백 보고서가 있는 수신 거부 알림

다음은 피드백 보고서와 원래 이메일 헤더가 포함되어 있는 수신 거부 알림의 예제입니다. 수신 거부 알림에 원래 이메일 헤더가 포함되도록 구성하지 않은 경우, 알림의 `mail` 객체에 `headersTruncated`, `headers` 및 `commonHeaders` 필드가 없습니다.

```
   {
      "notificationType":"Complaint",
      "complaint":{
         "userAgent":"AnyCompany Feedback Loop (V0.01)",
         "complainedRecipients":[
            {
               "emailAddress":"richard@example.com"
            }
         ],
         "complaintFeedbackType":"abuse",
         "arrivalDate":"2016-01-27T14:59:38.237Z",
         "timestamp":"2016-01-27T14:59:38.237Z",
         "feedbackId":"000001378603177f-18c07c78-fa81-4a58-9dd1-fedc3cb8f49a-000000"
      },
```

```
      "mail":{
         "timestamp":"2016-01-27T14:59:38.237Z",
         "messageId":"000001378603177f-7a5433e7-8edb-42ae-af10-f0181f34d6ee-000000",
         "source":"john@example.com",
         "sourceArn": "arn:aws:ses:us-east-1:888888888888:identity/example.com",
         "sourceIp": "127.0.3.0",
         "sendingAccountId":"123456789012",
         "callerIdentity": "IAM_user_or_role_name",
         "destination":[
            "jane@example.com",
            "mary@example.com",
            "richard@example.com"
         ],
          "headersTruncated":false,
          "headers":[
           {
             "name":"From",
             "value":"\"John Doe\" <john@example.com>"
           },
           {
             "name":"To",
             "value":"\"Jane Doe\" <jane@example.com>, \"Mary Doe\" <mary@example.com>,
 \"Richard Doe\" <richard@example.com>"
           },
           {
             "name":"Message-ID",
             "value":"custom-message-ID"
           },
           {
             "name":"Subject",
             "value":"Hello"
           },
           {
             "name":"Content-Type",
             "value":"text/plain; charset=\"UTF-8\""
           },
           {
             "name":"Content-Transfer-Encoding",
             "value":"base64"
           },
           {
             "name":"Date",
             "value":"Wed, 27 Jan 2016 14:05:45 +0000"
           }
```

```
         ],
         "commonHeaders":{
           "from":[
              "John Doe <john@example.com>"
           ],
           "date":"Wed, 27 Jan 2016 14:05:45 +0000",
           "to":[
              "Jane Doe <jane@example.com>, Mary Doe <mary@example.com>, Richard Doe
 <richard@example.com>"
           ],
           "messageId":"custom-message-ID",
           "subject":"Hello"
         }
      }
   }
```

피드백 보고서가 없는 수신 거부 알림

다음은 원래 이메일 헤더가 포함되어 있지만 피드백 보고서는 없는 수신 거부 알림의 예제입니다. 수신 거부 알림에 원래 이메일 헤더가 포함되도록 구성하지 않은 경우, 알림의 `mail` 객체에 `headersTruncated`, `headers` 및 `commonHeaders` 필드가 없습니다.

```
   {
      "notificationType":"Complaint",
      "complaint":{
         "complainedRecipients":[
            {
               "emailAddress":"richard@example.com"
            }
         ],
         "timestamp":"2016-01-27T14:59:38.237Z",
         "feedbackId":"0000013786031775-fea503bc-7497-49e1-881b-a0379bb037d3-000000"
      },
      "mail":{
         "timestamp":"2016-01-27T14:59:38.237Z",
         "messageId":"0000013786031775-163e3910-53eb-4c8e-a04a-f29debf88a84-000000",
         "source":"john@example.com",
         "sourceArn": "arn:aws:ses:us-east-1:888888888888:identity/example.com",
         "sourceIp": "127.0.3.0",
         "sendingAccountId":"123456789012",
         "callerIdentity": "IAM_user_or_role_name",
         "destination":[
            "jane@example.com",
```

```
            "mary@example.com",
            "richard@example.com"
         ],
         "headersTruncated":false,
         "headers":[
          {
            "name":"From",
            "value":"\"John Doe\" <john@example.com>"
          },
          {
            "name":"To",
            "value":"\"Jane Doe\" <jane@example.com>, \"Mary Doe\" <mary@example.com>,
 \"Richard Doe\" <richard@example.com>"
          },
          {
            "name":"Message-ID",
            "value":"custom-message-ID"
          },
          {
            "name":"Subject",
            "value":"Hello"
          },
          {
            "name":"Content-Type",
            "value":"text/plain; charset=\"UTF-8\""
          },
          {
            "name":"Content-Transfer-Encoding",
            "value":"base64"
          },
          {
            "name":"Date",
            "value":"Wed, 27 Jan 2016 14:05:45 +0000"
          }
          ],
          "commonHeaders":{
             "from":[
                "John Doe <john@example.com>"
             ],
             "date":"Wed, 27 Jan 2016 14:05:45 +0000",
             "to":[
                "Jane Doe <jane@example.com>, Mary Doe <mary@example.com>, Richard Doe
 <richard@example.com>"
             ],
```

```
            "messageId":"custom-message-ID",
            "subject":"Hello"
        }
    }
}
```

Amazon SNS 전송 알림 예제

다음은 원래 이메일 헤더가 포함되어 있는 전송 알림의 예제입니다. 전송 알림에 원래 이메일 헤더가 포함되도록 구성하지 않은 경우, 알림의 `mail` 객체에 `headersTruncated`, `headers` 및 `commonHeaders` 필드가 없습니다.

```
{
   "notificationType":"Delivery",
   "mail":{
      "timestamp":"2016-01-27T14:59:38.237Z",
      "messageId":"0000014644fe5ef6-9a483358-9170-4cb4-a269-f5dcdf415321-000000",
      "source":"john@example.com",
      "sourceArn": "arn:aws:ses:us-east-1:888888888888:identity/example.com",
      "sourceIp": "127.0.3.0",
      "sendingAccountId":"123456789012",
      "callerIdentity": "IAM_user_or_role_name",
      "destination":[
         "jane@example.com"
      ],
       "headersTruncated":false,
       "headers":[
        {
           "name":"From",
           "value":"\"John Doe\" <john@example.com>"
        },
        {
           "name":"To",
           "value":"\"Jane Doe\" <jane@example.com>"
        },
        {
           "name":"Message-ID",
           "value":"custom-message-ID"
        },
        {
           "name":"Subject",
           "value":"Hello"
        },
```

```
            {
               "name":"Content-Type",
               "value":"text/plain; charset=\"UTF-8\""
            },
            {
               "name":"Content-Transfer-Encoding",
               "value":"base64"
            },
            {
               "name":"Date",
               "value":"Wed, 27 Jan 2016 14:58:45 +0000"
            }
           ],
           "commonHeaders":{
             "from":[
                "John Doe <john@example.com>"
             ],
             "date":"Wed, 27 Jan 2016 14:58:45 +0000",
             "to":[
                "Jane Doe <jane@example.com>"
             ],
             "messageId":"custom-message-ID",
             "subject":"Hello"
           }
        },
       "delivery":{
          "timestamp":"2016-01-27T14:59:38.237Z",
          "recipients":["jane@example.com"],
          "processingTimeMillis":546,
          "reportingMTA":"a8-70.smtp-out.amazonses.com",
          "smtpResponse":"250 ok:  Message 64111812 accepted",
          "remoteMtaIp":"127.0.2.0"
       }
    }
```


Amazon SES에서 자격 증명 권한 부여 사용

자격 증명 권한 부여 정책은 자격 증명에 대해 허용되거나 거부되는 SES API 작업과 조건을 지정하여 확인된 개별 자격 증명이 Amazon SES를 사용할 수 있는 방법을 정의합니다.

이러한 권한 부여 정책을 통해 사용자는 언제든지 권한을 변경 또는 취소하여 자신의 자격 증명을 계속 관리할 수 있습니다. 또한 사용자는 소유한 자격 증명(도메인 또는 이메일 주소)을 다른 사용자가 자신의 SES 계정에서 사용하도록 권한을 부여할 수 있습니다.

주제

- Amazon SES 정책 구조
- Amazon SES에서 자격 증명 권한 부여 정책 생성
- Amazon SES의 자격 증명 정책 예제
- Amazon SES에서 자격 증명 권한 부여 정책 관리

Amazon SES 정책 구조

정책은 특정 구조를 준수하고 요소를 포함하며 특정 요구 사항을 충족해야 합니다.

정책 구조

각 권한 부여 정책은 특정 자격 증명에 연결되는 JSON 문서입니다. 각 정책에는 다음 단원이 포함되어 있습니다.

- 문서 상단의 정책 전반의 정보.
- 각각 하나의 권한 집합을 설명하는 하나 이상의 문.

다음 예제 정책은 확인된 도메인 example.com 작업 섹션에 지정된 AWS 계정 ID 123456789012 권한을 부여합니다.

JSON

```
{
  "Id":"ExampleAuthorizationPolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeAccount",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:123456789012:identity/example.com",
      "Principal":{
        "AWS":[
          "123456789012"
        ]
      },
      "Action":[
        "ses:GetEmailIdentity",
```

```
        "ses:UpdateEmailIdentityPolicy",
        "ses:ListRecommendations",
        "ses:CreateEmailIdentityPolicy",
        "ses:DeleteEmailIdentity"
      ]
    }
  ]
}
```

자격 증명 정책 예제에 더 많은 권한 부여 정책 예제가 나와 있습니다.

정책 요소

이 섹션에서는 자격 증명 권한 부여 정책에 포함되는 요소를 설명합니다. 먼저 전역 정책 요소를 설명한 후, 해당 요소가 포함된 문에만 적용되는 요소를 설명합니다. 그런 다음 문에 조건을 추가하는 방법을 알아봅니다.

요소의 구문에 대한 자세한 내용은 IAM 사용 설명서의 IAM 정책 언어의 문법을 참조하십시오.

정책 전반의 정보

전역 정책 요소는 2가지가 있습니다. `Id` 및 `Version`. 다음 표에 이들 요소에 대한 정보가 나와 있습니다.

명칭	설명	필수	유효값
`Id`	정책을 고유하게 식별합니다.	아니요	모든 문자열
`Version`	정책 액세스 언어 버전을 지정합니다.	아니요	임의의 문자열. 모범 사례는 이 필드를 "2012-10-17"의 값으로 포함시키는 것입니다.

정책 전용 설명문

자격 증명 권한 부여 정책은 적어도 하나의 문을 필요로 합니다. 각 문은 다음 표에서 설명하는 요소를 포함할 수 있습니다.

명칭	설명	필수	유효값
`Sid`	문을 고유하게 식별합니다.	아니요	임의의 문자열.
`Effect`	평가 시점에서 정책 문이 반환할 결과를 지정합니다.	예	"Allow" 또는 "Deny".
`Resource`	정책이 적용되는 자격 증명을 지정합니다. (전송 자격 증명의 경우 이는 자격 증명 소유자가 위임 발신자에게 사용 권한을 부여한 이메일 주소 또는 도메인입니다.)	예	자격 증명의 Amazon 리소스 이름(ARN)입니다.
`Principal`	문에서 권한을 수신하는 AWS 계정사용자 또는 AWS 서비스를 지정합니다.	예	유효한 AWS 계정 ID, 사용자 ARN 또는 AWS service. AWS 계정 IDs 및 사용자 ARNs은 `"AWS"` (예: `"AWS": ["123456789012"]` 또는 `"AWS": ["arn:aws:iam::123456789012:root"]`)를 사용하여 지정됩니다. AWS 서비스 이름은 `"Service"` (예:) 를 사용하여 지정됩니다`"Service": ["cognito`

명칭	설명	필수	유효값
			`-idp.amazonaws.com"]` . 사용자 ARN 형식에 대한 예시는 AWS 일반 참조 섹션을 참조하세요.

명칭	설명	필수	유효값
`Action`	문이 적용되는 작업을 지정합니다.	예	"ses:BatchGetMetricData", "ses:CancelExportJob", "ses:CreateDeliverabilityTestReport", "ses:CreateEmailIdentityPolicy", "ses:CreateExportJob", "ses:DeleteEmailIdentity", "ses:DeleteEmailIdentityPolicy", "ses:GetDomainStatisticsReport", "ses:GetEmailIdentity","ses:GetEmailIdentityPolicies", "ses:GetExportJob", "ses:ListExportJobs", "ses:ListRecommendations", "ses:PutEmailIdentityConfigurationSetAttributes", "ses:PutEmailIdentityDkimAttributes", "ses:PutEmailIdentityDkimSigningAttributes", "ses:PutEmailIdentityFeedbackAttributes", "ses:PutEmailIdentityMailFromAttributes", "ses:TagResource",

명칭	설명	필수	유효값
			"ses:UntagResource", "ses:UpdateEmailIdentityPolicy" (전송 권한 부여 작업: "ses:SendEmail", "ses:SendRawEmail", "ses:SendTemplatedEmail", "ses:SendBulkTemplatedEmail") 이러한 작업을 하나 이상 지정할 수 있습니다.
`Condition`	권한에 대한 제한 또는 세부 정보를 지정합니다.	아니요	이 표 다음에서 조건에 대한 자세한 내용을 참조하세요.

조건

조건(condition)은 문에 지정된 권한에 대한 제한입니다. 문에서 조건을 지정하는 부분이 가장 세부적일 수 있습니다. 키(key)는 요청의 날짜 및 시간과 같이 액세스 제한의 기준이 되는 구체적인 특성입니다.

조건과 키를 함께 사용하여 제한을 표현합니다. 예를 들어, 위임 발신자가 2019년 7월 30일 이후로는 사용자를 대신하여 Amazon SES에 요청을 하지 못하게 제한하려면 `DateLessThan` 조건을 사용합니다. `aws:CurrentTime`이라는 키를 사용하여 그 값을 `2019-07-30T00:00:00Z`로 설정합니다.

SES는 다음과 같은 AWS전체 정책 키만 구현합니다.

- `aws:CurrentTime`
- `aws:EpochTime`
- `aws:SecureTransport`
- `aws:SourceIp`
- `aws:SourceVpc`
- `aws:SourceVpce`

- `aws:UserAgent`
- `aws:VpcSourceIp`

이러한 키에 대한 자세한 내용은 IAM 사용 설명서를 참조하십시오.

정책 요구 사항

정책은 다음 요구 사항을 모두 충족해야 합니다.

- 각 정책은 하나 이상의 문을 포함해야 합니다.
- 각 정책은 하나 이상의 유효한 보안 주체를 포함해야 합니다.
- 각 정책은 리소스 하나를 지정해야 하며, 이 리소스는 정책이 연결된 자격 증명의 ARN이어야 합니다.
- 자격 증명 소유자는 각 고유 자격 증명에 최대 20개의 정책을 연결할 수 있습니다.
- 정책 크기는 4KB(킬로바이트) 이내여야 합니다.
- 정책 이름은 64자 이내여야 합니다. 영숫자 문자, 대시 및 밑줄만 포함할 수 있습니다.

Amazon SES에서 자격 증명 권한 부여 정책 생성

자격 증명 권한 부여 정책은 자격 증명에 대해 허용되거나 거부되는 API 작업과 조건을 지정하는 명령문으로 구성됩니다.

소유한 Amazon SES 도메인 또는 이메일 주소 자격 증명에 권한을 부여하려면 권한 부여 정책을 생성한 후 해당 자격 증명에 연결합니다. 자격 증명에는 정책이 없거나 하나 또는 여러 개의 정책을 포함할 수 있습니다. 하지만 단일 정책은 단일 자격 증명에만 연결할 수 있습니다.

자격 증명 권한 부여 정책에 사용할 수 있는 API 작업 목록은 the section called “정책 전용 설명문” 테이블의 Action 행을 참조하세요.

다음과 같은 방법으로 자격 증명 권한 부여 정책을 생성할 수 있습니다.

- 정책 생성기 사용 – SES 콘솔에서 정책 생성기를 사용하여 간단한 정책을 생성할 수 있습니다. SES API 작업에 대한 권한을 허용하거나 거부하는 것 외에도 조건을 사용하여 작업을 제한할 수 있습니다. 정책 생성기를 사용하여 정책의 기본 구조를 신속히 생성한 후 나중에 정책을 편집하여 사용자 지정할 수도 있습니다.
- 사용자 지정 정책 생성 - 고급 조건을 포함하거나 AWS 서비스를 보안 주체로 사용하려는 경우 SES 콘솔 또는 SES API를 사용하여 사용자 지정 정책을 생성하고 자격 증명에 연결할 수 있습니다.

주제

- 정책 생성기 사용
- 사용자 지정 정책 생성

정책 생성기 사용

다음 절차에 따라 정책 생성기를 사용하여 간단한 권한 부여 정책을 생성할 수 있습니다.

정책 생성기를 사용하여 정책을 생성하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. Identities(자격 증명) 컨테이너의 Verified identities(확인된 자격 증명) 화면에서 권한 부여 정책을 생성할 확인된 자격 증명을 선택합니다.
4. 이전 단계에서 선택한 확인된 자격 증명의 세부 정보 화면에서 권한 부여(Authorization) 탭을 선택합니다.
5. Authorization policies(권한 부여 정책) 창에서 Create policy(정책 생성)를 선택하고 드롭다운에서 Use policy generator(정책 생성기 사용)를 선택합니다.
6. 설명문 생성(Create statement) 창의 효과(Effect) 필드에서 허용(Allow)을 선택합니다. (이 자격 증명을 제한하는 정책을 만들려면 Deny(거부)를 선택합니다.)
7. 보안 주체 필드에 AWS 계정 ID, IAM 사용자 ARN 또는 AWS 서비스를 입력하여이 자격 증명에 대해 승인하려는 권한을 받은 다음 추가를 선택합니다. (두 명 이상에게 권한을 부여하려면 각각에 대해 이 단계를 반복합니다.)
8. Actions(작업) 필드에서 보안 주체에게 권한을 부여하려는 각 작업의 확인란을 선택합니다.
9. (선택 사항) 권한에 보조 설명문을 추가하려는 경우 Specify conditions(조건 지정)를 확장합니다.
   a. 연산자(Operator) 드롭다운에서 연산자를 선택합니다.
   b. 키(Key) 드롭다운에서 유형을 선택합니다.
   c. 선택한 키 유형에 따라 해당 값을 값(Value) 필드에 입력합니다. (조건을 더 추가하려면 새 조건 추가(Add new condition)를 선택하고 각 추가 조건에 대해 이 단계를 반복합니다.)
10. 설명문 저장(Save statement)을 선택합니다.
11. (선택 사항) 정책에 설명문을 더 추가하려면 다른 설명문 생성(Create another statement)을 확장하고 6~10단계를 반복합니다.

12. 다음(Next)을 선택하면 정책 사용자 지정(Customize policy) 화면의 정책 세부 정보 편집(Edit policy details) 컨테이너에 정책의 이름 및 정책 문서 자체를 변경하거나 사용자 지정할 수 있는 필드가 있습니다.
13. Next(다음)를 선택하면 Review and apply(검토 및 적용) 화면의 Overview(개요) 컨테이너에 권한을 부여하고 있는 확인된 자격 증명과 이 정책의 이름이 표시됩니다. 정책 문서(Policy document) 창에 추가한 조건으로 방금 작성한 실제 정책이 표시됩니다. 정책을 검토하고 문제가 없으면 정책 적용(Apply policy)을 선택합니다. (무언가를 변경하거나 수정해야 할 경우 이전(Previous)을 선택하고 정책 세부 정보 편집(Edit policy details) 컨테이너에서 변경합니다.)

사용자 지정 정책 생성

사용자 지정 정책을 생성하여 자격 증명에 연결할 수 있는 옵션은 다음과 같습니다.

- Amazon SES API 사용 – 텍스트 편집기에서 정책을 생성한 후 Amazon Simple Email Service API 참조에 설명된 `PutIdentityPolicy` API를 사용하여 정책을 자격 증명에 연결합니다.
- Amazon SES 콘솔 사용 – 텍스트 편집기에서 정책을 생성하고 Amazon SES 콘솔의 사용자 지정 정책 편집기에 붙여 넣어서 자격 증명에 연결합니다. 다음 절차에서는 이 방법을 설명합니다.

사용자 지정 정책 편집기를 사용하여 사용자 지정 정책을 생성하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. Identities(자격 증명) 컨테이너의 Verified identities(확인된 자격 증명) 화면에서 권한 부여 정책을 생성할 확인된 자격 증명을 선택합니다.
4. 이전 단계에서 선택한 확인된 자격 증명의 세부 정보 화면에서 권한 부여(Authorization) 탭을 선택합니다.
5. Authorization policies(권한 부여 정책) 창에서 Create policy(정책 생성)를 선택하고 드롭다운에서 Create custom policy(사용자 지정 정책 생성)를 선택합니다.
6. Policy document(정책 문서) 창에서 JSON 형식의 정책의 텍스트를 입력하거나 붙여 넣습니다. 또한 정책 생성기를 사용하여 정책의 기본 구조를 신속히 생성한 다음 여기에서 사용자 지정할 수 있습니다.
7. 정책 적용을 선택합니다. (사용자 지정 정책을 수정해야 하는 경우 권한 부여(Authorization) 탭에서 해당 확인란을 선택하고 편집(Edit)을 선택하여 정책 문서(Policy document) 창에서 변경한 후 변경 사항 저장(Save changes)을 선택합니다).

Amazon SES의 자격 증명 정책 예제

자격 증명 권한 부여를 사용하면 자격 증명에 대한 API 작업을 허용하거나 거부할 세부 조건을 지정할 수 있습니다.

다음 예제는 API 작업의 다양한 측면을 제어하는 정책을 작성하는 방법을 보여줍니다.

- 보안 주체 지정
- 작업 제한
- 복수의 설명문 사용

보안 주체 지정

권한을 부여하려는 엔터티인 보안 주체는 , AWS 계정 AWS Identity and Access Management (IAM) 사용자 또는 동일한 계정에 속한 AWS 서비스일 수 있습니다.

다음 예제는 AWS ID 123456789012가에서 소유한 AWS 계정 확인된 자격 증명 example.com 제어할 수 있도록 허용하는 간단한 정책을 보여줍니다123456789012.

JSON

```
{
  "Id":"SampleAuthorizationPolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeMarketer",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:123456789012:identity/example.com",
      "Principal":{
        "AWS":[
          "123456789012"
        ]
      },
      "Action":[
        "ses:DeleteEmailIdentity",
        "ses:PutEmailIdentityDkimSigningAttributes"
      ]
    }
  ]
}
```

다음 정책 예제는 두 사용자에게 확인된 자격 증명 example.com을 제어할 수 있는 권한을 부여합니다. 사용자는 Amazon 리소스 이름(ARN)으로 지정됩니다.

JSON

```
{
  "Id":"ExampleAuthorizationPolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeIAMUser",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:123456789012:identity/example.com",
      "Principal":{
        "AWS":[
          "arn:aws:iam::123456789012:user/John",
          "arn:aws:iam::123456789012:user/Jane"
        ]
      },
      "Action":[
        "ses:DeleteEmailIdentity",
        "ses:PutEmailIdentityDkimSigningAttributes"
      ]
    }
  ]
}
```

작업 제한

권한을 부여하려는 제어 수준에 따라 자격 증명 권한 부여 정책에서 여러 가지 작업을 지정할 수 있습니다.

```
"BatchGetMetricData",
"ListRecommendations",
"CreateDeliverabilityTestReport",
"CreateEmailIdentityPolicy",
"DeleteEmailIdentity",
"DeleteEmailIdentityPolicy",
"GetDomainStatisticsReport",
"GetEmailIdentity",
"GetEmailIdentityPolicies",
```

```
"PutEmailIdentityConfigurationSetAttributes",
"PutEmailIdentityDkimAttributes",
"PutEmailIdentityDkimSigningAttributes",
"PutEmailIdentityFeedbackAttributes",
"PutEmailIdentityMailFromAttributes",
"TagResource",
"UntagResource",
"UpdateEmailIdentityPolicy"
```

또한 자격 증명 권한 부여 정책을 통해 보안 주체를 이러한 작업 중 하나로만 제한할 수 있습니다.

JSON

```
{
    "Id": "ExamplePolicy",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ControlAction",
            "Effect": "Allow",
            "Resource": "arn:aws:ses:us-east-1:123456789012:identity/
example.com",
            "Principal": {
                "AWS": [
                    "123456789012"
                ]
            },
            "Action": [
                "ses:PutEmailIdentityMailFromAttributes"
            ]
        }
    ]
}
```

복수의 설명문 사용

자격 증명 권한 부여 정책에는 여러 문이 포함될 수 있습니다. 다음의 정책 예제에는 문이 2개입니다. 첫 번째 문은 두 명의 사용자가 동일한 계정 123456789012 내 sender@example.com에서 `getemailidentity`에 액세스하는 것을 거부합니다. 두 번째 문은 동일한 계정 123456789012 내에서 보안 주체 Jack에 대해 `UpdateEmailIdentityPolicy`를 거부합니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"DenyGet",
      "Effect":"Deny",
      "Resource":"arn:aws:ses:us-east-1:123456789012:identity/
sender@example.com",
      "Principal":{
        "AWS":[
          "arn:aws:iam::123456789012:user/John",
          "arn:aws:iam::123456789012:user/Jane"
        ]
      },
      "Action":[
        "ses:GetEmailIdentity"
      ]
    },
    {
      "Sid":"DenyUpdate",
      "Effect":"Deny",
      "Resource":"arn:aws:ses:us-east-1:123456789012:identity/
sender@example.com",
      "Principal":{
        "AWS":"arn:aws:iam::123456789012:user/Jack"
      },
      "Action":[
        "ses:UpdateEmailIdentityPolicy"
      ]
    }
  ]
}
```

Amazon SES에서 자격 증명 권한 부여 정책 관리

정책을 생성하여 자격 증명에 연결하는 것 이외에, 다음 섹션의 설명에 따라 자격 증명의 정책을 편집, 제거, 나열 및 검색할 수 있습니다.

Amazon SES 콘솔을 사용하여 정책 관리

Amazon SES 정책을 관리하려면 Amazon SES 콘솔을 사용하여 자격 증명에 연결된 정책을 보거나 편집하거나 삭제해야 합니다.

Amazon SES 콘솔을 사용하여 정책을 관리하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Verified identities(확인된 자격 증명)를 선택합니다.
3. 자격 증명 목록에서 관리할 자격 증명을 선택합니다.
4. 자격 증명의 세부 정보 페이지에서 권한 부여(Authorization) 탭을 선택합니다. 여기에서 이 자격 증명에 연결된 모든 정책 목록을 확인할 수 있습니다.
5. 해당 확인란을 선택하여 관리할 정책을 선택합니다.
6. 원하는 관리 작업에 따라 다음과 같이 해당 버튼을 선택합니다.
   a. 정책을 보려면 정책 보기(View policy)를 선택합니다. 복사본이 필요한 경우 Copy(복사) 버튼을 클릭하면 클립보드에 복사됩니다.
   b. 정책을 편집하려면 편집(Edit)을 선택합니다. 정책 문서(Policy document) 창에서 정책을 편집한 다음 변경 사항 저장(Save changes)을 선택합니다.

      > **Note**
      > 권한을 취소하려는 경우 정책을 편집하거나 제거할 수 있습니다.

   c. 정책을 제거하려면 삭제(Delete)를 선택합니다.

      > **Important**
      > 정책 제거는 영구적입니다. 정책을 제거하기 전에 정책을 복사하여 텍스트 파일에 붙여 넣어 백업하는 것이 좋습니다.

Amazon SES API를 사용하여 정책 관리

Amazon SES 정책을 관리하려면 Amazon SES API를 사용하여 자격 증명에 연결된 정책을 보거나 편집하거나 삭제해야 합니다.

Amazon SES API를 사용하여 정책을 나열하고 보려면

- ListIdentityPolicies API 작업을 사용하여 자격 증명에 연결된 정책을 나열할 수 있습니다. GetIdentityPolicies API 작업을 사용하여 정책 자체를 검색할 수도 있습니다.

Amazon SES API를 사용하여 정책을 편집하려면

- PutIdentityPolicy API 작업을 사용하여 자격 증명에 연결된 정책을 편집할 수 있습니다.

Amazon SES API를 사용하여 정책을 삭제하려면

- DeleteIdentityPolicy API 작업을 사용하여 자격 증명에 연결된 정책을 삭제할 수 있습니다.

Amazon SES에서 전송 권한 부여 사용

다른 사용자가 자신의 Amazon SES 계정을 사용하여 당신이 소유한 (도메인 또는 이메일 주소) 자격 증명에서 이메일을 보낼 수 있도록 Amazon SES를 구성할 수 있습니다. 전송 권한 부여 기능을 사용하여 사용자는 자신의 자격 증명을 계속 관리할 수 있으므로 언제라도 권한을 변경 또는 취소할 수 있습니다. 예를 들어 자영업자가 전송 권한 부여 기능을 사용하여 타사(예: 이메일 마케팅 회사)에게 사용자가 소유하고 있는 도메인에서 마케팅 이메일을 발송하도록 허용할 수 있습니다.

이 장에서는 기존 계정 간 알림 기능을 대체하는 전송 권한 부여에 대한 세부 사항을 다룹니다. 먼저 권한 부여 정책의 구조 및 정책 관리 방법과 같은 중요한 주제를 다루는 Amazon SES에서 자격 증명 권한 부여 사용에 설명된 권한 부여 정책을 사용한 자격 증명 기반 권한 부여의 기본 사항을 이해해야 합니다.

계정 간 알림 레거시 지원

ID 소유자가 확인된 ID 중 하나에서 보낼 수 있도록 승인한 위임 발신자로부터 보낸 이메일과 관련된 반송 메일, 수신 거부 및 배달에 대한 피드백 알림은 기존에 위임 발신자가 주제를 소유하지 않은 ID와 연결하는(즉, 계정 간 연결) 계정 간 알림을 사용하도록 구성되었습니다. 하지만 계정 간 알림은 위임 발신자가 확인된 ID 중 하나를 사용하여 이메일을 보내도록 ID 소유자가 승인하는 위임 발신과 관련하여 구성 세트 및 확인된 ID를 사용하는 것으로 바뀌었습니다. 이 새로운 방법을 사용하면 위임 발신자인지 확인된 ID의 소유자인지 여부에 따라 반송 메일, 수신 거부, 배달 및 기타 이벤트 알림을 다음 두 가지 구조별로 유연하게 구성할 수 있습니다.

- 구성 세트 - 위임 발신자는 자신이 소유하지 않지만 인증 정책을 통해 자격 증명 소유자가 보낼 수 있는 권한이 있는 확인된 자격 증명에서 이메일을 보낼 때 지정 가능한 자체 구성 세트에서 이벤트 게

시를 설정할 수 있습니다. 이벤트 게시를 사용하면 반송 메일, 수신 거부, 전송 및 기타 이벤트 알림을 Amazon CloudWatch, Amazon Data Firehose, Amazon Pinpoint 및 Amazon SNS에 게시할 수 있습니다. 이벤트 대상 만들기을(를) 참조하세요.

- 확인된 자격 증명 - 자격 증명 소유자는 위임 발신자가 확인된 자격 증명 중 하나를 사용하여 이메일을 보내도록 권한을 부여하는 것 외에도 위임 발신자의 요청에 따라 위임 발신자가 소유한 SNS 주제를 사용하도록 공유 자격 증명에 대한 피드백 알림을 구성할 수도 있습니다. 위임 발신자만 SNS 주제를 소유하고 있기 때문에 이러한 알림을 받습니다. '소유하지 않은 SNS 주제'를 구성하는 방법은 권한 부여 정책 절차에서 14단계를 참조하세요.

> **Note**
>
> 호환성을 위해 현재 계정에서 사용 중인 레거시 계정 간 알림에 대해서는 계정 간 알림이 계속 지원됩니다. 이 지원은 Amazon SES 클래식 콘솔에서 생성한 모든 현재 계정 간 알림을 수정하고 사용할 수 있지만 더 이상 새로운 계정 간 알림을 생성할 수는 없도록 제한됩니다. 새 Amazon SES 콘솔에서 새 계정 간 알림을 생성하려면 이벤트 게시를 사용하는 구성 세트 또는 자체 SNS 주제로 구성된 확인된 자격 증명을 사용하는 새 위임 발신 방법을 사용합니다.

주제

- Amazon SES 전송 권한 부여의 개요
- Amazon SES 전송 권한 부여를 위한 자격 증명 소유자 작업
- Amazon SES 전송 권한 부여를 위한 위임 발신자 작업

Amazon SES 전송 권한 부여의 개요

이 주제에서는 전송 권한 부여 프로세스의 개요를 살펴보고 어떻게 발신 할당량, 알림과 같은 Amazon SES의 이메일 전송 기능을 전송 권한 부여와 함께 사용하는지 설명합니다.

이 단원에서는 다음 용어를 사용합니다.

- 자격 증명 – Amazon SES 사용자가 이메일을 보내기 위해 사용하는 이메일 주소 또는 도메인입니다.
- 자격 증명 소유자 – 확인된 자격 증명에서 설명한 절차를 사용하여 이메일 주소 또는 도메인의 소유권을 확인한 Amazon SES 사용자입니다.
- 위임 발신자 - AWS 계정, AWS Identity and Access Management (IAM) 사용자 또는 자격 증명 소유자를 대신하여 이메일을 전송할 수 있는 권한 부여 정책을 통해 권한이 부여된 AWS 서비스입니다.

- 전송 권한 부여 정책 – 자격 증명에 연결되는 문서로, 누가 어떤 조건으로 해당 자격 증명을 사용하여 전송할 수 있는지 지정합니다.
- Amazon 리소스 이름(ARN) - 모든 AWS 서비스에서 AWS 리소스를 고유하게 식별하는 표준화된 방법입니다. 전송 권한 부여의 경우 리소스는 자격 증명 소유자가 위임 발신자에게 사용하도록 권한을 부여한 자격 증명입니다. arn:aws:ses:us-east-1:123456789012:identity/example.com은 ARN의 한 예입니다.

전송 권한 부여 프로세스

전송 권한 부여는 전송 권한 부여 정책을 기반으로 합니다. 위임 발신자에게 자신을 대신하여 이메일을 전송하도록 허용하려면 Amazon SES 콘솔 또는 Amazon SES API를 사용하여 전송 권한 부여 정책을 생성하여 자격 증명에 연결해야 합니다. 위임 발신자는 자격 증명 소유자를 대신하여 Amazon SES를 통해 이메일을 전송할 때 요청 또는 이메일 헤더에서 자격 증명의 ARN을 전달합니다.

이메일 전송 요청을 수신한 Amazon SES는 자격 증명 소유자의 정책(있는 경우)을 확인하여 자격 증명 소유자가 위임 발신자에게 해당 자격 증명을 대신하여 이메일을 전송하도록 허용했는지 판단합니다. 위임 발신자에게 권한이 부여된 경우 Amazon SES가 이메일을 수락하고, 그렇지 않으면 Amazon SES가 오류 메시지를 반환합니다.

다음 다이어그램은 전송 권한 부여 개념 사이의 상위 수준 관계를 보여줍니다.

전송 권한 부여 프로세스는 다음 단계로 구성됩니다.

1. 자격 증명 소유자는 위임 발신자가 사용할 확인된 자격 증명을 선택합니다. 확인된 자격 증명이 없는 경우 확인된 자격 증명 섹션을 참조하세요.

> **Note**
>
> 위임 발신자에 대해 선택한 확인된 자격 증명에는 기본 구성 세트를 할당할 수 없습니다.

2. 위임 발신자는 자격 증명 소유자가 전송에 사용할 AWS 계정 ID 또는 IAM 사용자 ARN을 알 수 있도록 합니다.
3. 자격 증명 소유자가 본인의 계정 중 하나에서 위임 발신자가 발신하도록 허용하는 데 동의할 경우, Amazon SES 콘솔 또는 Amazon SES API를 사용하여 전송 권한 부여 정책을 생성하여 선택한 자격 증명에 연결합니다.
4. 위임 발신자가 이메일 전송 시 Amazon SES에게 ARN을 제공할 수 있도록 자격 증명 소유자가 위임 발신자에게 권한이 부여된 자격 증명의 ARN을 제공합니다.
5. 위임 발신자는 이벤트 게시를 통해 위임 발신 시에 지정된 구성 세트에서 반송 메일 및 수신 거부 알림을 사용하도록 설정할 수 있습니다. 자격 증명 소유자는 반송 메일 및 수신 거부 이벤트에 대해 위임 발신자의 Amazon SNS 주제로 전송되도록 이메일 피드백 알림을 설정할 수도 있습니다.

> **Note**
>
> 자격 증명 소유자가 전송 이벤트 알림을 비활성화한 경우 위임 발신자는 Amazon SNS 주제 또는 Firehose 스트림에 반송 메일 및 수신 거부 이벤트를 게시하도록 이벤트 게시를 설정해야 합니다. 또한 발신자는 이벤트 게시 규칙이 포함된 구성 세트를 전송하는 각 이메일에 적용해야 합니다. 자격 증명 소유자도 위임 발신자도 반송 메일 및 수신 거부 이벤트에 대한 알림 전송 방법을 설정하지 않은 경우 자격 증명 소유자가 이메일 피드백 전달을 비활성화했더라도 Amazon SES는 이메일의 Return-Path 필드에 있는 주소(또는 Return-Path 주소를 지정하지 않은 경우 Source 필드의 주소)로 이메일을 통해 이벤트 알림을 자동으로 보냅니다.

6. 위임 발신자가 요청 또는 이메일 헤더에서 자격 증명 소유자의 ARN을 전달하여 자격 증명 소유자를 대신하여 Amazon SES를 통해 이메일 전송을 시도합니다. 위임 발신자는 Amazon SES SMTP 인터페이스 또는 Amazon SES API를 사용하여 이메일을 전송할 수 있습니다. 요청을 수신한 Amazon SES는 자격 증명에 연결된 모든 정책을 검사하여 위임 발신자가 지정된 "From" 주소 및 "Return Path" 주소를 사용할 권한이 부여된 경우 이메일을 수락합니다. 그렇지 않을 경우 Amazon SES가 오류 메시지를 반환하고 메시지를 수락하지 않습니다.

> **Important**
>
> 위임 발신자의 AWS 계정을 샌드박스에서 제거해야 확인되지 않은 주소로 이메일을 보내는 데 사용할 수 있습니다.

7. 자격 증명 소유자가 위임 발신자의 권한을 철회해야 하는 경우, 자격 증명 소유자는 전송 권한 부여 정책을 편집하거나 전체 정책을 삭제합니다. 자격 증명 소유자는 Amazon SES 콘솔 또는 Amazon SES API를 사용하여 각 작업을 수행할 수 있습니다.

자격 증명 소유자 또는 위임 발신자가 이러한 작업을 수행할 수 있는 방법에 대한 자세한 내용은 각각 자격 증명 소유자 작업 또는 위임 발신자 작업 단원을 참조하십시오.

이메일 전송 기능의 특성

일일 발신 할당량, 반송 메일 및 수신 거부, DKIM 서명, 피드백 전달 등의 Amazon SES 이메일 전송 기능과 관련하여 위임 발신자 및 자격 증명 소유자의 역할을 이해하는 것이 중요합니다. 특성은 다음과 같습니다.

- 발신 할당량 – 자격 증명 소유자의 자격 증명에서 전송된 이메일은 위임 발신자의 할당량에서 감산됩니다.
- 반송 메일 및 수신 거부 – 반송 메일 및 수신 거부 이벤트는 위임 발신자의 Amazon SES 계정에 기록되므로 위임 발신자의 평판에 영향을 줄 수 있습니다.
- DKIM 서명 – 자격 증명 소유자가 자격 증명에 대해 Easy DKIM 서명을 활성화한 경우 위임 발신자가 전송한 이메일을 포함하여 해당 자격 증명에서 전송된 모든 이메일이 DKIM 서명됩니다. 자격 증명 소유자는 이메일이 DKIM 서명될지 여부만 제어할 수 있습니다.
- 알림 – 자격 증명 소유자와 위임 발신자는 모두 반송 메일 및 수신 거부에 대한 알림을 설정할 수 있습니다. 이메일 자격 증명 소유자도 이메일 피드백 전달을 활성화할 수 있습니다. 알림 설정에 대한 자세한 내용은 Amazon SES 전송 활동 모니터링 단원을 참조하세요.
- 확인 – 자격 증명 소유자는 확인된 자격 증명의 절차에 따라 위임 발신자에게 사용 권한을 부여한 이메일 주소 및 도메인의 소유권을 확인할 책임이 있습니다. 위임 발신자는 전송 권한 부여만을 위해 이메일 주소 또는 도메인을 확인할 필요가 없습니다.

> **Important**
>
> 위임 발신자의 AWS 계정을 샌드박스에서 제거해야 확인되지 않은 주소로 이메일을 보내는 데 사용할 수 있습니다.

- AWS 리전 - 위임 발신자는 자격 증명 소유자의 자격 증명이 확인된 AWS 리전에서 이메일을 보내야 합니다. 위임 발신자에게 권한을 부여하는 전송 권한 부여 정책이 해당 리전의 자격 증명에 연결되어야 합니다.
- 결제 – 위임 발신자가 자격 증명 소유자의 주소를 사용해 보낸 이메일을 비롯해 위임 발신자의 계정에서 보낸 모든 메시지는 위임 발신자에게 청구됩니다.

Amazon SES 전송 권한 부여를 위한 자격 증명 소유자 작업

이 단원에서는 전송 권한 부여를 구성할 때 자격 증명 소유자가 취해야 하는 단계를 설명합니다.

주제

- Amazon SES 전송 권한 부여를 위한 자격 증명 확인
- Amazon SES 전송 권한 부여를 위한 자격 증명 소유자 알림 설정
- Amazon SES 전송 권한 부여에서 위임 발신자로부터 정보 얻기
- Amazon SES 전송 권한 부여 정책 생성
- 전송 정책 예시
- Amazon SES 전송 권한 부여에서 위임 발신자에게 자격 증명 정보 제공

Amazon SES 전송 권한 부여를 위한 자격 증명 확인

전송 권한 부여 구성의 첫 단계는 위임 발신자가 이메일을 보내는 데 사용할 이메일 주소나 도메인을 소유한다고 입증하는 것입니다. 확인 절차는 확인된 자격 증명 단원에 설명되어 있습니다.

https://console.aws.amazon.com/ses/ Verified Identity 섹션에서 상태를 확인하거나 `GetIdentityVerificationAttributes` API 작업을 사용하여 이메일 주소 또는 도메인이 확인되었는지 확인할 수 있습니다.

개발자나 위임 발신자가 확인되지 않은 이메일 주소로 이메일을 보내려면 먼저 Amazon SES 샌드박스에서 계정을 제거해 달라고 요청을 제출해야 합니다. 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하십시오.

> **Important**
>
> - 위임 발신자 AWS 계정 의를 샌드박스에서 제거해야 확인되지 않은 주소로 이메일을 보내거나 보내는 데 사용할 수 있습니다.

- 계정이 샌드박스에 있는 경우 계정에서 확인되지 않은 이메일 주소로 전송할 수 없습니다. 자격 증명 계정에서 도메인 또는 이메일 주소가 확인된 경우에도 마찬가지입니다.

Amazon SES 전송 권한 부여를 위한 자격 증명 소유자 알림 설정

사용자를 대신하여 이메일을 보내도록 위임 발신자에게 권한을 부여하면 Amazon SES는 해당 이메일로 인해 생성되는 모든 반송 메일 또는 수신 거부를 사용자가 아니라 위임 발신자의 반송 메일 및 수신 거부 한도에 반영합니다. 그러나 위임 발신자가 보낸 메시지로 인해 IP 주소가 타사 스팸 방지, DNS 블랙리스트(DNSBL)에 오를 경우, 자격 증명의 평판이 손상될 수 있습니다. 따라서 자격 증명 소유자일 경우 위임 발신을 승인한 자격 증명을 포함하여 모든 자격 증명에 대해 이메일 피드백 전달을 설정해야 합니다. 자세한 내용은 이메일을 통해 Amazon SES 알림 수신 단원을 참조하십시오.

위임 발신자는 사용자가 사용 권한을 부여한 자격 증명에 대해 자체 반송 메일 및 수신 거부 알림을 설정할 수 있고 그렇게 해야 합니다. 반송 메일 및 수신 거부 이벤트를 Amazon SNS 주제 또는 Firehose 스트림에 게시하도록 이벤트 게시를 설정할 수 있습니다.

자격 증명 소유자도 위임 발신자도 반송 메일 및 수신 거부 이벤트에 대한 알림 전송 방법을 설정하지 않은 경우 또는 발신자가 이벤트 게시 규칙을 사용하는 구성 세트를 적용하지 않은 경우 이메일 피드백 전달을 비활성화했더라도 Amazon SES는 이메일의 Return-Path 필드에 있는 주소(또는 Return-Path 주소를 지정하지 않은 경우 Source 필드의 주소)로 이메일을 통해 이벤트 알림을 자동으로 보냅니다. 이 프로세스는 다음 이미지에 설명되어 있습니다.

Amazon SES 전송 권한 부여에서 위임 발신자로부터 정보 얻기

발신 권한 부여 정책에 하나 이상의 보안 주체를 지정해야 합니다. 보안 주체는 확인된 자격 증명 중 하나를 대신하여 발신할 수 있도록 액세스 권한을 부여하는 위임 발신자의 엔터티입니다. Amazon SES 전송 권한 부여 정책의 경우 보안 주체는 위임 발신자의 AWS 계정, AWS Identity and Access Management (IAM) 사용자 ARN 또는 AWS 서비스일 수 있습니다.

쉽게 설명하자면 보안 주체(위임 발신자)는 피부여자이고 귀하(자격 증명 소유자)는 권한 부여 정책의 부여자입니다. 즉, 귀하는 보안 주체에게 여러분이 소유한 리소스(확인된 자격 증명)에서 이메일, 원시 이메일, 템플릿 이메일 또는 대량 템플릿 이메일의 어떤 조합이든 보낼 수 있는 허용 권한을 부여합니다.

가장 세부적인 제어를 원한다면 위임 발신자의 AWS 계정 내 임의의 사용자가 아니라 하나의 위임 발신자만 사용자를 대신하여 이메일을 전송하도록 위임 발신자에게 IAM 사용자를 설정하도록 요청하십시오. 위임 발신자는 IAM 사용 설명서의 AWS 계정에서 IAM 사용자 생성에서 IAM 사용자를 설정하는 방법에 대한 자세한 내용을 확인할 수 있습니다.

위임 발신자에게 AWS 계정 ID 또는 IAM 사용자의 Amazon 리소스 이름(ARN)을 요청하여 전송 권한 부여 정책에 포함할 수 있도록 합니다. 위임 발신자에게 자격 증명 소유자에게 정보 제공 내 지침을 사용하여 이 정보를 찾을 수 있음을 알려줄 수 있습니다. 위임 발신자가 AWS 서비스인 경우 해당 서비스에 대한 설명서를 참조하여 서비스 이름을 확인합니다.

다음 예제 정책은 위임 발신자가 자격 증명 소유자의 리소스에서 발신할 수 있도록 권한을 부여하기 위해 자격 증명 소유자가 만드는 정책에 필요한 기본 요소를 보여 줍니다. 자격 증명 소유자는 확인된 자격 증명 워크플로를 거치고 권한 부여에서 정책 생성기를 사용하여 가장 간단한 형식으로, 자격 증명 소유자가 소유한 리소스를 대신하여 위임 발신자가 발신할 수 있도록 하는 다음과 같은 기본 정책을 만듭니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSESSendEmail",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": [
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource": [
          "arn:aws:ses:us-east-1:444455556666:identity/bob@example.com"
      ],
      "Condition": {}
    }
  ]
}
```

위 정책에 대해 다음 범례는 핵심 요소와 해당 요소를 소유한 사용자를 설명합니다.

- 보안 주체(Principal) - 이 필드는 위임 발신자의 IAM 사용자 ARN으로 채워집니다.
- 작업(Action) - 이 필드는 위임 발신자가 자격 증명 소유자의 리소스에서 수행하도록 자격 증명 소유자가 허용하는 2개의 SES 작업(`SendEmail` 및 `SendRawEmail`)으로 채워집니다.
- 리소스(Resource) - 이 필드는 위임 발신자에게 발신 권한을 부여하는 자격 증명 소유자의 확인된 리소스로 채워집니다.

Amazon SES 전송 권한 부여 정책 생성

자격 증명 권한 부여 정책 생성에 설명된 대로 Amazon SES에서 권한 부여 정책을 생성하는 것과 유사하게 위임 발신자에게 이메일 주소 또는 도메인(자격 증명)을 사용하여 이메일을 전송하도록 권한을 부여하려면 SES 전송 API 작업이 지정된 정책을 생성한 후 해당 자격 증명에 연결합니다.

전송 권한 부여 정책에 지정할 수 있는 API 작업 목록은 the section called “정책 전용 설명문” 테이블의 Action 행을 참조하세요.

정책 생성기를 사용하거나 사용자 지정 정책을 생성하여 전송 권한 부여 정책을 만들 수 있습니다. 두 방법의 경우 모두 전송 권한 부여 정책을 만드는 절차가 제공됩니다.

> **Note**
>
> - 이메일 주소 자격 증명에 연결한 전송 권한 부여 정책은 해당 도메인 자격 증명에 연결한 정책보다 우선합니다. 예를 들어 사용자가 example.com에 대해서는 특정 위임 발신자를 허용하지 않는 정책을 생성하고 sender@example.com에 대해서는 이 위임 발신자를 허용하는 정책을 생성할 경우, 위임 발신자는 sender@example.com에서는 이메일을 전송할 수 있지만 example.com 도메인의 기타 모든 주소에서는 전송할 수 없습니다.
> - 사용자가 example.com에 대해서는 특정 위임 발신자를 허용하는 정책을 생성하고 sender@example.com에 대해서는 이 위임 발신자를 허용하지 않는 정책을 생성할 경우, 위임 발신자는 example.com 도메인에서는 이메일을 전송할 수 있지만 sender@example.com에서는 전송할 수 없습니다.
> - SES 권한 부여 정책의 구조에 익숙하지 않은 경우 정책 구조 섹션을 참조하세요.
> - 권한을 부여하려는 자격 증명이 글로벌 엔드포인트 기능의 일부로 보조 리전에서 복제되는 경우 위임 발신자가 두 리전 모두에서이 자격 증명을 사용하여 전송할 수 있는 권한을 갖도록 기본 및 보조 리전 모두에서 자격 증명에 대한 전송 권한 부여 정책을 생성해야 합니다.

정책 생성기를 사용하여 전송 권한 부여 정책 생성

다음 절차에 따라 정책 생성기를 사용하여 전송 권한 부여 정책을 생성할 수 있습니다.

정책 생성기를 사용하여 전송 권한 부여 정책을 생성하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 자격 증명을 선택합니다.
3. 자격 증명(Identities) 컨테이너의 확인된 자격 증명(Verified identities) 화면에서 위임 발신자가 귀하를 대신하여 발신할 수 있도록 승인할 확인된 자격 증명을 선택합니다.
4. 확인된 자격 증명의 권한 부여 탭을 선택합니다.
5. Authorization policies(권한 부여 정책) 창에서 Create policy(정책 생성)를 선택하고 드롭다운에서 Use policy generator(정책 생성기 사용)를 선택합니다.
6. 설명문 생성(Create statement) 창의 효과(Effect) 필드에서 허용(Allow)을 선택합니다. (위임 발신자를 제한하는 정책을 만들려면 거부(Deny)를 선택합니다.)
7. 보안 주체(Principal) 필드에 위임 발신자가 귀하와 공유한 AWS 계정 ID 또는 IAM 사용자 ARN을 입력하여 이 자격 증명에 해당하는 귀하의 계정을 대신하여 이메일을 보낼 수 있도록 권한을 부여한 다음 추가(Add)를 선택합니다. (두 명 이상의 위임 발신자에게 권한을 부여하려면 각 위임 발신자에 대해 이 단계를 반복합니다.)
8. 작업(Actions) 필드에서 위임 발신자에게 권한을 부여하려는 각 전송 유형에 대해 확인란을 선택합니다.
9. (선택 사항) 위임 발신자 권한에 보조 설명문을 추가하려는 경우 조건 지정(Specify conditions)을 확장합니다.
   a. 연산자(Operator) 드롭다운에서 연산자를 선택합니다.
   b. 키(Key) 드롭다운에서 유형을 선택합니다.
   c. 선택한 키 유형에 따라 해당 값을 값(Value) 필드에 입력합니다. (조건을 더 추가하려면 새 조건 추가(Add new condition)를 선택하고 각 추가 조건에 대해 이 단계를 반복합니다.)
10. 설명문 저장(Save statement)을 선택합니다.
11. (선택 사항) 정책에 설명문을 더 추가하려면 다른 설명문 생성(Create another statement)을 확장하고 6~10단계를 반복합니다.
12. 다음(Next)을 선택하면 정책 사용자 지정(Customize policy) 화면의 정책 세부 정보 편집(Edit policy details) 컨테이너에 정책의 이름 및 정책 문서 자체를 변경하거나 사용자 지정할 수 있는 필드가 있습니다.

13. 다음(Next)을 선택하면 검토 및 적용(Review and apply) 화면의 개요(Overview) 컨테이너에 위임 발신자에 대해 권한을 부여하고 있는 확인된 자격 증명과 이 정책의 이름이 표시됩니다. 정책 문서(Policy document) 창에 추가한 조건으로 방금 작성한 실제 정책이 표시됩니다. 정책을 검토하고 문제가 없으면 정책 적용(Apply policy)을 선택합니다. (무언가를 변경하거나 수정해야 할 경우 이전(Previous)을 선택하고 정책 세부 정보 편집(Edit policy details) 컨테이너에서 변경합니다.) 방금 만든 정책을 통해 위임 발신자가 귀하를 대신하여 발신할 수 있습니다.

14. (선택 사항) 위임 발신자가 소유한 SNS 주제를 사용하거나 반송 메일 또는 수신 거부를 수신할 때 피드백 알림을 받거나 이메일이 배달될 때, 이 확인된 자격 증명에서 SNS 주제를 구성해야 합니다. (위임 발신자는 구하와 SNS 주제 ARN을 공유해야 합니다.) 알림(Notifications) 탭을 선택하고 피드백 알림(Feedback notifications) 컨테이너에서 편집(Edit)을 선택합니다.

    a. SNS 주제 구성(Configure SNS topics) 창의 피드백 필드(반송 메일, 수신 거부 또는 배달)에서 소유하지 않은 SNS 주제(SNS topic you don’t own)를 선택한 다음, 위임 발신자가 소유하고 공유한 SNS 주제 ARN을 입력합니다. (위임 발신자만 SNS 주제를 소유하고 있기 때문에 이러한 알림을 수신합니다. 자격 증명 소유자는 이 알림을 수신하지 않습니다.)

    b. (선택 사항) 주제 알림에 원본 이메일의 헤더를 포함하려면 각 피드백 유형의 SNS 주제 이름 바로 아래에 있는 원본 이메일 헤더 포함(Include original email headers) 상자를 선택합니다. 이 옵션은 연결된 알림 유형에 Amazon SNS 주제를 지정한 경우에만 사용할 수 있습니다. 원래 이메일 헤더의 내용에 대한 자세한 내용은 `mail`에서 알림 내용 객체를 참조하세요.

    c. 변경 사항 저장을 선택합니다. 알림 설정의 변경 사항이 적용되려면 몇 분 정도 걸릴 수 있습니다.

    d. (선택 사항) 위임 발신자가 반송 메일 및 수신 거부에 대한 Amazon SNS 주제 알림을 받게 되므로 이 자격 증명 전송에 대한 피드백을 받지 않으려면 이메일 알림을 완전히 사용 중지할 수 있습니다. 반송 메일과 수신 거부에 대해 이메일 피드백을 사용 중지하려면, 알림(Notifications) 탭의 이메일 피드백 전달(Email Feedback Forwarding) 컨테이너에서 편집(Edit)을 선택하고 사용(Enabled) 상자를 선택 취소한 다음 변경 사항 저장(Save changes)을 선택합니다. 이제 전송 상태 알림이 위임 발신자가 소유한 SNS 주제로만 전송됩니다.

사용자 지정 전송 권한 부여 정책 생성

사용자 지정 전송 권한 부여 정책을 생성하여 자격 증명에 연결할 수 있는 옵션은 다음과 같습니다.

- Amazon SES API 사용 – 텍스트 편집기에서 정책을 생성한 후 Amazon Simple Email Service API 참조에 설명된 `PutIdentityPolicy` API를 사용하여 정책을 자격 증명에 연결합니다.
- Amazon SES 콘솔 사용 – 텍스트 편집기에서 정책을 생성하고 Amazon SES 콘솔의 사용자 지정 정책 편집기에 붙여 넣어서 자격 증명에 연결합니다. 다음 절차에서는 이 방법을 설명합니다.

사용자 지정 정책 편집기를 사용하여 사용자 지정 전송 권한 부여 정책을 생성하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 자격 증명을 선택합니다.
3. 자격 증명(Identities) 컨테이너의 확인된 자격 증명(Verified identities) 화면에서 위임 발신자가 귀하를 대신하여 발신할 수 있도록 승인할 확인된 자격 증명을 선택합니다.
4. 이전 단계에서 선택한 확인된 자격 증명의 세부 정보 화면에서 권한 부여(Authorization) 탭을 선택합니다.
5. Authorization policies(권한 부여 정책) 창에서 Create policy(정책 생성)를 선택하고 드롭다운에서 Create custom policy(사용자 지정 정책 생성)를 선택합니다.
6. Policy document(정책 문서) 창에서 JSON 형식의 정책의 텍스트를 입력하거나 붙여 넣습니다. 또한 정책 생성기를 사용하여 정책의 기본 구조를 신속히 생성한 다음 여기에서 사용자 지정할 수 있습니다.
7. 정책 적용을 선택합니다. (사용자 지정 정책을 수정해야 하는 경우 권한 부여(Authorization) 탭에서 해당 확인란을 선택하고 편집(Edit)을 선택하여 정책 문서(Policy document) 창에서 변경한 후 변경 사항 저장(Save changes)을 선택합니다).
8. (선택 사항) 위임 발신자가 소유한 SNS 주제를 사용하거나 반송 메일 또는 수신 거부를 수신할 때 피드백 알림을 받거나 이메일이 배달될 때, 이 확인된 자격 증명에서 SNS 주제를 구성해야 합니다. (위임 발신자는 구하와 SNS 주제 ARN을 공유해야 합니다.) 알림(Notifications) 탭을 선택하고 피드백 알림(Feedback notifications) 컨테이너에서 편집(Edit)을 선택합니다.

   a. SNS 주제 구성(Configure SNS topics) 창의 피드백 필드(반송 메일, 수신 거부 또는 배달)에서 소유하지 않은 SNS 주제(SNS topic you don’t own)를 선택한 다음, 위임 발신자가 소유하고 공유한 SNS 주제 ARN을 입력합니다. (위임 발신자만 SNS 주제를 소유하고 있기 때문에 이러한 알림을 수신합니다. 자격 증명 소유자는 이 알림을 수신하지 않습니다.)

   b. (선택 사항) 주제 알림에 원본 이메일의 헤더를 포함하려면 각 피드백 유형의 SNS 주제 이름 바로 아래에 있는 원본 이메일 헤더 포함(Include original email headers) 상자를 선택합니다. 이 옵션은 연결된 알림 유형에 Amazon SNS 주제를 지정한 경우에만 사용할 수 있습니다. 원래 이메일 헤더의 내용에 대한 자세한 내용은 `mail`에서 알림 내용 객체를 참조하세요.

   c. 변경 사항 저장을 선택합니다. 알림 설정의 변경 사항이 적용되려면 몇 분 정도 걸릴 수 있습니다.

   d. (선택 사항) 위임 발신자가 반송 메일 및 수신 거부에 대한 Amazon SNS 주제 알림을 받게 되므로 이 자격 증명 전송에 대한 피드백을 받지 않으려면 이메일 알림을 완전히 사용 중

지할 수 있습니다. 반송 메일과 수신 거부에 대해 이메일 피드백을 사용 중지하려면, 알림(Notifications) 탭의 이메일 피드백 전달(Email Feedback Forwarding) 컨테이너에서 편집(Edit)을 선택하고 사용(Enabled) 상자를 선택 취소한 다음 변경 사항 저장(Save changes)을 선택합니다. 이제 전송 상태 알림이 위임 발신자가 소유한 SNS 주제로만 전송됩니다.

전송 정책 예시

전송 권한 부여에서는 사용자가 위임 발신자에게 사용자를 대신하여 이메일을 전송하도록 허용하는 세부 조건을 지정할 수 있습니다.

다음 조건 및 예제는 전송의 다양한 측면을 제어하는 정책을 작성하는 방법을 보여줍니다.

- 전송 권한 부여와 관련된 조건
- 위임 발신자 지정
- "From" 주소 제한
- 위임자가 이메일을 전송할 수 있는 시간 제한
- 이메일 전송 작업 제한
- 이메일 발신자의 표시 이름 제한
- 복수의 설명문 사용

전송 권한 부여와 관련된 조건

조건(condition)은 문에 지정된 권한에 대한 제한입니다. 문에서 조건을 지정하는 부분이 가장 세부적일 수 있습니다. 키(key)는 요청의 날짜 및 시간과 같이 액세스 제한의 기준이 되는 구체적인 특성입니다.

조건과 키를 함께 사용하여 제한을 표현합니다. 예를 들어, 위임 발신자가 2019년 7월 30일 이후로는 사용자를 대신하여 Amazon SES에 요청을 하지 못하게 제한하려면 `DateLessThan` 조건을 사용합니다. `aws:CurrentTime`이라는 키를 사용하여 그 값을 `2019-07-30T00:00:00Z`로 설정합니다.

IAM 사용 설명서의 사용 가능한 키에 나열된 AWS전체 키를 사용하거나 권한 부여 정책을 전송하는 데 유용한 SES와 관련된 다음 키 중 하나를 사용할 수 있습니다.

조건 키	설명
`ses:Recipients`	To:, "CC" 및 "BCC" 주소가 포함된 수신자 주소를 제한합니다.

조건 키	설명
`ses:FromAddress`	"From" 주소를 제한합니다.
`ses:FromDisplayName`	"From" 표시 이름(때때로 "대화명"으로 부름)으로 사용되는 문자열의 내용을 제한합니다. 예를 들어 "John Doe <johndoe@example.com>"의 표시 이름은 John Doe입니다.
`ses:FeedbackAddress`	이메일 피드백 전달로 반송 메일과 수신 거부를 전송할 수 있는 주소인 "Return Path" 주소를 제한합니다. 이메일 피드백 전달에 대한 자세한 내용은 이메일을 통해 Amazon SES 알림 수신 단원을 참조하세요.

Amazon SES 키와 함께 `StringEquals` 및 `StringLike` 조건을 사용할 수 있습니다. 이러한 조건은 대/소문자 구분 문자열 매칭을 위한 것입니다. `StringLike`의 경우 문자열 어디에서나 다중 문자 매칭 와일드카드(*) 또는 단일 문자 매칭 와일드카드(?)를 값에 포함할 수 있습니다. 예를 들어 다음 조건은 위임 발신자가 invoicing으로 시작하고 @example.com으로 끝나는 "From" 주소에서만 이메일을 전송할 수 있게 지정합니다.

```
"Condition": {
    "StringLike": {
      "ses:FromAddress": "invoicing*@example.com"
    }
}
```

또한 위임 발신자가 특정 이메일 주소에서 이메일을 전송하지 못하도록 `StringNotLike` 조건을 사용할 수 있습니다. 예를 들어, 정책 설명문에 다음 조건을 포함하여 admin@example.com과 "admin"@example.com, admin+1@example.com, sender@admin.example.com 등의 유사 주소에서 전송을 허용하지 않을 수 있습니다.

```
"Condition": {
    "StringNotLike": {
      "ses:FromAddress": "*admin*example.com"
    }
 }
```

조건 지정 방법에 대한 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하십시오.

위임 발신자 지정

권한을 부여하려는 엔터티인 보안 주체는 AWS 계정, AWS Identity and Access Management (IAM) 사용자 또는 AWS 서비스일 수 있습니다.

다음 예제는 AWS ID 1234567890 12이 확인된 자격 증명 example.com(888888888888에서 소유 AWS 계정)에서 이메일을 보내도록 허용하는 간단한 정책을 보여줍니다. 이 정책의 `Condition` 문은 위임자(즉, AWS ID 123456789012)가 주소 marketing+.*@example.com에서 이메일을 보내도록 허용합니다. 여기서 *는 marketing+ 이후에 발신자가 추가하려는 문자열입니다.

JSON

```
{
  "Id":"SampleAuthorizationPolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeMarketer",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:888888888888:identity/example.com",
      "Principal":{
        "AWS":[
          "123456789012"
        ]
      },
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Condition":{
        "StringLike":{
          "ses:FromAddress":"marketing+.*@example.com"
        }
      }
    }
  ]
}
```

다음 정책 예제는 두 IAM 사용자에게 자격 증명 example.com에서 이메일을 전송할 수 있는 권한을 부여합니다. IAM 사용자는 Amazon Resource Name(ARN)으로 지정됩니다.

JSON

```
{
  "Id":"ExampleAuthorizationPolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeIAMUser",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:888888888888:identity/example.com",
      "Principal":{
        "AWS":[
          "arn:aws:iam::111122223333:user/John",
          "arn:aws:iam::444455556666:user/Jane"
        ]
      },
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ]
    }
  ]
}
```

다음 정책 예제는 Amazon Cognito에게 자격 증명 example.com에서 이메일을 전송할 수 있는 권한을 부여합니다.

JSON

```
{
  "Id":"ExampleAuthorizationPolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeService",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:888888888888:identity/example.com",
      "Principal":{
        "Service":[
          "cognito-idp.amazonaws.com"
```

```
        ]
      },
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "888888888888",
          "aws:SourceArn": "arn:aws:cognito-idp:us-east-1:888888888888:userpool/
your-user-pool-id-goes-here"
        }
      }
    }
  ]
}
```

다음 정책 예제는 AWS Organizations에게 자격 증명 example.com에서 이메일을 전송할 수 있는 권한을 부여합니다. AWS 조직은 PrincipalOrgID 전역 조건 키를 사용하여 지정됩니다.

JSON

```
{
  "Id":"ExampleAuthorizationPolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeOrg",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:888888888888:identity/example.com",
      "Principal":"*",
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Condition":{
        "StringEquals":{
          "aws:PrincipalOrgID":"o-xxxxxxxxxxx"
        }
      }
    }
```

```
  ]
}
```

"From" 주소 제한

확인된 도메인을 사용하면 위임 발신자만 지정된 이메일 주소에서 전송하도록 허용하는 정책을 생성할 수 있습니다. "From" 주소를 제한하려면 ses:FromAddress 키에서 조건을 설정합니다. 다음 정책은 AWS 계정 ID 123456789012가 자격 증명 example.com, 그러나 이메일 주소 sender@example.com에서만 전송하도록 허용합니다.

JSON

```
{
  "Id":"ExamplePolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeFromAddress",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:888888888888:identity/example.com",
      "Principal":{
        "AWS":[
          "123456789012"
        ]
      },
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Condition":{
        "StringEquals":{
          "ses:FromAddress":"sender@example.com"
        }
      }
    }
  ]
}
```

위임자가 이메일을 전송할 수 있는 시간 제한

위임 발신자가 특정 시간대 또는 특정 날짜 범위 내에서만 이메일을 전송할 수 있도록 발신자 권한 부여 정책을 구성할 수도 있습니다. 예를 들어 이메일 캠페인 전송이 2021년 9월 한 달 동안 예정되어 있는 경우 다음 정책을 사용하여 위임자가 해당 월에만 이메일을 전송할 수 있도록 제한할 수 있습니다.

JSON

```
{
  "Id":"ExamplePolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"ControlTimePeriod",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:888888888888:identity/example.com",
      "Principal":{
        "AWS":[
          "123456789012"
        ]
      },
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Condition":{
        "DateGreaterThan":{
          "aws:CurrentTime":"2021-08-31T12:00Z"
        },
        "DateLessThan":{
          "aws:CurrentTime":"2021-10-01T12:00Z"
        }
      }
    }
  ]
}
```

이메일 전송 작업 제한

발신자가 Amazon SES에서 이메일을 전송할 수 있는 작업은 발신자가 이메일의 형식을 어느 정도 제어하기를 원하는가에 따라 `SendEmail` 및 `SendRawEmail` 두 가지가 있습니다. 전송 권한 부여 정책

을 통해 위임 발신자가 두 작업 중 하나만 사용하도록 제한할 수 있습니다. 하지만 많은 자격 증명 소유자는 이메일 전송 호출의 세부 정보를 위임 발신자가 선택할 수 있도록 정책에서 두 작업을 모두 활성화합니다.

> **Note**
>
> 위임 발신자가 SMTP 인터페이스를 통해 Amazon SES에 액세스하도록 허용하려면 최소한 `SendRawEmail`을(를) 선택해야 합니다.

작업을 제한하려는 사용 사례라면 전송 권한 부여 정책에 작업 중 하나만 포함시킵니다. 다음 예제는 작업을 `SendRawEmail`로 제한하는 방법을 보여줍니다.

JSON

```
{
  "Id":"ExamplePolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"ControlAction",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:888888888888:identity/example.com",
      "Principal":{
        "AWS":[
          "123456789012"
        ]
      },
      "Action":[
        "ses:SendRawEmail"
      ]
    }
  ]
}
```

이메일 발신자의 표시 이름 제한

일부 이메일 클라이언트는 실제 "From" 주소가 아니라 이메일 발신자의 "대화명" 이름을 표시합니다(이메일 헤더가 제공하는 경우). 예를 들어 "John Doe <johndoe@example.com>"의 표시 이름은 John Doe입니다. 예를 들어, user@example.com에서 이메일을 보낼 수 있지만 수신자에게

user@example.com이 아니라 Marketing에서 발송된 것으로 보여지기를 원합니다. 다음 정책은 AWS 계정 ID 123456789012가 ID example.com 전송하도록 허용하지만 "From" 주소의 표시 이름에 Marketing이 포함된 경우에만 가능합니다.

JSON

```
{
  "Id":"ExamplePolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeFromAddress",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:888888888888:identity/example.com",
      "Principal":{
        "AWS":[
          "123456789012"
        ]
      },
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Condition":{
        "StringLike":{
          "ses:FromDisplayName":"Marketing"
        }
      }
    }
  ]
}
```

복수의 설명문 사용

전송 권한 부여 정책에는 여러 문이 포함될 수 있습니다. 다음의 정책 예제에는 문이 2개입니다. 첫 번째 문은 "From" 주소와 피드백 주소가 모두 도메인 example.com 사용하는 한 두 AWS 계정 명이 sender@example.com에서 전송할 수 있는 권한을 부여합니다. 두 번째 설명문은 수신자의 이메일 주소가 example.com 도메인에 속하는한 IAM 사용자가 sender@example.com에서 이메일을 전송하도록 권한을 부여합니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeAWS",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:999999999999:identity/
sender@example.com",
      "Principal":{
        "AWS":[
          "111111111111",
          "222222222222"
        ]
      },
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Condition":{
        "StringLike":{
          "ses:FromAddress":"*@example.com",
          "ses:FeedbackAddress":"*@example.com"
        }
      }
    },
    {
      "Sid":"AuthorizeInternal",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:999999999999:identity/
sender@example.com",
      "Principal":{
        "AWS":"arn:aws:iam::333333333333:user/Jane"
      },
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Condition":{
        "ForAllValues:StringLike":{
          "ses:Recipients":"*@example.com"
        }
```

```
        }
      }
    ]
  }
```

Amazon SES 전송 권한 부여에서 위임 발신자에게 자격 증명 정보 제공

전송 권한 부여 정책을 생성하여 자격 증명에 연결한 후, 위임 발신자에게 자격 증명의 Amazon 리소스 이름(ARN)을 제공해야 합니다. 위임 발신자는 이메일 전송 작업 또는 이메일 헤더에서 이 ARN을 Amazon SES로 전달합니다. 자격 증명의 ARN 찾으려면 다음 단계를 따릅니다.

자격 증명의 ARN을 찾으려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Verified identities(확인된 ID)를 선택합니다.
3. 자격 증명 목록에서 전송 권한 부여 정책에 연결한 자격 증명을 선택합니다.
4. 요약(Summary) 창에서, 두 번째 열, Amazon 리소스 이름(ARN)(Amazon Resource Name(ARN))에는 자격 증명의 ARN이 포함됩니다. 예를 들면 ARN은 arn:aws:ses:us-east-1:123456789012:identity/user@example.com과 같습니다. 전체 ARN을 복사하여 위임 발신자에게 제공합니다.

> **Important**
>
> 권한을 부여하려는 자격 증명이 글로벌 엔드포인트 기능의 일부로 보조 리전에서 복제되는 경우와 같은 리전 파라미터를 다음 예제 `us-east-1`와 `*` 같이 별표로 바꿉니다`arn:aws:ses:*:123456789012:identity/user@example.com`.

Amazon SES 전송 권한 부여를 위한 위임 발신자 작업

위임 발신자로서 귀하는 소유하지는 않지만 사용 권한을 부여받은 자격 증명을 대신하여 이메일을 전송합니다. 자격 증명 소유자를 대신하여 전송하는 경우에도 반송 메일 및 수신 거부는 AWS 계정의 반송 메일 및 수신 거부 지표에 포함되고 전송하는 메시지 수는 전송 할당량에 포함됩니다. 또한 자격 증명 소유자의 이메일을 전송하는 데 필요하다면 발신 할당량 증가를 요청할 책임이 있습니다.

위임 발신자는 다음 작업을 완료해야 합니다.

- 자격 증명 소유자에게 정보 제공
- 위임 발신자 알람 사용
- 자격 증명 소유자를 대신하여 이메일 전송

Amazon SES 전송 권한 부여에서 자격 증명 소유자에게 정보 제공

위임 발신자는 자격 증명 소유자를 대신하여 이메일을 보내게 되므로 자격 증명 소유자에게 AWS 계정 ID 또는 IAM 사용자 Amazon 리소스 이름(ARN)을 제공해야 합니다. 자격 증명 소유자는 확인된 자격 증명 중 하나에서 발신할 권한을 부여하는 정책을 만들기 위해 계정 정보가 필요합니다.

자체 SNS 주제를 사용하려는 경우 자격 증명 소유자가 반송 메일, 수신 거부 또는 배달에 대한 피드백 알림을 구성하여 하나 이상의 SNS 주제로 전송하도록 요청할 수 있습니다. 이렇게 하려면, 자격 증명 소유자가 발신 권한을 부여한 확인된 자격 증명에서 SNS 주제를 구성할 수 있도록 SNS 주제 ARN을 자격 증명 소유자와 공유해야 합니다.

다음 절차에서는 자격 증명 소유자와 공유할 계정 정보 및 SNS 주제 ARN을 찾는 방법에 대해 설명합니다.

AWS 계정 ID를 찾으려면

1. https://console.aws.amazon.com AWS Management Console 에 로그인합니다.
2. 콘솔의 오른쪽 상단에서 계정 이름/계정 번호를 확장한 후 드롭다운에서 내 계정(My Account)을 선택합니다.
3. 계정 설정 페이지가 열리고 계정 ID를 포함한 모든 AWS 계정 정보가 표시됩니다.

IAM 사용자 ARN을 찾으려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/iam/ IAM 콘솔을 엽니다.
2. 탐색 창에서 사용자를 선택합니다.
3. 사용자 목록에서 사용자 이름을 선택합니다. 요약(Summary) 섹션에 IAM 사용자 ARN이 표시됩니다. ARN은 arn:aws:iam::123456789012:user/John과 같습니다.

SNS 주제 ARN을 찾으려면

1. https://console.aws.amazon.com/sns/v3/home에서 Amazon SNS 콘솔을 엽니다.

2. 탐색 창에서 주제를 선택합니다.
3. 주제 목록의 ARN열에 SNS 주제 ARN이 표시됩니다. ARN은 arn:aws:sns:us-east-1:444455556666:my-sns-topic과 유사합니다.

Amazon SES 전송 권한 부여에서 위임 발신자 알림 사용

위임 발신자는 소유하지는 않지만 사용 권한을 부여받은 자격 증명을 대신하여 이메일을 전송하지만 반송 메일 및 수신 거부는 여전히 자격 증명 소유자가 아니라 귀하의 반송 메일 및 수신 거부 지표에 가산됩니다.

계정에 대한 반송 메일 또는 수신 거부 발생률이 너무 높으면 계정이 검토 대상이 되거나 계정의 이메일 전송 기능이 일시 중지될 위험이 있습니다. 따라서 알림을 설정하고 이를 모니터링하는 프로세스를 수립해야 합니다. 또한 반송했거나 수신 거부한 주소를 메일 그룹에서 제거하는 프로세스를 갖춰야 합니다.

따라서 위임 발신자는 소유하지는 않지만 자격 증명 소유자로부터 사용할 수 있도록 권한을 부여받은 자격 증명을 대신하여 보내는 이메일에 대해 반송 메일 및 수신 거부 이벤트가 발생할 때 알림을 보내도록 Amazon SES를 구성할 수 있습니다. 또한, 반송 메일 및 수신 거부 알림을 Amazon SNS 또는 Firehose에 게시하기 위해 이벤트 게시를 설정할 수 있습니다.

> **Note**
>
> Amazon SNS를 사용하여 알림을 보내도록 Amazon SES를 설정한 경우 수신하는 알림에 대해 표준 Amazon SNS 요금이 청구됩니다. 자세한 내용은 Amazon SNS 요금 페이지를 참조하십시오.

새 위임 발신자 알림 생성

이벤트 게시를 사용하는 구성 세트 또는 자체 SNS 주제로 구성된 확인된 자격 증명으로 위임 전송 알림을 설정할 수 있습니다.

두 방법 중 하나를 사용하여 새 위임 전송 알림을 설정하는 절차는 아래에 나와 있습니다.

- 구성 세트를 통한 이벤트 게시
- 소유한 SNS 주제에 대한 피드백 알림

위임 전송을 위해 구성 세트를 통한 이벤트 게시를 설정하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 이벤트 대상 만들기의 절차를 따르십시오.
3. 구성 세트에서 이벤트 게시를 설정한 후, 자격 증명 소유자가 귀하에게 전송하도록 권한을 부여한 확인된 자격 증명을 사용하여 위임 발신자로서 이메일을 보낼 때 구성 세트의 이름을 지정합니다. 자격 증명 소유자를 대신하여 이메일 전송을(를) 참조하세요.

위임 전송을 위해 소유한 SNS 주제에 대한 피드백 알림을 설정하려면

1. 피드백 알림에 사용할 SNS 주제를 결정한 후 해당 절차에 따라 SNS 주제 ARN을 찾고 전체 ARN을 복사하여 자격 증명 소유자와 공유합니다.
2. 자격 증명 소유자에게 귀하가 전송에 사용하도록 권한을 부여한 공유 자격 증명에 대한 피드백 알림을 위한 SNS 주제를 구성해 달라고 요청합니다. (자격 증명 소유자는 권한 부여 정책 절차에 명시된 SNS 주제 구성 절차를 따라야 합니다.)

Amazon SES 전송 권한 부여에서 자격 증명 소유자를 대신하여 이메일 전송

위임 발신자가 이메일을 전송하는 방식은 자격 증명 소유자로부터 사용 권한을 부여받은 자격 증명의 Amazon Resource Name(ARN)을 제공한다는 점을 제외하면 다른 Amazon SES 발신자와 동일합니다. 위임 발신자가 이메일 전송을 위해 Amazon SES를 호출할 때 Amazon SES는 위임 발신자가 지정한 자격 증명이 해당 위임 발신자에게 자신을 대신하여 이메일을 전송하도록 권한을 부여하는 정책을 포함하는지 확인합니다.

이메일을 전송할 때 자격 증명의 ARN을 지정하는 방법은 여러 가지가 있습니다. 사용하는 방법은 이메일을 Amazon SES API 작업 또는 Amazon SES SMTP 인터페이스를 사용하여 전송하는지 여부에 따라 결정됩니다.

Important

- 이메일을 성공적으로 보내려면 자격 증명 소유자가 자격 증명을 확인한 AWS 리전의 Amazon SES 엔드포인트에 연결해야 합니다.
- 또한 자격 증명 소유자와 위임 발신자 모두의 AWS 계정을 샌드박스에서 제거해야 두 계정 중 하나가 확인되지 않은 주소로 이메일을 보낼 수 있습니다. 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하십시오.

- 사용 권한이 부여된 자격 증명이 글로벌 엔드포인트 기능의 일부로 보조 리전에서 복제되는 경우:
  - 자격 증명 소유자는 다음 예와 * 같이와 같은 리전 파라미터가 별표로 us-east-1대체된 자격 증명 ARN을 제공했어야 합니다arn:aws:ses:*:123456789012:identity/user@example.com.
  - 자격 증명 소유자는 기본 리전과 보조 리전 모두에서 전송 권한 부여 정책을 생성했어야 합니다.

Amazon SES API 사용

모든 Amazon SES 이메일 발신자와 마찬가지로 Amazon SES API를 통해(HTTPS를 통해 직접 또는 AWS SDK를 통해 간접적으로) Amazon SES에 액세스하는 경우 SendEmail, 및 SendTemplatedEmail의 세 가지 이메일 전송 작업 중 하나를 선택할 수 있습니다SendRawEmail. Amazon Simple Email Service API 참조에서는 이러한 API를 상세히 설명하고 있지만, 여기서는 전송 권한 부여 파라미터의 개요만 설명합니다.

SendRawEmail

이메일의 형식을 제어할 수 있도록 SendRawEmail을(를) 사용하려는 경우 다음 두 방법 중 하나로 권한이 부여된 위임 자격 증명을 지정할 수 있습니다.

- 옵션 파라미터를 **SendRawEmail** API로 전달합니다. 필수 파라미터는 다음 표에 설명되어 있습니다.

파라미터	설명
SourceArn	SendRawEmail 의 Source 파라미터에 지정된 이메일 주소에 대해 이메일을 전송하도록 허용하는 전송 권한 부여 정책과 연결된 자격 증명의 ARN. **Note** SourceArn 만 지정하면 Amazon SES 가 "From" 주소와 "Return Path" 주소를 SourceArn 에 지정된 자격 증명으로 설정합니다.

파라미터	설명
`FromArn`	위임 발신자에게 원시 이메일의 헤더에 특정 "From" 주소를 지정하도록 허용하는 전송 권한 부여 정책과 연결된 자격 증명의 ARN.
`ReturnPathArn`	`SendRawEmail` 의 `ReturnPath` 파라미터에 지정된 이메일 주소를 사용하도록 허용하는 전송 권한 부여 정책과 연결된 자격 증명의 ARN.

- 이메일에 X-헤더를 포함합니다. X-헤더는 표준 이메일 헤더(예: From, Reply-To 또는 Subject 헤더) 외에 사용할 수 있는 사용자 지정 헤더입니다. Amazon SES는 전송 권한 부여 파라미터를 지정하는 데 사용할 수 있는 세 개의 X-헤더를 인식합니다.

> **Important**
>
> DKIM 서명에는 이러한 X-헤더를 포함하지 마십시오. Amazon SES가 이메일을 전송하기 전에 이들 헤더를 제거합니다.

X-헤더	설명
`X-SES-SOURCE-ARN`	`SourceArn` 에 해당합니다.
`X-SES-FROM-ARN`	`FromArn`에 해당합니다.
`X-SES-RETURN-PATH-ARN`	`ReturnPathArn` 에 해당합니다.

Amazon SES는 이메일을 전송하기 전에 해당 이메일에서 모든 X-헤더를 제거합니다. X-헤더의 여러 인스턴스를 포함하는 경우 Amazon SES는 첫 번째 인스턴스만 사용합니다.

다음 예제는 전송 권한 부여 X-헤더를 포함하는 이메일입니다.

```
X-SES-SOURCE-ARN: arn:aws:ses:us-east-1:123456789012:identity/example.com
X-SES-FROM-ARN: arn:aws:ses:us-east-1:123456789012:identity/example.com
X-SES-RETURN-PATH-ARN: arn:aws:ses:us-east-1:123456789012:identity/example.com

From: sender@example.com
```

```
To: recipient@example.com
Return-Path: feedback@example.com
Subject: subject
Content-Type: multipart/alternative;
 boundary="----=_boundary"

------=_boundary
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 7bit

body
------=_boundary
Content-Type: text/html; charset=UTF-8
Content-Transfer-Encoding: 7bit

body
------=_boundary--
```

SendEmail 및 SendTemplatedEmail

`SendEmail` 또는 `SendTemplatedEmail` 작업을 사용하는 경우 아래의 선택적 파라미터를 전달하여 권한이 부여된 위임 자격 증명을 지정할 수 있습니다. `SendEmail` 또는 `SendTemplatedEmail` 작업을 사용하면 X-헤더 방법을 사용할 수 없습니다.

파라미터	설명
`SourceArn`	`SendEmail` 또는 `SendTemplatedEmail` 의 `Source` 파라미터에 지정된 이메일 주소에 대해 이메일을 전송하도록 허용하는 전송 권한 부여 정책과 연결된 자격 증명의 ARN.
`ReturnPathArn`	`SendEmail` 또는 `SendTemplatedEmail` 의 `ReturnPath` 파라미터에 지정된 이메일 주소를 사용하도록 허용하는 전송 권한 부여 정책과 연결된 자격 증명의 ARN.

다음 예제에서는 `SendEmail` 또는 `SendTemplatedEmail` 작업과 SDK for Python을 사용하여 `SourceArn` 및 `ReturnPathArn` 속성이 포함된 이메일을 보내는 방법을 보여줍니다.

```
import boto3
from botocore.exceptions import ClientError

# Create a new SES resource and specify a region.
client = boto3.client('ses',region_name="us-east-1")

# Try to send the email.
try:
    #Provide the contents of the email.
    response = client.send_email(
        Destination={
            'ToAddresses': [
                'recipient@example.com',
            ],
        },
        Message={
            'Body': {
                'Html': {
                    'Charset': 'UTF-8',
                    'Data': 'This email was sent with Amazon SES.',
                },
            },
            'Subject': {
                'Charset': 'UTF-8',
                'Data': 'Amazon SES Test',
            },
        },
        SourceArn='arn:aws:ses:us-east-1:123456789012:identity/example.com',
        ReturnPathArn='arn:aws:ses:us-east-1:123456789012:identity/example.com',
        Source='sender@example.com',
        ReturnPath='feedback@example.com'
    )
# Display an error if something goes wrong.
except ClientError as e:
    print(e.response['Error']['Message'])
else:
    print("Email sent! Message ID:"),
    print(response['ResponseMetadata']['RequestId'])
```

Amazon SES SMTP 인터페이스 사용

위임 전송을 위해 Amazon SES SMTP 인터페이스를 사용하는 경우 메시지에 `X-SES-SOURCE-ARN`, `X-SES-FROM-ARN` 및 `X-SES-RETURN-PATH-ARN` 헤더를 포함해야 합니다. SMTP 대화에서 `DATA` 명령을 발급한 후에 이 헤더를 전달하세요.

시뮬레이터를 사용하여 Amazon SES에서 테스트 이메일 전송

Amazon SES 콘솔을 사용하여 Amazon SES에서 테스트 이메일을 보내는 것이 좋습니다. 콘솔에서는 사용자가 정보를 수동으로 입력해야 하므로 일반적으로 테스트 이메일을 보낼 때만 사용합니다. 일단 Amazon SES를 사용하기 시작하면 Amazon SES SMTP 인터페이스나 API를 사용하여 대부분의 이메일을 전송할 것입니다. 하지만 전송 활동을 모니터링하기에는 콘솔이 유용합니다.

다음 주제에서는 콘솔에서 메일박스 시뮬레이터를 사용하는 방법과 이메일을 전송하여 수동으로 사용하는 방법을 살펴봅니다.

- 콘솔에서 메일박스 시뮬레이터 사용
- 수동으로 메일박스 시뮬레이터 사용

콘솔에서 메일박스 시뮬레이터 사용

> **Important**
>
> - 이 자습서에서는 수신 여부를 확인할 수 있도록 콘솔에서 자신에게 이메일을 발송합니다. 추가적인 실험 또는 로드 테스트는 수동으로 메일박스 시뮬레이터 사용를 사용하세요.
> - 메일박스 시뮬레이터로 전송되는 이메일은 전송 할당량이나 반송 메일 및 수신 거부 발생률에 포함되지 않으며, 가상 전달 가능성 관리자 지표에도 영향을 주지 않습니다.

이러한 단계를 따르기 전에 Amazon Simple Email Service 설정에서 다음 작업을 완료합니다.

Amazon SES 콘솔에서 이메일 메시지 전송

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성에서 자격 증명을 선택합니다.

3. 자격 증명 표에서 확인된 이메일 자격 증명을 선택합니다(확인란을 선택하는 대신 자격 증명 이름을 직접 클릭). 확인된 이메일 자격 증명이 없는 경우 이메일 주소 자격 증명 생성 섹션을 참조하세요.
4. 선택한 이메일 자격 증명 세부 정보 페이지에서 테스트 이메일 보내기(Send test email)를 선택합니다.
5. Message details(메시지 세부 정보)에서 Email Format(이메일 형식)을 선택합니다. 다음과 같이 두 가지 옵션이 있습니다.
   - Formatted(서식 지정)—가장 단순한 옵션입니다. Body 텍스트 상자에 메시지 텍스트를 입력하려면 이 옵션을 선택합니다. 이메일을 전송할 때 Amazon SES가 사용자 대신 텍스트를 이메일 형식으로 전환합니다.
   - Raw(원시)—보다 복잡한 메시지(예: HTML 또는 첨부 파일을 포함하는 메시지)를 전송하려면 이 옵션을 선택합니다. 이러한 유연성 때문에, 사용자가 Amazon SES API v2를 사용하여 원시 이메일 보내기에서 설명된 대로 메시지를 서식 지정한 후, 헤더를 포함하여 서식 지정된 메시지 전체를 Body 텍스트 상자에 붙여 넣어야 합니다. HTML을 포함하는 다음 예제를 사용하여 Raw 이메일 형식으로 테스트 이메일을 전송할 수 있습니다. 이 메시지를 전부 복사하여 Body 텍스트 상자에 붙여 넣습니다. `MIME-Version` 헤더와 `Content-Type` 헤더 사이에 빈 줄이 있으면 안 됩니다. 빈 줄이 있으면 이메일이 HTML이 아니라 일반 텍스트로 서식 지정됩니다.

```
Subject: Amazon SES Raw Email Test
MIME-Version: 1.0
Content-Type: text/html

<!DOCTYPE html>
<html>
<body>
<h1>This text should be large, because it is formatted as a header in HTML.</h1>
<p>Here is a formatted link: <a href="https://docs.aws.amazon.com/ses/latest/
DeveloperGuide/Welcome.html">Amazon Simple Email Service Developer Guide</a>.</p>
</body>
</html>
```

6. 시나리오 목록 상자를 펼쳐 테스트할 시뮬레이션된 이메일 시나리오 유형을 선택합니다.
   - 사용자 지정을 선택했고 계속 Amazon SES 샌드박스 환경에 있는 경우 사용자 지정 수신자 필드의 주소가 확인된 이메일 주소여야 합니다. 자세한 내용은 이메일 주소 자격 증명 생성 단원을 참조하십시오.
7. 나머지 필드를 원하는 대로 채웁니다.

8. 테스트 이메일 보내기(Send a Test Email)를 선택합니다.
9. 이메일 수신 주소의 이메일 클라이언트에 로그인합니다. 보낸 메시지가 도착해 있을 것입니다.

수동으로 메일박스 시뮬레이터 사용

Amazon SES에는 애플리케이션이 여러 이메일 전송 시나리오를 어떻게 처리하는지 테스트하는 데 사용할 수 있는 메일박스 시뮬레이터가 포함되어 있습니다. 메일박스 시뮬레이터는 예를 들면 허구의 이메일 주소를 만들지 않고 이메일 전송 애플리케이션을 테스트하거나, 일일 발신 할당량에 영향을 주지 않고 시스템의 최대 처리량을 찾고 싶을 때 유용합니다.

중요 고려 사항

Amazon SES 메일박스 시뮬레이터를 사용할 때는 다음과 같은 기능과 한계를 고려하세요.

- Amazon SES 샌드박스 환경에서도 메일박스 시뮬레이터를 사용할 수 있습니다.
- 메일박스 시뮬레이터로 보내는 이메일은 계정의 최대 전송 속도에 의해 제한되지만, 일일 발신 할당량에 영향을 주지 않습니다. 예를 들어 24시간 주기로 10,000개의 메시지를 발송하기로 승인받은 계정에서 메일박스 시뮬레이터에 100개의 메시지를 발송한 경우, 발신 할당량을 초과할 걱정 없이 일반 수신자에게 최대 10,000개의 메시지를 보낼 수 있습니다.
- 메일박스 시뮬레이터에 보내는 이메일은 이메일 배달 가능성이나 평판 지표에 영향을 주지 않습니다. 예를 들어 이메일 시뮬레이터의 반송 메일 주소로 많은 메시지를 보내도 평판 지표 콘솔 페이지에 반송률이 너무 높다는 메시지 경고가 표시되지 않습니다.
- 결제를 목적으로 Amazon SES 메일박스 시뮬레이터로 보내는 이메일은 Amazon SES를 사용하여 보내는 다른 이메일과 동일합니다. 다시 말해 메일박스 시뮬레이터에 보내는 메시지에 대해 일반 수신자에게 보낼 때와 동일하게 청구합니다.
- 메일박스 시뮬레이터는 레이블 지정을 지원합니다. 레이블 지정을 통해 동일한 메일박스 시뮬레이터 주소에 다양한 방식으로 이메일을 보내거나 애플리케이션이 VERP(Variable Envelope Return Path)를 어떻게 지원하는지 알아볼 수 있습니다. 예를 들어 bounce+label1@simulator.amazonses.com과 bounce+label2@simulator.amazonses.com으로 이메일을 보내 애플리케이션이 반송 메일 메시지와 반송 메일을 초래한 이메일 주소에 연결되는지 확인할 수 있습니다.
- 메일박스 시뮬레이터를 사용해 동일한 발신 요청에서 온 여러 개의 반송 메일을 시뮬레이션하면 Amazon SES가 반송 응답을 단일 응답으로 결합합니다.

메일박스 시뮬레이터 사용

이메일 시뮬레이터를 사용하려면 다음 표에서 시나리오를 찾은 후, 그에 해당하는 이메일 주소로 이메일을 보내세요.

> **Note**
>
> 메일박스 시뮬레이터 주소로 이메일을 보낼 때는 , AWS CLI AWS SDK, Amazon SES 콘솔, Amazon SES SMTP 인터페이스 또는 Amazon SES API를 사용하여 Amazon SES를 통해 이메일을 보내야 합니다. 메일박스 시뮬레이터는 외부 소스로부터 받은 이메일에는 응답하지 않습니다.

시뮬레이션된 시나리오	이메일 주소
성공적인 전송—수신자의 이메일 공급자가 이메일을 수락합니다. Amazon SES에 대한 이벤트 알림 설정에서 설명한 바와 같이 전송 알림을 설정한 경우 Amazon SES가 Amazon Simple Notification Service(Amazon SNS)를 통해 전송 알림을 전송합니다.	success@simulator.amazonses.com
반송 메일—수신자의 이메일 공급자가 SMTP 550 5.1.1(“알 수 없는 사용자”) 응답 코드를 사용하여 이메일을 거부합니다. Amazon SES는 반송 메일 알림을 생성하고 계정 설정 방식에 따라 이메일로 사용자에게 보내거나 Amazon SNS 주제로 알림을 보냅니다. 메일박스 시뮬레이터 이메일 주소는 이메일 하드 바운스가 발생할 경우에 보통 그런 것처럼 Amazon SES 금지 목록에 추가되지 않습니다. 메일박스 시뮬레이터에서 받는 반송 메일 응답은 RFC 3464와 호환됩니다. 반송 메일 피드백을 받는 방법에 대한 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하세요.	bounce@simulator.amazonses.com
자동 응답—수신자의 이메일 공급자가 이메일을 수락하고 수신자의 받은 편지함으로 전송합	ooto@simulator.amazonses.com

시뮬레이션된 시나리오	이메일 주소
니다. 이메일 공급자가 이메일의 Return-Path 헤더에 있는 주소 또는 Return-Path 헤더가 없을 경우 envelope sender("MAIL FROM") 주소로 OOTO("out of the office") 메시지 같은 자동 응답을 보냅니다. 메일박스 시뮬레이터에서 받는 자동 응답은 RFC 3834와 호환됩니다.	
수신 거부—수신자의 이메일 공급자가 이메일을 수락하고 수신자의 받은 편지함으로 전송합니다. 수신자가 원치 않는 메시지라고 결정하고 자신의 이메일 클라이언트에서 '스팸으로 표시'를 클릭합니다. 그러면 Amazon SES가 계정 설정 방식에 따라 이메일 또는 Amazon SNS 주제 알림을 통해 수신 거부 알림을 전달합니다. 메일박스 시뮬레이터에서 받는 수신 거부 응답은 RFC 5965와 호환됩니다. 수신 거부 피드백을 받는 방법에 대한 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하세요.	complaint@simulator.amazonses.com
금지 목록의 수신자 주소—수신자의 주소가 금지 목록에 있는 것처럼 Amazon SES 에서 하드 바운스를 생성합니다.	suppressionlist@simulator.amazonses.com

거부 이벤트 테스트

Amazon SES를 통해 전송되는 모든 메시지는 바이러스 검사됩니다. 바이러스가 포함된 메시지를 보내는 경우 Amazon SES는 메시지를 수락하고 바이러스를 감지한 다음 전체 메시지를 거부합니다. 메시지가 거부되면 Amazon SES는 메시지 처리를 중지하고 해당 메시지를 수신자의 메일 서버로 전송하지 않습니다. 그리고 나서 거부 이벤트를 생성합니다.

Amazon SES 메일박스 시뮬레이터에는 거부 이벤트를 테스트하기 위한 주소가 포함되지 않습니다. 그러나 유럽 컴퓨터 바이러스 백신 연구소(EICAR) 테스트 파일을 사용하여 수신 거부 이벤트를 테스트할 수 있습니다. 이 파일은 안전한 방법으로 바이러스 백신 소프트웨어를 테스트하는 업계 표준 방법입니다. EICAR 테스트 파일을 생성하려면 다음 텍스트를 파일에 붙여 넣습니다.

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

파일을 `sample.txt`(으)로 저장하고 이메일에 첨부한 다음 확인된 주소로 이메일을 보냅니다. 이메일에 다른 문제가 없는 경우 Amazon SES는 메시지를 수락하지만 실제 바이러스가 포함된 경우와 마찬가지로 메시지를 거부합니다.

> **Note**
>
> 위의 절차를 사용하여 보낸 이메일이 포함된 거부된 이메일은 일일 전송 할당량에 불리하게 작용합니다. 거부된 메시지를 포함하여 보내는 각 메시지에 대해 요금이 청구됩니다.

EICAR 테스트 파일에 대해 자세히 알아보려면 Wikipedia의 EICAR 테스트 파일 페이지를 참조하세요.

Amazon SES에서 구성 세트 사용

구성 세트는 확인된 ID에 적용할 수 있는 규칙 그룹입니다. ID는 Amazon SES를 통해 이메일을 보낼 때 사용하는 도메인, 서브도메인 또는 이메일 주소입니다. 구성 세트를 이메일에 적용하면 해당 구성 세트의 모든 규칙이 이메일에 적용됩니다.

구성 세트를 사용하여 이메일 전송에 다음 유형의 규칙을 적용할 수 있으며, 이러한 유형을 하나 또는 모두 포함할 수 있고 하나도 포함되지 않을 수도 있습니다.

- 이벤트 대상 - 보내는 각 이메일의 다른 AWS 제품에 대한 전송, 전송, 열기, 클릭, 반송 메일 및 수신 거부 수를 포함한 이메일 전송 지표를 게시할 수 있습니다. 예를 들어 이메일 지표를 Amazon Data Firehose 대상으로 전송한 다음 Amazon Managed Service for Apache Flink를 사용하여 분석할 수 있습니다. 그 밖에 반송 메일 및 수신 거부 정보를 Amazon SNS로 전송한 후 이러한 이벤트가 발생할 때마다 즉시 알림 메시지를 수신할 수도 있습니다.
- IP 풀 관리 – Amazon SES에 사용할 전용 IP 주소를 임차한 경우 특정 유형의 이메일을 보내는 데 사용되는 전용 IP 풀이라는 이러한 주소 그룹을 생성할 수 있습니다. 예를 들어 이러한 전용 IP 풀을 구성 세트와 연결하여 마케팅 커뮤니케이션을 보내는 용도로 하나를 사용하고 거래 이메일을 보내는 용도로 다른 풀을 사용할 수 있습니다. 거래 이메일에 대한 발신자의 평판은 이제 마케팅 이메일에서 격리됩니다.

다음 방법으로 구성 세트를 확인된 ID와 연결할 수 있습니다.

- 이메일의 헤더에 구성 세트에 대한 참조를 포함합니다. 이메일에 구성 세트를 지정하는 방법에 대한 자세한 내용은 이메일을 보낼 때 구성 세트 지정 섹션을 참조하세요.
- ID 생성 시 또는 나중에 확인된 ID를 편집하는 동안 ID의 기본 구성 세트로 사용할 기존 구성 세트를 지정합니다. 기본 구성 세트 이해을(를) 참조하세요.

내용

- SES에서 구성 세트 생성
- Amazon SES에서 구성 세트 관리
- 이메일을 보낼 때 구성 세트 지정
- 평판 지표 보기 및 내보내기

SES에서 구성 세트 생성

SES 콘솔, Amazon SES API v2의 `CreateConfigurationSet` 작업, 또는 Amazon SES CLI v2의 `aws sesv2 create-configuration-set` 명령을 사용하여 새 구성 세트를 생성합니다. 이 섹션에서는 SES 콘솔과 Amazon SES CLI v2를 사용하여 구성 세트를 생성하는 방법을 보여줍니다.

구성 세트(콘솔) 생성

SES 콘솔을 사용하여 구성 세트를 생성하려면, 다음 절차를 따르세요.

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Configuration sets(구성 세트)를 선택합니다.
3. Create set(세트 생성)를 선택합니다.
4. 일반 세부 정보 -이 섹션에서는 구성 세트를 사용자 지정하는 옵션을 제공합니다.
   - 구성 세트 이름 – 구성 세트의 이름입니다. 이름에는 문자, 숫자, 하이픈(-), 밑줄(_)만을 포함하여 최대 64자의 영숫자를 사용할 수 있습니다.
   - IP 풀 전송 - 이 구성 세트를 사용하여 이메일을 보내면 할당된 풀의 전용 IP 주소에서 메시지가 전송됩니다. 목록에서 IP 풀을 선택합니다.

     > **Note**
     >
     > 아직 다른 풀에 할당하지 않은 전용 IP 주소는 기본값(ses-default-dedicated-pool)에 있습니다. IP 풀 관리에 대한 자세한 내용은 IP 풀 할당 섹션을 참조하세요.

   - 추적 옵션
     - 사용자 지정 리디렉션 도메인 사용 - 사용자 지정 리디렉션 도메인 사용 확인란을 선택하여 이 구성 세트에서 보낸 이메일의 열기 및 클릭 추적을 처리합니다.
     - 사용자 지정 리디렉션 도메인 - 사용자 지정 리디렉션 도메인으로 사용할 확인된 도메인 선택 목록에서 확인된 도메인을 선택합니다. 또한 하위 도메인 입력 필드에 하위 도메인을 입력할 수 있습니다.

       > **Note**
       >
       > 사용자 지정 리디렉션 도메인은 다음과 같이 지정할 수 있습니다.

- 먼저 이메일을 전송 및 추적 AWS 리전 하려는에서 사용자 지정 리디렉션 도메인을 생성 및 확인하고 콘텐츠 전송 네트워크(CDN)를 설정해야 합니다. 이는 확인 및 클릭 추적을 처리하기 위한 사용자 지정 도메인 구성 섹션에 설명되어 있습니다.
- 그런 다음 사용자 지정 리디렉션 도메인을 사용하여 이메일을 열고 클릭 추적을 수행하려면 이 단계에서 구성 세트를 만들거나 편집하면서 해당 도메인을 표시해야 합니다.
- 마지막으로 사용자 지정 리디렉션 도메인을 지정하면 구성 세트의 일반 세부 정보 컨테이너에 DNS 레코드 보기가 표시됩니다. 확장하면에서 사용 중인 추적 도메인이 포함된 CNAME 레코드가 표시됩니다 AWS 리전. 예를 들어 사용자 지정 하위 도메인이 marketing.example.com이라고 하고 AWS 리전 `us-east-1`에서 만들어진 경우 DNS 레코드 보기를 펼치면 이름 = marketing.example.com 및 값 = r.us-east-1.awstrack.me 값이 있는 CNAME 레코드가 표시됩니다.

  이 정보를 확인 및 클릭 추적을 처리하기 위한 사용자 지정 도메인 구성 섹션에 설명된 대로 CDN을 설정할 때 테이블에서 올바른 추적 도메인을 선택했다는 확인으로 사용하거나, 먼저 이 작업을 수행하고 여기에서 CNAME 레코드 값을 사용하여 CDN 설정에 사용할 수 있습니다.

- HTTPS 정책 - 열린 프로토콜에 대한 HTTPS 정책 옵션을 선택하고 사용자 지정 리디렉션 도메인에 대한 추적 링크를 클릭합니다.
  - 선택 사항 – (기본 동작) 열기 추적 링크는 HTTP를 사용하여 래핑됩니다. 클릭 추적 링크는 링크의 원본 프로토콜을 사용하여 래핑됩니다.
  - 필수 - 열기 및 클릭 추적 링크는 모두 HTTPS를 사용하여 래핑됩니다.
  - 열기에 필요 - HTTPS를 사용하여 열기 추적 링크가 래핑됩니다. 클릭 추적 링크는 링크의 원본 프로토콜을 사용하여 래핑됩니다.
- 고급 전송 옵션 - 왼쪽의 화살표를 선택하여 고급 전송 옵션 섹션을 펼칩니다.
  - Transport Layer Security(TLS) - SES가 수신 메일 서버와의 보안 연결을 설정하고 TLS 프로토콜을 사용하여 이메일을 보내도록 하려면 필수 확인란을 선택합니다.

> **Note**
>
> SES는 TLS 1.2를 지원하며 TLS 1.3을 권장합니다. 자세한 내용은 SES의 인프라 보안을 참조하세요.

  - 최대 전송 기간 - SES가 이 구성 세트를 통해 이메일 전송을 시도하는 시간 제한을 지정하려면 300~50,400초 범위의 값을 초 단위로 입력합니다.

> **Note**
>
> 사용자 지정 최대 전송 한도(SES 기본값인 14시간보다 짧음)를 설정하면 시간에 영향을 받는 이메일(예: 일회용 암호가 포함된 이메일), 트랜잭션 이메일, 업무 외 시간에 전송되지 않도록 하려는 이메일과 같은 경우에 유용할 수 있습니다.
>
> > **Tip**
> >
> > - 분에서 초로 계산하려면 60을 곱합니다. 예: 7분*60 = 420초
> > - 시간에서 초로 계산하려면 3,600을 곱합니다. 예: 2시간*3,600 = 7,200초.

5. 평판 옵션 -이 섹션에서는 평판 지표를 설정하는 방법을 제공합니다.

   - 평판 지표 - 이 구성 세트를 사용하여 전송된 이메일에 대한 CloudWatch의 반송 메일 및 수신 거부 지표를 추적하는 데 사용됩니다. (추가 요금이 적용되며, CloudWatch의 지표당 요금 참조)
     - 활성화됨 - 구성 세트에 대한 평판 지표의 사용을 설정하려면 이 확인란을 선택합니다.

6. 금지 목록 옵션 -이 섹션에서는이 구성 세트를 사용하여 계정 수준 금지를 재정의하는 옵션부터 시작하여 사용자 지정 금지를 정의하는 결정 세트를 제공합니다. 구성 세트 수준 금지 로직 맵을 사용하면 재정의 조합의 효과를 이해하는 데 도움이 됩니다. 이러한 다중 계층 재지정 선택을 결합하여 세 가지 다른 금지 수준을 구현할 수 있습니다.

   a. 계정 수준 금지 사용: 계정 수준 금지를 재정의하지 말고 구성 세트 수준 금지를 구현하지 마세요. 기본적으로, 이 구성 세트를 사용하여 보낸 이메일은 단순히 계정 수준 금지를 사용합니다. 방법:

      - 금지 목록 설정(Suppression list settings)에서 계정 수준 설정 재정의(Override account level settings) 상자를 선택 취소합니다.

   b. 금지 사용 안 함: 구성 세트 수준 금지를 사용하지 않고 계정 수준 금지를 재정의합니다. 즉, 이 구성 세트를 사용하여 전송된 이메일은 계정 수준 금지를 사용하지 않습니다. 다시 말해 모든 금지가 취소됩니다. 방법:

      i. 금지 목록 설정(Suppression list settings)에서 계정 수준 설정 재정의(Override account level settings) 상자를 선택합니다.

      ii. 금지 목록(Suppression list)에서 사용(Enabled) 상자를 선택 취소합니다.

c. 구성 세트 수준 금지 사용: 계정 수준 금지를 이 구성 세트에 정의된 사용자 지정 금지 목록 설정으로 재정의합니다. 즉, 이 구성 세트를 사용하여 보낸 이메일은 자체 금지 설정만 사용하고 계정 수준 금지 설정은 무시합니다. 방법:

   i. 금지 목록 설정(Suppression list settings)에서 계정 수준 설정 재정의(Override account level settings) 상자를 선택합니다.

   ii. 금지 목록(Suppression list)에서 사용(Enabled)을 선택합니다.

   iii. 이유 지정...(Specify the reason(s)...)에서 사용할 구성 세트에 대해 금지 이유 중 하나를 선택합니다.

7. 가상 배달 가능성 관리자 옵션 -이 섹션은 가상 배달 가능성 관리자 기능이 활성화된 경우에만 표시됩니다. 여기에서 계정 수준에서 가상 전달 가능성 관리자 설정에 정의된 방식을 재정의하여이 구성 세트가 참여 추적 및 최적화된 공유 전달을 사용하는 방법에 대한 사용자 지정 설정을 정의할 수 있습니다.

   a. 이 구성 세트에 대한 참여 추적과 최적화된 공유 전송을 모두 비활성화하려면 다음과 같이 하세요.

      i. 계정 수준 설정 재정의 상자를 선택합니다.

      ii. Engagement tracking(참여 추적)과 Optimized shared delivery(최적화된 공유 전송)에서 모두 Enabled(활성화)가 선택 취소되어 있는지 확인한 다음 Save changes(변경 사항 저장)를 선택합니다.

   b. 이 구성 세트에 대한 참여 추적 및 최적화된 공유 전송 중 하나 또는 둘 다를 활성화하거나 비활성화하려면 다음과 같이 하세요.

      i. Override account level settings(계정 수준 설정 재정의) 상자를 선택합니다.

      ii. Engagement tracking(참여 추적)과 Optimized shared delivery(최적화된 공유 전송) 중 하나 또는 둘 다에서 Enabled(활성화)를 선택하거나 선택 취소한 다음 Save changes(변경 사항 저장)를 선택합니다.

   c. 이 구성 세트에 대한 참여 추적 및 최적화된 공유 전송의 가상 전달 가능성 관리자 계정 수준 설정으로 되돌리려면 다음과 같이 하세요.

      - Override account level settings(계정 수준 설정 재정의) 상자를 선택 취소한 다음 Save changes(변경사항 저장)를 선택합니다.

8. 아카이빙 옵션 -이 섹션에서는이 구성 세트에서 보낸 이메일을 아카이브하는 옵션을 제공합니다.

   a. 활성 확인란을 선택합니다.

b. 아카이브 필드 내부를 클릭하고 목록에서 아카이브를 선택한 다음 변경 사항 저장을 선택하거나 아카이브 생성을 선택하고 나머지 단계를 계속합니다.

c. 아카이브 이름 필드에 고유한 이름을 입력합니다.

d. (선택 사항) 보존 기간 필드에서 보존 기간을 선택하여 기본 보존 기간인 180일을 재정의합니다.

e. (선택 사항) KMS AWS KMS 키 ARN 필드에 자체 키를 입력하거나 AWS KMS 키 생성을 선택하여 아카이브를 암호화할 수 있습니다.

f. 아카이브 만들기를 선택합니다.

9. 태그 -이 섹션에서는 선택적으로 구성 세트에 하나 이상의 태그를 추가할 수 있습니다.

   a. Add new tag(새 태그 추가가)를 선택합니다.

   b. 태그 키를 입력합니다.

   c. 태그 값을 입력합니다(선택 사항).

   입력한 태그를 제거하려면 해당 태크에 대하여 Remove(제거)를 선택합니다. 최대 50개의 태그를 입력할 수 있습니다.

10. Create set(세트 만들기)를 선택하여 구성 세트를 만듭니다.

구성 세트를 만들었으므로 구성 집합에 대한 이벤트 대상을 정의할 수 있습니다. 그러면 이벤트 대상에 대해 지정한 이벤트 유형으로 트리거되는 이벤트 게시가 사용 설정됩니다. 구성 세트에는 여러 이벤트 유형이 정의된 여러 이벤트 대상이 있을 수 있습니다. Amazon SES 이벤트 대상 만들기을(를) 참조하세요.

구성 세트를 만듭니다(AWS CLI).

JSON 파일을 AWS CLI의 `aws sesv2 create-configuration-set` 명령에 대한 입력으로 사용하여 구성 집합을 만들 수 있습니다.

1. CLI 입력 JSON 파일 생성

   자주 사용하는 파일 편집 도구를 사용하여 다음 키와 사용자 환경에 유효한 값을 포함하는 JSON 파일을 만들거나 값이 지정되지 않은 `--generate-cli-skeleton` 옵션을 포함하는 SES API v2 `aws sesv2 create-configuration-set` 명령을 사용하여 샘플 JSON 구조를 표준 출력으로 인쇄합니다.

   이 예제에서는 `create-configuration-set.json`(이)라는 이름의 파일이 사용됩니다.

```
{
    "ConfigurationSetName": "sample-configuration-set",
    "TrackingOptions": {
        "CustomRedirectDomain": "some.domain.com",
        "HttpsPolicy": "REQUIRE"
    },
    "DeliveryOptions": {
        "TlsPolicy": "REQUIRE",
        "SendingPoolName": "sending pool",
        "MaxDeliverySeconds": 300
    },
    "ReputationOptions": {
        "ReputationMetricsEnabled": true,
        "LastFreshStart": timestamp
    },
    "SendingOptions": {
        "SendingEnabled": true
    },
    "Tags": [
        {
            "Key": "tag key",
            "Value": "tag value"
        }
    ],
    "SuppressionOptions": {
        "SuppressedReasons": ["BOUNCE","COMPLAINT"]
    },
    "ArchivingOptions": {
        "ArchiveArn": "arn:aws:ses:us-east-1:123456789012:mailmanager-
archive/MyArchiveID"
    }
}
```

Note

- JSON 파일 경로의 시작 부분에 `file://` 표기법을 포함시켜야 합니다.
- JSON 파일의 경로는 명령을 실행하는 기본 운영 체제에 대한 적절한 규칙을 따라야 합니다. 예를 들어 Windows에서는 디렉터리 경로를 참조하기 위해 백슬래시(\)를 사용하고 Linux에서는 슬래시(/)를 사용합니다.

2. 작성한 파일을 입력으로 사용하여 다음 명령을 실행합니다.

```
aws sesv2 create-configuration-set --cli-input-json file://create-configuration-set.json
```

> **Note**
>
> 이 명령에 대한 AWS CLI 참조를 검토하려면 create-configuration-set를 참조하세요.

Amazon SES에서 구성 세트 관리

구성 세트를 만든 후, SES 콘솔, Amazon SES API v2 및 Amazon SES CLI v2를 사용하여 보기, 업데이트, 삭제 옵션으로 구성 세트를 관리할 수 있습니다. 또한 구성 세트는 ID에서 이메일을 보낼 때마다 적용되는 기본 구성 세트로 확인된 ID에 할당할 수도 있습니다.

이 섹션의 주제:

- 구성 세트 보기, 편집 및 삭제(콘솔)
- 구성 세트 나열(AWS CLI)
- 구성 세트 세부 정보 가져오기(AWS CLI)
- 구성 세트 삭제(AWS CLI)
- 구성 세트에서 이메일 전송 중지(AWS CLI)
- 기본 구성 세트 이해
- Amazon SES 이벤트 대상 만들기
- Amazon SES에서 IP 풀 할당
- 확인 및 클릭 추적을 처리하기 위한 사용자 지정 도메인 구성

구성 세트 보기, 편집 및 삭제(콘솔)

기존 구성 세트의 세부 정보 페이지 액세스

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Configuration sets(구성 세트)를 선택합니다.

3. 구성 세트 목록에서 이름을 선택하여 선택한 옵션을 보거나 편집하거나 비활성화할 수 있는 개요 탭에서 세부 정보 페이지를 엽니다. 탭을 선택하여 이벤트 대상 옵션에 대해서도 동일한 작업을 수행할 수 있습니다. 각 옵션 및 해당 필드에 대한 자세한 내용은의 관련 섹션을 참조하세요구성 세트(콘솔) 생성.
4. 개요 또는 이벤트 대상 탭에서 볼 수 있는 각 구성 세트의 세부 정보 페이지 상단에는 다음 옵션이 있습니다.
   - Delete(삭제) - 이 버튼을 클릭하면 구성 세트가 삭제됩니다.
   - Disable sending(전송 비활성화) - 이 버튼은 구성 세트에서 이메일 전송을 중지합니다.

구성 세트 나열(AWS CLI)

에서 list-configuration-sets 명령을 사용하여 다음과 같이 현재 리전의 계정과 연결된 모든 구성 세트 목록을 AWS CLI 생성할 수 있습니다.

```
aws sesv2 list-configuration-sets
```

구성 세트 세부 정보 가져오기(AWS CLI)

에서 get-configuration-set 명령을 사용하여 다음과 같이 특정 구성 세트에 대한 세부 정보를 AWS CLI 가져올 수 있습니다.

```
aws sesv2 get-configuration-set --configuration-set-name name
```

구성 세트 삭제(AWS CLI)

다음과 같이에서 delete-configuration-set 명령을 사용하여 특정 구성 세트를 AWS CLI 삭제할 수 있습니다.

```
aws sesv2 delete-configuration-set --configuration-set-name name
```

구성 세트에서 이메일 전송 중지(AWS CLI)

다음과 같이의 put-configuration-set-sending-options 명령을 사용하여 특정 구성 세트에서 이메일 전송을 중지 AWS CLI 할 수 있습니다.

```
aws sesv2 put-configuration-set-sending-options --configuration-set-name name --no-sending-enabled
```

다시 전송을 시작하려면, `--sending-enabled` 옵션과 동일한 명령을 실행합니다.

```
aws sesv2 put-configuration-set-sending-options --configuration-set-name name --sending-enabled
```

기본 구성 세트 이해

이 섹션에서 설명하는 구성 세트를 확인된 ID에 사용할 기본값으로 할당하는 개념은 이점과 사용 사례를 이해하는 데 도움이 됩니다.

기본 구성 세트는 해당 구성 세트와 연결된 이메일 ID에서 보내는 모든 메시지에 해당 규칙을 자동으로 적용합니다. ID를 만드는 동안 또는 사후에 기존 ID의 편집 기능으로 이메일 주소와 도메인 ID 모두에 기본 구성 세트를 적용할 수 있습니다.

기본 구성 세트 고려 사항

- ID와 연결하기 전에 구성 세트를 먼저 만들어야 합니다.
- 기본 구성 세트는 ID가 확인된 경우에만 적용됩니다.
- 이메일 ID는 한 번에 하나의 구성 세트에만 연결될 수 있습니다. 그러나 동일한 구성 세트를 여러 ID에 적용할 수 있습니다.
- 이메일 주소 수준에서 설정된 기본 구성은 도메인 수준에서 설정된 기본 구성을 재정의합니다. 예를 들어, joe@example.com에 연결된 기본 구성 세트는 example.com 도메인의 구성 세트를 재정의합니다.
- 도메인 수준에서 설정된 기본 구성은 도메인의 특정 주소를 확인하지 않는 한 해당 도메인의 모든 이메일 주소에 적용됩니다.
- ID에 대한 기본 구성 세트로 지정된 구성 세트를 삭제한 다음 해당 ID를 통해 이메일을 보내려고 하면 Amazon SES에 대한 호출이 실패하고 '잘못된 요청' 오류가 발생합니다.
- 위임 발신자가 사용 중인 확인된 ID에는 기본 구성 세트를 할당할 수 없습니다.
- ID의 기본 구성 세트로 사용할 기존 구성 세트를 지정하는 방법은 실제로 확인된 ID의 기능이므로 그에 따라 ID 워크플로에 지침이 제공됩니다.
  - ID를 만드는 동안 기본 구성 세트 지정 - Amazon SES에서 자격 증명 생성 및 확인 장에서 도메인 ID 기본 구성 세트 또는 이메일 ID 기본 구성 세트에 대한 선택적인 6단계에 제시된 지침을 따릅니다.

- 기존 ID의 기본 구성 세트 지정 - 5단계에 대한 다음 세부 정보에 따라 콘솔을 사용하여 ID 편집의 단계를 따릅니다.
  a. 구성 세트(Configuration set) 탭을 선택합니다.
  b. 기본 구성 세트(Default configuration set) 컨테이너에서 편집(Edit)을 선택합니다.
  c. 목록 상자를 선택하고 기본값으로 사용할 기존 구성 세트를 선택합니다.
  d. 콘솔을 사용하여 ID 편집의 나머지 단계를 계속 진행합니다.

> **Note**
> 기본값으로 할당한 구성 세트에 평판 지표가 사용 설정된 경우 기본 구성 세트를 사용하여 전송된 모든 메일에 대해 추가 요금이 발생합니다. CloudWatch의 지표당 요금을 참조하세요.

Amazon SES 이벤트 대상 만들기

이벤트 대상을 사용하면 모니터링을 위해 다음과 같은 발신 이메일 추적 작업을 다른 AWS 서비스에 게시할 수 있습니다.

- 전송
- 렌더링 오류
- 거부
- 전달
- 하드 바운스
- 수신 거부
- 배달 지연
- 구독
- 열기
- 클릭

이벤트 게시 설정에 대한 자세한 내용은 the section called “이벤트 게시를 사용하여 이메일 전송 모니터링” 섹션을 참조하세요.

이벤트 대상 만들기

구성 세트를 만든 후에는 구성 집합에 대한 이벤트 대상을 만들 수 있습니다. 그러면 이벤트 대상에 대해 지정한 이벤트 유형으로 트리거되는 이벤트 게시가 사용 설정됩니다. 구성 세트에는 여러 이벤트 유형이 정의된 여러 이벤트 대상이 있을 수 있습니다.

아직 구성 세트를 만들지 않은 경우 the section called “구성 세트를 생성합니다.” 섹션을 참조하세요.

다음 단계에서는 구성 세트에 이벤트 대상을 만들거나 추가하는 방법을 보여줍니다.

SES 콘솔을 사용하여 이벤트 대상을 만들거나 추가하는 방법:

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Configuration sets(구성 세트)를 선택합니다.
3. 이름(Name) 열에서 세부 정보에 액세스할 구성 세트의 이름을 선택합니다.
4. 이벤트 대상(Event destinations) 탭을 선택합니다.
5. Add destination(대상 추가)을 선택합니다.
6. Event types(이벤트 유형)을 선택합니다.

   이메일 전송 이벤트는 Amazon SES를 사용하여 측정할 수 있는 전송 활동과 관련된 지표입니다. 이 단계에서는 Amazon SES가 이벤트 대상에 게시할 이메일 전송 이벤트 유형을 선택합니다.

   이벤트 유형에 대한 자세한 내용은 Amazon SES 전송 활동 모니터링을(를) 참조하세요.

   a. Event types(이벤트 유형)을 선택하여 게시

      - 전송 및 배달 - 게시할 이벤트 유형을 선택하거나 해당 확인란을 선택하거나 Select all(모두 선택)을 클릭하여 모든 이벤트 유형을 게시합니다.

        이벤트 유형

        - 전송 - 전송 요청이 성공했으며 Amazon SES는 메시지를 수신자의 메일 서버로 전송합니다.
        - 렌더링 오류 – 템플릿 렌더링 문제로 인해 이메일이 전송되지 않았습니다. 이 이벤트 유형은 템플릿 데이터가 누락되었을 때 또는 템플릿 파라미터와 데이터 사이에 불일치가 있을 때 발생할 수 있습니다. (이 이벤트 유형은 `SendTemplatedEmail` 또는 `SendBulkTemplatedEmail` API 작업을 사용하는 이메일을 전송할 때만 발생합니다.)

- Rejects(거부) - Amazon SES가 이메일을 수락했으나 이메일에 바이러스가 포함된 것으로 판단되어 수신자의 메일 서버로 전송하려고 시도하지 않았습니다.
- Deliveries(배달) – Amazon SES에서 이메일을 수신자의 메일 서버로 성공적으로 배달했습니다.
- Hard bounces(하드 바운스) – 수신자의 메일 서버가 이메일을 영구적으로 거부했습니다. (Soft bounces(소프트 바운스)는 Amazon SES가 일정 시간 동안 재시도한 후 이메일을 배달하는 데 실패한 경우에만 포함됩니다.)
- Complaints(수신 거부) - 이메일이 수신자의 메일 서버로 성공적으로 전송되었지만 수신자가 이를 스팸으로 표시했습니다.
- 배달 지연 – 일시적인 문제가 발생하여 수신자의 메일 서버에 이메일을 전송할 수 없습니다. 예를 들어 수신자의 받은 편지함이 가득 찼거나 이메일 수신 서버에 일시적인 문제가 발생했을 때 전송 지연이 발생할 수 있습니다. (이 이벤트 유형은 Amazon Pinpoint에서 지원하지 않습니다.)
- Subscriptions(구독) - 이메일이 성공적으로 배달되었지만 수신자가 이메일 헤더에서 `List-Unsubscribe`를 클릭하거나, 바닥글에서 `Unsubscribe` 링크를 선택하여 구독 기본 설정을 업데이트했습니다. (이 이벤트 유형은 Amazon Pinpoint에서 지원하지 않습니다.)

- 확인 및 클릭 추적 - 구독자 참여도를 측정하려면 확인란 중 하나 또는 모두를 선택하여 확인 및 클릭을 추적할 수 있습니다.
  - 확인 – 수신자가 메시지를 수신하여 자신의 이메일 클라이언트에서 열었습니다.
  - Clicks(클릭) – 수신자가 이메일의 링크를 1개 이상 클릭했습니다.

> **Note**
>
> 여기 또는 다른 구성 세트에 정의된 열기 및 클릭 이벤트 게시는 Virtual Deliverability Manager 대시보드의 참여 추적 옵션에 영향을 주지 않습니다. 이러한 옵션은 Virtual Deliverability Manager의 계정 설정 또는 구성 세트 재정의를 통해 정의됩니다. 예를 들어 Virtual Deliverability Manager를 통해 참여 추적을 비활성화한 경우 SES 이벤트 대상에서 설정한 열기 및 클릭 이벤트 게시가 비활성화되지 않습니다.

  - 구성 세트 리디렉션 도메인(Configuration set redirect domain) - 이 필드는 구성 세트를 만들 때 사용자 지정 리디렉션 도메인의 이름을 할당한 경우 표시되며 미리 채워집니다.

> **Note**
>
> 구성 세트에서 Custom redirect domain(사용자 지정 리디렉트 도메인)을 업데이트하여 해당 도메인에 대하여 열기 및 클릭 여부를 추적할 수 있습니다. 구성 세트를 생성합니다.의 4단계에 나온 추적 옵션을 참조하세요. 사용자 지정 확인 및 클릭 도메인 구성에 대한 자세한 내용은 확인 및 클릭 추적을 처리하기 위한 사용자 지정 도메인 구성 섹션을 참조하세요.

b. 다음을 선택하여 계속 진행합니다.

7. 대상 지정

이벤트 대상은 이메일 전송 이벤트를 게시할 수 있는 AWS 서비스입니다. 적절한 대상을 선택하는 것은 캡처할 세부 정보 수준과 데이터 수신 방법에 따라 달라집니다.

a. 대상 옵션

- 대상 유형 - 이벤트를 게시할 AWS 서비스 옆의 라디오 버튼을 선택하면 서비스에 해당하는 필드와 함께 세부 정보 패널이 나타납니다. 아래 링크를 선택하면 서비스의 세부 정보 패널에 대한 지침이 제공됩니다.
  - Amazon CloudWatch(추가 요금이 적용되며 CloudWatch의 지표당 요금 참조)
  - Amazon Data Firehose
  - Amazon EventBridge
  - Amazon Pinpoint(배달 지연 또는 구독 이벤트 유형은 지원하지 않습니다.)
  - Amazon SNS

  이벤트 게시 모델을 사용하여 이메일 작업을 모니터링하는 방법에 대한 자세한 내용은 Amazon SES 이벤트 게시를 사용하여 이메일 전송 모니터링 섹션을 참조하세요.
- 이름 - 이 구성 세트에 대한 대상의 이름을 입력합니다. 이름에는 문자, 숫자, 대시 및 하이픈만 포함될 수 있습니다.
- 이벤트 게시 - 이 대상에 대한 이벤트 게시를 설정하려면 Enabled(활성화됨) 확인란을 선택합니다.

b. Next(다음)를 선택하여 계속 진행합니다.

8. 검토

입력사항이 올바르다고 생각되면 Add destination(대상 추가)를 사용하여 이벤트 대상을 추가합니다.

또한 Amazon SES 콘솔, Amazon SES API v2 또는 Amazon SES CLI v2를 사용하여 이벤트 대상을 만들 수 있습니다.

SES API를 사용하여 이벤트 대상을 만드는 방법:

- SES API를 사용하여 이벤트 대상을 만드는 방법은 `CreateConfigurationSetEventDestination` 섹션을 참조하세요.

이벤트 대상 편집, 사용 중지, 사용 설정 또는 삭제

SES 콘솔을 사용하여 이벤트 대상을 편집, 사용 중지, 사용 설정 또는 삭제하려면 다음 단계를 따르세요.

SES 콘솔을 사용하여 이벤트 대상을 편집, 사용 중지, 사용 설정 또는 삭제하는 방법:

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Configuration sets(구성 세트)를 선택합니다.
3. 이름(Name) 열에서 세부 정보에 액세스할 구성 세트의 이름을 선택합니다.
4. 구성 세트의 이벤트 대상(Event destinations) 탭을 선택합니다.
5. 이름(Name) 열에서 이벤트 대상의 이름을 선택합니다.
6.
   - 편집 - 편집하려는 필드 세트에 해당하는 패널의 편집(Edit) 버튼을 선택하고 변경한 다음 변경 사항 저장(Save changes)을 선택합니다.
   - 사용 중지 또는 사용 설정 - 오른쪽 상단에서 사용 중지(Disable) 또는 사용 설정(Enable)이라는 레이블이 지정된 버튼을 선택합니다.
   - 삭제 - 오른쪽 상단에 있는 삭제(Delete) 버튼을 선택합니다.

또한 Amazon SES 콘솔, Amazon SES API v2 또는 Amazon SES CLI v2를 사용하여 이벤트 대상을 편집, 사용 중지/사용 설정 또는 편집할 수 있습니다.

SES API를 사용하여 이벤트 대상을 편집, 사용 중지, 사용 설정 또는 삭제하는 방법:

1. SES API를 사용하여 이벤트 대상을 사용 중지/사용 설정하는 방법은 `UpdateConfigurationSetEventDestination` 섹션을 참조하세요.
2. SES API를 사용하여 이벤트 대상을 삭제하는 방법은 `DeleteConfigurationSetEventDestination` 섹션을 참조하세요.

Amazon SES에서 IP 풀 할당

IP 풀을 사용하여 특정 유형의 이메일을 전송하기 위한 전용 IP 주소 그룹을 만들 수 있습니다. 모든 Amazon SES 고객이 공유하는 IP 주소의 풀을 사용할 수도 있습니다.

구성 세트에 IP 풀을 할당할 때 다음 옵션 중에서 선택할 수 있습니다.

- 특정 전용 IP 풀 – 기존의 전용 IP 풀을 선택하면 구성 세트를 사용하는 이메일이 해당 풀에 속한 전용 IP 주소만을 사용하여 전송됩니다. 만드는 절차는 다음과 같습니다.
  - 새 표준 IP 풀은 전용 IP(표준)용 표준 전용 IP 풀 생성 섹션을 참조하세요.
  - 새 관리형 IP 풀은 전용 IP(관리형) 사용을 위한 관리형 IP 풀 만들기 섹션을 참조하세요.
- ses-default-dedicated-pool – 이 풀에는 계정에서 IP 풀에 아직 속하지 않은 전용 IP 주소가 모두 포함되어 있습니다. 풀에 연결되지 않은 구성 세트를 사용하여 이메일을 보내거나 구성 세트를 지정하지 않고 이메일을 보내는 경우 이 기본 풀에 있는 주소 중 하나에서 이메일이 전송됩니다. 이 풀은 SES에서 자동으로 관리되며 편집할 수 없습니다.
- ses-shared-pool – 이 풀에는 모든 Amazon SES 고객이 공유하는 많은 IP 주소가 포함되어 있습니다. 이 옵션은 일반적인 전송 동작과 일치하지 않는 이메일을 보내야 할 때 유용할 수 있습니다.

구성 세트에 IP 풀 할당

이 섹션에서는 Amazon SES 콘솔을 사용하여 구성 세트에 IP 풀을 할당하고 수정하는 절차에 대해 설명합니다.

- 콘솔을 사용하여 구성 세트에 IP 풀을 할당하려면...
  - 새 구성 세트를 만드는 중에 - 구성 세트를 생성합니다.의 단계 4에 있는 전송 IP 풀을 참조하세요.
  - 기존 구성 세트를 수정하는 동안 - 선택한 구성 세트의 일반 세부 정보 패널에서 Edit(편집) 버튼을 선택하고 구성 세트를 생성합니다.의 단계 4에 나온 전송 IP 풀의 지침을 따릅니다.

확인 및 클릭 추적을 처리하기 위한 사용자 지정 도메인 구성

이벤트 게시 기능을 사용하여 열기 및 클릭 이벤트를 캡쳐할 때 Amazon SES는 전송하는 이메일을 약간 변경합니다. 열기 이벤트를 캡처하기 위해 SES는 SES를 통해 전송되는 각 이메일에 1x1픽셀 크기의 투명한 GIF 이미지를 추가합니다. 이 이미지에는 각 이메일에 고유한 파일 이름이 포함되며, SES가 운영하는 서버에서 호스팅됩니다. 이미지를 다운로드하면 SES는 누가 어떤 메시지를 열었는지 정확히 알 수 있습니다.

기본적으로 이 픽셀은 이메일 하단에 삽입됩니다. 그러나 일부 이메일 공급자의 애플리케이션에서는 특정 크기를 초과하면 이메일의 미리 보기가 잘리고 메시지의 나머지 부분을 볼 수 있는 링크가 제공될 수 있습니다. 이 시나리오에서는 SES 픽셀 추적 이미지가 로드되지 않으며 추적하려는 오픈율에 집계되지 않습니다. 이 문제를 해결하려면 필요에 따라 이메일의 시작 부분에 픽셀을 배치하거나 이메일 본문에 `{{ses:openTracker}}` 자리표시자를 삽입하여 다른 위치에 픽셀을 배치하면 됩니다. SES가 자리표시자가 있는 메시지를 받으면 열기 추적 픽셀 이미지로 대체됩니다.

> **Important**
>
> - SES에서 전송 시 1개를 초과하는 `{{ses:openTracker}}` 자리 표시자는 모두 제거됩니다.
> - 이메일 템플릿에서 자리 표시자를 사용하는 경우 `{{ses:openTracker}}` 자리 표시자를 하나만 추가합니다. 둘 이상 사용하면 `400 BadRequestException` 오류 코드가 반환됩니다.

링크 클릭 이벤트를 캡처하기 위해 SES는 이메일의 링크를 SES에서 운영하는 서버에 대한 링크로 대체합니다. 이렇게 하면 수신자를 원하는 대상으로 즉시 리디렉션합니다. 쿠키를 포함하여 이 서버에 적용된 요청의 총 헤더 크기는 8,192바이트를 초과해서는 안 됩니다. 그렇지 않으면 `400 BadRequestException` 오류 코드가 반환됩니다.

또한 SES가 소유하고 운영하는 도메인이 아닌 사용자의 자체 도메인을 사용하면 모든 SES 표시가 제거되어 수신자에게 더 일관된 경험을 제공할 수 있습니다. 확인 및 클릭 추적 이벤트를 처리하기 위해 여러 사용자 지정 도메인을 구성할 수 있습니다. 이 사용자 지정 도메인은 구성 세트와 연결됩니다. 구성 세트를 사용하여 이메일을 보낼 때 해당 구성 세트가 사용자 지정 도메인을 사용하도록 구성된 경우, 해당 이메일의 링크를 확인하거나 클릭하면 해당 구성 세트에 지정된 사용자 지정 도메인이 자동으로 사용됩니다.

이 섹션에는 SES에서 운영하는 열기 및 클릭 추적 서버로 사용자를 자동 리디렉션하기 위해 자체 소유한 서버에 하위 도메인을 설정하는 절차가 포함되어 있습니다. 이러한 도메인을 설정하기 위해서는 세 가지 단계가 필요합니다. 먼저 하위 도메인 자체를 구성하고, 사용자 지정 도메인을 사용하도록 구성 세트를 설정한 후, 구성 세트의 이벤트 대상이 열기와 클릭 이벤트를 게시하도록 설정합니다. 이 주제에는 이 모든 단계를 완료하는 절차가 포함되어 있습니다.

그러나 사용자 지정 도메인을 설정하지 않고 열기 또는 클릭 추적을 사용 설정하려면 구성 세트에 대한 이벤트 대상 정의로 곧바로 진행하면 됩니다. 그러면 열기 및 클릭 이벤트를 포함하여 사용자가 지정한 이벤트 유형에 대해 트리거되는 이벤트 게시가 사용 설정됩니다. 구성 세트에는 여러 이벤트 유형이 정의된 여러 이벤트 대상이 있을 수 있습니다. Amazon SES 이벤트 대상 만들기을(를) 참조하세요.

1부: 확인 및 클릭 링크 리디렉션 처리를 위한 도메인 설정

리디렉션 도메인을 설정하는 데 대한 특정 절차는 웹 호스팅 공급자(및 HTTPS 서버를 사용하는 경우 콘텐츠 전송 네트워크)에 따라 다릅니다. 다음 섹션의 절차는 특정 단계가 아닌 일반적인 지침을 제공합니다.

옵션 1: HTTP 도메인 구성

HTTP 도메인을 사용하여 링크 열기 및 클릭을 처리하려는 경우(HTTPS 도메인과 대조), 하위 도메인을 구성하는 프로세스는 간단합니다.

> **Note**
>
> HTTP 프로토콜을 사용하는 사용자 지정 도메인을 설정하고 HTTPS 프로토콜을 사용하는 링크가 포함된 이메일을 보내는 경우, 고객은 이메일의 링크를 클릭할 때 경고 메시지를 확인하게 될 수 있습니다. HTTPS 프로토콜을 사용하는 링크가 포함된 이메일을 보내려는 경우 클릭 추적 이벤트를 처리하기 위해 HTTPS 도메인을 사용해야 합니다.

확인 및 클릭 링크를 처리하기 위한 HTTP 하위 도메인을 설정하려면

1. 확인 및 클릭 추적 링크에 사용할 하위 도메인을 만듭니다. SES는이 하위 도메인이 이러한 링크 처리 전용이며 추적하려는에서 이메일을 AWS 리전 보내는 각에 대해 하위 도메인이 생성되도록 권장합니다.
2. SES에서 사용할 하위 도메인을 확인합니다. 자세한 내용은 도메인 자격 증명 생성 단원을 참조하십시오.
3. 요청을 SES 추적 도메인으로 리디렉션하는 하위 도메인의 DNS 설정에 새 CNAME 레코드를 추가합니다. 리디렉션할 주소는 사용자 지정 하위 도메인 AWS 리전 과 동일한의 이어야 합니다.
   - 의 추적 도메인 테이블 AWS 일반 참조 을 사용하여 사용자 지정 도메인과 동일한 리전에 있는 추적 도메인을 선택합니다.

> **Note**
>
> 웹 호스팅 공급자에 따라 하위 도메인의 DNS 레코드 변경 사항이 적용되려면 몇 분이 걸릴 수 있습니다. 웹 호스팅 공급자 또는 IT 조직은 이러한 지연에 대한 추가 정보를 제공할 수 있습니다.

옵션 2: HTTPS 도메인 구성

열기 및 링크 클릭 추적을 위해 HTTPS 도메인을 사용할 수도 있습니다. 열기 및 링크 클릭 추적을 위해 HTTPS 도메인을 설정하려면 HTTP 도메인 설정에 필요한 단계 외에 몇 가지 추가 단계를 수행해야 합니다.

확인 및 클릭 링크를 처리하기 위한 HTTPS 하위 도메인을 설정하려면

1. 확인 및 클릭 추적 링크에 사용할 하위 도메인을 만듭니다. SES는이 하위 도메인이 이러한 링크 처리 전용이며 추적하려는에서 이메일을 AWS 리전 보내는 각에 대해 하위 도메인이 생성되도록 권장합니다.
2. SES에서 사용할 하위 도메인을 확인합니다. 자세한 내용은 도메인 자격 증명 생성 단원을 참조하십시오.
3. Amazon CloudFront와 같은 콘텐츠 전송 네트워크(CDN)를 사용하여 새 계정을 만듭니다. 기본 CloudFront 배포 시작하기를 참조하세요.
4. 예를 들어 `r.us-east-1.awstrack.me`와 같은 SES 추적 도메인인 오리진에 CDN을 구성합니다. CDN은 사용자 지정 도메인과 동일한 리전에 있는 AWS 추적 도메인을 가리켜야 합니다. CDN은 요청자가 제공한 `Host` 헤더를 오리진에 전달해야 하며 자세한 내용은 이 AWS re:Post 문서를 참조하세요.

   - 의 추적 도메인 테이블 AWS 일반 참조 을 사용하여 사용자 지정 도메인과 동일한 리전에 있는 추적 도메인을 선택합니다.
5. Route 53을 사용하여 도메인에 대한 DNS 구성을 관리하고 CloudFront를 CDN으로 사용하는 경우 Route 53에 해당 CloudFront 배포를 나타내는 별칭 레코드를 만드세요(예: d111111abcdef8.cloudfront.net). 자세한 내용은 Amazon Route 53 개발자 가이드에서 Amazon Route 53 콘솔을 사용하여 레코드 만들기를 참조하세요.

   그 외의 경우에는, 해당 하위 도메인의 DNS 구성에 CDN의 주소를 나타내는 CNAME 레코드를 추가합니다.
6. 신뢰할 수 있는 인증 기관으로부터 SSL 인증서를 얻습니다. 인증서는 1단계에서 만든 하위 도메인과 3~5단계에서 구성한 CDN을 모두 포함해야 합니다. 인증서를 CDN으로 업로드합니다.
7. 다음 curl 명령을 사용하여 새로 만들어진 사용자 지정 도메인이 올바른 리전 및 HTTPS 프로토콜을 사용하고 있는지 확인할 수 있습니다. 다음 예제에서는 도메인 이름을 제외한 모든 항목이 리터럴입니다.

```
curl --head https://custom.domain.com/favicon.ico
```

다음 예제와 같이 응답이 반환됩니다.

```
(python-sdk-test) jdoe@12a34567b89c BaconRedirectService % curl --head https://
custom.domain.com/favicon.ico
HTTPS/1.1 200 OK
x-amz-ses-region: us-east-1
x-amz-ses-request-protocol: https
Content-Type: image/x-icon
Transfer-Encoding: chunked
Date: Fri, 30 Aug 2024 13:50:14 GMT
```

이 응답에는 다음 속성이 포함됩니다.

- `x-amz-ses-region` 헤더 값은 요청을 받은 SES 리전입니다.
- `x-amz-ses-request-protocol` 헤더 값은 헤더의 CDN과 SES 간의 요청에 사용되는 프로토콜입니다.

설정이 올바르면 리전에 도메인이 만들어진 리전이 반영되어야 하며 프로토콜은 HTTPS여야 합니다.

2부: 구성 세트를 통해 사용자 지정 리디렉션 도메인 및 HTTPS 정책 지정

열기 및 클릭 추적 리디렉션을 처리하도록 도메인을 구성한 후, 구성 세트에서 사용자 지정 도메인과 HTTPS 정책을 지정해야 합니다.

구성 세트를 사용하여 이메일을 보낼 때 해당 구성 세트가 사용자 지정 도메인을 사용하도록 구성된 경우, 해당 이메일의 링크를 열거나 클릭하면 구성 세트에 지정된 사용자 지정 도메인과 HTTPS 정책 옵션이 자동으로 사용됩니다.

SES 콘솔 또는 `CreateConfigurationSet` v2 API 작업을 사용하여 완료할 수 있습니다.

콘솔을 사용하여 사용자 지정 리디렉션 도메인과 HTTPS 정책을 지정하는 방법

- 구성 세트를 만들거나 편집하는 동안 구성 세트를 생성합니다.의 4단계에 있는 추적 옵션을 사용하여 사용자 지정 리디렉션 도메인 및 HTTPS 정책 옵션을 지정합니다.

를 사용하여 사용자 지정 리디렉션 도메인 및 HTTPS 정책을 지정하려면 AWS CLI

SES API v2에서 `CreateConfigurationSet` 작업을 사용하고 `TrackingOptions` 속성을 사용하여 사용자 지정 리디렉션 도메인과 HTTPS 정책을 지정할 수 있습니다. 다음 예제 AWS CLI 와 같이에서이 작업을 호출할 수 있습니다.

- 이메일을 보내고 추적하려는 AWS 리전 에서 구성 세트를 생성합니다.

```
aws sesv2 create-configuration-set --cli-input-json file://create.json
```

- 이 예제에서 입력 파일은 `TrackingOptions` 속성의 파라미터를 사용합니다. `CustomRedirectDomain`은 열기 및 클릭 링크를 추적하는 데 사용할 사용자 지정 도메인을 지정하고 `HttpsPolicy`는 HTTPS 정책 옵션을 지정합니다.

```
{
    "ConfigurationSetName": "my-config-set",
    "TrackingOptions": {
        "CustomRedirectDomain": "marketing.example.com",
        "HttpsPolicy": "REQUIRE"
    },
    "SendingOptions": {
        "SendingEnabled": true
    }
}
```

  `HttpsPolicy` 파라미터의 경우 다음 값을 지정하여 사용자 지정 리디렉션 도메인에 대한 열기 및 클릭 추적 링크의 프로토콜을 설정할 수 있습니다.

  - `OPTIONAL` – (기본 동작) 열기 추적 링크는 HTTP를 사용하여 래핑됩니다. 클릭 추적 링크는 링크의 원본 프로토콜을 사용하여 래핑됩니다.
  - `REQUIRE` – 열기 및 클릭 추적 링크는 모두 HTTPS를 사용하여 래핑됩니다.
  - `REQUIRE_OPEN_ONLY` – HTTPS를 사용하여 열기 추적 링크가 래핑됩니다. 클릭 추적 링크는 링크의 원본 프로토콜을 사용하여 래핑됩니다.

3부: 구성 세트를 통해 열기 및 클릭 이벤트 유형 지정

이전 단계의 구성 세트에서 사용자 지정 도메인과 HTTPS 정책을 지정한 후, 구성 세트를 통해 이벤트 대상에서 열기 또는 클릭 이벤트 유형을 선택해야 합니다.

SES 콘솔 또는 `CreateConfigurationSetEventDestination` v2 API 작업을 사용하여 완료할 수 있습니다.

콘솔을 사용하여 열기 또는 클릭 이벤트 유형을 선택하는 방법

- 이벤트 대상을 만들거나 수정하는 동안 the section called "이벤트 대상 만들기"의 6단계에 있는 열기 및 클릭 추적을 사용하여 이벤트 유형을 지정합니다.

이메일을 보낼 때 구성 세트 지정

이메일을 보낼 때 구성 세트를 사용하려면 구성 세트 이름을 이메일 헤더에 전달해야 합니다. 모든 Amazon SES 이메일 전송 방법(예: AWS CLI, AWS SDK, Amazon SES SMTP 인터페이스)을 사용하여 보내는 이메일 헤더에 구성 세트를 전달할 수 있습니다.

SMTP 인터페이스 또는 `SendRawEmail` API 작업을 사용하는 경우 이메일에 다음 헤더를 포함하여 (*`ConfigSet`*를 사용할 구성 세트의 이름으로 바꿈) 구성 세트를 지정할 수 있습니다.

```
X-SES-CONFIGURATION-SET: ConfigSet
```

이 안내서에는 AWS SDK 및 Amazon SES SMTP 인터페이스를 사용하여 이메일을 보내는 코드 예제가 포함되어 있습니다. 이러한 각 예제에는 구성 세트를 지정하는 방법이 포함되어 있습니다. 구성 세트에 대한 참조를 포함하는 이메일을 보내는 단계별 절차를 확인하려면 다음을 참조하세요.

- AWS SDK를 사용하여 Amazon SES를 통해 이메일 전송
- Amazon SES SMTP 인터페이스를 사용하여 이메일 보내기

평판 지표 보기 및 내보내기

Amazon SES는 전체 계정의 전체 반송 메일 및 수신 거부 발생률에 대한 정보를 Amazon CloudWatch로 자동으로 내보냅니다. 이러한 지표를 사용하여 CloudWatch에서 경보를 만들거나 Lambda 함수를 사용한 이메일 전송을 자동으로 일시 중지할 수 있습니다.

또한 개별 구성 세트의 평판 지표를 CloudWatch로 내보낼 수 있습니다. 구성 세트 수준에서 평판 데이터를 내보내면 발신자 평판을 더욱 세부적으로 제어할 수 있습니다.

이 섹션에는 Amazon SES API를 사용하여 개별 구성 세트의 평판 데이터를 CloudWatch로 내보내는 절차가 포함되어 있습니다.

평판 지표 내보내기 활성화

구성 세트의 평판 지표 내보내기를 시작하려면 `UpdateConfigurationSetReputationMetricsEnabled` API 작업을 사용합니다. Amazon SES API에 액세스하려면 AWS CLI 또는 AWS SDKs 중 하나를 사용하는 것이 좋습니다.

이 절차에서는 AWS CLI 가 컴퓨터에 설치되어 있고 올바르게 구성되어 있다고 가정합니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

구성 세트의 평판 지표 내보내기를 활성화하려면

- 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses update-configuration-set-reputation-metrics-enabled --configuration-set-
name ConfigSet --enabled
```

  앞의 명령에서 *ConfigSet*를 평판 지표 내보내기를 시작할 구성 세트의 이름으로 바꿉니다.

평판 지표 내보내기 비활성화

`UpdateConfigurationSetReputationMetricsEnabled` API 작업을 사용하여 구성 세트의 평판 지표 내보내기를 비활성화할 수도 있습니다.

구성 세트의 평판 지표 내보내기를 비활성화하려면

- 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses update-configuration-set-reputation-metrics-enabled --configuration-set-
name ConfigSet --no-enabled
```

  앞의 명령에서 *ConfigSet*를 평판 지표 내보내기를 비활성화할 구성 세트의 이름으로 바꿉니다.

Amazon SES에서 글로벌 엔드포인트 사용

Amazon SES Global 엔드포인트는 이메일 전송 작업의 연속성과 신뢰성을 향상시키는 기능입니다. 이 장에서는 글로벌 엔드포인트의 개념, 설정 및 사용을 안내하여 다중 리전 전송(MRS)을 활용하여 이메일 워크로드의 가용성을 높이고 재해 복구 기능을 개선하는 데 도움이 됩니다.

글로벌 엔드포인트란 무엇입니까?

글로벌 엔드포인트는 SES 아웃바운드 워크로드를 둘 다 배포할 수 있는 리소스입니다 AWS 리전. 구성되면 SES는 선택한 기본 리전과 보조 리전 간에 전송 트래픽을 자동으로 분할합니다. 어느 리전이든 장애가 발생하면 SES는 전송 작업의 연속성을 유지하기 위해 영향을 받는 리전에서 트래픽을 자동으로 이동합니다.

글로벌 엔드포인트 사용의 주요 이점은 다음과 같습니다.

- 이메일 전송 연속성 개선
- 리전 간 자동 장애 조치
- 간소화된 다중 리전 구성

글로벌 엔드포인트 작동 방식

글로벌 엔드포인트를 설정할 때 기본 리전(엔드포인트가 생성되는 리전)과 보조 리전을 선택합니다. 그런 다음 SES는 이메일 전송 요청의 진입점 역할을 하는 다중 리전 엔드포인트(MREP)를 생성합니다.

글로벌 엔드포인트 설정 프로세스는 주요 아티팩트와 기본 리전에서 보조 리전으로의 전송 제한을 동기화합니다. 이렇게 하면 두 리전 모두 모든 예상 볼륨에 충분한 동등한 확인된 자격 증명, 구성 세트 및 승인된 전송 한도를 가질 수 있습니다.

글로벌 엔드포인트가 준비되고 엔드포인트 ID가 SendEmail API 호출에 지정되면 SES는 기본 리전과 보조 리전 간에 아웃바운드 트래픽을 균등하게 자동으로 라우팅합니다. 두 리전 중 하나가 손상된 경우 장애가 해결될 때까지 트래픽이 해당 리전에서 다른 리전으로 가중치가 부여됩니다.

글로벌 엔드포인트 설정

주제

- 사전 조건

- 글로벌 엔드포인트 생성
- 글로벌 엔드포인트 상태

사전 조건

글로벌 엔드포인트를 생성하기 전에 먼저 계정에서 서비스 연결 역할(SLRs)을 생성할 수 있는 SES 권한을 부여해야 합니다. 이러한 역할을 통해 글로벌 엔드포인트를 생성, 사용 및 모니터링하는 데 필요한 필수 서비스 기능과 리소스 액세스를 사용할 수 있습니다. 이 작업은 다음 정책을 구현하여 수행할 수 있습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "ses.amazonaws.com"
        }
      }
    }
  ]
}
```

글로벌 엔드포인트 생성

새 글로벌 엔드포인트를 생성하려면:

1. https://console.aws.amazon.com/ses/ SES 콘솔을 엽니다.
2. 탐색 창에서 글로벌 엔드포인트를 선택합니다.
3. 글로벌 엔드포인트 생성을 선택하고 이름 필드에 이름을 입력합니다.
4. 드롭다운 메뉴에서 보조 리전을 선택합니다. (기본 리전은 기본적으로 콘솔에 로그인한 리전으로 설정됩니다.)

5. (선택 사항) 글로벌 엔드포인트에 하나 이상의 태그를 추가합니다.
6. 구성을 검토하고 글로벌 엔드포인트 생성을 선택합니다.

생성 프로세스는 몇 초 정도 걸릴 수 있습니다. 완료되면 글로벌 엔드포인트의 상태가 "Ready"로 변경됩니다.

사용 AWS CLI:

```
aws sesv2 create-multi-region-endpoint --primary-region us-west-2 --secondary-region us-east-1 --endpoint-name MyGlobalEndpoint
```

이전 예제에서:

- *us-west-2*를 글로벌 엔드포인트의 기본 리전으로 바꿉니다.
- *us-east-1*을 글로벌 엔드포인트의 보조 리전으로 바꿉니다.
- *MyGlobalEndpoint*를 표시 이름으로 바꾸어 글로벌 엔드포인트를 지정합니다.

글로벌 엔드포인트 상태

글로벌 엔드포인트의 상태는 다음과 같을 수 있습니다.

- 생성 - 리소스가 프로비저닝되고 있습니다.
- 준비 - 리소스를 사용할 준비가 되었습니다.
- 실패 - 리소스를 프로비저닝하지 못했습니다.
- 삭제 중 - 요청에 따라 리소스가 삭제되고 있습니다.

보조 리전 준비

이제 글로벌 엔드포인트를 생성했으므로 이메일 전송에 글로벌 엔드포인트를 활용하기 전에 모든 구성 요소(자격 증명, 구성 세트, 이메일 템플릿 및 전송 한도)를 포함한 이메일 전송 구성이 기본 및 보조 리전에서 일관되는지 확인해야 합니다. 이 조정은 잠재적 문제를 방지하고 적절한 이메일 전송 및 추적을 보장하는 데 매우 중요합니다.

콘솔의 리전 복제 기능은 리소스를 자동으로 복제하고 계정 수준 설정을 기본 리전에서 보조 리전으로 복제하여 두 리전의 구성이 동일한지 빠르게 확인할 수 있도록 지원합니다.

리소스 종속성에 따라 리소스를 복제하는 순서가 중요합니다. 충돌을 방지하려면 아래 주제 순서를 따르세요.

주제

- 구성 세트 복제
- 확인된 도메인 자격 증명 복제
- 프로덕션 제한 복제

구성 세트 복제

기본 리전의 여러 구성 세트를 선택하여 보조 리전의 설정과 함께 복제할 수 있습니다.

"구성 세트 복제" 기능을 사용하면 다음을 수행할 수 있습니다.

- 여러 구성 세트를 보조 리전에 한 번에 복제합니다.
- 기본 리전과 보조 리전의 구성 세트 간에 차이가 있는지 확인합니다.

구성 세트를 복제하려면:

1. 글로벌 엔드포인트 페이지의 이름 열에서 글로벌 엔드포인트를 선택하여 복제할 글로벌 엔드포인트를 선택합니다.
2. 중복 구성 세트 카드에서 구성 세트 작업을 확장하고 중복을 선택합니다.
3. 최대 10개의 구성 세트를 선택한 다음 확인을 선택합니다.
4. 상태가 성공하지 못하면 보고서 보기를 선택하여 문제를 식별합니다.
5. (선택 사항) 이전에 복제된 구성 세트의 경우 마지막 세 단계를 반복하면서 차이 확인을 선택하여 기본 리전과 보조 리전 간의 차이를 확인할 수 있습니다.

> **Note**
>
> - 복제한 구성 세트에 이벤트 대상, 평판 옵션, 보관 옵션이 포함되어 있거나 이메일 템플릿에서 참조되는 경우 보조 리전에서 이러한 설정을 수동으로 구성해야 합니다.
> - 기본 리전의 구성 세트에서 전송된(아웃바운드) 이메일에 대해 아카이브를 활성화한 경우 동일한 이름의 보조 리전에서 생성된 아카이브를 사용하여 보조 리전의 구성 세트에서 전송된(아웃바운드) 이메일에 대해 아카이브를 수동으로 활성화해야 합니다.

확인된 도메인 자격 증명 복제

글로벌 엔드포인트 구성이 효과적으로 작동하려면 기본 리전과 보조 리전 모두에서 전송 도메인 자격 증명을 확인해야 합니다. SES는 Deterministic Easy DKIM(DEED)를 사용하여이 프로세스를 간소화합니다.

DEED(Deterministic Easy DKIM)는 Easy DKIM으로 구성된 상위 도메인을 AWS 리전 기반으로 모든에서 일관된 DKIM 토큰을 생성하는 기능입니다. ??? 이러한 일관성을 통해 SES는 추가 DNS 레코드 업데이트 없이 기본 리전에서 확인되면 보조 리전의 도메인을 자동으로 확인할 수 있습니다. 따라서 복제하려는 도메인 자격 증명, 즉 상위 도메인 자격 증명이 Easy DKIM으로 이미 구성되어 있는지 확인해야 합니다.

“검증된 도메인 자격 증명 복제” 기능을 사용하면 다음을 수행할 수 있습니다.

- 여러 도메인 자격 증명을 보조 리전에 한 번에 복제합니다.
- Deterministic Easy DKIM(DEED)을 사용하여 자동으로 확인합니다.
- 기본 리전과 보조 리전의 자격 증명 간 차이를 확인합니다.

SES 콘솔에서 자격 증명을 복제하려면:

1. 글로벌 엔드포인트 페이지의 이름 열에서 글로벌 엔드포인트를 선택하여 복제할 글로벌 엔드포인트를 선택합니다.
2. 확인된 도메인 자격 증명 중복 카드에서 자격 증명 작업을 확장하고 중복을 선택합니다.
3. 최대 10개의 자격 증명을 선택한 다음 확인을 선택합니다.
4. 상태가 성공하지 못하면 보고서 보기를 선택하여 문제를 식별합니다.
5. (선택 사항) 이전에 복제된 자격 증명의 경우 마지막 세 단계를 반복하면서 차이 확인을 선택하여 기본 리전과 보조 리전 간의 차이를 확인할 수 있습니다.

> **Note**
>
> - BYODKIM으로 확인되거나 자체 서명된 도메인 자격 증명은이 경우 DEED를 적용할 수 없으므로 보조 리전에서 수동으로 생성해야 합니다.
> - 메일 발신 속성, 정책 또는 피드백 전달 및 알림을 사용하는 도메인 자격 증명은 보조 리전에서 이러한 기능을 수동으로 구성해야 합니다.

프로덕션 제한 복제

SES는 전송 한도가 리전 간에 정렬되어 있는지 확인하고 필요한 경우 보조 리전에서 한도 증가를 요청할 수 있습니다.

"프로덕션 한도 복제" 기능을 사용하면 다음을 수행할 수 있습니다.

- 프로덕션 한도가 기본 리전과 보조 리전 간에 정렬되어 있는지 확인합니다.
- 필요한 경우 보조 리전에서 한도 증가를 요청합니다.

프로덕션 제한을 복제하려면:

1. 글로벌 엔드포인트 페이지의 이름 열에서 글로벌 엔드포인트를 선택하여 복제할 글로벌 엔드포인트를 선택합니다.
2. 중복 프로덕션 제한 카드에서 상태에 발신 제한 정렬되지 않음이 표시되면 발신 제한 작업을 확장합니다.
3. 보조 리전의 전송 제한 관리를 선택합니다.
4. 보조 리전에서 Service Quotas 페이지가 열리고 기본 리전의 값과 일치하도록 "보내기 할당량" 및 "보내기 속도"로 증가를 요청할 수 있습니다.

   > **Tip**
   > 두 리전 모두에서 사용할 수 있는 최대 할당량을 요청하는 것이 좋습니다. 이메일 트래픽은 정상 작동 조건에서 두 리전에 분산되어 있지만 장애 조치 이벤트 중에 이메일 트래픽의 전체 볼륨이 한 리전으로 전송되고 전체 볼륨 로드를 처리하기에 충분한 한도가 있어야 합니다.

5. (선택 사항) 이전 두 단계를 반복하면서 기본 리전에 대한 전송 제한 관리를 선택하여 기본 리전에 대한 프로덕션 증가를 요청할 수도 있습니다.

> **Important**
> 두 리전 모두 글로벌 엔드포인트의 적절한 기능을 보장하기 위해 일치하는 전송 제한과 함께 이메일을 보내려는 동등한 확인된 자격 증명 및 구성 세트를 가지고 있어야 합니다. 불일치로 인해 전송 실패, 장애 조치 신뢰성 저하, 지표 누락이 발생할 수 있습니다.

글로벌 엔드포인트 사용

주제

- 애플리케이션과 통합
- 모니터링 및 지표

애플리케이션과 통합

애플리케이션에서 글로벌 엔드포인트를 사용하려면 엔드포인트 ID를 얻어야 합니다.

글로벌 엔드포인트의 엔드포인트 ID를 검색하려면:

1. SES 콘솔에서 글로벌 엔드포인트 페이지로 이동하여 이름 열에서 글로벌 엔드포인트를 선택하여 사용할 글로벌 엔드포인트를 선택합니다.
2. 글로벌 엔드포인트 세부 정보 페이지의 엔드포인트 ID에서 복사 아이콘을 선택합니다.

사용 AWS CLI:

```
aws sesv2 get-multi-region-endpoint --endpoint-name MyGlobalEndpoint --region us-west-2
```

이전 예제에서:

- *MyGlobalEndpoint*를 생성 중에 글로벌 엔드포인트에 부여한 표시 이름으로 바꿉니다.
- *us-west-2*를 글로벌 엔드포인트를 생성한 기본 리전으로 바꿉니다.
- API 응답에는와 같은 엔드포인트 ID의 값이 포함됩니다`"EndpointId": "abcdef12.g3h"`.

글로벌 엔드포인트의 엔드포인트 ID를 가져온 후에는 `endpoint-id` 파라미터의 엔드포인트 ID 값을 포함하도록 `SendEmail` 또는 `SendBulkEmail` API 호출을 업데이트할 수 있습니다. 다음은를 사용하여 SendEmail API 호출에서 엔드포인트 ID를 지정하는 방법의 예입니다 AWS CLI.

```
aws sesv2 send-email \
                --from-email-address "sender@example.com" \
                --destination "ToAddresses=recipient@example.com" \
                --content "Subject={Data=Test
 email,Charset=UTF-8},Body={Text={Data=This is a test email sent using Amazon SES
 Global endpoints.,Charset=UTF-8}}" \
```

```
                --endpoint-id "abcdef12.g3h"
```

*abcdef12.g3h*를 콘솔 또는 API를 통해 얻은 실제 엔드포인트 ID로 바꿉니다.

모니터링 및 지표

글로벌 엔드포인트 기능은 기본 리전과 보조 리전 모두에서 이메일 전송 볼륨의 통합 보기를 제공합니다. SES 콘솔의 글로벌 엔드포인트 세부 정보 페이지에 있는 교차 리전 지표 탭을 통해 이러한 지표에 액세스할 수 있습니다.

두 리전의 전송 지표에 액세스하려면:

1. SES 콘솔의 글로벌 엔드포인트 페이지로 이동하여 이름 열에서 지표를 선택하여 지표를 보려는 글로벌 엔드포인트를 선택합니다.
2. 글로벌 엔드포인트 세부 정보 페이지에서 리전 간 지표 탭을 선택하고 최대 31일의 날짜 범위를 입력합니다. 지정된 날짜 범위에 대해 두 리전의 지표가 표시됩니다.

사용 AWS CLI:

```
aws cloudwatch get-metric-statistics \
                --namespace AWS/SES \
                --metric-name SendCount \
                --dimensions Name=ses:multi-region-endpoint-id,Value=abcdef12.g3h \
                --start-time 2024-10-01T00:00:00Z \ --end-time 2024-10-31T23:59:59Z \
                --period 86400 \
                --statistics Sum
```

*abcdef12.g3h*를 실제 엔드포인트 ID로 바꿉니다.

모범 사례 및 고려 사항

이러한 모범 사례 및 고려 사항을 따르면 여러에서 글로벌 엔드포인트의 효과적인 사용률, 모니터링 및 비용 최적화 AWS 리전 를 보장하여 이메일 전송 기능의 가용성과 신뢰성을 개선할 수 있습니다.

- 리전 간에 아티팩트(예: 구성 세트, 확인된 자격 증명)에 대한 변경 사항을 정기적으로 동기화하여 전송 무결성을 유지합니다.
- 교차 리전 지표를 모니터링하여 균형 잡힌 트래픽 분산을 보장하고 잠재적 문제를 식별합니다.
- 글로벌 엔드포인트는 가용성을 개선하지만 SES 아웃바운드의 리전 가용성의 물리적 상태는 변경되지 않습니다.

- 시작 시 글로벌 엔드포인트는 SMTP 또는 VPC 엔드포인트 액세스를 지원하지 않습니다.
- AWS 주소 변환 게이트웨이를 사용하는 경우 잠재적 송신 요금을 고려하세요.
- MREP 지원 원격 리전을 호출할 때 API 지연 시간이 약간 증가할 수 있습니다.

요금

정확한 요금 세부 정보는 변경될 수 있지만 글로벌 엔드포인트는 동일한 양의 메일에 대해 단일 리전 전송보다 가격 프리미엄을 부과할 것으로 예상됩니다. 이러한 증가에도 불구하고 전체 비용은 다른 이메일 서비스 공급자에 비해 경쟁을 유지할 것으로 예상됩니다.

up-to-date 요금 정보는 Amazon SES 요금 페이지를 참조하세요.

Amazon SES를 위한 전용 IP 주소

새로운 Amazon SES 계정을 만들면 기본적으로 다른 SES 사용자와 공유하고 있는 IP 주소에서 이메일이 전송됩니다. 추가 월별 요금을 지불하고 임대하여 귀하만을 위해 독점적으로 예약된 전용 IP 주소를 사용할 수 있습니다. 이를 통해 발신자 평판을 완벽하게 제어할 수 있으며 이메일 프로그램 내에서 서로 다른 세그먼트에 대한 평판을 격리할 수 있습니다. Amazon SES는 전용 IP 주소를 프로비저닝하고 관리하는 두 가지 방법을 제공합니다.

- 표준 - 수동으로 워밍업 및 스케일 아웃하고 IP 풀 내부 및 외부로 수동으로 이동하는 옵션을 포함하여 수동으로 설정하고 관리하는 전용 IP 주소를 말합니다. (SES에서는 이러한 주소를 이전에 전용 IP 주소라고 했습니다.)
- 관리형 - SES에서 관리하는 전용 IP 주소를 빠르고 쉽게 사용할 수 있는 방법을 제공하기 위해 SES가 사용자를 대신하여 자동으로 설정하는 전용 IP 주소를 말합니다. 각 ISP에 대해 개별적으로 자동으로 워밍업되고 전송 볼륨에 따라 자동으로 크기 조정되므로 이메일을 보내는 방식에 따라 전용 IP 주소 사용을 최적화하는 데 도움이 됩니다.

공유 IP 주소와 위에서 정의한 두 가지 유형의 전용 IP 주소 중 하나를 결정할 때는 보내는 이메일의 유형, 볼륨 및 패턴에 가장 많은 이점을 제공하는 주소를 선택하세요. 결정을 돕기 위해 이점이 다음 표에 요약되어 있습니다. 추가 정보를 보려면 혜택 열에서 항목을 선택하세요.

혜택	공유 IP 주소	전용 IP 주소(표준)	전용 IP 주소(관리형)
즉시 사용 가능	예	아니요	아니요
추가 설정이 필요함	아니요	예	예
다른 SES 고객으로부터 분리된 IP 주소 및 평판	아니요	예	예
트래픽이 증가함에 따라 자동으로 용량 증가	아니요	아니요	예
연속적이고 예측 가능한 전송 패턴을 보이는 고객에게 유용	예	예	예

혜택	공유 IP 주소	전용 IP 주소(표준)	전용 IP 주소(관리형)
예측이 비교적 어려운 전송 패턴을 보이는 고객에게 유용	예	아니요	예
대량 발신자에게 유용	예	예	예
소량 발신자에게 유용	예	아니요	아니요
추가 월별 비용	아니요	예	예
발신자 평판에 대한 완벽한 제어	아니요	예	예
이메일 유형, 수신자 또는 기타 요인에 따른 평판 격리	아니요	예	예
이미 알려져 있어 변경할 수 없는 IP 주소 제공	아니요	예	아니요

> **Important**
>
> 예측 기반에 따라 정기적으로 대량의 이메일을 보내려는 경우가 아니면 공유 IP 주소를 사용하는 것이 좋습니다. 전송 패턴이 매우 불규칙한 상황에서 전용 IP 주소를 사용하려는 경우 전용 IP(관리형)를 사용하는 것이 더 좋습니다.

설정 용이성

공유 IP 주소 - 추가로 구성할 필요가 없습니다. 이메일 주소를 확인하고 샌드박스에서 나가자마자 SES 계정에서 이메일을 보낼 준비가 됩니다.

전용 IP 주소(표준) - AWS 지원 센터를 통해 요청을 제출하고 선택적으로 전용 IP 풀을 구성해야 합니다.

전용 IP 주소(관리형) - 전용 IP 주소 요청을 제출할 필요가 없습니다. 옵트인하고 일회성 안내를 통해 관리형 전용 풀을 만들면 자동으로 할당됩니다.

평판 관리

IP 주소 평판은 주로 이전 전송 패턴과 볼륨에 따라 결정됩니다. 장기간 일정한 볼륨의 이메일을 전송하는 IP 주소는 일반적으로 평판이 좋습니다.

공유 IP 주소 - 여러 SES 고객 간에 공유되는 이러한 주소는 대량의 이메일을 한꺼번에 보내고 AWS 가 아웃바운드 트래픽을 신중하게 관리하여 공유 IP 주소의 평판을 극대화합니다.

전용 IP 주소(표준) - 워밍업 후 IP 주소가 SES 공유 풀에서 분리되고 예측할 수 있는 볼륨의 이메일을 일정하게 전송하여 사용자가 자체 발신자 평판을 유지합니다.

> **Note**
>
> 전용 IPs(표준)의 스마트 네트워크 데이터 서비스(SNDS) 데이터에 대한 자세한 내용은 섹션을 참조하세요전용 IP에 대한 SNDS 지표.

전용 IP 주소(관리형) - 새 IP 워밍업 후 새 IP가 SES 공유 풀에서 분리되고 사용자가 자체 발신자의 평판을 유지합니다. 각 ISP의 평판을 추적하고 그에 따라 발신 전송을 최적으로 예약하는 추가적인 이점이 있습니다. 따라서 발신자 평판을 유지하면서 이 자동화를 통해 수동으로 구성된 전용 IP 주소의 동등한 워크로드에 비해 전반적인 전송 가능성을 개선하고 이탈률을 줄일 수 있습니다.

전송 패턴의 예측 가능성

이메일 전송 기록이 일정한 IP 주소는 이전 전송 기록도 없이 갑자기 대량의 이메일을 전송하는 IP 주소보다 평판이 좋습니다.

공유 IP 주소 - 예측 가능한 패턴을 따르지 않는 이메일 전송 패턴에 적합합니다. 공유 IP 주소를 사용하면 상황에 따라 이메일 전송 패턴을 늘리거나 줄일 수 있기 때문입니다.

전용 IP 주소(표준) 매일 점차 늘려가면서 이메일을 전송하여 주소를 워밍업해야 합니다. 새로운 IP 주소의 워밍업 프로세스는 전용 IP 주소(표준) 워밍업 섹션에 설명되어 있습니다. 전용 IP 주소에 대한 워밍업이 끝나면 일정한 전송 패턴을 유지하는 것이 중요합니다.

전용 IP 주소(관리형) - 각 ISP의 워밍업을 개별적으로 최적화하기 위해 실제 전송 패턴을 고려하는 적응형 워밍업 전략(SES 공유 풀과 함께)을 사용하여 관리형 풀의 각 IP에 대해 전용 IP 주소가 자동으로

워밍업됩니다. 관리형 IP 풀은 ISP별 정책의 사용 및 고려 사항에 따라 ISP별로 자동으로 스케일 아웃됩니다.

아웃바운드 이메일 볼륨

공유 IP 주소 - 적은 볼륨의 이메일을 보내는 고객에게 가장 적합합니다.

전용 IP 주소(표준) | 전용 IP 주소(관리형) - 둘 다 대량의 이메일을 보내는 고객에게 적합합니다. 대부분의 ISP는 전용 IP 주소에서 대량의 이메일을 수신하는 경우 임의의 IP 주소에 대한 평판만 추적합니다. 평판을 높이고 싶은 ISP가 있다면 각각 월 1회 이상 24시간 이내에 수백 개 이메일을 전송해야 합니다. 경우에 따라 두 가지 유형의 전용 IP 주소 모두 소량의 이메일에도 사용할 수 있습니다. 예를 들어 메일 서버가 IP 주소 평판이 아닌 특정 IP 주소 목록을 기준으로 이메일을 허용하거나 거부하여 쉽게 알 수 있는 소규모 수신자 그룹에게 전송하는 경우에 효과적일 수 있습니다.

추가 요금

공유 IP 주소 - 사용은 표준 SES 요금에 포함됩니다.

전용 IP 주소(표준) - 임대하는 IP 주소당 월 사용료를 추가로 지불하면 사용할 수 있습니다. 요금에 대한 자세한 내용은 SES 요금 페이지를 참조하세요.

전용 IP 주소(관리형) - 필요한 IP의 양과 상관없이 월 표준 요금 및 메시지당 사용 요금으로 사용할 수 있습니다. 요금에 대한 자세한 내용은 SES 요금 페이지를 참조하세요.

발신자 평판 관리

공유 IP 주소 - 발신자 평판은 SES가 제어합니다.

전용 IP 주소(표준) | 전용 IP 주소(관리형) - 사용자가 발신자 평판을 완전히 제어할 수 있습니다. SES 계정이 이 주소에서 이메일을 전송할 수 있는 유일한 계정입니다. 따라서 발신자 평판은 이메일 전송 사례를 통해 결정됩니다. 또한 전용 IP(관리형)는 최고 성능의 IP 주소를 사용하여 이메일 전송에 사용되는 아웃바운드 IP 주소를 능동적으로 모니터링하여 수신자에게 보내는 이메일의 전달 가능성을 개선합니다. Amazon CloudWatch 지표 및 Amazon SES에 있는 내장 대시보드와 같은 추가 서비스를 사용하여 사용률 데이터를 표시할 수 있습니다.

발신자 평판 격리

공유 IP 주소 - 발신자 평판은 계정 수준에서 설정되며 격리될 수 없습니다.

전용 IP 주소(표준) | 전용 IP 주소(관리형) - 특정 유형의 이메일을 전송하는 데 사용할 수 있는 전용 IP 주소의 그룹인 전용 IP 풀을 만들어 이메일 프로그램 내의 서로 다른 구성 요소에 대한 발신자 평판을 격리할 수 있습니다. 예를 들어 마케팅 이메일을 보내는 용도로 전용 IP 주소 풀을 하나 만들고, 거래 이메일을 보내기 위한 용도로 다른 풀을 만을 수 있습니다.

변경할 수 없는, 알려진 IP 주소

공유 IP 주소 - SES가 메일 전송에 사용하는 IP 주소를 모를 뿐만 아니라 IP 주소가 언제든지 바뀔 수 있습니다.

전용 IP 주소(표준) - 메일을 보내는 주소의 값을 SES 콘솔의 Dedicated IPs(전용 IP) 페이지에서 확인할 수 있습니다. 이는 전용 IP 주소가 정적이기 때문입니다.

전용 IP 주소(관리형) - SES는 전송 패턴에 따라 최적의 전용 IP 주소 수를 자동으로 구성합니다. SES가 IP 풀의 자동 조정을 관리하는 동안 SES 콘솔 또는 API를 통해 계정에 현재 할당된 모든 전용 IP 주소를 볼 수 있습니다. 풀의 IPs 수는 전송 수요에 따라 계속 동적으로 증가하거나 감소합니다.

Amazon SES의 전용 IP 주소(표준)

전용 IP 주소(표준)는 SES에서 수동으로 설정하고 관리하는 전용 IP 주소입니다. SES 기능인 the section called “전용 IP 주소(관리형)”을 사용하여 자동으로 설정 및 관리되는 것과는 다릅니다. 전용 IP(표준)를 사용하면 전용 IP 주소를 사용하여 전송 평판을 완벽하게 제어할 수 있을 뿐만 아니라, 워밍업, 스케일 아웃, IP 풀 관리 등 전용 IP를 완전하게 관리할 수 있습니다.

전용 IP(표준)와 전용 IP(관리형)는 모두 추가 비용을 지불하고 SES에서 임대하는 전용 IP 주소를 의미하지만 구현 및 관리 방식이 다릅니다. 두 가지 모두에 공통된 혜택이 있지만 전용 IP 주소에서 설명한 것처럼 이메일 전송 유형에 따라 각각 고유한 이점이 있습니다.

이 섹션의 주제에서는 SES에서 전용 IP(표준)를 수동으로 설정하고 관리하는 방법을 설명합니다.

주제

- 전용 IP 주소(표준) 요청 및 해제
- 전용 IP 주소(표준) 워밍업
- 전용 IP(표준)용 표준 전용 IP 풀 생성

전용 IP 주소(표준) 요청 및 해제

전용 IP 주소(표준)를 사용하려면 먼저 요청해야 합니다. 더 이상 필요 없으면 해제해야 합니다. AWS Support Center를 통해 전용 IP (표준)를 요청하고 해제할 수 있습니다. Amazon SES용으로 임대한 각 전용 IP 주소에 대한 추가 월 사용료가 계정에 청구됩니다. 전용 IP(표준)를 사용할 때는 최소 약정이 없습니다.

전용 IP(표준)에 연결된 비용에 대한 자세한 내용은 Amazon SES 요금을 참조하세요.

현재 Amazon SES를 사용할 수 있는 모든 리전 목록은 Amazon Web Services 일반 참조의 AWS 리전 및 엔드포인트를 참조하세요. 각에서 사용할 수 있는 가용 영역 수에 대한 자세한 내용은 AWS 글로벌 인프라를 AWS 리전참조하세요.

전용 IPs 요청 또는 해제(표준)

AWS 지원 센터에서 서비스 할당량 증가 사례를 생성하여 필요한 만큼 전용 IPs(표준)를 요청할 수 있습니다.

전용 IPs 요청하거나 해제하려면(표준)

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽의 탐색 창에서 Dedicated IPs(전용 IP)를 선택합니다.
3. 다음 중 하나를 수행합니다.
   a. 계정에 기존 전용 IP가 없는 경우:
      - Dedicated IPs(전용 IP) 온보딩 페이지가 표시됩니다. Dedicated IPs (standard) overview(전용 IP(표준) 개요) 패널에서 Request dedicated IPs(전용 IP 요청)를 선택합니다.
   b. 계정에 기존 전용 IP가 있는 경우:
      i. Dedicated IPs(전용 IP) 페이지에서 Standard IP pools(표준 IP 풀) 탭을 선택합니다.
      ii. Standard overview(표준 개요) 패널에서 Request or relinquish Standard dedicated IPs(표준 전용 IP 요청 또는 해제)를 선택합니다.
4. 안녕하세요! 지원 콘솔에서 AWS 도움말 페이지가 열립니다. 이 페이지의 모든 필드에는 다음 값이 미리 선택되어 있습니다.
   - 사례에 대한 관련 문제 선택 - 계정 및 결제

- 서비스 - Service Quotas
- 범주 - Amazon SES
- 심각도 - 일반 질문

이러한 값을 확인한 후 다음 단계: 추가 정보를 선택합니다.

5. 추가 정보에서 다음 선택을 완료합니다.

- 리전에서 요청이 적용되는 AWS 리전 를 선택합니다.
- 할당량 제목에서 원하는 전용 IP를 선택합니다.
- 값에서 선택한 리전에서 요청하거나 해지하는 전용 IPs 수를 선택합니다.
- 다른에서 전용 IPs(표준)를 요청하거나 해제하려면 다른 제한 추가를 AWS 리전선택하고 그에 따라 필드를 입력합니다. 각 추가 항목에 대해 반복합니다 AWS 리전.
- 설명의 경우 다음 예제에서 로 지정된 각 리전에서 무엇을 하고 싶은지 명확히 합니다.

  요청 - “밀라노 리전에 DIPs가 두 개 있지만 총 3개에 대해 한 개를 더 추가하려고 합니다.”

  해제 - "오하이오 리전에 두 개의 DIPs가 있지만 그 중 하나를 제거하고 싶습니다. 주소가 23.251.228.95인 DIP를 제거하세요."

  > **Important**
  >
  > 전용 IP 주소를 해제하는 프로세스는 되돌릴 수 없습니다. 매월 중순 즈음에 전용 IP 주소를 해제할 경우, 월별 전용 IP 사용 요금은 당월에 경과한 일수를 기준으로 일할 계산됩니다.

- 다음 단계: 지금 해결하거나 문의하기를 선택합니다.
- 지금 해결 또는 문의 페이지에서 원하는 연락처 언어를 선택하고 제출을 선택합니다.

양식을 제출하면 지원 센터에서 요청을 평가합니다. 요청을 부여하면 지원 센터의 사례에 회신하여 요청에 따라 전용 IP 주소가 계정에 추가되거나 계정에서 제거되었는지 확인합니다.

전용 IP 주소(표준) 워밍업

이메일 서비스 공급자가 이메일 허용 또는 거부 여부를 결정할 때는 전송 IP 주소의 평판을 고려합니다. IP 주소의 평판을 결정하는 요인 중 하나는 주소에서 양질의 이메일을 상당수 전송한 이력이 있는지 여부입니다. 이메일 공급자는 이력이 적거나 없는 새로운 IP 주소로부터 오는 이메일을 수락할 가능

성이 낮습니다. 이력이 적거나 없는 IP 주소로부터 전송된 이메일은 수신자의 정크 메일 폴더에 들어가거나 모두 차단될 수 있습니다.

새 전용 IP 주소에서 이메일을 전송하기 시작할 때 이를 전면적으로 사용하기 전에 해당 주소로부터의 이메일 전송량을 점진적으로 늘려야 합니다. 이러한 프로세스를 IP 주소 워밍업이라고 합니다.

IP 주소의 워밍업에 필요한 시간은 이메일 공급자에 따라 다릅니다. 일부 이메일 공급자의 경우 긍정적인 평판을 확립하는 데 2주가 소요될 수 있는 반면, 최대 6주가 걸리는 경우도 있습니다. 새로운 전용 IP 주소를 워밍업할 때는 수신 거부 발생률을 낮게 유지할 수 있도록 활성 상태가 가장 높은 사용자에게 이메일을 전송해야 합니다. 또한 반송 메시지를 주의 깊게 살펴본 후 차단 또는 병목 알림 수가 높은 경우에는 전송하는 이메일 수를 줄여야 합니다. 반송 메일의 모니터링에 대한 자세한 내용은 Amazon SES 전송 활동 모니터링 섹션을 참조하세요.

전용 IP(표준)를 위한 자동 워밍업

전용 IP 주소(표준)를 요청할 때는 Amazon SES에서 이를 자동으로 워밍업하여 전송 이메일의 전송을 개선합니다. IP 주소 자동 워밍업 기능은 기본적으로 사용 설정되어 있습니다. SES는 사전 정의된 워밍업 계획을 기반으로 전용 IP를 통해 보내는 이메일 수를 점차 늘려 전용 IP를 자동으로 워밍업합니다. 이러한 점진적인 증가는 IP가 인터넷 서비스 제공업체(ISP)로부터 긍정적인 평판을 쌓는 데 도움이 됩니다.

자동 워밍업 프로세스 도중의 단계는 전용 IP 주소를 이미 보유 중인지 여부에 따라 다릅니다.

- 전용 IP(표준)를 처음 요청할 때 SES는 전용 IP 주소와 기타 SES 고객과 공유하는 주소 세트 사이에서 이메일 전송을 분산합니다. SES는 시간에 따라 전용 IP 주소에서 전송하는 메시지의 양을 점진적으로 늘립니다.
- 전용 IP 주소가 이미 있는 경우 SES는 기존 전용 IP(이미 워밍업되어 있음)와 새로운 전용 IP(워밍업되지 않음) 사이에서 이메일 전송을 분산합니다. SES는 시간에 따라 새로운 전용 IP 주소에서 전송하는 메시지의 양을 점진적으로 늘립니다.

> **Note**
>
> 자동 IP 워밍업은 시간 기반 프로세스입니다. 워밍업 비율은 전송 볼륨과는 별도로 45일 동안 꾸준히 증가합니다.

전용 IP 주소를 워밍업한 후 긍정적인 평판을 유지하고자 하는 각 이메일 공급자로 매일 약 1,000개의 이메일을 전송해야 합니다. SES와 함께 사용할 각 전용 IP 주소에서 이 작업을 수행해야 합니다.

워밍업 프로세스가 완료된 직후 대규모의 이메일을 보내는 것은 피해야 합니다. 대신, 대상 볼륨에 이를 때까지 이메일의 수를 천천히 늘려야 합니다. 이메일 공급자가 한 IP 주소에서 전송되는 이메일의 수가 크게 급증할 경우 해당 주소로부터의 메시지를 차단 또는 병목시킬 수 있습니다.

전용 IP(표준)에서 자동 워밍업 프로세스 사용 중지

새 표준 전용 IP 주소를 구매할 때 계정에 자동 IP 주소 워밍업 기능이 기본적으로 사용 설정되어 있으므로 Amazon SES에서 자동 워밍업을 수행합니다. 직접 전용 IP 주소를 워밍업하고자 하는 경우 계정 수준에서 모든 IP 주소의 자동 워밍업 기능을 사용 중지할 수 있습니다.

자동 워밍업 기능을 사용 중지하면 이후에 임대하는 모든 전용 IP의 워밍업 상태가 Complete(완료)으로 계정에 추가되어 워밍업 없이 사용할 수 있습니다. 즉, 일반적인 전송 작업에 이러한 IP를 사용하기 전에 제대로 워밍업했는지 사용자가 직접 확인해야 합니다. 자동 워밍업 기능을 사용 중지했을 때 워밍업 중이었던 IP는 영향을 받지 않습니다.

> **Important**
>
> 자동 워밍업 기능을 비활성화한 경우 전용 IP 주소를 직접 워밍업해야 합니다. 워밍업되지 않은 주소로부터 이메일을 전송하는 경우 전송 속도가 느릴 수 있습니다.

계정의 모든 전용 IP(표준)에 대한 자동 워밍업 기능을 사용 중지(또는 다시 사용)하는 방법

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽의 탐색 창에서 Dedicated IPs(전용 IP)를 선택합니다.
3. Dedicated IPs(전용 IP) 페이지에서 Standard IP pools(표준 IP 풀) 탭을 선택합니다.
4. Standard overview(표준 개요) 패널에서 Disable auto warm-up(자동 워밍업 사용 중지)을 선택하여 자동 워밍업을 사용 중지하거나 Enable auto warm-up(자동 워밍업 사용)을 선택하여 자동 워밍업을 다시 사용하도록 설정합니다.

전용 IP(표준) 수동 워밍업

워밍업 비율을 편집하여 전용 IP(표준) 현재 전송 볼륨을 수동으로 늘리거나 줄이고, 워밍업 프로세스를 조기에 종료하고, 현재 전송 볼륨을 0%로 설정하고 워밍업 프로세스를 다시 시작할 수 있습니다.

전용 IP(표준)를 수동으로 워밍업하는 방법

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽의 탐색 창에서 Dedicated IPs(전용 IP)를 선택합니다.
3. Dedicated IPs(전용 IP) 페이지에서 Standard IP pools(표준 IP 풀) 탭을 선택합니다.
4. All Standard dedicated IPs(모든 표준 전용 IP) 패널에서 IP 주소를 선택하고 Edit warm up(워밍업 편집)을 선택하고 다음 옵션 중 하나를 선택합니다.

   a. Edit percentage(편집 비율) - Warm-up percentage(워밍업 비율) 필드에 값을 입력하여 워밍업 비율을 편집한 다음 Save changes(변경 사항 저장)를 선택하여 IP의 현재 전송 볼륨을 늘리거나 줄입니다.

      Warm-up status(워밍업 상태) 열에 `In progress` 상태가 표시되고 Warm-up percentage(워밍업 비율) 열에 입력한 값이 표시됩니다.

   b. Mark as Complete(완료로 표시) - Mark warm-up as Complete?(워밍업을 완료로 표시하시겠습니까?) 대화 상자를 읽고 자동 워밍업 프로세스를 조기에 종료하는 데 따른 영향을 이해했는지 확인한 다음 Mark as Complete(완료로 표시)을 선택합니다.

      Warm-up status(워밍업 상태) 열에 `Complete` 상태가 표시되고 Warm-up percentage(워밍업 비율) 열에 입력한 `100%`가 표시됩니다.

   c. 비율 재설정 - 워밍업 비율을 재설정하시겠습니까? 대화 상자를 읽고 IP의 현재 전송 볼륨을 1%로 설정하는 것이 맞는지 확인한 다음, 자동 워밍업 프로세스를 다시 시작하거나 워밍업 비율을 수동으로 설정하고 재설정을 선택합니다.

      Warm-up status(워밍업 상태) 열에 `In progress` 상태가 표시되고 Warm-up percentage(워밍업 비율) 열에 입력한 `1%`가 표시됩니다.

전용 IP(표준)용 표준 전용 IP 풀 생성

Amazon SES에서 사용할 전용 IP 주소(표준)를 여러 개 구매한 경우 해당 주소의 그룹(전용 IP 풀)을 생성할 수 있습니다. 전용 IP(표준)를 풀에 그룹화하면 관리하기가 더 쉬워집니다. 일반 시나리오는 마케팅 서신을 보내는 용도로 풀을 하나 만들고, 거래 이메일을 보내기 위한 용도로 다른 풀을 만드는 것입니다. 거래 이메일에 대한 발신자의 평판은 이제 마케팅 이메일에서 격리됩니다. 이 시나리오에서는 마케팅 캠페인에서 많은 수의 수신 거부가 발생했을 때도 거래 이메일 전송은 영향을 받지 않습니다.

이 단원에는 전용 IP 풀 생성을 위한 절차가 포함되어 있습니다.

> **Note**
>
> 또한 모든 SES 고객이 공유하는 IP 주소의 풀을 사용하여 구성 세트를 생성할 수도 있습니다. 공유 IP 풀은 일반적인 전송 동작과 일치하지 않는 이메일을 보내야 하는 상황에서 유용할 수 있습니다. 구성 세트가 있는 공유 IP 풀을 사용하는 방법에 대한 내용은 Amazon SES에서 IP 풀 할당 단원을 참조하세요.

SES 콘솔을 사용하여 전용 IP(표준)용 전용 IP 풀 생성

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽의 탐색 창에서 Dedicated IPs(전용 IP)를 선택합니다.

   > **Note**
   >
   > 현재 계정에 전용 IP(표준)가 없는 경우 Dedicated IPs(전용 IP) 온보딩 페이지가 표시되어 전용 IP(표준)를 구매할 수 있습니다. 자세한 내용은 the section called “전용 IPs 요청 또는 해제(표준)”을 참조하세요.

3. Dedicated IPs(전용 IP) 페이지에서 Standard IP pools(표준 IP 풀) 탭을 선택합니다.
4. All Dedicated IP (standard) pools(모든 전용 IP(표준) 풀) 패널에서 Create Standard IP pool(표준 IP 풀 생성)을 선택합니다.

   Create IP Pool(IP 풀 생성) 페이지가 열립니다.
5. Pool details(풀 세부 정보) 패널에서 다음과 같이 구성합니다.

   a. Scaling mode(스케일링 모드) 필드에서 Standard (self managed)(표준(셀프 매니지드))를 선택합니다.

   b. IP pool name(IP 풀 이름) 필드에 IP 풀의 이름을 입력합니다.

      > **Note**
      >
      > IP 풀의 이름은 고유해야 하며 사용자 계정의 관리형 IP 풀의 이름과 같을 수 없습니다.

   c. (선택 사항) 이 IP 풀에 추가하려는 기존 표준 전용 IP 주소가 있는 경우 Dedicated IP addresses(전용 IP 주소) 필드의 드롭다운 목록에서 해당 주소를 선택합니다.

> **Note**
>
> IP 풀에 이미 연결된 IP 주소를 선택하면 이제 이 IP 풀과만 연결됩니다.

6. (선택 사항) Configuration sets(구성 세트) 필드의 드롭다운 목록에서 하나를 선택하여 이 IP 풀을 구성 세트와 연결할 수 있습니다.

> **Note**
>
> - IP 풀에 이미 연결된 구성 세트를 선택하면 이제 이 IP 풀과만 연결됩니다.
> - 이 IP 풀이 생성된 후 관련 구성 세트를 추가하거나 제거하려면 구성 세트의 Sending IP pool(전송 IP 풀) 파라미터를 편집합니다.
> - 아직 구성 세트를 만들지 않은 경우 구성 세트 섹션을 참조하세요.

7. (선택 사항) 하나 이상의 태그를 태그 키와 키 값(선택 사항)을 포함시켜 이 IP 풀에 추가합니다.

   a. Add new tag(새 태그 추가)를 선택하고 키를 입력합니다. 또한, 태그에 선택적 값을 추가할 수 있습니다.

   b. 태그를 추가하려면 Save changes(변경 사항 저장)를 선택합니다.

      최대 50개의 태그를 추가할 수 있습니다. Remove(제거)를 선택하여 태그를 제거할 수 있습니다.

8. Create IP Pool(IP 풀 생성)을 선택합니다.

> **Note**
>
> 표준 IP 풀을 생성한 후에 관리형 IP 풀로 변환할 수 있습니다. 관리형 IP 풀 만들기을(를) 참조하세요.

Amazon SES를 위한 전용 IP 주소(관리형)

전용 IP 주소(관리형) 는 사용자를 대신하여 전용 IP 주소를 자동으로 설정하고 관리하여 SES에서 관리하는 전용 IP 주소를 빠르고 쉽게 사용할 수 있는 Amazon SES 기능입니다. 이를 통해 전용 IP 주소를 이메일 전송 방식에 효율적이고 최적으로 사용할 수 있습니다.

계정에서 전용 IP(관리형)를 사용 설정하려는 경우 관리형 IP 풀을 만들기만 하면 SES가 나머지 작업을 모두 처리합니다. SES는 발신 패턴에 따라 필요한 전용 IP의 수를 결정하고, 전용 IP를 만든 다음, 발신 요구 사항에 따라 전용 IP가 확장되는 방식을 관리합니다.

활성화되면 관리형 IP 풀을 구성 세트와 연결한 다음 이메일을 보낼 때 해당 구성 세트를 지정하여 이메일 발신에 전용 IP(관리형)를 활용할 수 있습니다. 기본 구성 세트를 사용하여 전송 자격 증명에 구성 세트를 적용할 수도 있습니다.

전용 IP(관리형)의 이점 및 특징

전용 IP(관리형)로 만드는 전용 IP 주소는 관리 작업을 자동화하여 전용 IP 주소가 이메일 전송 방식에 최적화된 방식으로 사용되도록 합니다.

- 간편한 온보딩 - 전용 IP(관리형) 사용을 시작하려면 SES 콘솔에서 바로 관리형 IP 풀을 만듭니다. 전용 IP 주소는 풀에 자동으로 할당됩니다. AWS 지원 센터를 통해 요청 사례를 열지 않고도 관리형 IP 풀로 전송을 시작할 수 있습니다.
- ISP별 오토 스케일링 - 관리형 IP 풀은 사용량에 따라 자동으로 스케일 아웃되므로, 전용 IP 풀을 수동으로 모니터링하거나 규모를 조정할 필요가 없습니다. 또한 ISP별 정책도 고려합니다. 예를 들어, SES가 ISP가 낮은 일일 전송 할당량을 지원한다고 감지하면 풀이 스케일 아웃되어 더 많은 IP 주소에 걸쳐 해당 ISP로 트래픽을 더 잘 보낼 수 있습니다.
- 지능형 워밍업 - 전용 IP(관리형)는 용량에 따라 ISP에 메일을 보내기 시작합니다. 즉, 현재 워밍업된 양을 고려하는 것입니다. 각 ISP의 워밍업 수준을 개별적으로 자동으로 추적합니다. 또한 일일 유효 요금을 지불하면 전용 IP(관리형) 기능은 Amazon CloudWatch 지표 및 내장 대시보드의 형태로 주요 ISP 관련 평판에 대한 정보를 제공합니다.
  - ISP별 워밍업 - SES는 각 ISP에 대해 관리형 IP 풀의 각 IP에 대한 평판을 개별적으로 추적합니다. 예를 들어 모든 트래픽을 Gmail로 보낸 경우 IP 주소는 Gmail에서만 워밍업된 것으로 간주되고 다른 ISP는 콜드 상태로 간주됩니다. Hotmail로 전송되는 이메일을 늘려 트래픽 패턴을 변경하면 이 IP 주소가 아직 워밍업되지 않았으므로 SES는 Hotmail의 트래픽을 천천히 늘립니다.
  - 적응형 워밍업 및 공유 풀 전환 - 워밍업 조정은 적응형이며 실제 전송 패턴을 고려합니다. 어떤 ISP로 보내는 볼륨이 떨어지면 해당 ISP의 워밍업 비율도 떨어집니다. 워밍업 초기 단계에서 현재 워밍업 수준을 기준으로 과도한 전송은 다른 Amazon SES 사용자(SES 공유 풀)와 공유되는 IP 주소를 통해 전송됩니다. 워밍업의 이후 단계에서는 과도 전송이 선제적으로 느려지고 나중에 다시 시도됩니다.

> **Important**
>
> 전용 IP(관리형)는 전용 IP 주소를 자동으로 워밍업하지만, 자동 프로세스의 일부는 SES 공유 IP 풀과 대화형으로 작동합니다.
>
> - 새 전용 IP가 워밍업되는 동안 전송 속도가 너무 공격적인 경우 SES는 전송의 일부를 SES 공유 IP 풀에 자동으로 유출하여 새 전용 IP의 평판을 보호합니다.
> - 새로운 전용 IP가 완전히 워밍업된 후에도 모든 전송이 100% 완료된다는 보장은 없습니다. 예를 들어, 전송 속도가 갑자기 상승하고 전용 IP(관리형)가 추가 전용 IP 주소를 할당해야 한다고 판단하는 경우 공유 풀 사용을 포함하는 워밍업 프로세스가 시작됩니다. 마찬가지로 전송 속도가 갑자기 매우 낮아지면 모든 전송이 SES 공유 IP 풀로 전환될 수 있습니다. the section called “워밍업의 중요성” 섹션을 참조하세요.

- 전용 IP 주소 자동 요청 및 해제 - 전용 IP(표준)를 사용할 때 필요에 따라 AWS 지원 센터를 통해 관리형 전용 IPs 주소를 요청하거나 해제할 필요가 없습니다. SES 콘솔, CLI 또는 API에서 직접 전용 IP(관리형)로 온보딩하는 경우 자동으로 전용 IP 주소가 할당되고 전송하는 메시지 양에 따라 요금이 부과됩니다. 전용 IP(관리형)로 만든 IP 풀을 삭제하거나 전용 IP(관리형)를 옵트아웃하면 할당된 IP 주소가 자동으로 해제되고 요금 부과가 즉시 중단됩니다.
  - 첫 번째 전용 IP 주소 받기 - 전용 IP(관리형) 기능은 발신량이 며칠 동안 수백 개의 이메일에 도달하면 첫 번째 전용 IP 주소를 자동으로 할당합니다. 이를 통해 발신 IP가 발신 평판을 쌓고 발송률을 높일 수 있습니다. (발신량이 이 수준이 안 될 것으로 예상되는 경우 공유 IP 주소를 사용해야 합니다. 전용 IP 주소의 비교 표에서 해당 이메일 발신 방식에 가장 적합한 IP 주소 유형을 확인하세요.)

적절한 IP 워밍업이 중요한 이유

전용 IP 주소를 통해 이메일이 전송되려면 수신 ISP에서 좋은 평판을 얻어야 합니다. ISP는 인식할 수 없는 IP에서 소량의 이메일만 수락합니다. 처음 할당된 IP는 새 IP로서 평판이 없기 때문에 수신 ISP에서 인식되지 못합니다. IP의 평판이 확립되기 위해서는 수신 ISP에서 점진적으로 신뢰를 쌓아야 합니다. 이러한 점진적인 신뢰 구축 프로세스를 워밍업이라고 합니다. 전용 IP(관리형)가 IP를 할당한 직후 지능형 워밍업 프로세스가 시작됩니다.

ISP별 워밍업 및 전용 IP(관리형)의 적응형 워밍업 기능을 사용하면 이메일이 제대로 전달되어 워밍업 주기 내내 비즈니스 연속성이 유지됩니다. 워밍업 단계가 완료되면 초과 용량이 대기열에 추가되어 전용 IP 풀을 통해서만 전송됩니다. 그러나 전용 IP 주소가 하나 있고 발신량이 IP 평판 유지에 필요한 최소 볼륨보다 낮으면 전용 IP(관리형)가 전용 IP를 제거할 수 있으며 SES 공유 IP 풀을 통해 전송이 라우팅됩니다.

> **Note**
>
> 소량의 이메일(며칠 동안 하루에 수백 개 미만)을 보내는 경우 SES 공유 IP 풀을 통해 보내는 것이 더 낫습니다. 전용 IP 주소의 비교 표를 검토하여 전용 IP(관리형)가 메일 발송 방식에 적합한지 확인하세요.

전용 IPs(관리형)를 사용할 때 사용자와 SES 간의 공동 책임 이해

전용 IP 주소(관리형)는 전용 IP 관리, 조정 및 워밍업을 위한 다양한 자동 기능을 제공하지만이 자동화의 범위와 SES의 책임을 이해해야 합니다. "관리형"은 SES가 IP 평판 및 목록 문제의 모든 측면을 완전히 처리한다는 의미라고 가정하는 것은 올바르지 않습니다. 이러한 오해를 명확히 하려면 서비스가 규모 조정 및 워밍업과 같은 기술적 측면을 자동화하지만 전송 평판을 유지하고 평판 블록 목록(RBL)에 등록하는 것과 같은 평판 관련 문제를 관리할 책임은 사용자에게 있다는 점을 강조해야 합니다.

다음 FAQs는 사용자와 SES 간의 공동 책임 모델을 명확히 하고 서비스 범위에 대한 일반적인 오해를 해결합니다. 이 FAQ에서는 "관리형" 측면이 기술 인프라 관리를 의미하지만 전송 평판을 적극적으로 모니터링 및 유지하고 반송률을 낮게 유지하며 대부분의 RBL 등록 취소 요청을 직접 처리해야 한다고 강조합니다.

- the section called “관리형 DIPS FAQs”

전용 IP(관리형) 사용을 위한 관리형 IP 풀 만들기

전용 IP(관리형)를 활성화하려면 먼저 관리형 IP 풀을 만들어야 합니다. 관리형 풀을 만들면 이 기능은 발신 패턴에 따라 필요한 전용 IP의 수를 결정하고 요구 사항에 맞게 전용 IP를 동적으로 확장합니다.

관리형 풀을 사용하여 이메일을 보내려면 관리형 풀을 구성 세트와 연결한 다음 이메일을 보낼 때 해당 구성 세트를 지정해야 합니다. 기본 구성 세트를 사용하여 전송 자격 증명에 구성 세트를 적용할 수도 있습니다.

관리형 IP 풀을 만드는 방법에는 다음 두 가지가 있습니다.

- 새 풀을 만듭니다.
- 기존 풀을 표준 풀에서 관리형 풀로 변환합니다.

다음 절차에서는 두 방법에 대한 지침을 제공합니다.

SES 콘솔을 사용하여 관리형 IP 풀을 만들거나 관리형 IP 풀로 변환하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽의 탐색 창에서 Dedicated IPs(전용 IP)를 선택합니다.
3. 새 관리형 IP 풀을 만들지 표준 전용 IP 풀을 관리형 IP 풀로 변환할지에 따라 해당 지침을 따릅니다.

   Create new pool

   새 관리형 IP 풀을 만들려면

   1. 다음 중 하나를 수행합니다.

      a. 계정에 기존 전용 IP가 없는 경우:

         - Dedicated IPs(전용 IP) 온보딩 페이지가 표시됩니다. Dedicated IPs (managed) overview(전용 IP(관리형) 개요) 패널에서 Enable dedicated IPs(전용 IP 활성화)를 선택합니다.

           Create IP Pool(IP 풀 만들기) 페이지가 열립니다.

      b. 계정에 기존 전용 IP가 있는 경우:

         i. Dedicated IPs(전용 IP) 페이지에서 Managed IP pools(관리형 IP 풀) 탭을 선택합니다.

         ii. All Dedicated IP (managed) pools(모든 전용 IP(관리형) 풀) 패널에서 Create Managed IP pool(관리형 IP 풀 만들기)을 선택합니다.

           Create IP Pool(IP 풀 만들기) 페이지가 열립니다.

   2. Pool details(풀 세부 정보) 패널에서 다음과 같이 구성합니다.

      a. Scaling mode(크기 조정 모드) 필드에서 Managed (auto managed)(관리형(자동 관리형)를 선택합니다.

      b. IP pool name(IP 풀 이름) 필드에 관리형 풀의 이름을 입력합니다.

         > **Note**
         >
         > - IP 풀 이름은 고유해야 합니다. 계정의 표준 전용 IP 풀 이름과 중복되어서는 안 됩니다.

- 관리형 IP 풀과 표준 IP 풀을 모두 포함하여 계정당 AWS 리전 별로 50개가 넘는 전용 IP 풀을 보유할 수 없습니다.

3. (선택 사항) Configuration sets(구성 세트) 필드의 드롭다운 목록에서 하나를 선택하여 이 관리형 IP 풀을 구성 세트와 연결할 수 있습니다.

> **Note**
>
> - 이미 IP 풀에 연결된 구성 세트를 선택하면 해당 구성 세트는 이 관리형 풀과 연결되며 더 이상 이전 풀과 연결되지 않습니다.
> - 이 관리형 풀이 만들어진 후 관련 구성 세트를 추가하거나 제거하려면 구성 세트의 General details(일반 세부 정보) 패널에서 Sending IP pool(전송 IP 풀) 파라미터를 편집합니다.
> - 아직 구성 세트를 만들지 않은 경우 구성 세트 섹션을 참조하세요.

4. (선택 사항) 하나 이상의 태그를 태그 키와 키 값(선택 사항)을 포함시켜 IP 풀에 추가합니다.

   a. Add new tag(새 태그 추가)를 선택하고 키를 입력합니다. 또한, 태그에 선택적 값을 추가할 수 있습니다. 최대 50개의 태그를 추가할 수 있습니다. 실수를 한 경우 Remove(제거)를 선택합니다.

   b. 태그를 추가하려면 Save changes(변경 사항 저장)를 선택합니다.

   풀을 만들고 나면 관리형 풀을 선택하고 Edit(편집)을 선택하여 태그를 추가, 제거 또는 편집할 수 있습니다.

5. Create IP Pool(IP 풀 만들기)을 선택합니다.

> **Note**
>
> - 관리형 IP 풀을 만든 후에는 표준 IP 풀로 변환할 수 없습니다.
> - 전용 IPs(관리형)를 사용하는 경우 계정 AWS 리전 의에 대해 10,000개 이상의 발신 자격 증명(도메인 및 이메일 주소 조합)을 가질 수 없습니다.

Convert standard to managed

표준 전용 IP 풀을 관리형 IP 풀로 변환하려면

1. Dedicated IPs(전용 IP) 페이지에서 Standard IP pools(표준 IP 풀) 탭을 선택합니다.
2. 모든 전용 IP(표준) 풀 패널에서, 표준에서 관리형으로 변환하려는 전용 IP 풀의 확인란을 선택합니다.
3. 관리형 풀로 변환을 선택합니다. 관리형 IP 풀로 변환 대화 상자를 읽고 표준 전용 IP 풀을 관리형 IP 풀로 변환하는 조건을 이해했는지 확인합니다.

> **Note**
>
> 전용 IP 풀을 표준에서 관리형으로 변환하기 전에 다음 사항에 유의하세요.
>
> 1. 기존의 모든 전용 IP(표준)가 관리형 풀로 이동됩니다.
> 2. 현재 전송 볼륨에 사용할 전용 IP(표준)를 너무 많이 리스하고 있는 경우 전용 IP(관리형)에서 중복 IP가 제거됩니다.
> 3. 전용 IP(표준)가 다른 애플리케이션의 허용 목록에 포함되어 있다면 중복된 경우 제거되므로 관리형 풀로 이전해서는 안 됩니다(2번 항목 참조).
> 4. 더 이상 IP당 요금이 부과되지 않으며 대신 관리형 풀을 통해 전송하는 볼륨에 따라 요금이 부과됩니다. Amazon SES 요금을 참조하세요.

4. 명시된 조건에 동의하는 경우 확인을 선택합니다. 표준 전용 IP 풀이 관리형 풀로 변환되었음을 확인하는 배너가 나타납니다.

> **Note**
>
> 변환 전에 표준 풀과 연결했던 모든 구성 세트 또는 태그가 이제 관리형 풀과 연결되므로 해당 구성 세트를 사용하는 모든 이메일 전송을 원활하게 전환할 수 있습니다.

이벤트 게시를 사용하여 관리형 풀의 발신 성능을 추적할 수 있습니다. 자세한 내용은 the section called “이벤트 게시를 사용하여 이메일 전송 모니터링” 단원을 참조하십시오.

Amazon SES 콘솔에서 관리형 IP 풀 전송 및 용량 보기

만들어진 관리형 IP 풀의 경우 SES 콘솔에서 전송 지표와 ISP 사용률 및 용량을 보여주는 카드와 시계열 그래프를 사용하여 해당 풀이 이메일 전송에 어떻게 사용되는지 쉽게 확인할 수 있습니다.

SES 콘솔을 사용하여 관리형 IP 풀 전송 및 용량을 확인하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽의 탐색 창에서 Dedicated IPs(전용 IP)를 선택합니다.
3. Dedicated IPs(전용 IP) 페이지에서 Managed IP pools(관리형 IP 풀) 탭을 선택합니다.
4. 전송 및 용량 지표를 Amazon SES 콘솔에서 볼지 Amazon CloudWatch 콘솔에서 볼지에 따라 해당 지침을 따릅니다.

   Amazon SES console

   Amazon SES 콘솔에서 전송 및 용량 지표를 보려면

   1. 모든 전용 IP(관리형) 풀 테이블에서 IP 풀 열에 나열된 관리형 IP 풀의 이름을 선택하여 세부 정보를 확인합니다.

      선택한 IP 풀의 세부 정보 페이지가 열리고 다음 카드 및 시계열 그래프가 표시됩니다.

      a. 카드:

         - 전송 상태 - 다음 두 가지 상태 중 하나를 표시하여 전송 볼륨과 빈도가 전용 IP를 활용하기에 충분한지 여부를 나타냅니다.
           - 볼륨 부족 - 전송 볼륨이 너무 작습니다.
           - 전용 IP를 통한 전송 - 하나 이상의 전용 IP가 관리형 풀에서 사용되고 있습니다.
         - 관리형 전용 IP 전송 볼륨 - 지난 7일 동안 관리형 풀의 전용 IP를 통해 전송된 이메일의 양입니다.
         - 관리형 전용 IP 전송 비율 - 지난 7일 동안 관리형 풀의 전용 IP를 통해 전송된 이메일의 비율입니다.

      b. 그래프:

         - 전송 볼륨 - 지난 7일 동안 관리형 전용 IP를 통해 전송된 이메일 양을 공유 IP를 통해 전송된 것과 비교한 양입니다.

- 전송 볼륨 비율 - 지난 7일 동안 관리형 전용 IP를 통해 전송된 이메일 비율을 공유 IP를 통해 전송된 것과 비교한 비율입니다.
- ISP 용량 - 가장 널리 사용되는 상위 10개 ISP별로 관리형 풀의 전용 IP를 통해 전송되는 이메일의 양과 전송 시 사용 가능한 용량을 표시합니다.
  - ISP에 대한 전송(빨간색 막대) - 선택한 ISP를 통해 지난 24시간 동안 전송된 이메일의 양입니다.
  - ISP에 대한 용량(파란색 선) - 선택한 ISP에서 지난 24시간 동안 사용 가능한 용량입니다.

2. 특정 ISP를 기준으로 ISP 용량 그래프를 필터링하려면 ISP 목록 상자를 선택하고 ISP를 선택합니다. 그래프가 선택한 ISP에 대한 지표로 업데이트됩니다. (ISP로 필터링하지 않으면 기본적으로 Gmail이 표시됩니다.)

Amazon CloudWatch console

Amazon CloudWatch 콘솔에서 전송 및 용량 지표를 보려면

- 모든 전용 IP(관리형) 풀 테이블에서 CloudWatch 지표 열의 <pool_name> CloudWatch 지표 보기 링크를 선택하여 세부 정보를 확인합니다.

  선택한 IP 풀의 페이지가 CloudWatch 콘솔에서 열리고 다음 지표가 표시됩니다.

  - 전송 - 관리형 전용 IP와 공유 IP를 통해 전송된 이메일의 양입니다.
  - ApproximateDedicatedSendingPercentage - 전용 IP를 통해 전송된 트래픽의 대략적인 비율을 나타냅니다.
  - SentLast24Hours - 선택한 ISP를 통해 지난 24시간 동안 전송된 이메일의 양입니다. (SES 콘솔에서 ISP에 대한 전송 레이블이 지정됨)
  - Available24HourSend - 선택한 ISP에서 지난 24시간 동안 사용 가능한 용량입니다. (SES 콘솔에서 ISP에 대한 용량 레이블이 지정됨)

관리형 IP 풀 삭제 및 전용 IP(관리형) 옵트아웃

관리형 IP 풀을 삭제하면 할당된 모든 IP 주소가 자동으로 해제됩니다. 관리형 IP 풀이 하나뿐인 상태에서 이를 삭제하거나 마지막으로 남은 관리형 IP 풀을 삭제하면 전용 IP(관리형) 기능이 옵트아웃되고 청구가 즉시 중단됩니다.

SES 콘솔을 사용하여 관리형 IP 풀 삭제

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽의 탐색 창에서 Dedicated IPs(전용 IP)를 선택합니다.
3. Dedicated IPs(전용 IP) 페이지에서 Managed IP pools(관리형 IP 풀) 탭을 선택합니다.
4. All Dedicated IP (managed) pools(모든 전용 IP(관리형) 풀) 표에서 제거하려는 관리형 풀의 IP 풀 이름 옆에 있는 라디오 버튼을 선택하고 Delete(삭제)를 선택합니다.
5. 팝업 모달에서 선택을 확인할 기회가 주어지며, 이때 Delete(삭제) 또는 Cancel(취소)을 선택하면 관리형 풀을 유지할 수 있습니다.

> **Note**
>
> 관리형 풀이 하나뿐이거나 마지막 관리형 풀을 제거하려는 경우 팝업 모달에서 나머지 관리형 풀을 삭제하면 전용 IP(관리형) 기능이 옵트아웃되며 관련 요금이 더 이상 청구되지 않는다는 메시지가 표시됩니다. Delete(삭제)를 선택하려면 먼저 확인 필드에 *Disable*을 입력해야 합니다.

고유 IP 주소를 사용하여 Amazon SES를 통해 이메일 보내기

Amazon SES에는 자체 IP 사용(BYOIP)이라는 기능이 있어 Amazon SES를 통해 이메일을 전송할 수 있도록 자체 IP 주소를 사용할 수 있습니다. 이메일 전송에 이미 일정 범위의 IP 주소를 사용하고 있는 경우 Amazon SES를 통해 이메일을 전송할 수 있도록 IP 범위를 사용 가능하게 해 달라고 당사에 요청할 수 있습니다.

> **Note**
>
> BYOIP는 수동으로 구성한 전용 IP 주소에만 사용할 수 있으며 전용 IP(관리형)와 함께 사용할 수 없습니다.

BYOIP는 예를 들어 사내 이메일 발송 시스템을 사용하여 긍정적인 IP 평판을 구축했지만 Amazon SES로 마이그레이션하려는 경우에 유용합니다. BYOIP를 사용하면 IP 주소의 평판을 다시 구축할 필요 없이 Amazon SES를 통해 즉시 이메일 발송을 시작할 수 있습니다.

요구 사항

BYOIP를 사용하려면 IP 주소 범위가 다음 요구 사항을 충족해야 합니다.

- 주소 범위를 ARIN(미국 인터넷 번호 등록 협회), RIPE NCC(Réseaux IP Européens Network Coordination Centre) 또는 APNIC(아시아 태평양 지역 네트워크 정보 센터)와 같은 RIR(지역 인터넷 등록 기관)에 등록해야 합니다. 주소 범위는 기업 또는 기관에 등록해야 하며 개인에게 등록할 수 없습니다.
- 서명된 인증 메시지를 제출하여 주소 범위를 소유하고 있다는 증거를 제공할 수 있어야 합니다.
- IP 주소 범위의 주소에는 명확한 기록이 있어야 합니다. 당사는 IP 주소 범위의 평판을 조사할 수 있으며, 좋지 않은 평판이 있거나 악의적인 동작과 연관된 IP 주소가 포함될 경우 IP 주소 범위를 거부할 권한을 보유합니다.
- IP 주소 범위에는 Amazon EC2와 같이 BYOIP를 AWS 서비스 위해 다른 로 가져온 IP 주소 범위가 포함될 수 없습니다.

고려 사항

Amazon SES로 IP 범위를 이전하도록 요청하기 전에 고려해야 할 몇 가지 요소가 있습니다.

- 지정 가능한 가장 구체적인 주소 범위는 /24입니다. 즉, 203.0.113.0/24의 IP 범위를 사용자의 Amazon SES 계정으로 전송하면 203.0.113.0부터 203.0.113.255까지의 총 256개 주소에서 전송할 수 있습니다. 전체 범위는 전송해야 합니다. 현재 Amazon SES에서는 개별 IP 주소 전송을 허용하지 않습니다.
- 특정 IP 주소 범위에 BYOIP를 사용하는 경우 단일 AWS 리전에서만 해당 범위에 액세스할 수 있습니다.
- 리전당 5개의 주소 범위를 AWS 계정으로 가져올 수 있습니다.
- 고유 IP 주소를 사용하는 경우 공유 Amazon SES IP 주소 풀에 있는 주소를 사용할 수 없습니다. 이러한 공유 IP 주소를 사용해야 하는 경우 다른에서 Amazon SES를 사용하거나 새를 AWS 리전생성할 수 있습니다 AWS 계정.
- BYOIP에서 사용하는 각 IP 주소에 대해 월별 요금이 부과됩니다. 자세한 내용은 Amazon SES 요금을 참조하십시오.

Amazon SES에서 고유 IP 주소 사용

당사는 시스템을 통해 원치 않는 콘텐츠 또는 악성 콘텐츠가 전송되지 않도록 하기 위해 각 BYOIP 요청을 신중하게 고려해야 합니다.

Amazon SES에서 자체 IP 범위를 사용하려면 ses-byoip-request@amazon.com으로 다음 정보를 보내 주십시오.

- AWS 계정 ID입니다.
- ap-south-1과 같이 IP 범위를 사용 AWS 리전 하려는 입니다.
- 사용 사례에 대한 설명.
- Amazon SES에서 사용할 IP 범위.
- 해당 범위가 등록된 인터넷 등록 기관의 이름.

업무 시간 기준 48시간 이내에 답변해 드리겠습니다. 귀사와의 커뮤니케이션에서 IP 범위 소유권을 증명하는 문서를 포함하여 추가 정보를 요청할 수도 있습니다.

Amazon SES용 가상 전달 가능성 관리자

전달 가능성, 즉 이메일이 스팸이나 정크 폴더 대신 수신자의 받은 편지함으로 전달되도록 하는 것은 성공적인 이메일 전략의 핵심 요소입니다.

가상 전달 가능성 관리자는 발신 및 전달 데이터에 대한 인사이트를 제공하고 전달 성공률 및 평판에 부정적인 영향을 미치는 문제를 해결하는 방법에 대한 조언을 제공하여 받은 편지함 전달 가능성 및 이메일 전환율을 높이는 등 이메일 전달 가능성을 향상시키는 데 도움이 되는 Amazon SES 기능입니다.

받은 편지함 전달 가능성과 발신자 평판이 중요한 이유

받은 편지함 전달 가능성은 이메일 전환(수신자가 이메일을 연 후 조치를 취하는 경우)의 핵심 요소입니다. 메시지를 받지 못한 고객은 메시지를 볼 수 없고 메시지에 참여할 수도 없습니다.

발신 평판은 고객 경험 수준에서 받은 편지함 전달 가능성에 가장 큰 영향을 미칩니다. 발신 평판은 원치 않는 메시지가 수신자에게 도달하는지, 필요한 메시지가 스팸 폴더로 라우팅되는지 또는 수신자의 메일박스에 도달하기 전에 차단되는지 여부를 결정합니다.

가상 전달 가능성 관리자가 전달 가능성 및 평판을 개선하는 데 도움이 되는 방법

가상 전달 가능성 관리자를 사용하면 계정의 이메일 프로그램을 개괄적 및 세부적인 수준에서 모두 확인하여 문제가 있는 부분에 집중할 수 있도록 해 주는 대시보드와 이메일 전달 가능성 및 평판에 부정적인 영향을 미치는 인프라 문제를 해결하기 위한 솔루션을 제공하는 어드바이저를 통해 전달 가능성과 평판을 모두 높일 수 있습니다.

- 대시보드 – 계정, ISP, 전송 보안 인증 및 구성 세트 수준을 중심으로 전달 가능성 데이터에 대한 인사이트를 제공합니다. 이렇게 하면 신속하게 문제가 있는 영역과 추세를 확인하고 잠재적인 문제가 일시적 거부(지연) 또는 차단과 같은 더 큰 전달 가능성 문제로 이어지기 전에 문제를 파악할 수 있습니다. 이러한 인사이트는 이메일 캠페인에 대한 고객 참여 및 전환율을 높이기 위한 이상적인 시간과 날짜를 계산하여 발신자 평판을 높이는 데도 도움이 됩니다.
- 어드바이저 – 이메일 전달 가능성 및 평판에 부정적인 영향을 미치는 구성 문제에 플래그를 지정하여 이메일 발신을 개선하기 위한 권장 사항을 제공합니다. SPF, DMARC 또는 DKIM 레코드가 없거나 DKIM 키 길이가 너무 짧은 경우 등 발신 도메인, IP 공간 및 인증 레코드의 인프라에서 발생하는 특정 문제를 해결하기 위한 솔루션을 추천합니다.

가상 전달 가능성 관리자 시작하기

가상 전달 가능성 관리자를 사용하기 시작하려면 Amazon SES 콘솔의 온보딩 마법사가 안내하는 계정에 가상 전달 가능성 관리자를 활성화하는 단계를 따릅니다. the section called “시작”을(를) 참조하세요.

주제

- 가상 전달 가능성 관리자 시작하기
- 가상 배달 가능성 관리자 대시보드
- 가상 배달 가능성 관리자 어드바이저
- 가상 전달 가능성 관리자 설정

가상 전달 가능성 관리자 시작하기

계정에서 가상 전달 가능성 관리자를 사용하려면 Amazon SES 콘솔의 온보딩 마법사를 사용하여 활성화해야 합니다. 여기서 참여 추적 및 최적화된 공유 전송을 설정할 수 있습니다. 가상 전달 가능성 관리자는 참여 추적 및 최적화된 공유 전송을 사용하여 전송을 모니터링하고 전달 가능성과 평판을 개선할 수 있도록 지원합니다.

- 참여 추적 – 래핑된 링크 내의 추적 픽셀을 사용하여 조회 및 클릭 이벤트를 통해 수신자 참여 행동을 모니터링할 수 있습니다. 추적 픽셀이 트리거되면 메시지가 열린 시점의 타임스탬프를 제공하고 수신자가 클릭한 링크를 표시합니다. 이 기능을 켜면 Amazon SES 참여 추적 래퍼를 포함하도록 URL과 링크가 변경됩니다.
- 최적화된 공유 전송 – 이메일을 보낼 때 사용할 최적의 IP를 자동으로 선택하므로 대상 이메일 수신자에게 메시지가 전송되는 엔드포인트가 향상됩니다. 전용 IP 주소에는 적용되지 않습니다.

참여 추적과 최적화된 공유 전송은 온보딩 마법사에서 기본적으로 모두 켜져 있지만 끌 수도 있습니다. 가상 배달 가능성 관리자를 최대한 활용하려면 두 기능을 모두 활성화하는 것이 좋습니다.

Amazon SES 콘솔을 사용하여 가상 전달 가능성 관리자 시작하기

다음 절차에서는 Amazon SES 콘솔을 사용하여 가상 전달 가능성 관리자를 시작하는 방법을 보여 줍니다.

Amazon SES 콘솔을 사용하여 가상 전달 가능성 관리자 시작

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.

2. 왼쪽 탐색 창에서 Virtual Deliverability Manager(가상 전달 가능성 관리자)를 선택합니다.
3. Virtual Deliverability Manager overview(가상 전달 가능성 관리자 개요) 페이지에서 Get started with Virtual Deliverability Manager(가상 전달 가능성 관리자 시작하기) 버튼 중 하나를 선택합니다.
4. Select Engagement tracking(참여 추적 선택) 페이지에서 기본값을 그대로 사용하거나 Turn off engagement tracking(참여 추적 끄기)을 선택한 후 Next(다음)를 선택합니다.

   > **Note**
   >
   > 참여 추적을 켜면 Amazon SES 참여 추적 래퍼를 포함하도록 URL과 링크가 변경됩니다.

5. Select Optimized shared delivery(최적화된 공유 전송 선택) 페이지에서 기본값을 그대로 사용하거나 Turn off optimized shared delivery(최적화된 공유 전송 끄기)를 선택한 후 Next(다음)를 선택합니다.

   > **Important**
   >
   > 공유 전송을 최적화하면 전송 평판을 보호하기 위해 이메일 발신이 사전에 지연될 수 있습니다. 지연 없이 발신해야 하는 중요한 워크로드가 있는 경우 이 설정을 활성화하지 않는 것이 좋습니다. 대신 발신에 구성 세트를 사용하고 지연을 감당할 수 있는 구성 세트에 대해서만 최적화된 공유 전송을 활성화하세요.

6. Review and enable(검토 및 활성화) 페이지에서 참여 추적 및 최적화된 공유 전송에 대한 선택 사항을 검토하세요. 돌아가서 변경하려면 Previous(이전)를 선택하고, 그렇지 않으면 Enable Virtual Deliverability Manager(가상 전달 가능성 관리자 활성화)를 선택합니다.

   Virtual Deliverability Manager settings(가상 전달 가능성 관리자 설정) 페이지가 열립니다. Subscription overview(구독 개요) 패널에는 가상 전달 가능성 관리자의 상태가 표시되고 Additional settings(추가 설정) 패널에는 참여 추적 및 최적화된 공유 전송의 상태가 표시됩니다.

계정에 가상 전달 가능성 관리자를 활성화한 후에는 가상 전달 가능성 관리자에서 정의한 방식을 재정의하여 구성 세트가 참여 추적 및 최적화된 공유 전송을 사용하는 방법에 대한 사용자 지정 설정을 정의할 수 있습니다. 이를 통해 특정 이메일 캠페인에 맞게 이메일 전송을 유연하게 조정할 수 있습니다. 예를 들어 마케팅 이메일에 대해 참여 추적 및 최적화된 공유 전송을 활성화하고 트랜잭션 이메일에 대해서는 비활성화할 수 있습니다. 구성 세트를 만들거나 편집할 때 가상 전달 가능성 관리자 옵션을 참조하세요.

를 사용하여 가상 배달 가능성 관리자 시작하기 AWS CLI

다음 예제에서는 AWS CLI를 사용하여 가상 전달 가능성 관리자를 시작하는 방법을 보여 줍니다.

를 사용하여 가상 배달 가능성 관리자를 시작하려면 AWS CLI

Amazon SES API v2에서 `PutAccountVdmAttributes` 작업을 사용하여 가상 전송 가능성 관리자를 시작할 수 있습니다. 다음 예제와 AWS CLI같이에서이 작업을 호출할 수 있습니다.

- 계정에서 가상 전달 가능성 관리자 활성화:

```
aws --region us-east-1 sesv2 put-account-vdm-attributes --vdm-attributes
 VdmEnabled=ENABLED
```

- 다음과 같이 입력 파일을 사용하여 참여 추적과 최적화된 공유 전송을 모두 활성화합니다.

```
aws --region us-east-1 sesv2 put-account-vdm-attributes --cli-input-json file://
attributes.json
```

입력 파일은 다음과 유사합니다.

```
{
    "VdmAttributes": {
        "VdmEnabled": "ENABLED",
        "DashboardAttributes": {
            "EngagementMetrics": "ENABLED"
        },
        "GuardianAttributes": {
            "OptimizedSharedDelivery": "ENABLED"
        }
    }
}
```

Amazon SES API v2 참조의 `VdmAttributes` 데이터 유형에서 연결하여 파라미터 값 및 관련 데이터 유형에 대한 자세한 정보를 찾을 수 있습니다.

> **Note**
>
> 참여 추적을 켜면 Amazon SES 참여 추적 래퍼를 포함하도록 URL과 링크가 변경됩니다.

> **Important**
>
> 공유 전송을 최적화하면 전송 평판을 보호하기 위해 이메일 발신이 사전에 지연될 수 있습니다. 지연 없이 발신해야 하는 중요한 워크로드가 있는 경우 이 설정을 활성화하지 않는 것이 좋습니다. 대신 발신에 구성 세트를 사용하고 지연을 감당할 수 있는 구성 세트에 대해서만 최적화된 공유 전송을 활성화하세요.

- 결과 확인:

```
aws --region us-east-1 sesv2 get-account
```

- 가상 전달 가능성 관리자에서 정의된 방식을 재정의하여 구성 세트가 참여 추적 및 최적화된 공유 전달을 사용하는 방법에 대한 사용자 지정 설정을 정의하려면의 AWS CLI 예제를 참조하세요the section called “설정”.

가상 배달 가능성 관리자 대시보드

대시보드는 조회/클릭, 전송률 및 반송 메일/수신 거부 통계를 통해 전달 가능성과 평판을 나타내는 읽기 쉬운 카드와 시계열 그래프 등 계정의 전달 가능성 프로그램을 개괄적으로 보여 줍니다. 또한 대시보드는 더 자세한 보기도 제공하므로 이메일 캠페인과 관련된 특정 ISP, 전송 보안 인증 또는 구성 세트와 관련된 문제가 있을 때 보다 자세한 특정 테이블 데이터를 심층 분석할 수 있습니다.

전체를 개괄적으로 살펴볼 뿐만 아니라 구체적인 세부 정보도 볼 수 있는 기능을 통해 이메일 프로그램을 전체적으로 검토할 필요 없이 전달 가능성에서 문제가 되는 부분에 집중할 수 있습니다. 또한 이러한 수준의 인사이트를 통해 추세 및 잠재적인 문제가 지연이나 차단과 같은 더 큰 전달 가능성 문제로 이어지기 전에 이를 파악할 수 있습니다.

카드 및 시계열 그래프를 보여주는 가상 배달 가능성 관리자 대시보드의 계정 개요입니다.

날짜 범위 및 필터 기준과 일치하는 전송된 메시지를 보여주는 가상 배달 가능성 관리자 대시보드의 선택된 메시지 테이블입니다.

대시보드에서 제공되는 상세 데이터를 사용하면 다음과 같은 특정 데이터 세트를 심층 분석하여 발신자 평판을 높이고 이메일 프로그램의 참여도와 전환율을 높이기에 이상적인 시간과 날짜를 계산할 수 있습니다.

- ISP 데이터 – 특정 ISP 또는 메일박스 공급자에 전달 가능성 문제가 발생했을 때 유용합니다. 그 외에는 정상적으로 작동할 수 있는 전체 계정을 조정하는 대신 문제가 있는 엔드포인트에 집중하고 모범 사례를 따름으로써 해당 ISP에 대한 발신자 평판을 개선하고 양호한 받은 편지함 전달 가능성을 복원하여 수신자에게 도달할 수 있습니다. 또한 ISP 분포를 이해하는 것도 중요합니다. 한 ISP 또는 메일박스 공급자에게 발신하는 양이 다른 ISP 또는 메일박스 공급자보다 많을 수 있기 때문입니다. 이메일 전환에 긍정적인 영향을 미치려면 트래픽이 항상 전송되고 최종 수신자가 트래픽에 참여하도록 해야 합니다.
- 전송 보안 인증 및 구성 세트 데이터 – 전반적인 계정 전달 가능성 문제에 영향을 미치는 전송 보안 인증 및 구성 세트를 식별하는 데 유용합니다. 문제가 해결될 때까지 발신 ID 및 구성 세트 데이터에 집중하고 구성을 조정하고 특정 ID를 사용한 발신을 줄일 수 있습니다. 예를 들어 전송 보안 인증이 실수로 금지 목록에 전송되어 모든 트래픽이 해당 ID를 거치게 되는 경우가 있습니다. 이 ID는 구성 세트와 연결되어 있어 전달 가능성 문제가 발생합니다. 이러한 경우 전달 가능성 문제의 근본 원인을 파악하려고 전체 계정을 모두 살피는 대신 전송 보안 인증 또는 구성 세트를 식별함으로써 해당 문제를 특정하여 해결하는 데 집중할 수 있다는 점에서 유용합니다.

선택한 전송 보안 인증인 example.com에 대해 가상 배달 가능성 관리자 대시보드에 표시되는 심층 데이터로, 카드에 전달 가능성 및 평판 지표가 표시되어 있습니다. 표에는 해당 전송 보안 인증이 메일을 보낸 모든 ISP가 입력한 날짜 범위 내 각 ISP에 대한 지표 비율과 함께 표시되어 있습니다.

Amazon SES 콘솔에서 가상 배달 가능성 관리자 대시보드 사용

다음 절차는 Amazon SES 콘솔의 가상 배달 가능성 관리자 대시보드를 사용하여 전반적인 전달 가능성 및 평판 통계를 보고 문제가 있는 영역을 심층 분석하는 방법을 보여 줍니다.

가상 배달 가능성 관리자 대시보드를 사용하여 계정의 전달 가능성 지표 데이터를 개괄적 및 세부적 수준에서 확인

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 가상 배달 가능성 관리자 아래의 대시보드를 선택합니다.

> **Note**
>
> - 계정에서 가상 배달 가능성 관리자를 활성화하지 않은 경우 대시보드가 표시되지 않습니다. 자세한 내용은 the section called “시작” 단원을 참조하십시오.

- 대시보드 지표는 거의 실시간으로 표시됩니다.
- 대시보드 메시지는 전송 후 몇 분 이내에 표시됩니다.

3. 전체 계정 개요 패널에서 카드, 시계열 그래프, 심층 분석 테이블의 모든 지표에 사용할 날짜 범위를 선택합니다.

   - Date range(날짜 범위) 필드에서 Relative range(상대 범위)(기본값) 또는 Absolute range(절대 범위)를 선택합니다.

     - Relative range(상대 범위) – 원하는 일수에 해당하는 라디오 버튼을 선택합니다.
       - 사용자 지정 범위 – 일(최대 60일), 주(최대 8주) 또는 월(최대 2개월) 단위로 범위를 입력합니다.
     - 절대 범위 – 선택한 첫 번째 날짜는 시작 날짜이고 두 번째 날짜는 종료 날짜이며 총 60일을 초과하지 않아야 합니다. 히루의 날짜를 지정하려면 시작 날짜와 종료 날짜 모두에 대해 해당 날짜를 선택합니다.

> **Note**
>
> 다음은 대시보드의 모든 날짜 범위에 적용됩니다.
>
> - 모든 날짜 및 시간은 UTC 기준입니다.
> - Relative range(상대 범위) 날짜의 경우 마지막 날은 UTC 자정 타임스탬프로 끝납니다. 예를 들어, Last 7 days(지난 7일)를 선택하면 일곱째 날은 어제 자정에 끝나는 날이 됩니다.
> - 날짜 범위가 30일을 초과하는 경우 계정 통계 테이블의 비율 차이 열과 카드의 변경 비율에 값(대시 -로 표시)이 표시되지 않습니다.

4. 카드, 시계열 그래프, 모든 심층 분석 테이블, 계정 통계, ISP, 발신 ID 및 구성 세트에는 입력한 날짜 범위에서 계산된 지표 합계가 표시되며 대시보드 지표가 계산되는 방법에 설명된 지표 계산을 사용합니다.

   - ISP, 전송 자격 증명 또는 구성 세트 테이블에서 현재 보고 있는 데이터의 로컬 `.csv` 파일을 만들려면 해당 내보내기 버튼을 선택합니다.

5. 입력한 날짜 범위의 볼륨 및 비율 추이를 차트로 표시하는 시계열 그래프가 지표 창에 표시됩니다. 그래프에서 날짜 간격을 마우스오버하면 일일 집계를 기준으로 정확한 거래량 수 또는 비율(%) 이 표시됩니다. 지표 선택 드롭다운을 사용하여 보려는 지표를 필터링할 수 있습니다.

6. Accounts(계정) 탭을 선택하면 Accounts statistics(계정 통계) 표가 표시됩니다.

   - 이 표는 전달 가능성 및 평판 지표의 개요를 보여 주며 입력한 날짜 범위에서 계산된 발신됨, 전송됨, 수신 거부, 일시적 및 영구 반송, 조회 및 클릭에 대한 총 볼륨, 비율(%) 및 차이(%)를 표시합니다.

   > **Note**
   >
   > 날짜 범위가 30일을 초과하는 경우 비율 차이 열에 값(대시 -로 표시)이 표시되지 않습니다.

7. ISP 탭을 선택하여 ISP 테이블을 표시합니다.

   - 이 표는 사용자가 메일을 보낸 각 ISP의 발신 볼륨, 전송됨, 일시적 및 영구 반송, 수신 거부, 조회 및 클릭에 대한 지표를 입력된 날짜 범위에서 계산하여 표시합니다.
   - 특정 ISP를 필터링하려면 ISP 비교 검색 상자에서 포함할 각 ISP에 해당하는 확인란을 선택합니다.
   - 이 테이블에서 현재 보고 있는 데이터의 로컬 `.csv` 파일을 만들려면 해당 내보내기 버튼을 선택합니다.

8. Sending identities(발신 ID) 탭을 선택하면 Sending identities(발신 ID) 표가 표시됩니다.

   - 이 표는 사용한 각 전송 보안 인증의 발신 볼륨, 전송됨, 일시적 및 영구 반송, 수신 거부, 조회 및 클릭에 대한 지표를 입력된 날짜 범위에서 계산하여 표시합니다.
   - 특정 발신 ID를 필터링하려면 보안 인증 비교 검색 상자에서 포함할 각 ID에 해당하는 확인란을 선택합니다.
   - 특정 전송 보안 인증을 심층 분석하려면 Sending identity(전송 보안 인증) 열에서 해당 이름을 선택합니다.
     - 선택된 전송 보안 인증에 대한 전송 속도, 수신 거부, 일시적 및 영구 반송, 공개 및 클릭 속도를 입력된 날짜 범위에서 계산하여 보여주는 카드가 표시됩니다.
     - 시계열 그래프가 새로 고쳐지고 입력한 날짜 범위에서 계산된 대로 선택한 전송 ID에 대한 모든 지표가 표시됩니다.
     - 전송 보안 인증이 메일을 보낸 모든 ISP를 나열하는 ISP 표가 각 ISP의 지표를 입력된 날짜 범위에서 계산한 값과 함께 표시됩니다.
   - 이 테이블에서 현재 보고 있는 데이터의 로컬 `.csv` 파일을 만들려면 해당 내보내기 버튼을 선택합니다.

9. Configuration sets(구성 세트) 탭을 선택하면 Configuration sets(구성 세트) 표가 표시됩니다.

- 이 표는 사용한 각 구성 세트의 발신 볼륨, 전송됨, 일시적 및 영구 반송, 수신 거부, 조회 및 클릭에 대한 지표를 입력된 날짜 범위에서 계산하여 표시합니다.
- 특정 구성 세트를 필터링하려면 구성 세트 비교 검색 상자에서 포함할 각 구성 세트에 해당하는 확인란을 선택합니다.
- 특정 구성 세트를 심층 분석하려면 Configuration set(구성 세트) 열에서 해당 이름을 선택합니다.
  - 선택된 구성 세트에 대한 전송률, 수신 거부, 일시적 및 영구 반송, 공개 및 클릭 속도을 입력된 날짜 범위에서 계산하여 보여 주는 카드가 표시됩니다.
  - 시계열 그래프가 새로 고쳐지고 입력한 날짜 범위에서 계산된 대로 선택한 구성 집합에 대한 모든 지표가 표시됩니다.
  - 구성 세트가 메일을 보내는 데 사용된 모든 ISP를 나열하는 ISP 표가 각 ISP의 지표를 입력된 날짜 범위에서 계산한 값과 함께 표시됩니다.
- 이 테이블에서 현재 보고 있는 데이터의 로컬 `.csv` 파일을 만들려면 해당 내보내기 버튼을 선택합니다.

10. 메시지 탭을 선택하여 메시지 테이블을 표시합니다.

    이 테이블은 보낸 메시지를 검색하고 찾을 수 있는 대화형 테이블입니다. 각 메시지에 대해 현재 배달 및 참여 상태와 이벤트 기록을 추적하고 사서함 공급자가 반환한 응답을 볼 수 있습니다. 특정 메시지를 검색하는 방법은 다음과 같습니다.

    - 날짜 범위 선택기 내에서 선택하면 지난 30일 이내에 보낸 메시지를 필터링할 수 있습니다. 날짜 범위를 선택하지 않으면 기본적으로 해당 시간대 내의 현재 날짜를 포함한 지난 7일이 검색됩니다.
    - 메시지 검색 필드에서는 수신자, 발신 주소, 제목 줄, ISP, 참여 이벤트, 배달 이벤트 및 메시지 ID로 필터링할 수 있습니다. 다음 속성이 적용됩니다.
      - 필터 유형에 따라 대소문자를 구분하는 텍스트 문자열을 입력하거나 목록에서 값을 선택합니다.
      - 참여 이벤트는 단일 값으로 제한되며, 제목 줄은 최대 2개의 값을 가질 수 있고, 다른 모든 필터는 검색당 최대 5개의 값을 가질 수 있습니다. 메시지 ID로 필터링하면 날짜 범위를 포함하여 선택한 다른 모든 필터가 제외됩니다.
      - 메시지 ID 열은 기본적으로 숨겨져 있지만 기어 모양 아이콘을 선택해 메시지 테이블을 보는 방법을 사용자 지정하여 표시할 수 있습니다.

- 필터와 날짜 범위를 선택한 후 검색을 선택하면 검색 기준과 일치하는 메시지로 테이블이 채워집니다. 테이블에는 최대 100개의 메시지가 로드될 수 있습니다. 검색 결과 100개가 넘는 메시지가 반환되는 경우 테이블의 100개 메시지는 반환된 전체 메시지 중 무작위 샘플입니다.
- 메시지의 라디오 버튼을 선택한 후 세부 정보 보기를 선택하면 메시지의 전체 이벤트 기록에 대한 세부 정보가 최신순으로 포함된 메시지 정보 사이드바가 생성되고, 사서함 제공업체가 반환한 응답 또는 진단 코드가 표시됩니다.
- 이 테이블에서 현재 보고 있는 데이터의 로컬 `.csv` 파일을 만들려면 해당 내보내기 버튼을 선택합니다.

AWS CLI를 사용하여 가상 배달 가능성 관리자 지표 데이터에 액세스

다음 예에서는 AWS CLI를 사용하여 가상 배달 가능성 관리자 지표 데이터에 액세스하는 방법을 보여줍니다. 이는 콘솔의 가상 배달 가능성 관리자 대시보드에서 사용되는 것과 동일한 데이터입니다.

를 사용하여 전송 가능성 지표 데이터에 액세스하려면 AWS CLI

Amazon SES API v2에서 `BatchGetMetricData` 작업을 사용하여 전달 가능성 지표 데이터에 액세스할 수 있습니다. 다음 예제에 표시된 대로 AWS CLI 에서 이 작업을 호출할 수 있습니다.

- 전달 가능성 지표 데이터에 액세스:

```
aws --region us-east-1 sesv2 batch-get-metric-data --cli-input-json file://sends.json
```

- 입력 파일은 다음과 유사합니다.

```
{
 "Queries": [
   {
     "Id": "Retrieve-Account-Sends",
     "Namespace": "VDM",
     "Metric": "SEND",
     "StartDate": "2022-11-04T00:00:00",
     "EndDate": "2022-11-05T00:00:00"
    }
 ]
}
```

Amazon SES API v2 참조의 `BatchGetMetricDataQuery` 데이터 유형에서 연결하여 파라미터 값 및 관련 데이터 유형에 대한 자세한 정보를 찾을 수 있습니다.

를 사용하여 전송 가능성 지표 데이터 필터링 및 내보내기 AWS CLI

이 예제에서는 `CreateExportJob` 작업을 사용하여 AWS CLI를 통해 배달 가능성 지표 데이터를 필터링하고 .csv 또는 .json 파일로 내보내는 방법을 보여줍니다. 이는 가상 배달 가능성 관리자 대시보드의 ISP, 전송 자격 증명 및 구성 세트 테이블에 사용되는 것과 동일한 데이터입니다.

를 사용하여 배달 가능성 지표 데이터를 필터링하고 .csv 또는 .json 파일로 내보내려면 AWS CLI

Amazon SES API v2의 `MetricsDataSource` 데이터 유형과 함께 `CreateExportJob` 작업을 사용하여 지표 데이터를 필터링하고 .csv 또는 .json 파일로 내보낼 수 있습니다. 다음 예제 AWS CLI 와 같이에서이 작업을 호출합니다.

- 입력 파일을 사용하여 배달 가능성 지표 데이터를 필터링하고 내보냅니다.

```
aws --region us-east-1 sesv2 create-export-job --cli-input-json file://metric-export-
input.json
```

- 이 예제에서 입력 파일은 `MetricsDataSource` 파라미터를 사용해 메일을 보낸 모든 ISP로 필터링하고 지정된 날짜 범위 내의 배달 성공률을 표시하며 출력 파일로 .csv 형식을 지정합니다.

```
{
    "ExportDataSource": {
        "MetricsDataSource": {
            "Dimensions": {
                "ISP": ["*"]
            },
            "Namespace": "VDM",
            "Metrics": [
                {
                    "Name": "DELIVERY",
                    "Aggregation": "RATE"
                }
            ],
            "StartDate": "2023-06-13T00:00:00",
            "EndDate": "2023-06-20T00:00:00"
        }
    },
    "ExportDestination": {
        "DataFormat": "CSV"
    }
}
```

Amazon SES API v2 참조에 있는 `ExportDataSource` 유형 객체의 `MetricsDataSource`에서 파라미터 값 및 관련 데이터 유형에 대한 자세한 정보를 찾을 수 있습니다.

를 사용하여 전송된 메시지, 전송 및 참여 상태 찾기 및 결과 내보내기 AWS CLI

이 예제에서는 AWS CLI를 통해 `CreateExportJob` 작업을 사용하여 보낸 특정 메시지를 검색하고 찾으며, 현재 배달 및 참여 상태를 확인하고, 검색 결과를 .csv 또는 .json 파일로 내보내는 방법을 보여줍니다. 이는 가상 배달 가능성 관리자 대시보드의 메시지 테이블에서 사용되는 것과 동일한 데이터입니다.

를 사용하여 전송된 메시지, 전송 및 참여 상태를 찾고 결과를 .csv 또는 .json 파일로 내보내려면 AWS CLI

Amazon SES API v2의 `MessageInsightsDataSource` 데이터 유형과 함께 `CreateExportJob` 작업을 사용하면 필터를 적용하여 보낸 특정 메시지를 찾고, 메시지의 배달 및 참여 상태를 확인하고, 결과를 .csv 또는 .json으로 내보낼 수 있습니다. 다음 예제 AWS CLI 와 같이에서이 작업을 호출합니다.

> **Note**
>
> 필터링된 검색 결과 10,000개가 넘는 메시지가 반환되는 경우 API 결과 세트의 10,000개 메시지는 반환된 전체 메시지 중 무작위 샘플입니다.

- 입력 파일을 사용하여 보낸 메시지를 찾고, 현재 상태를 확인하고, 결과를 내보냅니다.

```
aws --region us-east-1 sesv2 create-export-job --cli-input-json file://message-
insights-export-input.json
```

- 이 예제에서 입력 파일은 `MessageInsightsDataSource` 파라미터를 사용하여 제목이 "Sale Ends Tonight!"인 메시지를 필터링하고 출력 파일로 .csv 형식을 지정합니다.

```
{
    "ExportDataSource": {
        "MessageInsightsDataSource": {
            "StartDate": "2023-07-01T00:00:00",
            "EndDate": "2023-07-10T00:00:00",
            "Include": {
                "Subject": [
```

```
                    "Sale Ends Tonight!"
                ]
            }
        }
    },
    "ExportDestination": {
        "DataFormat": "CSV"
    }
}
```

- 이 예제에서 입력 파일은 `MessageInsightsDataSource` 파라미터를 사용하여 제목이 "Hello"로 시작하고 "information"을 포함한 FromEmailAddress에서 "@example.com"으로 끝나는 대상에 전송된 메시지를 필터링하며, 출력 파일로 .json 형식을 지정합니다.

```
{
    "ExportDataSource": {
        "MessageInsightsDataSource": {
            "StartDate": "2023-07-01T00:00:00",
            "EndDate": "2023-07-10T00:00:00",
            "Include": {
                "Subject": [
                    "Hello*"
                ],
                "FromEmailAddress": [
                    "*information*"
                ],
                "Destination": [
                    "*@example.com"
                ]
            }
        }
    },
    "ExportDestination": {
        "DataFormat": "JSON"
    }
}
```

- 이 예제에서 입력 파일은 `MessageInsightsDataSource` 파라미터를 사용하여 제목이 "Hello"로 시작하는 메시지를 필터링하고 FromEmailAddress가 "noreply@example.com"인 결과는 제외하며, 출력 파일로 .csv 형식을 지정합니다.

```
{
    "ExportDataSource": {
        "MessageInsightsDataSource": {
            "StartDate": "2023-07-01T00:00:00",
            "EndDate": "2023-07-10T00:00:00",
            "Include": {
                "Subject": [
                    "Hello*"
                ]
            },
            "Exclude": {
                "FromEmailAddress": [
                    "noreply@example.com"
                ]
            }
        }
    },
    "ExportDestination": {
        "DataFormat": "CSV"
    }
}
```

- 이 예제에서 입력 파일은 `MessageInsightsDataSource` 파라미터를 사용하여 제목이 "Hello"로 시작하고 "information"을 포함한 FromEmailAddress에서 "@example.com"으로 끝나는 대상에 전송되고 Gmail을 ISP로 사용하고 마지막 배달 이벤트가 "DELIVERY"이고 마지막 참여 이벤트가 "OPEN" 또는 "CLICK"인 메시지를 필터링하며, 출력 파일로 .json 형식을 지정합니다.

```
{
    "ExportDataSource": {
        "MessageInsightsDataSource": {
            "StartDate": "2023-07-01T00:00:00",
            "EndDate": "2023-07-10T00:00:00",
            "Include": {
                "Subject": [
                    "Hello*"
                ],
                "FromEmailAddress": [
                    "*information*"
                ],
                "Destination": [
```

```
                    "*@example.com"
                ],
                "Isp": [
                    "Gmail"
                ],
                "LastDeliveryEvent": [
                    "DELIVERY"
                ],
                "LastEngagementEvent": [
                    "OPEN", "CLICK"
                ]
            }
        }
    },
    "ExportDestination": {
        "DataFormat": "JSON"
    }
}
```

- 이 예제에서 입력 파일은 `MessageInsightsDataSource` 파라미터를 사용하여 대상이 "@example1.com" 또는 "@example2.com" 또는 "@example3.com"으로 끝나는 메시지를 필터링하고 LastDeliveryEvent가 "SEND" 또는 "DELIVERY"인 메시지는 제외하며, 출력 파일로 .csv 형식을 지정합니다.

```
{
    "ExportDataSource": {
        "MessageInsightsDataSource": {
            "StartDate": "2023-07-01T00:00:00",
            "EndDate": "2023-07-10T00:00:00",
            "Include": {
                "Destination": [
                    "*@example1.com",
                    "*@example2.com",
                    "*@example3.com"
                ]
            },
            "Exclude": {
                "LastDeliveryEvent": [
                    "SEND",
                    "DELIVERY"
                ]
            }
```

```
        }
    },
    "ExportDestination": {
        "DataFormat": "CSV"
    }
}
```

Amazon SES API v2 참조에 있는 ExportDataSource 유형 객체의 MessageInsightsDataSource에서 파라미터 값 및 관련 데이터 유형에 대한 자세한 정보를 찾을 수 있습니다.

AWS CLI를 사용하여 내보내기 작업 관리

다음 예제에서는 AWS CLI를 사용하여 내보내기 작업을 나열하고, 관련 정보를 얻고, 작업을 취소하는 등 내보내기 작업을 관리하는 방법을 보여줍니다.

를 사용하여 내보내기 작업을 나열하려면 AWS CLI

Amazon SES API v2에서 ListExportJobs 작업을 사용하여 내보내기 작업을 나열할 수 있습니다. 다음 예제 AWS CLI 와 같이에서이 작업을 호출할 수 있습니다.

- 내보내기 작업을 나열합니다.

```
aws --region us-east-1 sesv2 list-export-jobs --export-source-type=METRICS_DATA
```

```
aws --region us-east-1 sesv2 list-export-jobs --job-status=CREATED
```

```
aws --region us-east-1 sesv2 list-export-jobs --cli-input-json file://list-export-jobs-input.json
```

- 입력 파일은 다음과 유사합니다.

```
{
  "NextToken": "",
  "PageSize": 0,
  "ExportSourceType": "METRICS_DATA",
  "JobStatus": "CREATED"
}
```

ListExportJobs 작업의 파라미터 값에 대한 자세한 내용은 Amazon SES API v2 참조에서 찾을 수 있습니다.

를 사용하여 내보내기 작업에 대한 정보를 가져오려면 AWS CLI

Amazon SES API v2에서 GetExportJob 작업을 사용하여 내보내기 작업에 대한 정보를 얻을 수 있습니다. 다음 예제 AWS CLI 와 같이에서이 작업을 호출할 수 있습니다.

- 내보내기 작업에 대한 정보를 얻습니다.

```
aws --region us-east-1 sesv2 get-export-job --job-id=<JobId>
```

```
aws --region us-east-1 sesv2 get-export-job --cli-input-json file://get-export-job-input.json
```

- 입력 파일은 다음과 유사합니다.

```
{
    "JobId": "e2220d6b-dce5-45f2-bf60-3287a465b732"
}
```

GetExportJob 작업의 파라미터 값에 대한 자세한 내용은 Amazon SES API v2 참조에서 찾을 수 있습니다.

를 사용하여 내보내기 작업을 취소하려면 AWS CLI

Amazon SES API v2에서 CancelExportJob 작업을 사용하여 내보내기 작업을 취소할 수 있습니다. 다음 예제 AWS CLI 와 같이에서이 작업을 호출할 수 있습니다.

- 내보내기 작업을 취소합니다.

```
aws --region us-east-1 sesv2 cancel-export-job --job-id=<JobId>
```

```
aws --region us-east-1 sesv2 cancel-export-job --cli-input-json file://cancel-export-job-input.json
```

- 입력 파일은 다음과 유사합니다.

```
{
    "JobId": "e2220d6b-dce5-45f2-bf60-3287a465b732"
}
```

CancelExportJob 작업의 파라미터 값에 대한 자세한 내용은 Amazon SES API v2 참조에서 찾을 수 있습니다.

를 사용하여 메시지의 전체 이벤트 기록 및 ISP 응답 보기 AWS CLI

다음 예제에서는 AWS CLI를 사용하여 메시지의 전체 이벤트 기록에 대한 세부 정보와 사서함 제공업체가 반환한 응답 또는 진단 코드를 보는 방법을 보여줍니다. 이는 가상 배달 가능성 관리자 대시보드의 메시지 테이블에서 메시지의 라디오 버튼을 선택한 후 메시지 정보 사이드바에 사용되는 것과 동일한 데이터입니다.

를 사용하여 메시지의 이벤트 기록 및 ISP 응답을 보려면 AWS CLI

Amazon SES API v2에서 GetMessageInsights 작업을 통해 보낸 메시지의 세부 정보를 볼 수 있습니다. 다음 예제 AWS CLI 와 같이에서이 작업을 호출할 수 있습니다.

- message-id로 식별되는 보낸 이메일에 대한 메시지 세부 정보를 확인합니다.

```
aws --region us-east-1 sesv2 get-message-insights --message-id
 01000100001000dd-2a19190d-99d4-0000-9f00-deb5bbf2bfbe-000001
```

GetMessageInsights 작업의 파라미터 값에 대한 자세한 내용은 Amazon SES API v2 참조에서 찾을 수 있습니다.

가상 배달 가능성 관리자 대시보드 지표가 계산되는 방법

가상 배달 가능성 관리자 대시보드에 표시된 모든 비율 카드와 심층 분석 표는 전체 계정 개요 패널에 입력된 날짜 범위에 대한 지표를 계산합니다.

대시보드에 표시된 지표 비율 백분율은 표에 설명된 대로 계산됩니다. 마지막 네 개의 열은 표시된 지표를 도출하는 데 사용되는 기본 수식에 대한 한정자를 나타냅니다. 예를 들어, 공개 속도는 참여 추적이 켜진 상태에서 전달된 HTML 메시지의 총 조회 횟수를 총 전달 횟수로 나눈 값으로 계산됩니다. 참여 추적 및 HTML 인코딩 없이 발신한 메시지는 반영되지 않습니다.

비율(%)	계산 방법	참여 추적 활성화 및 HTML 사용	및 추적되는 링크 1개 이상	SES FBL을 통해 ISP에 전송	계정 수준 금지 목록에 있는 경우 제외
공개 속도	총 조회 횟수/총 전송 횟수	✓			
클릭률	총 클릭 횟수/총 전송 횟수	✓	✓		
수신 거부율	총 수신 거부 횟수/총 전송 횟수			✓	✓
전송률	총 전송 횟수/총 발신 횟수				
일시 반송률	총 일시 반송 횟수/총 발신 횟수				✓
영구 반송률	총 영구 반송 횟수/총 발신 횟수				✓
총 발신 볼륨	비율(%)이 표시되지 않음(발신한 모든 항목, 항상 100%)				

모든 지표에 대한 차이 비율 및 볼륨 합계가 계산되는 방법:

- 차이(%) – 지정된 날짜 범위에 대해 이전 지표 합계와 비교한 지표 합계의 차이입니다. 예를 들어, 지난 7일이 지정된 날짜 범위인 경우 지난 7일간의 지표 비율 - 이전 7일간의 지표 비율입니다.
  - 총 발신 볼륨의 차이(%)는 다르게 계산됩니다. 예: (지난 7일간의 발신 볼륨 - 이전 7일간의 발신 볼륨)/이전 7일간의 발신 볼륨
- 볼륨 - 각 지표의 총 수입니다.

> **Note**
>
> - 심층 분석 테이블의 Delivered(전송됨) 열에는 조회율, 클릭율 및 수신 거부율을 계산하는 데 사용된 전송됨 구분자 없이 전송됨 볼륨이 바로 표시됩니다.
> - 가상 배달 가능성 관리자는 수신자가 한 명인 이메일의 지표만 추적합니다. 수신자가 여러 명인 이메일은 가상 배달 가능성 관리자 대시보드 지표에 포함되지 않습니다.

- 이러한 경우 CloudWatch 지표에는 수신자가 여러 명인 이메일이 포함되므로 가상 배달 가능성 관리자 지표 수가 Amazon CloudWatch 지표 수보다 적어집니다.
- SES 메일박스 시뮬레이터로 발신된 이메일은 가상 배달 가능성 관리자 대시보드 지표에 포함되지 않습니다.
- 위임 발신자 계정을 통해 전송된 이메일(이전의 교차 계정 전송)은 가상 배달 가능성 관리자 대시보드 지표에 포함되지 않습니다.

Important

Apple Mail의 개인 정보 보호 및 참여율에 미치는 영향: iOS15부터 Apple이 Apple 기기에 메일 개인 정보 보호(MPP) 기능을 구현한 결과, 수신자가 메일을 열 때 및/또는 클릭할 때가 아니라 Apple Mail 앱을 시작할 때 MPP가 조회를 트리거하므로 참여 수가 부풀렸습니다. 이로 인해 참여 데이터가 평소보다 훨씬 더 높게 표시되므로 이메일 마케터는 참여를 검토할 때 이를 고려해야 합니다. 웹 활동, 앱/포털 사용, Apple 외 기기의 프록시 데이터를 사용하여 집계 지표를 구축하는 등 참여를 식별하는 여러 가지 다른 방법이 있습니다. 집중해야 할 중요한 것은 이메일 발신에 문제가 있는지 여부를 나타낼 수 있는 참여 트렌드입니다. 자세한 내용은 Apple Mail의 개인 정보 보호를 참조하세요.

가상 배달 가능성 관리자 어드바이저

가상 배달 가능성 관리자 어드바이저는 이메일 전달 가능성 및 평판에 부정적인 영향을 미치는 계정 및 전송 보안 인증 수준의 주요 성능 및 인프라 문제를 식별하여 이메일 전달 가능성 및 참여를 최적화하도록 도와줍니다. 식별된 문제를 해결하는 방법에 대한 구체적인 지침을 통해 솔루션을 제공합니다.

어드바이저의 인프라 권장 사항은 Open recommendations(미해결 권장 사항) 표에 나와 있습니다. 권장 사항은 SPF, DKIM, DMARC 또는 BIMI 레코드의 부재나 형식 오류, 너무 짧은 키 길이와 같은 구성 문제가 있는 경우와 같은 표준 이메일 인증 문제를 식별합니다. 문제는 영향의 심각도, 발신 도메인의 자격 증명 이름 및 알림 발생 후 경과 시간별로 분류됩니다. 검색 표시줄의 목록 상자는 영향 수준, 인프라 범주 또는 전송 보안 인증 이름을 기준으로 필터링할 수 있는 옵션을 제공합니다. Last checked(마지막 확인) 열에는 권장 사항이 마지막으로 업데이트된 상대적 시간(예: 'Just now'(방금) 또는 '15 minutes ago'(15분 전))가 표시됩니다. 마지막 열인 Resolve issue(문제 해결)는 식별된 문제를 해결하는 방법에 대한 지침과 함께 Amazon SES 개발자 안내서의 관련 섹션으로 연결되는 링크를 제공합니다.

미해결 권장 사항은 영향 수준별로 정렬되어 가상 배달 가능성 관리자 어드바이저에 표시됩니다.

진행 중인 어드바이저 알림이 없는 경우 진행 중인 권장 사항이 없다는 메시지가 표시됩니다. 정기적으로 어드바이저를 확인하는 것이 좋습니다. 필요에 따라 이러한 어드바이저 알림 이벤트를 Amazon EventBridge와 통합하여 EventBridge를 사용하여 모니터링에 설명된 대로 확장 가능한 이벤트 기반 애플리케이션을 구축할 수 있습니다.

가상 배달 가능성 관리자 어드바이저 페이지에서 해결된 권장 사항 표에 액세스할 수도 있습니다. 이 표에는 어드바이저 지침을 구현하여 해결한 인프라 문제가 나열되어 있습니다. 해결된 권장 사항은 해결되기 전의 문제를 설명하는 초기 상태와 함께 나열됩니다. 해결된 권장 사항은 30일 후에 만료됩니다.

Virtual Deliverability Manager 어드바이저가 찾는 항목

이전 섹션에서는 Virtual Deliverability Manager의 어드바이저가 전송 도메인에 대한 검사를 수행하여 이메일 전달률을 높이고 발신자 평판을 양호하게 유지할 수 있도록 안전하게 인증된 인프라를 구성했는지를 판단한다고 설명했습니다. Virtual Deliverability Manager 어드바이저를 활성화하기 전에 어드바이저가 무엇을 확인하고, 검사 과정에서 무슨 항목을 찾는지 정확히 알면 도움이 됩니다.

이 테이블을 참조하여 전송 도메인의 구성을 살펴보고, 여기에 나와 있는 표준에 맞지 않는 요소가 있으면 어드바이저가 사용자에게 알려야 하는 문제가 되기 전에 수정할 수 있습니다.

검사 유형	어드바이저 메시지	어드바이저가 알림을 보내는 이유	자세히 알아보기
수신 거부 비율 확인	*ISP_name* ISP의 수신 거부 비율이 *##/##/##*입니다.	자격 증명이 이 ISP에 대한 수신 거부 권장 임계값을 초과했습니다.	발신자 평판 모니터링
DKIM 구성	DKIM 확인이 활성화되지 않았습니다.	DKIM은 자격 증명별로 활성화되지 않습니다.	SES의 Easy DKIM
DKIM 키 강도	DKIM 서명 키 길이가 2048비트 미만입니다.	DKIM 서명 키 길이는 2048비트 이상을 사용하지 않습니다.	SES의 Easy DKIM
DKIM DNS 레코드 검증	DKIM 확인에 실패했습니다.	키를 찾아 검증한 후 DKIM CNAME 레코드가 유효하지 않은 것으로 판단되었습니다.	DNS 공급자로 DKIM 도메인 자격 증명 확인
DMARC 구성	DMARC 구성을 찾을 수 없습니다.	DMARC TXT 레코드가 누락되었습니다.	도메인의 DMARC 정책 설정
DMARC DNS 레코드 형식 검사	DMARC 구성을 구문 분석할 수 없습니다.	DMARC TXT 레코드에 잘못된 형식이 발견되었습니다.	도메인의 DMARC 정책 설정
DMARC의 DKIM 구성	DKIM 레코드를 찾을 수 없습니다.	DMARC를 준수하기 위한 DKIM 레코드를 찾을 수 없습니다.	DKIM을 통해 DMARC 준수
DMARC의 DKIM 구성	DKIM 레코드가 정렬되지 않았습니다.	DKIM 서명에 지정된 도메인이 From 주소의 도메인과 정렬(일치)되지 않습니다.	DKIM을 통해 DMARC 준수
SPF 구성	SPF 레코드를 찾을 수 없습니다.	사용자 지정 MAIL FROM 도메인에 대한	사용자 지정 MAIL FROM 도메인 구성

검사 유형	어드바이저 메시지	어드바이저가 알림을 보내는 이유	자세히 알아보기
		SPF TXT 레코드가 누락되었습니다.	
SPF '포함' 구성됨	Amazon SES에 대한 SPF 레코드를 찾을 수 없습니다.	`include:amazonses.com` 이 SPF TXT 레코드에서 누락되었습니다.	사용자 지정 MAIL FROM 도메인 구성
SPF 적용이 구성됨	모든 SPF 한정자가 누락되었습니다.	`~all`이 SPF TXT 레코드에서 누락되었습니다.	사용자 지정 MAIL FROM 도메인 구성
SPF 적용 검증	SPF 구성 문제가 발견되었습니다.	72시간 내에 필요한 SPF MX 레코드를 감지하려는 시도가 실패했습니다.	사용자 지정 MAIL FROM 도메인 설정 상태
BIMI 구성됨	BIMI 레코드를 찾을 수 없거나 기본 선택기 없이 구성되었습니다.	BIMI TXT 레코드가 누락되었거나 선택기 속성이 없습니다.	BIMI 설정
BIMI 형식 검증	BIMI의 TXT 레코드 형식이 잘못되었습니다.	버전, 인증서 URL 및 로고 URL의 존재 및 유효한 형식을 확인한 후 BIMI TXT 레코드가 잘못 구성된 것으로 확인되었습니다.	BIMI 설정

Amazon SES 콘솔에서 가상 배달 가능성 관리자 어드바이저 사용

다음 절차에서는 Amazon SES 콘솔의 가상 배달 가능성 관리자 어드바이저를 사용하여 Amazon SES 콘솔을 통해 식별된 전달 문제를 해결하는 방법을 보여 줍니다.

가상 배달 가능성 관리자 어드바이저를 사용하여 전달 및 평판 문제를 해결하려면 다음과 같이 하세요.

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 가상 배달 가능성 관리자 아래의 어드바이저를 선택합니다.

> **Note**
> 계정에서 가상 배달 가능성 관리자를 활성화하지 않은 경우 어드바이저가 표시되지 않습니다. 자세한 내용은 the section called "시작" 단원을 참조하십시오.

3. 기본적으로 미해결 권장 사항 테이블이 표시됩니다. 권장 사항은 영향(높음/낮음), ID 이름(발신 도메인), 알림 발생 후 경과 시간 및 권장 사항/설명(식별된 문제)별로 분류됩니다. 검색 표시줄에서 영향 수준, 인프라 문제 범주 또는 발신 도메인의 ID 이름을 기준으로 필터링합니다.
4. Recommendation/Description(권장 사항/설명) 열에 설명된 문제를 해결하려면 해당 행의 Resolve issue(문제 해결) 열에서 링크를 선택하고 제안된 솔루션을 구현하세요.

> **Note**
> 솔루션을 구현한 후 해결된 문제가 반영되기까지 최대 6시간이 걸릴 수 있습니다. Resolved recommendations(해결된 권장 사항) 탭에서 해결된 문제를 볼 수 있습니다.

AWS CLI를 사용하여 가상 배달 가능성 관리자 권장 사항에 액세스

다음 예제에서는 AWS CLI를 사용하여 가상 배달 가능성 관리자 권장 사항에 액세스하는 방법을 보여줍니다.

를 사용하여 가상 배달 가능성 관리자 권장 사항에 액세스하려면 AWS CLI

Amazon SES API v2에서 `ListRecommendations` 작업을 사용하여 전달 가능성 권장 사항을 나열할 수 있습니다. 다음 예제에 표시된 대로 AWS CLI에서 이 작업을 호출할 수 있습니다.

- 권장 사항을 나열하여 전달 가능성 문제를 확인합니다.

```
aws --region us-east-1 sesv2 list-recommendations
```

- 필터를 적용하여 소유한 특정 도메인에 대한 추천을 검색합니다.

```
aws --region us-east-1 sesv2 list-recommendations --cli-input-json file://list-
recommendations.json
```

- 입력 파일은 다음과 유사합니다.

```
{
  "PageSize":100,
  "Filter":{
    "RESOURCE_ARN": "arn:aws:ses:us-east-1:123456789012:identity/example.com"
   }
}
```

가상 전달 가능성 관리자 설정

언제든지 계정에서 가상 전달 가능성 관리자 설정을 보거나 변경할 수 있습니다. 가상 배달 가능성 관리자를 활성화 또는 비활성화할 수 있으며 Amazon SES 콘솔 또는를 통해 가상 배달 가능성 관리자 계정 수준에서 참여 추적 및 최적화된 공유 전송을 위한 켜기 또는 끄기 모드를 지정할 수 있습니다. AWS CLI

가상 전달 가능성 관리자 옵션도 구성 세트 수준에서 제공되므로 가상 전달 가능성 관리자에서 정의한 방식을 재정의하여 구성 세트가 참여 추적 및 최적화된 공유 전송을 사용하는 방법에 대한 사용자 지정 설정을 정의할 수 있습니다. 이를 통해 특정 이메일 캠페인에 맞게 이메일 전송을 유연하게 조정할 수 있습니다. 예를 들어 마케팅 이메일에 대해 참여 추적 및 최적화된 공유 전송을 활성화하고 트랜잭션 이메일에 대해서는 비활성화할 수 있습니다.

Amazon SES 콘솔을 사용하여 가상 전달 가능성 관리자 계정 설정 변경

다음 절차에서는 Amazon SES 콘솔을 사용하여 가상 전달 가능성 관리자 계정 설정을 변경하는 방법을 보여 줍니다.

Amazon SES 콘솔을 사용하여 가상 전달 가능성 관리자 계정 설정 변경

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Virtual Deliverability Manager(가상 전달 가능성 관리자) 아래의 Settings(설정)를 선택합니다.

Virtual Deliverability Manager settings(가상 전달 가능성 관리자 설정) 페이지가 열립니다. Subscription overview(구독 개요) 패널에는 가상 전달 가능성 관리자의 상태가 표시되고 Additional settings(추가 설정) 패널에는 Engagement tracking(참여 추적) 및 Optimized shared delivery(최적화된 공유 전송)의 상태가 표시됩니다.

3. Engagement tracking(참여 추적) 또는 Optimized shared delivery(최적화된 공유 배달) 설정을 변경하려면:

   a. Additional settings(추가 설정) 패널에서 Edit(편집)을 선택합니다.

   b. 해당 라디오 버튼을 선택하여 기능을 켜거나 끈 다음 Submit settings(설정 제출)를 선택합니다.

   Virtual Deliverability Manager settings(가상 전달 가능성 관리자 설정) 페이지에는 Additional settings(추가 설정) 패널의 변경 내용이 요약되어 표시됩니다.

   > **Note**
   >
   > 여기 또는 Virtual Deliverability Manager의 구성 세트 재정의에서 정의하는 참여 추적 옵션은 Virtual Deliverability Manager 대시보드에서 열기 및 클릭을 보고할지 여부를 제어하며, 열기 및 클릭 이벤트를 게시하는 이벤트 대상 구성에는 영향을 미치지 않습니다. 예를 들어 여기에서 참여 추적을 비활성화한 경우 SES 이벤트 대상에서 설정한 열기 및 클릭 이벤트 게시가 비활성화되지 않습니다.

4. (선택 사항) 가상 전달 가능성 관리자에서 정의된 방식을 재정의하여 구성 세트가 참여 추적 및 최적화된 공유 전송을 사용하는 방식에 대한 사용자 지정 설정을 정의하려면 구성 세트를 만들거나 편집할 때 가상 전달 가능성 관리자 옵션을 참조하세요.

5. 가상 전달 가능성 관리자를 비활성화하려면 다음과 같이 하세요.

   a. Subscription overview(구독 개요) 패널에서 Disable Virtual Deliverability Manager(가상 전달 가능성 관리자 비활성화)를 선택합니다.

   b. Disable Virtual Deliverability Manager?(가상 전달 가능성 관리자를 비활성화할까요?) 팝업 창의 확인 필드에 `Disable`을 입력하고 Disable Virtual Deliverability Manager(가상 전달 가능성 관리자 비활성화)를 선택합니다.

   c. 가상 전달 가능성 관리자를 비활성화했음을 확인하는 배너가 나타납니다.

6. 가상 전달 가능성 관리자를 다시 활성화하려면 the section called “시작” 섹션을 참조하세요.

AWS CLI를 사용하여 가상 전달 가능성 관리자 계정 설정 변경

AWS CLI를 사용하여 가상 전달 가능성 관리자 계정 설정을 변경할 수 있습니다.

를 사용하여 가상 배달 가능성 관리자 계정 설정을 변경하려면 AWS CLI

Amazon SES API v2의 `PutAccountVdmAttributes` 및 `PutConfigurationSetVdmOptions` 작업을 사용하여 가상 전달 가능성 관리자 설정을 변경할 수 있습니다. 다음 예제와 AWS CLI같이에서이 작업을 호출할 수 있습니다.

- 다음과 같이 입력 파일을 사용하여 참여 추적, 최적화된 공유 전송 또는 두 가지 모두를 활성화하거나 비활성화합니다.

  ```
  aws --region us-east-1 sesv2 put-account-vdm-attributes --cli-input-json file:// attributes.json
  ```

  참여 추적은 `ENABLED`, 최적화된 공유 전송은 `DISABLED` 상태인 이 예제에서 입력 파일은 다음과 유사합니다.

  ```
  {
      "VdmAttributes": {
          "VdmEnabled": "ENABLED",
          "DashboardAttributes": {
              "EngagementMetrics": "ENABLED"
          },
          "GuardianAttributes": {
              "OptimizedSharedDelivery": "DISABLED"
          }
      }
  }
  ```

  Amazon SES API v2 참조의 `VdmAttributes` 데이터 유형에서 연결하여 파라미터 값 및 관련 데이터 유형에 대한 자세한 정보를 찾을 수 있습니다.

- 가상 전달 가능성 관리자에서 정의한 방식을 재정의하여 구성 세트가 참여 추적 및 최적화된 공유 전송을 사용하는 방식에 대한 사용자 지정 설정을 정의하세요.

  ```
  aws --region us-east-1 sesv2 put-configuration-set-vdm-options --cli-input-json file://config-set.json
  ```

example이라는 이름의 구성 세트에 참여 추적과 최적화된 공유 전송이 모두 활성화된 상태의 이 예제에서 입력 파일은 다음과 유사합니다.

```
{
    "ConfigurationSetName": "example",
    "VdmOptions": {
        "DashboardOptions": {
            "EngagementMetrics": "ENABLED"
        },
        "GuardianOptions": {
            "OptimizedSharedDelivery": "ENABLED"
        }
    }
}
```

파라미터 값 및 관련 데이터 유형에 대한 자세한 내용은 Amazon SES API v2 참조의 `VdmOptions` 데이터 유형을 참조하세요.

- 결과 확인:

```
aws –region us-east-1 sesv2 get-configuration-set –configuration-set-name example
```

- 구성 세트 수준에서 `DashboardOptions` 또는 `GuardianOptions` 옵션을 지정하지 않으면 가상 전달 가능성 관리자의 계정 수준 설정이 해당 구성 세트를 통해 전송된 트래픽에 적용됩니다.

Amazon SES용 Mail Manager

Mail Manager는 조직의 이메일 인프라를 강화하고, 이메일 워크플로 관리를 단순화하고, 이메일 규정 준수 제어를 간소화하는 데 도움이 되도록 설계된 Amazon SES 이메일 게이트웨이 기능입니다. 기존 인프라와 통합되고, 다양한 비즈니스 애플리케이션을 연결할 수 있으며, 인바운드 이메일 처리를 자동화합니다. 나아가 Mail Manager는 이메일 트래픽을 효율적으로 관리하고 이메일 아카이브 기능 준수를 개선하여 양호한 이메일 시스템을 유지하는 1차 방어선 역할을 합니다.

현재 Amazon SES 기능과 함께 Mail Manager는 인바운드 트래픽을 지원하는 다음 기능으로 구성됩니다.

- 수신 엔드포인트 - 필터링 정책 및 규칙을 활용하는 주요 인프라 구성 요소로, 조직에 허용해야 하는 이메일과 거부해야 하는 이메일을 결정하도록 구성할 수 있습니다.
- 트래픽 정책 및 규칙 세트 - 사용자가 정의한 다양한 조건 및 예외를 기반으로 이메일 관리자가 이메일을 정렬하고, 분류하고, 우선순위를 지정하고, 이메일에 대한 작업을 수행할 수 있는 맞춤 설정 가능한 정책 및 규칙을 통해 인바운드 이메일 트래픽을 관리하기 위한 규칙을 정의하고 적용할 수 있습니다. 이 지능형 필터링과 자동화된 워크플로를 결합하면 이메일 더욱 쉽게 관리하고 효율성을 높이며 조직 이메일 정책을 준수할 수 있습니다.
- SMTP 릴레이 - 내부 이메일 시스템을 연결하여 규칙에 정의한 기준에 따라 이메일 트래픽을 다른 SMTP 서버로 리디렉션하고 자동 전달을 통해 이메일 관리를 간소화합니다. 여러 서버와 게이트웨이에 트래픽을 분산할 수 있으므로, 조직은 하이브리드 환경에서도 대량의 이메일 트래픽을 효과적으로 관리할 수 있습니다.
- 이메일 아카이브 - 영구적이고 안전한 장기 스토리지에 데이터를 저장하여 이메일을 저장하고 보호하며, 이메일을 빠르게 검색하고 아카이브할 수 있는 방법을 제공합니다. 메일박스 서버의 스토리지 요구 사항을 늘리지 않고 풀타임 엔터프라이즈 수준 아카이브를 제공합니다.
- 이메일 추가 기능 - 수신 엔드포인트로 들어오는 이메일을 관리하고 보안 결과에 따라 라우팅 옵션을 제공하는 데 사용할 수 있는 SES 승인 공급자의 특수 보안 도구 모음입니다. 이러한 도구는 이메일 워크플로에 통합할 준비가 되어 있고 Mail Manager 콘솔에서 직접 활성화할 수 있는 인증된 보안 인텔리전스 및 적용 솔루션입니다.

Mail Manager 시작하기

Mail Manager를 사용하기 시작하려면 Amazon SES 콘솔의 온보딩 마법사가 안내하는 계정에 Mail Manager를 활성화하는 단계를 따릅니다. the section called “시작”을(를) 참조하세요.

주제

- Mail Manager 시작하기
- 수신 엔드포인트
- 트래픽 정책 및 정책 스테이트먼트
- 규칙 세트 및 규칙
- SMTP 릴레이
- 주소 목록
- 이메일 아카이빙
- 이메일 추가 기능
- Mail Manager에 대한 권한 정책
- Mail Manager 로깅

Mail Manager 시작하기

Amazon SES Mail Manager 사용을 시작하려면 Amazon SES 콘솔에서 Mail Manager 시작 마법사를 사용하여 수신 엔드포인트를 만들고 트래픽 정책 및 규칙 세트로 이를 구성할 수 있습니다.

수신 엔드포인트는 Mail Manager를 설정하는 첫 번째 빌딩 블록이며 다음을 활용하는 주요 인프라 구성 요소입니다.

- 트래픽 정책 - 트래픽 정책에는 정책 스테이트먼트 조건이 충족될 경우 특정 유형의 이메일을 허용하거나 차단하여 수신 메일을 정렬하도록 정의하는 정책 스테이트먼트가 포함되어 있습니다.
- 규칙 세트 - 규칙 세트에는 규칙 조건이 충족될 경우 허용하는 이메일에 대한 작업을 수행하도록 정의하는 규칙이 포함되어 있습니다.

그러나 수신 엔드포인트를 만드는 과정의 일부로 이미 생성된 트래픽 정책 및 규칙 세트를 선택한 다음, 수신 엔드포인트에 할당합니다. 다음 절차의 단계에서는 올바른 순서를 통해 첫 번째 수신 엔드포인트를 구성하는 방법을 알아봅니다.

SES 콘솔을 사용하여 Mail Manager 시작하기

다음 절차에서는 SES 콘솔을 사용하여 Mail Manager를 시작하는 방법을 보여줍니다.

Amazon SES 콘솔을 사용하여 Mail Manager 시작하기

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.

2. 왼쪽 탐색 패널에서 Mail Manager를 선택하고 Mail Manager 개요 페이지의 Mail Manager 시작하기 버튼 중 하나를 선택합니다.
3. 설정 시작 페이지의 트래픽 정책 만들기 카드에서 트래픽 정책 만들기를 선택합니다.
   a. 트래픽 정책 만들기 페이지에서 워크플로를 완료합니다. 추가 정보가 필요한 경우 the section called "트래픽 정책 및 정책 스테이트먼트 만들기(콘솔)" 섹션을 참조하세요.
   b. 첫 번째 트래픽 정책 및 정책 스테이트먼트를 만든 후 브라우저의 뒤로 버튼을 사용하여 설정 시작 페이지로 돌아가거나 왼쪽 탐색 패널의 Mail Manager에서 설정 시작을 선택합니다.
4. 설정 시작 페이지의 규칙 세트 만들기 카드에서 규칙 세트 만들기를 선택합니다.
   a. 규칙 세트 만들기 페이지에서 워크플로를 완료합니다. 추가 정보가 필요한 경우 the section called "규칙 세트 및 규칙 만들기(콘솔)" 섹션을 참조하세요.
   b. 첫 번째 규칙 세트 및 규칙을 만든 후 브라우저의 뒤로 버튼을 사용하여 설정 시작 페이지로 돌아가거나 왼쪽 탐색 패널의 Mail Manager에서 설정 시작을 선택합니다.
5. 이제 첫 번째 트래픽 정책 및 규칙 세트를 만들었으므로 첫 번째 수신 엔드포인트를 만들 수 있습니다. 설정 시작 페이지의 수신 엔드포인트 만들기 카드에서 수신 엔드포인트 만들기를 선택합니다.
   - 이메일 수신 엔드포인트 페이지의 워크플로에서 수행할 작업에는 방금 만든 트래픽 정책 및 규칙 세트를 수신 엔드포인트에 할당하는 것이 포함됩니다. 추가 정보가 필요한 경우 the section called "수신 엔드포인트 만들기(콘솔)" 섹션을 참조하세요.

첫 번째 수신 엔드포인트가 만들어지면 Mail Manager 사용을 시작하고 SMTP 릴레이 및 이메일 아카이빙 등의 다른 기능을 활용할 수 있습니다. 또한 고유한 트래픽 정책 및 규칙 세트를 사용하여 추가 수신 엔드포인트를 만들어 모든 수신되는 이메일을 관리하는 방법을 추가로 사용자 지정할 수 있습니다.

수신 엔드포인트

수신 엔드포인트는 Mail Manager의 주요 인프라 구성 요소로, 사용자가 구성한 정책 및 규칙을 활용하여 거부해야 할 이메일, 허용해야 할 이메일, 조치해야 할 이메일을 결정함으로써 이메일을 수신, 라우팅, 관리합니다.

각 수신 엔드포인트에는 차단하거나 허용할 이메일을 결정하는 자체 트래픽 정책과 사용자가 허용하는 이메일에 대한 작업을 수행하도록 설정된 자체 규칙이 있습니다. 따라서 여러 수신 엔드포인트를 만들어 각 엔드포인트를 위임하여 특정 유형의 이메일을 관리하고 라우팅할 수 있습니다. 이렇게 세분화하면 비즈니스 요구 사항에 맞춤화된 이메일 관리 시스템을 구축할 수 있습니다.

수신 엔드포인트를 만들기 위한 사전 워크플로

수신 엔드포인트를 만들 때 트래픽 정책 및 이미 만들어진 규칙 세트를 할당해야 합니다. 따라서 수신 엔드포인트를 만드는 데 필요한 워크플로는 다음 순서로 진행되어야 합니다.

1. 먼저 트래픽 정책을 만들어 차단하거나 허용할 이메일을 결정합니다. 자세한 내용은 the section called “트래픽 정책 및 정책 스테이트먼트 만들기(콘솔)”을 참조하세요.
2. 그런 다음 규칙 세트를 만들어 사용자가 허용하는 이메일에 대한 작업을 수행합니다. 자세한 내용은 the section called “규칙 세트 및 규칙 만들기(콘솔)”을 참조하세요.
3. 마지막으로 수신 엔드포인트를 만들고 방금 만든 트래픽 정책 및 규칙 세트 또는 이전에 만든 다른 버전을 해당 엔드포인트에 할당합니다.

수신 엔드포인트를 만든 후에는 온프레미스 SMTP 클라이언트의 구성이든 웹 기반 DNS 도메인 호스트의 구성이든 이메일을 수신하는 데 사용하는 환경으로 구성해야 합니다. 자세한 내용은 the section called “퍼블릭 엔드포인트 구성” 섹션을 참조하세요.

수신 엔드포인트를 사용하도록 환경 구성

SES는 수신 엔드포인트가 수신 이메일을 수락할 수 있도록 퍼블릭 엔드포인트와 Amazon Virtual Private Cloud(VPC) 엔드포인트를 모두 지원합니다. 다음 섹션에서는 이러한 옵션 중 하나를 사용하도록 수신 엔드포인트를 구성하는 방법을 설명합니다.

주제

- 퍼블릭 엔드포인트를 통해 이메일 수신
- Amazon VPC 엔드포인트를 통해 이메일 수신

퍼블릭 엔드포인트를 통해 이메일 수신

'A' 레코드 사용

수신 엔드포인트를 만들 때 엔드포인트에 대한 'A' 레코드가 생성되고 해당 값이 SES 콘솔의 수신 엔드포인트 요약 화면에 표시됩니다. 이 레코드의 값을 사용하는 방법은 만든 엔드포인트 유형과 사용 사례에 따라 달라집니다.

- 개방형 엔드포인트 - 도메인으로 전송된 메일은 수신 엔드포인트로 직접 확인되므로 인증이 필요하지 않습니다.
  - 'A' 레코드의 값을 직접 복사하여 온프레미스 SMTP 클라이언트의 SMTP 구성에 붙여넣거나 DNS 구성의 도메인에 대한 MX 레코드에 붙여넣습니다.

  - 지원되는 포트: 25
  - STARTTLS 지원: 예
- 인증된 엔드포인트 - 도메인으로 전송된 메일은 온프레미스 이메일 서버와 같이 SMTP 자격 증명을 공유한 인증된 발신자가 보낸 것이어야 합니다.
  - 'A' 레코드의 값을 직접 복사하여 온프레미스 SMTP 클라이언트의 SMTP 구성과 사용자 이름 및 암호에 붙여넣습니다.
  - 지원되는 포트: 25, 587(RFC 2476)
  - STARTTLS 지원: 예

구성에서 MX 레코드를 사용하는 경우 각 DNS 공급자가 레코드를 구성하기 위한 다양한 절차와 인터페이스를 보유하고 있지만 DNS 설정에 입력해야 하는 주요 정보가 다음 예제에 나열되어 있다는 것을 참고하시기 바랍니다.

도메인의 DNS 설정에서 수신 엔드포인트의 'A' 레코드를 MX 레코드 값으로 입력했으므로 recipient@marketing.example.com으로 전송된 모든 이메일은 수신 엔드포인트로 이동합니다.

- 도메인 – `marketing.example.com`
- MX 레코드 값 - `890123abcdef.ghijk.mail-manager-smtp.amazonaws.com`(이 값은 수신 엔드포인트에서 복사된 'A' 레코드 값입니다.)
- 우선순위 – `10`

인증된 엔드포인트에 연결

인증된 엔드포인트에 연결하기 위해 SMTP 자격 증명을 공유한 인증된 발신자의 경우 서버에 성공적으로 연결하려면 사용자 이름과 암호에 대해 다음 프로토콜을 따라야 합니다.

- 사용자 이름 - 수신 엔드포인트 ID이며 Base64에 인코딩되어야 합니다. (수신 엔드포인트 ID를 찾는 방법을 알아보려면 콘솔 절차의 11단계를 참조하세요.)
- 암호 - 수신 엔드포인트를 만드는 중에 사용되는 암호이며 Base64에서 인코딩되어야 합니다.

다음 예제에서는 일반적인 SMTP AUTH 서버 및 클라이언트 교환 연결 설정 방법을 보여줍니다.

```
S: 250 AUTH LOGIN PLAIN
C: AUTH LOGIN
S: 334 VXNlcm5hbWU6
C: SW5ncmVzc1BvaW50
```

```
S: 334 UGFzc3dvcmQ6
C: SW5ncmVzc1Bhc3N3b3Jk
S: 235 Authentication successful
```

이 예제에는 다음 속성이 포함됩니다.

- `S`는 메시지를 수락하는 SMTP 서버인 '서버'를 의미합니다.
- `C`는 '클라이언트'를 의미하며, SMTP 클라이언트는 서버와의 연결을 설정하고 서버에 메시지를 전송합니다.
- `250 AUTH LOGIN PLAIN`은 AUTH 메서드가 지원되는 서버의 응답이거나, `AUTH PLAIN` 또는 `AUTH LOGIN`, 발신자는 이중 하나를 선택하고 SMTP Service Extension for Authentication 사양 RFC 2554를 준수하는 SMTP 명령을 보낼 수 있습니다. `AUTH LOGIN`는 여기에서 사용됩니다.
- `334 VXNlcm5hbWU6` – Base64에서 사용자 이름을 프롬프트하는 서버입니다.
- `SW5ncmVzc1BvaW50` - Base64에서 수신 엔드포인트 ID로 응답하는 클라이언트입니다.
- `334 UGFzc3dvcmQ6` – Base64에서 암호를 프롬프트하는 서버입니다.
- `SW5ncmVzc1Bhc3N3b3Jk` – Base64에서 수신 엔드포인트 암호로 응답하는 클라이언트입니다.

Amazon VPC 엔드포인트를 통해 이메일 수신

퍼블릭 수신 엔드포인트 외에도 VPC 엔드포인트를 SES 수신 엔드포인트와 함께 사용하여 프라이빗 네트워크 인프라 내에서 안전한 프라이빗 이메일 수집을 수행할 수 있습니다.

퍼블릭 수신 엔드포인트 사용과 비교한 구성 차이

- 퍼블릭 엔드포인트에 일반적으로 사용할 수 있는 "A" 레코드는 제공되지 않습니다.
- VPC 엔드포인트에서 제공하는 DNS 이름을 사용하여 수신 엔드포인트에 연결해야 합니다.
- 모든 연결은 VPC 내에서 프라이빗 네트워킹을 사용합니다.

VPC 엔드포인트를 통해 지원되는 수신 엔드포인트 유형

SES는 VPC 엔드포인트를 통해 두 가지 유형의 수신 지점을 지원합니다.

- 수신 엔드포인트 열기 - 발신자 인증 없이 VPC 엔드포인트를 통해 도메인으로 직접 전송되는 이메일입니다.

  구성 요구 사항:

  - 소유한 VPC 엔드포인트 ID와 연결하여 프라이빗 오픈 수신 엔드포인트를 생성합니다.
  - 지원되는 포트: 25, 587
  - STARTTLS 지원: 예
- 인증된 수신 엔드포인트 - 도메인으로 전송된 메일은 온프레미스 이메일 서버와 같이 SMTP 자격 증명을 공유한 승인된 발신자로부터 와야 합니다.

  구성 요구 사항:

  - 소유한 VPC 엔드포인트 ID와 연결하여 프라이빗 인증 수신 엔드포인트를 생성합니다.
  - 지원되는 포트: 25, 587
  - STARTTLS 지원: 예
  - 인증은 퍼블릭 인증 엔드포인트와 동일한 base64 인코딩 사용자 이름 및 암호 메커니즘을 사용합니다.

VPC 엔드포인트 요구 사항

VPC 엔드포인트를 SES 수신 엔드포인트와 함께 사용하려면 다음 요구 사항을 충족해야 합니다.

- VPC 엔드포인트는 활성 상태여야 하며 사용 가능해야 합니다.
- VPC 엔드포인트는 수신 엔드포인트와 동일한 AWS 계정이 소유해야 합니다(교차 계정 액세스는 지원되지 않음).
- VPC 엔드포인트는 수신 엔드포인트 유형에 따라 적절한 서비스 이름에 대해 생성해야 합니다.
  - 수신 엔드포인트 열기 - `com.amazonaws.`*`region`*`.mail-manager-smtp.open`
  - 인증된 수신 엔드포인트 - `com.amazonaws.`*`region`*`.mail-manager-smtp.auth`
  - FIPS 열린 수신 엔드포인트 - `com.amazonaws.`*`region`*`.mail-manager-smtp.open.fips`
  - FIPS 인증 수신 엔드포인트 – `com.amazonaws.`*`region`*`.mail-manager-smtp.auth.fips`

중요 구성 정보

- 미국 및 캐나다 리전 - FIPS 엔드포인트는 미국 및 캐나다 리전에서 사용할 수 있는 유일한 VPC 엔드포인트이며 이러한 리전에서 사용할 수 있는 올바른 엔드포인트입니다.
- One-to-one 관계 - 각 VPC 엔드포인트는 단일 수신 엔드포인트에만 연결할 수 있습니다. 여러 수신 엔드포인트에 동일한 VPC 엔드포인트를 사용할 수 없습니다.

- VPC 엔드포인트 정책 없음 - 다른 AWS 서비스와 달리 수신 엔드포인트와 함께 사용되는 VPC 엔드포인트는 VPC 엔드포인트 정책을 지원하지 않습니다. SES는 VPC 엔드포인트 소유자와 수신 엔드포인트 소유자가 동일한 AWS 계정인지 자동으로 확인합니다.
- 프라이빗 DNS만 - VPC 엔드포인트에서 제공하는 모든 DNS 이름은 VPC 내에서만 액세스할 수 있는 프라이빗 DNS 이름입니다.
- 생성 시 검증 - SES는 리소스 생성 중에 검증을 수행하여 VPC 엔드포인트가 모든 요구 사항을 충족하는지 확인합니다.

VPC 엔드포인트를 통해 수신 엔드포인트에 연결

VPC 엔드포인트 및 수신 엔드포인트를 구성한 후:

1. VPC 엔드포인트에 대해 생성된 DNS 이름을 검색합니다.
2. 이러한 DNS 이름을 연결에 사용하도록 SMTP 클라이언트 또는 이메일 서버를 구성합니다.
3. 인증된 엔드포인트를 사용하는 경우 인증된 수신 엔드포인트에 사용되는 적절한 base64 인코딩 자격 증명으로 SMTP 클라이언트를 구성합니다.

SES 콘솔에서 수신 엔드포인트 만들기

다음 절차에서는 SES 콘솔의 수신 엔드포인트 페이지를 사용하여 수신 엔드포인트를 만들고 이미 만든 엔드포인트를 관리하는 방법을 보여줍니다.

콘솔을 사용하여 수신 엔드포인트 관리를 만드는 방법

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 패널의 Mail Manager에서 수신 엔드포인트를 선택합니다.
3. 수신 엔드포인트 페이지에서 수신 엔드포인트 만들기를 선택합니다.
4. 새 수신 엔드포인트 만들기 페이지에서 수신 엔드포인트의 고유한 이름을 입력합니다.
5. 공개 또는 인증됨 엔드포인트인지 여부를 선택합니다.
   - 인증됨을 선택한 경우 SMTP 암호 및 암호(인증된 발신자와 공유)를 선택하거나 보안 암호를 선택하고 보안 암호 ARN에서 보안 암호 중 하나를 선택합니다. 이전에 생성한 보안 암호를 선택하는 경우 새 보안 암호를 만들려면 다음 단계에 표시된 정책이 포함되어 있어야 합니다.
   - 새로 생성을 선택하여 새 보안 암호를 생성할 수 있습니다. AWS Secrets Manager 콘솔이 열리고 새 키를 계속 생성할 수 있습니다.

a. 보안 암호 유형에서 다른 유형의 보안 암호를 선택합니다.

b. 키/값 페어에서 키에 `password`를 입력하고 값에는 실제 암호를 입력합니다.

> **Note**
>
> 키에 `password`만 입력해야 합니다(다른 입력이 있으면 인증 실패).

c. 새 키 추가를 선택하여 암호화 키에서 KMS 고객 관리형 키(CMK)를 생성합니다. 그러면 AWS KMS 콘솔이 열립니다.

d. 고객 관리형 키 페이지에서 키 생성을 선택합니다.

e. 키 구성 페이지에 기본값을 유지하고 다음을 선택합니다.

f. 별칭(필요에 따라 설명 및 태그 추가 가능)에 키 이름을 입력하고 다음을 선택합니다.

g. 키 관리자에서 키를 관리할 수 있도록 허용할 사용자(본인 제외) 또는 역할을 선택한 후 다음을 선택합니다.

h. 키 사용자에서 키를 사용할 수 있도록 허용할 사용자(본인 제외) 또는 역할을 선택한 후 다음을 선택합니다.

i. KMS CMK 정책를 복사하고 쉼표로 구분된 추가 스테이트먼트로 추가하여 `"statement"` 수준의 키 정책 JSON 텍스트 편집기에 붙여넣습니다. 리전 및 계정 번호를 자체 정보로 대체합니다.

j. 마침을 클릭합니다.

k. AWS Secrets Manager 새 보안 암호 저장 페이지가 열려 있는 브라우저의 탭을 선택하고 암호화 키 필드 옆의 새로 고침 아이콘(원형 화살표)을 선택한 다음 필드 내부를 클릭하고 새로 만든 키를 선택합니다.

l. 보안 암호 구성 페이지의 보안 암호 이름 필드에 이름을 입력합니다.

m. 리소스 권한에서 권한 편집을 선택합니다.

n. 보안 암호 리소스 정책를 복사하여 리소스 권한 JSON 텍스트 편집기에 붙여넣고, 리전 및 계정 번호를 자체 정보로 대체합니다. (편집기에서 예제 코드를 모두 삭제해야 합니다.)

o. 저장을 선택한 후, 다음을 선택합니다.

p. 필요에 따라 교체를 구성하고 다음을 선택합니다.

q. 새 보안 암호를 검토하고 저장을 선택하여 저장합니다.

r. SES 새 수신 엔드포인트 만들기 페이지가 열려 있는 브라우저의 탭을 선택하고 목록 새로 고침을 선택한 다음, 보안 암호 ARN에서 새로 만들어진 보안 암호를 선택합니다.

6. 허용한 이메일에 대해 수행할 규칙 작업이 포함된 규칙 세트를 선택합니다.
7. 트래픽 정책을 선택하여 차단하거나 허용할 이메일을 결정합니다.
8. 퍼블릭 네트워크인지 프라이빗 네트워크인지 선택합니다.
   - 퍼블릭 네트워크의 경우 IPv4 전용 또는 듀얼 스택(IPv4 및 IPv6) 주소 지정을 선택합니다.
   - 프라이빗 네트워크의 경우 IAM 사용자 또는 역할과 같은 동일한 계정의 승인된 발신자와 공유한 VPC 엔드포인트를 선택하거나 입력합니다. 선택적으로 VPC 엔드포인트 생성을 선택하여 Amazon VPC 콘솔을 열어 새 VPC 엔드포인트를 생성할 수 있습니다.
9. 수신 엔드포인트 만들기를 선택합니다.
10. 일반 세부 정보에서 수신 엔드포인트가 만들어지는 동안 '프로비저닝'이 표시됩니다. '활성'이 표시되고 ARecord 필드에 값이 포함될 때까지 페이지를 새로 고침합니다. 'A' 레코드 값을 복사하여 퍼블릭 엔드포인트 구성에 설명된 대로 DNS 구성 또는 SMTP 클라이언트에 붙여넣습니다.
11. 콘솔의 일반 세부 정보 컨테이너 바로 위에는 inp-1abc2de3fghi4jkl5mnop6qr과 같이 'inp' 접두사가 붙은 레이블이 지정되지 않은 큰 번호가 있습니다(페이지 상단의 브레드크럼 추적에서도 복제됨). 이를 수신 엔드포인트 ID라고 하며, 해당 값은 수신 서버에 로그인하기 위한 사용자 이름으로 사용됩니다. (엔드포인트에 연결하려면 승인된 발신자와 공유해야 함)
12. 수신 엔드포인트 페이지에서 이미 만든 수신 엔드포인트를 보고 관리할 수 있습니다. 제거하려는 수신 엔드포인트가 있는 경우 라디오 버튼을 선택한 다음 삭제를 선택합니다.
13. 수신 엔드포인트를 편집하려면 이름을 선택하여 요약 페이지를 엽니다.
   - 일반 세부 정보에서 편집을 선택한 다음 변경 사항 저장을 선택하여 엔드포인트의 활성 상태를 변경할 수 있습니다.
   - 규칙 세트 또는 트래픽 정책에서 편집을 선택한 다음 변경 사항 저장을 선택하여 다른 규칙 세트 또는 트래픽 정책을 선택할 수 있습니다.

트래픽 정책 및 정책 스테이트먼트

트래픽 정책은 수신 엔드포인트에 할당하는 정책 스테이트먼트의 컨테이너로, 정책 설명의 조건이 충족될 때 특정 유형의 이메일을 허용하거나 차단하여 수신 메일을 정렬할 수 있습니다. 트래픽 정책은 여러 수신 엔드포인트에서 사용할 수 있습니다.

> **Tip**
>
> 트래픽 정책을 '필터 세트'로, 정책 스테이트먼트를 '필터'로 생각할 수 있습니다. 트래픽 정책(필터 세트)에는 수신 메일을 필터링하는 데 사용하는 정책(필터)이 포함되어 있습니다.

트래픽 정책을 만들 때 최대 메시지 크기(바이트 단위)를 설정할 수 있습니다. 메시지가 해당 크기를 초과하면 삭제됩니다. 정책의 기본 작업을 설정하여 정책 설명의 조건을 벗어나는 이메일을 허용하거나 차단할 수도 있습니다. 이를 트래픽 정책에 대한 '모두 캐치' 작업으로 간주합니다.

또한 정책 스테이트먼트는 스테이트먼트 조건이 충족될 때 수행되는 허용 또는 차단 작업으로 만들어집니다. 정책 스테이트먼트에서 허용하거나 차단하기 전에 수신 메시지와 일치해야 하는 이메일 프로토콜과 입력한 값에 대한 조건부 연산자를 선택하여 조건을 작성합니다. 각 정책 스테이트먼트에는 여러 조건이 있을 수 있습니다.

트래픽 정책에는 여러 정책 스테이트먼트가 포함될 수 있으며 이메일 평가 방식의 암시적 계층 구조를 기반으로 하는 순서로 실행됩니다.

- 차단하는 정책 스테이트먼트 - 이러한 스테이트먼트는 먼저 평가되며 스테이트먼트 조건을 충족하는 모든 메시지를 차단합니다.
- 허용되는 정책 스테이트먼트 - 이러한 스테이트먼트가 다음으로 평가되며 스테이트먼트 조건을 충족하는 모든 메시지를 허용합니다.
- 트래픽 정책의 기본 작업 - 정책 스테이트먼트에 포함되지 않는 나머지 메시지는 이 파라미터를 정의한 방식에 따라 허용되거나 차단됩니다.
- 최대 메시지 크기 -이 선택적 파라미터를 설정하면이 크기보다 큰 모든 메시지가 삭제됩니다.

트래픽 정책은 하나 이상의 수신 엔드포인트에서 사용할 수 있는 독립 리소스이지만 정책 스테이트먼트는 생성된 트래픽 정책에만 포함됩니다. 따라서 수신 엔드포인트로 들어오는 이메일을 평가하기 위한 정책 스테이트먼트를 만들려면 먼저 트래픽 정책을 생성하거나 기존 정책을 편집해야 합니다.

다음 섹션의 절차에서는 SES 콘솔에서 트래픽 정책 및 정책 스테이트먼트를 만드는 방법을 설명합니다.

SES 콘솔에서 트래픽 정책 및 정책 스테이트먼트 만들기

다음 절차에서는 SES 콘솔의 트래픽 정책 페이지를 사용하여 트래픽 정책 및 정책 스테이트먼트를 만들고 이미 만든 정책을 관리하는 방법을 보여줍니다.

콘솔을 사용하여 트래픽 정책 및 정책 스테이트먼트를 만들고 관리하는 방법

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 창의 Mail Manager에서 트래픽 정책을 선택합니다.
3. 트래픽 정책 페이지에서 트래픽 정책 만들기를 선택합니다.

4. 트래픽 정책 만들기 페이지에서 트래픽 정책의 고유한 이름을 입력합니다.
5. (선택 사항) 특정 크기를 초과하는 메시지를 삭제하려면 최대 메시지 크기 필드에 바이트 단위로 값을 입력합니다.
6. 기본 작업에서 트래픽 정책이 정책 스테이트먼트의 조건을 벗어나는(해결되지 않는) 메시지 허용 또는 거부(차단) 중 하나를 선택합니다.
7. 새 정책 문 추가를 선택하여 트래픽 정책에 대한 스테이트먼트를 만듭니다.
8. 스테이트먼트 조건이 충족될 경우 수행할 작업에 대해 허용 또는 거부(차단)를 선택합니다.
9. 입력한 값에 대한 이메일 프로토콜과 조건부 연산자를 선택하여 조건을 작성합니다. 이 정책 스테이트먼트에 조건을 더 추가하려면 새 조건 추가를 선택합니다. 조건 속성과 해당 연산자 및 유효한 값에 대한 자세한 내용은 정책 스테이트먼트 조건 참조를 참조하세요.
   - 이메일 추가 기능을 구독한 경우 여기에서 이메일 프로토콜로 선택할 수 있습니다.
   - IPv6 또는 듀얼 스택 수신 엔드포인트와 연결된 트래픽 정책은 IPv6 연결을 통해 수신된 메시지에 이메일 추가 기능 조건을 평가하거나 적용하지 않습니다. 이메일 추가 기능 조건은 듀얼 스택 엔드포인트의 IPv4 연결을 통해 수신된 메시지에만 적용됩니다.
10. 정책 스테이트먼트 및 조건을 더 추가하려면 위의 7~9단계를 반복합니다.
11. 정책 스테이트먼트 및 해당 조건 생성을 완료했으면 트래픽 정책 만들기를 선택합니다.
12. 트래픽 정책 페이지에서 이미 만든 트래픽 정책을 보고 관리할 수 있습니다. 제거하려는 트래픽 정책이 있는 경우 라디오 버튼을 선택한 다음 삭제를 선택합니다.
13. 트래픽 정책의 속성 또는 정책 스테이트먼트를 편집하려면 이름을 선택하여 개요 페이지를 열고, 여기에서 편집을 선택합니다.
14. 트래픽 정책 세부 정보에서 최대 메시지 크기 및 기본 작업을 변경할 수 있습니다.
15. 정책 스테이트먼트 컨테이너에서 허용/거부 속성을 변경하고 조건을 편집할 수 있습니다. 또한 정책 스테이트먼트 및 조건을 제거하고 새 정책 스테이트먼트 및 조건을 추가하는 것도 가능합니다.
16. 모든 편집을 마치면 변경 사항 저장을 선택하여 변경 내용을 저장합니다.

정책 스테이트먼트 조건에 대한 참조

정책 스테이트먼트 조건

다음 참조 테이블에는 정책 스테이트먼트 조건을 작성하는 데 사용할 수 있는 모든 정책 스테이트먼트 프로토콜이 나열되어 있습니다. 프로토콜의 표현식 유형을 선택하면 해당 프로토콜에 사용 가능한 모든 연산자와 유효한 값을 나열하는 SES Mail Manager API 참조의 참조 페이지로 이동합니다.

정책 스테이트먼트 조건: 프로토콜, 연산자, 값

프로토콜	표현식 유형
수신자 주소 Abusix Mail Intelligence(구독한 경우) • 에 나열됨 Spamhaus Domain Block List(구독한 경우) • 에 나열됨	문자열 표현식에 유효한 연산자 및 값
발신자 IP 범위	IP 표현식에 유효한 연산자 및 값
TLS 프로토콜 버전	TLS 프로토콜 표현식에 유효한 연산자 및 값
Abusix Mail Intelligence(구독한 경우) • 나열됨 Spamhaus Domain Block List(구독한 경우) • 나열됨	부울 표현식에 유효한 연산자 및 값

규칙 세트 및 규칙

규칙 세트는 수신 엔드포인트의 트래픽 정책에서 허용된 이메일에 대한 작업을 수행할 수 있도록 수신 엔드포인트에 할당하는 규칙의 컨테이너입니다. 규칙 세트는 여러 수신 엔드포인트에서 사용할 수 있습니다.

규칙은 메시지가 규칙의 조건을 충족할 때 규칙에 정의된 작업을 실행하여 수신 이메일을 처리하는 방법을 수신 엔드포인트에 알려줍니다. 각 규칙에는 여러 조건과 작업이 있을 수 있습니다. 규칙 세트 내에서 만드는 규칙은 규칙 세트 내에서 지정한 순서대로 실행됩니다.

규칙이 작업을 실행하기 전에 메시지와 일치해야 하는 입력 값에 대한 조건부 연산자와 이메일 속성을 선택하여 규칙의 조건을 작성합니다. 수행할 작업과 실행 순서를 정의합니다.

세분성을 높이기 위해 규칙에는 조건과 유사하게 정의된 예외도 포함될 수 있지만, 여기서는 메시지가 일치하지 않아야 하는 조건을 정의합니다. 조건 및 예외는 독립적으로 작동합니다. 원할 경우 예외뿐 아니라 혼합 조건 및 예외도 포함하는 규칙을 구성할 수 있습니다.

규칙 세트 내에서 규칙을 정의할 수 있는 방법이 세밀하게 세분화되어 있으므로, 규칙 세트 구성 요소의 관계를 설명하는 데 도움이 되도록 다음 목록이 제공됩니다.

- 규칙 세트에는 다음이 포함됩니다.
  - 규칙 - 규칙 세트 내에서 규칙이 실행되는 순서를 정의할 수 있습니다.

    규칙에는 다음이 포함됩니다.
    - 조건 - 메시지가 하나 이상의 조건 평가와 일치하는 경우 규칙이 적용됩니다. 규칙에 예외가 있는 경우 아래를 참조하세요.
    - 예외 - 메시지가 하나 이상의 예외 평가와 일치하지 않는 경우 규칙이 적용됩니다. 규칙에 조건이 있는 경우 위를 참조하세요.
    - 작업 - 규칙이 적용되면 작업이 트리거됩니다. 모든 조건이 일치하고 예외가 없습니다.

      규칙 내에서 작업이 실행되는 순서를 정의할 수 있습니다.

각 규칙에는 여러 가지 조건, 예외 및 작업이 있을 수 있으며 규칙과 작업이 실행되는 방식의 순서를 정의할 수 있습니다. 따라서 특정 비즈니스 요구 사항에 적합한 고도로 맞춤화되고 자동화된 이메일 처리 솔루션을 구축할 수 있습니다.

규칙 세트는 둘 이상의 수신 엔드포인트에서 사용할 수 있는 독립 리소스이지만, 규칙은 규칙이 만들어진 규칙 세트에만 속합니다. 따라서 수신 엔드포인트로 들어오는 이메일에 대해 조치를 취할 규칙을 만들려면 먼저 규칙 세트를 만들거나 기존 세트를 편집해야 합니다.

다음 섹션의 절차는 SES 콘솔에서 규칙 세트 및 해당 규칙을 만드는 방법을 안내합니다.

SES 콘솔에서 규칙 세트 및 규칙 만들기

다음 절차에서는 SES 콘솔의 규칙 세트 페이지를 사용하여 규칙 세트와 해당 규칙을 만들고 이미 만든 규칙 세트와 규칙을 관리하는 방법을 보여줍니다.

콘솔을 사용하여 규칙 세트 및 규칙을 만들고 관리하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 패널의 Mail Manager에서 규칙 세트를 선택합니다.

3. 규칙 세트 페이지에서 규칙 세트 만들기를 선택하고 규칙 세트의 고유한 이름을 입력합니다.
4. 규칙 세트의 개요 페이지에서 편집을 선택한 다음 편집 페이지에서 새 규칙 만들기를 선택합니다.
5. 규칙 세부 정보 사이드바에 규칙의 고유한 이름을 입력합니다.
6. 새 조건 추가를 선택하여 메시지와 일치해야 하는 조건을 만들거나, 다음의 경우 예외 확인란을 선택한 다음 새 예외 추가를 선택하여 메시지가 일치하지 않는 조건을 만듭니다.
7. 입력한 값에 대한 이메일 속성과 조건부 연산자를 선택하여 조건 또는 예외를 작성합니다. 이 규칙에 조건이나 예외를 더 추가하려면 새 조건 추가 또는 새 예외 추가를 선택합니다. 조건 속성과 해당 연산자 및 유효한 값에 대한 자세한 내용은 규칙 조건 참조를 참조하세요.

   - 이메일 추가 기능을 구독한 경우 여기에서 이메일 속성으로 선택할 수 있습니다.
8. 새 작업 추가를 선택하여 규칙 조건이 일치하거나 예외가 일치하지 않을 때 수행할 작업을 정의합니다. 수행할 작업을 더 추가하려면 새 작업 추가를 선택합니다. 둘 이상의 작업을 생성하면 실행 순서를 설정할 수 있도록 위쪽/아래쪽 화살표가 표시됩니다.

   > **Note**
   >
   > 이러한 규칙 작업을 실행하려면 계정에 대해 해당 권한 정책을 활성화해야 합니다. 그렇지 않으면 규칙 작업이 실패합니다.

9. 작업을 선택한 후 규칙 세부 정보 패널에서 이러한 작업에 대한 권한 정책을 직접 적용합니다.

   a. IAM 역할 필드에서 새 역할 생성을 선택하고 이름을 입력한 다음 역할 생성을 선택합니다. (이 역할에 대한 IAM 신뢰 정책은 백그라운드에서 자동으로 생성됩니다.)

   b. IAM 신뢰 정책이 자동으로 생성되었으므로 역할에 작업의 권한 정책만 추가하면 됩니다. IAM 역할 필드에서 역할 보기를 선택하여 IAM 콘솔을 엽니다.

   c. 권한 탭에서 권한 추가를 선택하고 인라인 정책 만들기를 선택합니다.

   d. 권한 지정 페이지의 정책 편집기에서 JSON을 선택합니다.

   e. 에서 해당 정책을 복사하여 정책 편집기Mail Manager에 대한 권한 정책에 붙여넣고 빨간색 텍스트의 데이터를 사용자 고유의 데이터로 바꿉니다. (편집기에서 예제 코드를 모두 삭제해야 합니다.)

   f. 다음을 선택합니다.

   g. 정책 만들기를 선택하여 IAM 역할에 대한 권한 정책을 검토하고 만듭니다.

   h. SES Mail Manager 규칙 세트 편집 페이지가 열려 있는 브라우저의 탭을 선택하고 규칙을 만드는 나머지 단계를 계속합니다.

10. 규칙에 대한 조건, 예외 및 작업을 모두 만들면 왼쪽의 규칙 세트 편집 패널에 있는 규칙 세트 저장을 선택하여 규칙 세트에 저장합니다.
11. 규칙 세트에 규칙을 더 추가하려면 위의 4~9단계를 반복합니다.
    - 둘 이상의 규칙을 만들면 실행 순서를 설정할 수 있도록 규칙 세트의 재정렬 열에 위/아래 화살표가 표시됩니다.
12. 규칙 세트 페이지에서 이미 만든 규칙 세트를 보고 관리할 수 있습니다. 제거하려는 규칙 세트가 있는 경우 라디오 버튼을 선택한 다음 삭제를 선택합니다.
13. 규칙 세트를 편집하려면 이름을 선택하여 개요 페이지를 엽니다. 여기서 규칙 실행 순서를 재정렬하려면 편집을, 더 많은 규칙을 추가하려면 새 규칙 만들기를, 규칙을 삭제하려면 라디오 버튼과 삭제를 차례로 선택합니다.
14. 규칙을 편집하려면 라디오 버튼을 선택합니다. 규칙 세부 정보 사이드바의 컨테이너에서 조건 또는 예외를 편집하고 작업을 변경하거나 재정렬할 수 있습니다. 조건, 예외 및 작업을 제거하고 새 조건, 예외 및 작업을 추가할 수도 있습니다.
15. 모든 편집을 마치면 왼쪽의 규칙 세트 편집 패널에 있는 규칙 세트 저장을 선택하여 변경 사항을 저장합니다.

규칙 조건 및 작업에 대한 참조

규칙 조건

다음 참조 테이블에는 규칙 조건(또는 예외)을 구성하는 데 사용할 수 있고 표현식 유형별로 분류되는 모든 규칙 속성이 나열되어 있습니다. 같은 표현식 유형을 공유하는 규칙 속성은 동일한 연산자와 값도 공유합니다. 속성의 표현식 유형을 선택하면 해당 속성에 대해 사용 가능한 모든 연산자와 유효한 값을 나열하는 SES Mail Manager API 참조의 참조 페이지로 이동합니다.

규칙 조건: 속성, 연산자 및 값

속성	표현식 유형
From 주소	
To 주소	
CC 주소	문자열 표현식에 유효한 연산자 및 값
Mail from	
수신자 주소	

속성	표현식 유형
제목 Helo MIME 헤더 Vade Advanced Email Security(구독한 경우) • 범주 • 판정 Trend Micro 바이러스 검사(구독한 경우) • 범주	
IP 범위	IP 표현식에 유효한 연산자 및 값
메시지 최대 크기	숫자 표현식에 유효한 연산자 및 값
DKIM SPF	판정 표현식에 유효한 연산자 및 값
TLS 래핑된 TLS 수신 확인 Vade Advanced Email Security(구독한 경우) • 전달됨 Trend Micro 바이러스 검사(구독한 경우) • 전달됨	부울 표현식에 유효한 연산자 및 값
DMARC 정책	DMARC 표현식에 유효한 연산자 및 값

규칙 작업

다음 참조 테이블에는 규칙의 조건이 충족되거나 예외가 충족되지 않을 때 수행할 수 있는 모든 규칙 작업이 나열되어 있습니다. 작업을 선택하면 작업에 대한 파라미터와 형식을 나열하는 SES Mail Manager API 참조의 작업 참조 페이지로 이동합니다. 테이블에서는 Mail Manager 콘솔에서 채택한 작업 이름을 사용합니다. API 이름은 약간 다를 수 있습니다.

> **Note**
>
> 일부 API 참조에는 작업이 실패할 경우 계속 또는 삭제로 설정할 수 있는 `ActionFailurePolicy` 파라미터가 있습니다. 이는 API를 사용할 때만 적용되며, 콘솔을 사용할 때는 `ActionFailurePolicy`가 계속의 기본값으로 설정됩니다.

규칙 작업: 작업 및 파라미터

작업 및 해당 파라미터	설명
S3에 쓰기	이메일의 MIME 콘텐츠를 S3 버킷에 기록합니다.
SMTP 릴레이 작업	SMTP를 통해 이메일을 다른 특정 SMTP 서버로 릴레이합니다.
아카이브 작업	이메일을 Amazon SES 아카이브에 전달하여 아카이브합니다.
헤더 추가	수신한 이메일에 맞춤 헤더를 추가합니다.
이메일 수신자 재작성	이메일 봉투 수신자를 지정된 수신자 목록으로 바꿉니다. 이 작업의 조건이 수신자 하위 집합에만 적용되는 경우 해당 수신자만 대체됩니다.
메일박스로 전송	Amazon WorkMail 메일박스로 이메일을 전송합니다.
Q Business로 전송	Amazon Q Business 애플리케이션에 이메일을 전송하여 지식 기반에 수집합니다.

작업 및 해당 파라미터	설명
SNS에 게시	Amazon SNS 주제에 이메일 콘텐츠를 게시합니다.
인터넷으로 전송	SES를 사용하여 이메일의 수신자 목록에 있는 1명 이상의 수신자에게 이메일을 보냅니다.
삭제 작업	수신자가 여러 명인 이메일의 경우, 이 작업이 1명 이상의 수신자(전부는 아님)에 적용되면 해당 수신자가 이메일의 수신자 목록에서 삭제되고 나머지 수신자를 대상으로는 규칙이 계속 처리됩니다. 이 작업이 모든 수신자에게 적용되는 경우, 모든 수신자가 수신자 목록에서 삭제되어 이메일이 수신되지 않으므로 규칙 처리가 중지됩니다.

SMTP 릴레이

Mail Manager는 이메일 환경(예: Microsoft 365, Google Workspace 또는 온프레미스 Exchange)과 인터넷 사이에 배포되므로, Mail Manager는 SMTP 릴레이를 사용하여 Mail Manager에서 처리하는 수신 이메일을 이메일 환경으로 라우팅합니다. 또한, 최종 수신자에게 보내기 전에 아웃바운드 이메일을 다른 Exchange 서버나 타사 이메일 게이트웨이 등의 여타 이메일 인프라로 라우팅할 수 있습니다.

SMTP 릴레이는 이메일 인프라의 중요한 구성 요소이며, 규칙 세트에 정의된 규칙 작업으로 지정된 경우 서버 간에 이메일을 효율적으로 라우팅하는 역할을 합니다.

특히 SMTP 릴레이는 SES Mail Manager와 Exchange, 온프레미스 또는 타사 이메일 게이트웨이 등의 외부 이메일 인프라 간에 수신 이메일을 리디렉션할 수 있습니다. 수신 엔드포인트로 들어오는 이메일은 특정 이메일을 지정된 SMTP 릴레이로 라우팅하는 규칙에 의해 처리되며, 그러면 지정된 이메일이 SMTP 릴레이에 정의된 외부 이메일 인프라로 전달됩니다.

수신 엔드포인트에 이메일이 수신되면 트래픽 정책을 사용하여 어떤 이메일을 차단하거나 허용할지 결정합니다. 여기서 허용하는 이메일은 조건부 규칙을 적용하여 특정 유형의 이메일에 대해 정의한 작업을 실행하는 규칙 세트로 전달됩니다. 정의할 수 있는 규칙 작업 중 하나는 SMTPRelay 작업입니다. 이 작업을 선택하면 이메일이 SMTP 릴레이에 정의된 외부 SMTP 서버로 전달됩니다.

예를 들어, SMTPRelay 작업을 사용하여 수신 엔드포인트에서 온프레미스 Microsoft Exchange Server로 이메일을 보낼 수 있습니다. 특정 자격 증명으로만 액세스할 수 있는 퍼블릭 SMTP 엔드포인트가 있도록 Exchange 서버를 설정합니다. SMTP 릴레이를 만들 때 Exchange 서버의 서버 이름, 포트, 자격 증명을 입력하고 SMTP 릴레이에 고유한 이름, 즉 'RelayToMyExchangeServer'를 지정합니다. 그런 다음 수신 엔드포인트의 규칙 세트에 "From 주소에 'gmail.com'이 포함된 경우, RelayToMyExchangeServer라는 SMTP 릴레이를 사용하여 SMTPRelay 작업을 수행합니다."라는 규칙을 만듭니다.

> **Note**
>
> 프라이빗 SMTP 엔드포인트는 지원되지 않으므로 Amazon SES SMTP 릴레이로 구성하는 모든 SMTP 엔드포인트는 퍼블릭이어야 합니다.

gmail.com의 이메일이 수신 엔드포인트에 도착하면, 이 규칙에 따라 SMTPRelay 작업이 트리거되고 SMTP 릴레이를 만들 때 제공한 자격 증명을 통해 Exchange 서버에 연락하여 이메일을 Exchange 서버로 전달합니다. 따라서 gmail.com에서 수신한 이메일은 Exchange 서버로 릴레이됩니다.

먼저 SMTP 릴레이를 만든 다음 규칙 작업을 지정해야 합니다. 다음 섹션의 절차에서는 SES 콘솔에서 SMTP 릴레이를 만드는 방법을 안내합니다.

SES 콘솔에서 SMTP 릴레이 만들기

다음 절차에서는 SES 콘솔의 SMTP 릴레이 페이지를 사용하여 SMTP 릴레이를 만들고 이미 만든 SMTP 릴레이를 관리하는 방법을 보여줍니다.

콘솔을 사용하여 SMTP 릴레이를 만들고 관리하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 패널의 Mail Manager에서 SMTP 릴레이를 선택합니다.
3. SMTP 릴레이 페이지에서 SMTP 릴레이 만들기를 선택합니다.
4. SMTP 릴레이 만들기 페이지에서 SMTP 릴레이의 고유한 이름을 입력합니다.
5. 인바운드(미인증) 또는 아웃바운드(인증) SMTP 릴레이를 구성할지에 따라 해당 지침을 따릅니다.

Inbound

인바운드 SMTP 릴레이를 구성하려면

1. SMTP 릴레이를 인바운드 게이트웨이로 사용하여 Mail Manager에서 처리하는 수신 이메일을 외부 이메일 환경으로 라우팅하는 경우 먼저 이메일 호스팅 환경을 구성해야 합니다. 모든 이메일 호스팅 공급자는 고유한 GUI 및 구성 워크플로를 가지고 있지만, Mail Manager SMTP 릴레이와 같은 인바운드 게이트웨이에서 작동하도록 구성하는 보안 주체는 비슷합니다.

   이를 설명하기 위해 다음 섹션에서 SMTP 릴레이를 인바운드 게이트웨이로 사용하도록 Google Workspaces 및 Microsoft Office 365를 구성하는 방법에 대한 예를 제공했습니다.

   - Google Workspaces 설정
   - Microsoft Office 365 설정

   > **Note**
   >
   > 의도한 수신자 대상이 SES 확인 이메일 ID인지 확인합니다. 예를 들어 수신자 abc@example.com, , admin@example.com postmaster@acme.com및 support@acme.com에게 이메일을 전송하려면 SES에서 `example.com` 및 `acme.com` 도메인을 확인하는 것이 좋습니다. 수신자 대상이 확인되지 않으면 SES는 퍼블릭 SMTP 서버로 이메일을 전송하려고 시도하지 않습니다.

2. 인바운드 게이트웨이와 함께 작동하도록 Google Workspaces 또는 Microsoft Office 365를 구성한 후 퍼블릭 SMTP 서버의 호스트 이름을 공급자에 해당하는 아래 값으로 입력합니다.

   - Google Workspaces: `aspmx.l.google.com`
   - Microsoft Office 365: `<your_domain>.mail.protection.outlook.com`

     점을 도메인 이름의 '-' 기호로 바꿉니다. 예를 들어, 도메인이 acme.com이면 `acme-com.mail.protection.outlook.com`을 입력합니다.

3. 퍼블릭 SMTP 서버에 포트 번호 25를 입력합니다.
4. 인증 섹션을 비워 둡니다. 보안 암호 ARN을 선택하거나 만들지 마세요.

Outbound

아웃바운드 SMTP 릴레이를 구성하려면

1. 릴레이를 연결할 퍼블릭 SMTP 서버의 호스트 이름을 입력합니다.
2. 퍼블릭 SMTP 서버의 포트 번호를 입력합니다.
3. 보안 암호 ARN에서 보안 암호 중 하나를 선택하여 퍼블릭 SMTP 서버에 대한 인증을 설정합니다. 이전에 생성한 보안 암호를 선택하는 경우 새 보안 암호를 만들려면 다음 단계에 표시된 정책이 포함되어 있어야 합니다.

   - 새로 만들기를 선택하여 새 보안 암호를 만들 수 있는 옵션을 표시합니다. AWS Secrets Manager 콘솔이 열리고 새 키를 계속 만들 수 있습니다.

   a. 보안 암호 유형에서 다른 유형의 보안 암호를 선택합니다.
   b. 키/값 페어에 다음 키와 값을 입력합니다.

키	값
`username`	my_username
`password`	my_password

> **Note**
> 두 키 모두에 대해 표시된 대로 `username` 및 `password`만 입력해야 합니다. 다르게 입력한 경우 인증이 실패합니다. 값에 사용자 이름과 암호를 각각 입력합니다.

   c. 새 키 추가를 선택하여 암호화 키에서 KMS 고객 관리형 키(CMK)를 생성합니다. 그러면 AWS KMS 콘솔이 열립니다.
   d. 고객 관리형 키 페이지에서 키 생성을 선택합니다.
   e. 키 구성 페이지에 기본값을 유지하고 다음을 선택합니다.
   f. 별칭(필요에 따라 설명 및 태그 추가 가능)에 키 이름을 입력하고 다음을 선택합니다.

g. 키 관리자에서 키를 관리할 수 있도록 허용할 사용자(본인 제외) 또는 역할을 선택한 후 다음을 선택합니다.

h. 키 사용자에서 키를 사용할 수 있도록 허용할 사용자(본인 제외) 또는 역할을 선택한 후 다음을 선택합니다.

i. KMS CMK 정책를 복사하고 쉼표로 구분된 추가 스테이트먼트로 추가하여 `"statement"` 수준의 키 정책 JSON 텍스트 편집기에 붙여넣습니다. 리전 및 계정 번호를 자체 정보로 대체합니다.

j. 마침을 클릭합니다.

k. AWS Secrets Manager 새 보안 암호 저장 페이지가 열려 있는 브라우저의 탭을 선택하고 암호화 키 필드 옆의 새로 고침 아이콘(원형 화살표)을 선택한 다음 필드 내부를 클릭하고 새로 만든 키를 선택합니다.

l. 보안 암호 구성 페이지의 보안 암호 이름 필드에 이름을 입력합니다.

m. 리소스 권한에서 권한 편집을 선택합니다.

n. 보안 암호 리소스 정책를 복사하여 리소스 권한 JSON 텍스트 편집기에 붙여넣고, 리전 및 계정 번호를 자체 정보로 대체합니다. (편집기에서 예제 코드를 모두 삭제해야 합니다.)

o. 저장을 선택한 후, 다음을 선택합니다.

p. 필요에 따라 교체를 구성하고 다음을 선택합니다.

q. 새 보안 암호를 검토하고 저장을 선택하여 저장합니다.

r. SES SMTP 릴레이 생성 페이지가 열려 있는 브라우저의 탭을 선택하고 목록 새로 고침을 선택한 다음 보안 암호 ARN에서 새로 생성된 보안 암호를 선택합니다.

6. SMTP 릴레이 만들기를 선택합니다.
7. SMTP 릴레이 페이지에서 이미 만든 SMTP 릴레이를 보고 관리할 수 있습니다. 제거하려는 SMTP 릴레이가 있는 경우 라디오 버튼을 선택한 다음 삭제를 선택합니다.
8. SMTP 릴레이를 편집하려면 이름을 선택합니다. 세부 정보 페이지에서 해당 편집 또는 업데이트 버튼을 선택한 다음 변경 사항 저장을 선택하여 릴레이의 이름, 외부 SMTP 서버의 이름, 포트 및 로그인 자격 증명을 변경할 수 있습니다.

인바운드 (미인증) SMTP 릴레이에 대한 Google Workspaces 설정

다음 연습 예제에서는 Mail Manager 인바운드 (미인증) SMTP 릴레이에서 작동하도록 Google Workspaces를 설정하는 방법을 보여줍니다.

사전 조건

- Google 관리자 콘솔(Google 관리자 콘솔 > 앱 > Google Workspace > Gmail)에 액세스합니다.
- Mail Manager 설정에 사용할 도메인의 MX 레코드를 호스팅하는 도메인 이름 서버에 액세스합니다.

인바운드 SMTP 릴레이에서 작동하도록 Google Workspaces를 설정하려면

- 인바운드 게이트웨이 구성에 Mail Manager IP 주소 추가

  a. Google 관리자 콘솔에서 앱 > Google Workspace > Gmail로 이동합니다.

  b. 스팸, 피싱 및 맬웨어를 선택한 다음 인바운드 게이트웨이 구성으로 이동합니다.

  c. 인바운드 게이트웨이를 활성화하고 다음 세부 정보로 구성합니다.

  - 게이트웨이 IPs에서 추가를 선택하고 SMTP 릴레이 IPs 범위 테이블에서 리전별 수신 엔드포인트 IP를 추가합니다. https://docs.aws.amazon.com/general/latest/gr/ses.html#ses_mm_relay_ip_ranges
  - 외부 IP 자동 감지를 선택합니다.
  - 위에 나열된 이메일 게이트웨이의 연결에 TLS 필요를 선택합니다.

- 대화 상자 하단에서 저장을 선택하여 구성을 저장합니다. 저장되면 관리자 콘솔에 인바운드 게이트웨이가 활성화된 것으로 표시됩니다.

인바운드 (미인증) SMTP 릴레이에 대한 Microsoft Office 365 설정

다음 연습 예제에서는 Mail Manager 인바운드 (미인증) SMTP 릴레이에서 작동하도록 Microsoft Office 365를 설정하는 방법을 보여줍니다.

사전 조건

- Microsoft Security 관리 센터(Microsoft Security 관리 센터 > 이메일 및 협업 > 정책 및 규칙 > 위협 정책)에 액세스합니다.
- Mail Manager 설정에 사용할 도메인의 MX 레코드를 호스팅하는 도메인 이름 서버에 액세스합니다.

인바운드 SMTP 릴레이에서 작동하도록 Microsoft Office 365를 설정하려면

1. 허용 목록에 Mail Manager IP 주소 추가
   a. Microsoft Security 관리 센터에서 이메일 및 협업 > 정책 및 규칙 > 위협 정책으로 이동합니다.
   b. 정책에서 스팸 방지를 선택합니다.
   c. 연결 필터 정책을 선택한 다음 연결 필터 정책 편집을 선택합니다.
      - 다음 IP 주소 또는 주소 범위의 메시지 항상 허용 대화 상자에서 SMTP 릴레이 IPs 범위 테이블의 리전별 수신 엔드포인트 IP를 추가합니다. https://docs.aws.amazon.com/general/latest/gr/ses.html#ses_mm_relay_ip_ranges
      - 저장을 선택합니다.
   d. 스팸 방지 옵션으로 돌아가 스팸 방지 인바운드 정책을 선택합니다.
      - 대화 상자 하단에서 스팸 임계값 및 속성 편집을 선택합니다.

- 스팸으로 표시로 스크롤하여 SPF 레코드: 하드 실패가 꺼짐으로 설정되어 있는지 확인합니다.
- 저장을 선택합니다.

2. 향상된 필터링 구성(권장)

   이 옵션을 사용하면 Microsoft Office 365가 SES Mail Manager에서 메시지를 수신하기 전에 원래 연결 IP를 올바르게 식별할 수 있습니다.

a. 인바운드 커넥터 만들기

- 새 Exchange 관리 센터에 로그인하고 메일 흐름 > 커넥터로 이동합니다.
- 커넥터 추가를 선택합니다.
- 다음에서 연결에서 파트너 조직을 선택한 후 다음을 선택합니다.
- 다음과 같이 필드를 입력합니다.
  - 이름 - Simple Email Service Mail Manager 커넥터
  - 설명 - 필터링용 커넥터

- 다음을 선택합니다.
- 전송된 이메일 인증에서 전송 서버의 IP 주소가 파트너 조직에 속하는 다음 IP 주소 중 하나와 일치하는지 확인하고 SMTP 릴레이 IPs 범위 테이블에서 리전별 수신 엔드포인트 IP를 추가합니다. https://docs.aws.amazon.com/general/latest/gr/ses.html#ses_mm_relay_ip_ranges

- 다음을 선택합니다.
- 보안 제한에서 기본 TLS를 통해 이메일 메시지가 전송되지 않은 경우 이메일 메시지 거부 설정을 수락하고 다음을 수락합니다.
- 설정을 검토하고 커넥터 만들기를 선택합니다.

b. 향상된 필터링 활성화

이제 인바운드 커넥터가 구성되었으므로, Microsoft Security 관리 센터에서 커넥터의 향상된 필터링 구성을 활성화해야 합니다.

- Microsoft Security 관리 센터에서 이메일 및 협업 > 정책 및 규칙 > 위협 정책으로 이동합니다.
- 규칙에서 향상된 필터링을 선택합니다.

- 이전에 만든 Simple Email Service Mail Manager 커넥터를 선택하여 구성 파라미터를 편집합니다.
- 마지막 IP 주소 자동 감지 및 건너뛰기와 전체 조직에 적용을 모두 선택합니다.

- 저장을 선택합니다.

주소 목록

주소 목록은 메시지의 수신자 또는 발신자가 특정 목록에 속하는지 여부에 따라 수신 메일을 처리하기 위해 트래픽 정책 및 규칙 세트에 사용할 수 있는 이메일 주소 및 도메인 목록을 생성하고 관리할 수 있는 Mail Manager 기능입니다. 주소 목록은 이메일 흐름을 보다 세부적으로 제어하고 복잡한 이메일 라우팅 시나리오 관리를 간소화하는 데 도움이 됩니다.

주소 목록이란 무엇입니까?

주소 목록은 이메일 메시지를 필터링하고 처리하는 데 사용할 수 있는 이메일 주소 및 도메인의 컨테이너입니다. 관련 주소를 그룹화하고 라우팅 규칙 및 트래픽 정책을 집합적으로 적용하는 편리한 방법을 제공합니다.

주소 목록의 주요 사용 사례는 다음과 같습니다.

- 알려진 스팸 발신자 또는 도메인 차단을 위한 거부 목록
- 신뢰할 수 있는 발신자로부터의 전송을 보장하기 위한 허용 목록
- 존재하지 않는 수신자에게 보내는 이메일을 조기에 거부하기 위한 수신자 검증
- 수신자 역할에 따라 다양한 규칙을 적용하기 위한 역할 기반 라우팅
- 특정 사용자 그룹에 정책을 적용하기 위한 그룹 기반 정책

주소 목록 작동 방식

SES의 주소 목록은 이메일 주소 및 도메인 컬렉션을 생성하고 유지 관리할 수 있도록 하여 이메일 관리를 간소화합니다. 생성된 목록은 트래픽 정책 및 규칙을 통해 이메일 워크플로에 통합됩니다.

SES는 이메일을 처리할 때 관련 주소 목록을 확인하여 발신자 또는 수신자가 멤버인지 확인합니다. 이 멤버십과 구성된 정책 및 규칙에 따라 SES는 이메일 라우팅, 필터링 또는 거부와 같은 적절한 작업을 수행합니다. 이 프로세스를 통해 이메일 트래픽을 효율적이고 세밀하게 제어할 수 있습니다.

주소 목록 설정

주제

- 주소 목록 생성 및 채우기
- 주소 목록 관리

주소 목록 생성 및 채우기

콘솔에서 주소 목록 생성의 일부는 하나 이상의 주소로 주소 목록을 채우는 것입니다. Mail Manager APIs를 사용하여 빈 주소 목록을 생성하고 나중에 채울 수 있습니다. 이 섹션에서는 콘솔 절차와 AWS CLI 예제를 모두 사용하여를 수행하는 방법을 보여줍니다.

주소 목록을 생성하고 채우려면:

1. https://console.aws.amazon.com/ses/ SES 콘솔을 엽니다.
2. 탐색 창의 Mail Manager에서 주소 목록을 선택합니다.
3. 주소 목록 생성을 선택하고 주소 목록 이름 필드에 이름을 입력합니다.
4. 수동 항목 또는 대량 업로드를 선택하고 해당 단계를 따릅니다.
5. 수동 입력의 경우 - 콘솔에 하나 이상의 이메일 주소 또는 도메인을 입력합니다.

   별표(*) 와일드카드를 사용하는 경우 다음 형식이 적용됩니다.

   1. 주소에는 하나만 * 허용됩니다.
      - 항목이 이메일 주소인 경우는 @ 이전 또는 이후*여야 합니다.
      - *가 로컬 부분에 있는 경우 로컬 부분은를 제외하고 0 또는 3~19자일 수 있습니다*.
      - *가 도메인에 있는 경우 하위 도메인 수준은를 제외하고 2~9가 될 수 있습니다*.
   2. 유효한 와일드카드 형식의 예:
      - *.domain1.com ~ *.domain8.domain7...domain1.com
      - *@domain.com
      - 123*@domain.com에서 1234567890123456789*@domain.com으로
      - local@*.domain1.com에서 local@*.domain8.domain7...domain1.com
6. 대량 업로드의 경우 - 파일 선택을 선택하고 업로드할 주소가 포함된 컴퓨터에서 CSV 또는 JSON 파일을 선택합니다.

   각 파일 유형에 대해 예제에 표시된 형식을 사용합니다.

   1. CSV 파일 예제(헤더 `address`는 필수입니다.):

```
address
user1@domain.com
user2@*.domain.com
*@domain.com
```

2. JSON 파일 예제:

```
{
  "items": [
    {
      "address": "user1@domain.com"
    },
    {
      "address": "user2@*.domain.com"
    },
    {
      "address": "*@domain.com"
    }
  ]
}
```

7. 주소 추가를 완료했거나 대량 파일을 선택한 후에는 주소 목록 생성을 선택합니다.

사용 AWS CLI:

주소 목록 생성

```
aws mailmanager create-address-list --address-list-name "MyDenyList"
```

주소 목록을 채웁니다.

- 단일 업로드

```
aws mailmanager register-member-to-address-list \
                --address-list-id al-123456789abc \
                --address "user@example.com"
```

- 대량 업로드

  대량 업로드의 경우 먼저 CSV 또는 JSON 형식을 지정하는 가져오기 작업을 생성해야 합니다.

```
aws mailmanager create-address-list-import-job \
                --address-list-id "al-123456789abc" \
                --name "MyImportJob" \
                --import-data-format ImportDataType=CSV
```

그러면 작업 ID와 미리 서명된 URL이 반환됩니다. curl 명령을 사용하여 다음 예제와 같이이 미리 서명된 URL을 사용하여 CSV 또는 JSON 파일을 S3 버킷에 업로드합니다.

```
curl -X PUT -T "/path/to/file" "pre-signed URL"
```

업로드한 후 이전 명령에서 반환된 작업 ID를 사용하여 가져오기 작업을 시작합니다.

```
aws mailmanager start-address-list-import-job --job-id "job-123456789"
```

주소 목록 관리

필요에 따라 주소 목록을 업데이트, 확인 및 삭제할 수 있습니다.

주제

- 주소 목록 업데이트
- 주소 목록 세부 정보 보기
- 주소 목록 삭제

주소 목록 업데이트

주소를 추가 또는 제거하고 선택적으로 태그를 추가 또는 제거하여 주소 목록을 업데이트할 수 있습니다.

주소 목록을 업데이트하려면:

1. 주소 목록 페이지에서 편집하려는 주소 목록의 이름을 선택합니다.
2. 주소를 추가하려면 이메일 주소 추가를 선택하고에 설명된 대로 수동 항목 또는 대량 업로드 방법을 진행합니다주소 목록 생성 및 채우기.
3. 주소를 제거하려면 제거하려는 각 주소 옆의 확인란을 선택한 다음 이메일 주소 제거를 선택하고 삭제를 확인합니다.
4. (선택 사항) 태그 관리를 선택하여 주소 목록에 태그를 추가하거나 제거합니다.

사용 AWS CLI:

Add

```
aws mailmanager register-member-to-address-list \
                --address-list-id al-123456789abc \
                --address "user@example.com"
```

Remove

```
aws mailmanager deregister-member-from-address-list \
                --address-list-id al-123456789abc \
                --address "user@example.com"
```

주소 목록 세부 정보 보기

주소 목록 세부 정보를 보려면:

- 주소 목록 페이지에서 주소 목록의 이름을 선택하여 세부 정보를 확인합니다.

사용 AWS CLI:

```
aws mailmanager list-members-of-address-list --address-list-id al-123456789abc
```

주소 목록 삭제

주소 목록을 삭제하려면:

1. 주소 목록 페이지에서 삭제하려는 주소 목록 옆의 라디오 버튼을 선택한 다음 삭제를 선택합니다.
2. 확인과 삭제를 차례로 입력하여 목록 삭제를 확인합니다.

사용 AWS CLI:

```
aws mailmanager delete-address-list --address-list-id al-123456789abc
```

트래픽 정책 및 규칙 세트에서 주소 목록 사용

주소 목록은 트래픽 정책 설명 및 규칙 조건에 사용하여 이메일 흐름을 제어할 수 있는 목록 멤버십을 기반으로 이메일을 처리할 수 있습니다. 다음 섹션에서는 각 트래픽 정책 및 규칙 세트에 사용되는 주소 목록의 예를 제공합니다.

주제

- 트래픽 정책 설명에서 주소 목록 사용
- 규칙에서 주소 목록 사용

트래픽 정책 설명에서 주소 목록 사용

수신 엔드포인트로 들어오는 이메일을 허용하거나 거부하도록 트래픽 정책 설명의 조건을 빌드할 때 주소 목록을 선택할 수 있습니다.

다음 콘솔 절차와 이에 AWS CLI 상응하는 절차는 수신자가 지정된 주소 목록에 있는 경우 수신 엔드포인트로 메시지를 허용하는 정책 설명을 생성하는 예를 보여줍니다.

트래픽 정책 설명에서 주소 목록을 사용하려면:

1. 에 설명된 대로 새 트래픽 정책을 생성하거나 기존 트래픽 정책을 편집합니다. 트래픽 정책 및 정책 스테이트먼트 만들기(콘솔)
2. 정책 문 컨테이너에서 문 조건이 충족될 때 작업을 수행할 수 있도록 허용을 선택합니다.
3. 다음과 같이 문 조건을 빌드합니다.
   - 프로토콜 필드에 수신자 주소를 선택합니다.
   - 연산자 필드의 주소 목록에 있음을 선택합니다.
   - 값 필드에 대한 주소 목록의 이름을 선택합니다.
4. 이는 한 가지 예에 불과하지만 주소 목록이 있는 다양한 연산자를 기반으로 할 수 있는 정책 조건을 더 추가할 수 있습니다.

사용 AWS CLI:

```
aws mailmanager create-traffic-policy \
  --default-action ALLOW \
  --traffic-policy-name "testpolicy" \
  --policy-statements '[{
    "Action": "ALLOW",
    "Conditions": [{
      "BooleanExpression": {
        "Evaluate": {
          "IsInAddressList": {
            "Attribute": "RECIPIENT",
            "AddressLists": [
              "arn:aws:ses:eu-west-3:123456789012:mailmanager-address-
list/al-123456789abc"
```

```
            ]
          }
        },
        "Operator": "IS_TRUE"
      }
  }]
}]'
```

규칙에서 주소 목록 사용

규칙 세트 중 하나에 사용되는 규칙의 조건을 빌드하여 규칙의 작업을 트리거할 때 주소 목록을 선택할 수 있습니다.

다음 콘솔 절차와 이에 AWS CLI 상응하는 절차는 수신자가 지정된 주소 목록에 있는 경우 삭제 작업을 호출하는 규칙을 생성하는 예를 보여줍니다.

규칙 조건에서 주소 목록을 사용하려면:

1. 에 설명된 대로 새 규칙을 생성하거나 기존 규칙을 편집합니다. 규칙 세트 및 규칙 만들기(콘솔)
2. 규칙 조건 컨테이너에서 다음과 같이 규칙의 조건을 빌드합니다.
   - 속성 선택 필드에서 수신자 주소를 선택합니다.
   - 연산자 선택 필드에서 주소 목록에 있음을 선택합니다.
   - 값 필드에 대한 주소 목록의 이름을 선택합니다.
3. 작업 컨테이너에서 새 작업 추가를 선택하고 작업 삭제를 선택합니다.
4. 이는 한 가지 예에 불과하지만, 수행할 다양한 작업에 대해 주소 목록이 있는 다양한 연산자를 기반으로 할 수 있는 규칙 조건을 더 추가할 수 있습니다.

사용 AWS CLI:

```
aws mailmanager create-rule-set \
  --rule-set-name "testruleset2" \
  --rules '[{
    "Name": "addresslist",
    "Conditions": [{
      "BooleanExpression": {
        "Evaluate": {
          "IsInAddressList": {
            "Attribute": "RECIPIENT",
```

```
            "AddressLists": [
              "arn:aws:ses:us-east-1:123456789012:mailmanager-address-
list/al-123456789abc"
            ]
          }
        },
        "Operator": "IS_TRUE"
      }
    }],
    "Actions": [{
      "Drop": {}
    }]
  }]'
```

모범 사례 및 고려 사항

- 목록 크기에 유의하세요. 목록이 매우 크면 성능에 영향을 미칠 수 있습니다.
- 주소 목록은 계정별로 다르며 동일한 AWS 계정 내에서만 사용할 수 있습니다.
- 중첩 주소 목록은 현재 지원되지 않습니다.
- 리전당 최대 100개의 주소 목록이 지원됩니다.
- 주소 목록당 최대 100,000개의 주소가 지원됩니다.

이메일 아카이빙

이메일 아카이빙은 수신 엔드포인트로 들어오는 지정된 이메일 유형 또는 구성 세트를 통해 보내는 이메일을 아카이빙하는 방법과 풍부한 고급 검색 필터 세트와 결과 내보내기 기능을 통해 아카이빙된 메시지를 찾는 방법을 제공합니다.

이메일 아카이빙은 영구적이고 안전한 장기 스토리지에 데이터를 저장하여 이메일을 저장하고 보호하며, 이메일을 빠르게 검색하고 아카이브할 수 있는 방법을 제공합니다. 메일박스 서버의 스토리지 요구사항을 늘리지 않고 풀타임 엔터프라이즈 수준 아카이브를 제공합니다.

아카이브에는 전송된 이메일과 수신된 이메일의 조합이 포함될 수 있습니다.

- 전송된 이메일 아카이브 - 아카이브 옵션이 활성화된 구성 세트를 통해 (아웃바운드) 이메일을 전송하면 해당 구성 세트를 사용하여 전송된 모든 이메일이 지정한 이메일 아카이브에 아카이브됩니다.
- 수신된 이메일 아카이브 - 수신 엔드포인트가 (인바운드) 이메일을 수신하면 트래픽 정책을 사용하여 차단하거나 허용할 이메일을 결정합니다. 여기서 허용하는 이메일은 조건부 규칙을 적용하여 특

정 유형의 이메일에 대해 정의한 작업을 실행하는 규칙 세트로 전달됩니다. 정의할 수 있는 규칙 작업 중 하나는 아카이브 작업입니다. 이 작업을 선택하면 이메일이 사용자가 지정한 이메일 아카이브에 보관 처리됩니다.

규칙 작업 또는 구성 세트에서 지정하려면 먼저 아카이브를 생성해야 합니다. 다음 섹션의 절차에서는 SES 콘솔에서 아카이브를 만드는 방법을 안내합니다.

Amazon SES 콘솔에서 이메일 아카이빙 사용

SES 콘솔의 이메일 아카이브 페이지는 아카이브 검색, 검색 기록, 내보내기 기록, 아카이브 관리라는 대화형 테이블 네 개로 구성되며, 이를 통해 아카이브에서 이메일을 검색하고, 결과를 내보내고, 아카이브를 관리할 수 있습니다. 다음 절차에서는 각 테이블에 대한 설명을 제공합니다.

이메일 아카이브 페이지를 사용하여 아카이브를 검색, 내보내기, 관리하는 방법

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 패널의 Mail Manager에서 이메일 아카이브를 선택합니다.
3. 이메일 아카이브 페이지는 테이블 네 개로 구성되며, 아카이브 검색, 검색 기록, 내보내기 기록, 아카이브 관리입니다. 이러한 각 테이블과 관련된 설명은 아래에서 해당 탭을 선택해 확인하세요.

Search archive

아카이브 검색은 대화형 테이블로, 특정 이메일부터 광범위한 범주와 일치하는 많은 이메일에 이르기까지 모든 항목을 찾을 수 있는 세부 검색 기준을 제공해 주는 다양한 필터 및 날짜 모음으로 아카이브된 메시지를 검색하고 찾을 수 있습니다. 검색 기준과 일치하는 메시지는 개별로 다운로드하거나 S3 버킷으로 대량으로 내보낼 수 있습니다.

아카이브된 이메일을 검색, 다운로드, 내보내는 방법

1. 이메일 아카이브 페이지에서 아카이브 검색 탭을 선택하여 아카이브 검색 테이블을 표시합니다.
2. 아카이브 필드 내부를 클릭하고 목록에서 아카이브를 선택한 다음 검색을 선택하거나 다음 단계를 사용하여 검색을 구체화합니다.
3. 날짜 범위 필드를 선택하여 검색의 날짜 범위 옵션을 펼칩니다.
   - 상대 범위(기본값) - 원하는 일수에 해당하는 라디오 버튼을 선택하거나 시간 단위와 최대 30일의 날짜 단위를 선택하여 사용자 지정 범위를 선택합니다.

- 절대 범위 - 최대 30일까지 시작 날짜 및 종료 날짜(필요한 경우 시간)를 입력합니다.

> **Note**
>
> - 아카이브 내에서 검색은 한 번에 30일로 제한됩니다. 예를 들어 6월 1일부터 7월 31일까지 메시지를 검색하려면 다음과 같이 세 가지 검색으로 나누어야 합니다.
> 1. 6월의 30일
> 2. 7월의 첫 30일
> 3. 7월 31일
> - 상대 범위 날짜의 경우 마지막 날은 자정에 종료됩니다. 예를 들어, Last 7 days(지난 7일)를 선택하면 일곱째 날은 어제 자정에 끝나는 날이 됩니다.

4. (선택 사항) 필터 필드를 선택하여 보내는 사람, 받는 사람, 참조, 제목 줄, 첨부 파일 있음 필터 중에서 선택하고 다음 속성을 적용합니다.
   - 최대 10개의 필터를 만들 수 있습니다.
   - 필터를 클릭하여 편집하거나 X를 선택하여 제거할 수 있습니다.
5. 검색을 선택하면 검색 기준과 일치하는 아카이브된 이메일이 검색 결과 테이블에 채워집니다.
   - 메시지 ID 열은 기본적으로 숨겨져 있지만 기어 모양 아이콘을 선택해 메시지 테이블을 보는 방법을 사용자 지정하여 표시할 수 있습니다.
   - 실행하는 모든 검색은 고유한 검색 ID와 함께 자동으로 저장되며 검색 기록 테이블에 나열됩니다.
6. 메시지의 텍스트를 봉투 및 헤더 정보와 함께 보려면 메시지의 라디오 버튼을 선택한 다음 세부 정보 보기를 선택하여 메시지 세부 정보 사이드바를 엽니다.
7. 메시지의 로컬 파일을 만들려면 메시지의 라디오 버튼을 선택한 다음 다운로드 메시지를 선택합니다.
8. 필터링된 검색은 S3로 내보내기를 선택하여 Amazon S3 버킷에 저장할 수 있습니다.
   a. 사용하려는 S3 버킷의 URI를 알고 있는 경우 S3 URI 필드에 입력합니다. 정보를 모르는 경우 S3 찾아보기를 선택하고 S3 페이지에서 사용할 S3 버킷 및 폴더를 선택합니다.
   b. (선택 사항) KMS AWS KMS 키 ARN 필드에 자체 키를 입력하거나 새 키 생성을 선택하여 내보낸 메시지를 암호화할 수 있습니다. 그렇지 않으면 암호화가 대상 S3 버킷(없는 경우에도)에서 사용되는 모든 메서드로 설정됩니다.

c. 내보내기를 선택하면 필터링된 검색에서 찾은 모든 메시지가 선택한 S3 폴더에 개별 파일로 저장됩니다.

> **Note**
> 아카이브에 포함할 수 있는 메시지 수에는 제한이 없지만 검색 결과는 검색 결과 테이블에 1,000개 행으로 제한됩니다.

Search history

검색 기록이 이 테이블에 나열되어 있으므로 결과 세트를 복원하거나 이전에 만들어진 복잡한 필터 세트에 액세스할 수 있습니다. 또한 필터와 날짜를 편집하여 원래 검색을 기반으로 새 검색을 만들 수 있습니다. 모든 새 검색은 고유한 검색 ID와 함께 자동으로 저장되며 이 테이블에 나열됩니다.

이전 검색을 보고 작업하는 방법

1. 이메일 아카이브 페이지에서 검색 기록 탭을 선택하여 검색 기록 테이블을 표시하면 상단에 가장 최근 검색을 포함하여 모든 아카이브된 이메일 검색 기록이 나열됩니다. 이 테이블은 처음 방문할 때 데이터가 로드됩니다. 탭을 전환하고 다시 돌아오면 새로 고침 아이콘을 사용하여 최신 데이터를 검색합니다.
2. 아카이브 필드 내부를 클릭하고 목록에서 아카이브를 선택합니다. 해당 아카이브에 속하는 모든 검색이 테이블에 채워집니다. 아래 단계에서 각 검색을 보고 더 많은 작업을 수행할 수 있습니다.
3. 이전 검색의 라디오 버튼을 선택한 다음 검색 결과 보기를 선택하여 원래 검색 결과를 복원합니다. 아카이브 검색 페이지가 열리고 원래 검색에 사용된 필터 세트와 날짜 범위가 해당 기준에 따라 이전에 확인된 모든 메시지와 함께 표시됩니다. 다음 방법으로 원래 검색을 확장할 수 있습니다.
   - 날짜 범위와 필터를 수정한 다음 검색을 선택하여 새 검색을 만듭니다.
   - 수행하는 모든 새 검색은 고유한 검색 ID와 함께 자동으로 저장되며 검색 기록 테이블에 나열됩니다.

Export history

이 테이블에는 내보내기 기록이 나열되어 있으므로 S3 콘솔에서 내보내기 폴더의 내용에 쉽게 액세스할 수 있습니다.

최근 내보내기를 보는 방법

1. 이메일 아카이브 페이지에서 내보내기 기록 탭을 선택하여 지난 30일 이내에 S3 버킷에 내보낸 모든 아카이브된 이메일 검색을 나열하는 내보내기 기록 테이블을 표시합니다. 이 테이블은 처음 방문할 때 데이터가 로드됩니다. 탭을 전환하고 다시 돌아오면 새로 고침 아이콘을 사용하여 최신 데이터를 검색합니다.
2. 내보내기 상태가 대기 중, 전처리 중, 처리 중이면 취소를 선택하여 취소할 수 있습니다.
3. S3 URI를 선택하여 포함된 파일을 볼 수 있는 S3 콘솔에서 내보내기의 버킷 폴더를 엽니다.

Manage archives

이 테이블에는 새 아카이브를 만들거나, 특정 아카이브를 검색하고, 세부 정보를 보거나, 아카이브를 편집하거나, 아카이브를 삭제할 수 있는 옵션이 있는 아카이브가 나열되어 있습니다.

아카이브를 만들고 관리하는 방법

1. 이메일 아카이브 페이지에서 아카이브 관리 탭을 선택하여 모든 이메일 아카이브를 나열하는 아카이브 테이블을 표시합니다. 이 테이블은 처음 방문할 때 데이터가 로드됩니다. 탭을 전환하고 다시 돌아오면 새로 고침 아이콘을 사용하여 최신 데이터를 검색합니다.
2. 특정 아카이브를 검색하려면 아카이브 필드에 입력을 시작합니다.
3. 아카이브의 세부 정보를 보려면 아카이브 이름 열에서 해당 이름을 선택합니다.
4. 아카이브를 만들려면 아카이브 만들기를 선택합니다.

   a. 아카이브 이름 필드에 고유한 이름을 입력합니다.

   b. (선택 사항) 보존 기간 필드에서 보존 기간을 선택하여 기본 보존 기간인 180일을 재정의합니다.

   c. (선택 사항) KMS AWS KMS 키 ARN 필드에 자체 키를 입력하거나 새 키 생성을 선택하여 아카이브를 암호화할 수 있습니다.

   아카이브 만들기를 선택합니다.
5. 이제 아카이브를 생성했으므로 규칙 세트에서 아카이브를 사용하여 수신(인바운드) 이메일을 아카이브하거나 구성 세트에서 이를 지정하여 전송(아웃바운드) 이메일을 아카이브할 수 있습니다.
6. 아카이브를 편집하려면 라디오 버튼을 선택한 다음 편집을 선택합니다.

   a. 아카이브 이름 필드에서 이름을 편집하거나 변경합니다.

b. 보존 기간 필드에서 보존 기간을 변경합니다.

아카이브 업데이트를 선택합니다.

7. 아카이브를 삭제하려면 라디오 버튼을 선택한 다음 삭제를 선택합니다.

   - 확인 필드에 `delete`를 입력하고 삭제를 선택합니다.

     아카이브 테이블에서 아카이브 상태가 삭제 보류 중으로 전환되고 30일 후에 자동으로 삭제됩니다.

이메일 추가 기능

Email 추가 기능은 수신 엔드포인트에 허용하는 이메일 유형을 관리하고 특정 유형의 이메일에 대해 수행할 작업을 결정하는 데 사용할 수 있는 SES 승인 공급자의 전문 보안 도구 제품군입니다. 이러한 도구는 이메일 워크플로에 통합할 준비가 되어 있고 Mail Manager 콘솔에서 직접 활성화할 수 있는 인증된 보안 인텔리전스 및 적용 솔루션입니다.

이러한 추가 기능은 필요에 맞게 최적화되지 않을 수 있는 대규모 단일 제품 솔루션을 구매하는 대신 개별 사용 사례에 적합한 검증된 이메일 보안 솔루션 중에서 선택할 수 있는 유연성을 제공하며, 이는 사용한 만큼 요금이 부과될 수 있습니다. 이메일 추가 기능은 주요 위협 인텔리전스 및 보안 적용 기능을 워크로드별로 확장하므로 필요한 용량을 추측할 필요가 없습니다. 이러한 이점을 통해 이메일 보안 문제를 미리 대비하고 조직의 높은 서비스 표준을 유지하는 데 집중할 수 있습니다.

Mail Manager 콘솔에 있는 이메일 추가 기능 페이지에서 직접 각 추가 기능에 대해 자세히 알아볼 수 있으며, 콘솔에서 제품 설명, 주요 이점, 요금 정보에 액세스할 수 있습니다. 사용하려는 추가 기능을 결정하면 Mail Manager 콘솔에서 구독하기만 하면 됩니다. 구독하면 수신 엔드포인트로 허용되는 이메일을 결정할 때 트래픽 정책 조건으로 선택하거나 특정 이메일에 대해 수행할 작업을 결정하는 규칙 세트 조건으로 선택할 수 있습니다. 모든 추가 기능에 대한 기본 지원은에서 제공하며 Mail Manager 콘솔에서도 액세스할 AWS 수 있습니다.

다음 섹션의 절차에서는 Mail Manager 콘솔에서 이메일 추가 기능을 구독하는 방법을 안내합니다.

Mail Manager 콘솔에서 이메일 추가 기능 구독

다음 절차에서는 트래픽 정책 또는 규칙 세트에서 사용할 수 있도록 Mail Manager 콘솔의 이메일 추가 기능 페이지를 사용하여 추가 기능을 구독하는 방법을 보여줍니다.

콘솔을 사용하여 이메일 추가 기능을 구독하는 방법

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 왼쪽 탐색 패널의 Mail Manager에서 이메일 추가 기능을 선택합니다.
3. 이메일 추가 기능 페이지에서 추가 기능 카드의 제목을 선택하여 개요 페이지를 엽니다. 여기에서 기능, 주요 이점, 요금 정보를 자세히 알아볼 수 있습니다. 이 추가 기능을 사용하려면 구독을 선택합니다.

   - 제시된 이용 약관을 읽고 동의함 상자를 선택한 다음 구독을 선택합니다.
4. 추가 기능을 구독하면 수신 엔드포인트에 이메일을 거부하거나 허용하는 트래픽 정책 조건 또는 규칙 세트 조건으로 선택하여 적격 메시지에 대해 수행할 작업을 결정하여 이메일 워크플로에 통합할 수 있습니다.

   다음 예제에서는 정책 스테이트먼트 조건 및 규칙 조건에서 추가 기능을 사용하는 방법을 보여줍니다.

   추가 기능을 사용한 트래픽 정책 예제

   정책 스테이트먼트 조건에서 Spamhaus Domain Block List 추가 기능을 사용하여 Spamhaus에 나열된 도메인에서 시작된 수신 엔드포인트로 들어오는 이메일을 차단합니다.

   이메일 추가 기능을 사용하여 트래픽 정책을 만들고 정책 스테이트먼트 조건을 빌드하는 방법에 대한 자세한 내용은 the section called “트래픽 정책 및 정책 스테이트먼트 만들기(콘솔)” 섹션을 참조하세요.

   추가 기능을 사용한 규칙 조건 예제

   규칙 조건에서 Trend Micro 바이러스 검사 추가 기능을 사용하여 바이러스 검사를 통과하는 이메일에 대한 규칙 작업을 결정합니다.

이메일 추가 기능을 사용하여 규칙 세트를 만들고 규칙 조건을 작성하는 방법에 대한 자세한 내용은 the section called "규칙 세트 및 규칙 만들기(콘솔)" 섹션을 참조하세요.

5. 구독 중인 추가 기능에 대한 일반 세부 정보를 보거나 지원에 액세스하려면 이메일 추가 기능 페이지에서 이름을 선택하여 개요 페이지를 엽니다.

   - 일반 세부 정보에서 구독 날짜 및 추가 기능의 Amazon 리소스 이름(ARN)을 볼 수 있습니다.
   - 지원 탭을 선택하여 AWS 지원으로 연결되는 링크에 액세스합니다.

6. 추가 기능 구독을 해지하는 방법은 다음과 같습니다.

   a. 먼저 조건에 정의된 트래픽 정책 또는 규칙 세트에서 제거해야 합니다. 이렇게 하지 않으면 다음 구독 해지 단계를 진행할 수 없습니다.

   b. 이메일 추가 기능 페이지에서 이름을 선택하여 개요 페이지를 연 다음 구독 해지를 선택합니다.

   c. 확인 필드에 `confirm`을 입력하고 구독 해지를 선택합니다.

Mail Manager에 대한 권한 정책

이 장에 나와 있는 정책은 Mail Manager의 모든 다양한 기능을 활용하는 데 필요한 정책에 대한 단일 참조 자료로 제공됩니다.

Mail Manager 기능 페이지에서는 기능을 활용하는 데 필요한 정책이 포함된 이 페이지의 각 섹션으로 연결되는 링크가 제공됩니다. 필요한 정책의 복사 아이콘을 선택하고 해당 기능 설명에 지시된 대로 붙여넣습니다.

다음 정책은 리소스 권한 정책 및 AWS Secrets Manager 정책을 통해 Amazon SES Mail Manager에 포함된 다양한 기능을 사용할 수 있는 권한을 부여합니다. 권한 정책을 처음 사용하는 경우 the section called “정책 구조” 및 AWS Secrets Manager에 대한 권한 정책을 참조하세요.

수신 엔드포인트에 대한 권한 정책

이 섹션의 두 정책 모두 수신 엔드포인트를 만드는 데 필요합니다. 수신 엔드포인트를 만드는 방법과 이러한 정책을 사용하는 경우를 알아보려면 the section called “수신 엔드포인트 만들기(콘솔)” 섹션을 참조하세요.

수신 엔드포인트에 대한 Secrets Manager 보안 암호 리소스 권한 정책

SES가 수신 엔드포인트 리소스를 사용하여 보안 암호에 액세스하도록 허용하려면 다음 Secrets Manager 보안 암호 리소스 권한 정책이 필요합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Id": "Id",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "000000000000"
```

```
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:ses:us-
east-1:000000000000:mailmanager-ingress-point/*"
                }
            }
        }
    ]
}
```

수신 엔드포인트에 대한 KMS 고객 관리형 키(CMK) 관련 키 정책

SES가 보안 암호를 사용하는 동안 키를 사용하도록 허용하려면 다음 KMS 고객 관리형 키(CMK) 관련 키 정책이 필요합니다.

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "ses.amazonaws.com"
    },
    "Action": "kms:Decrypt",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
           "kms:ViaService": "secretsmanager.us-east-1.amazonaws.com",
            "aws:SourceAccount": "000000000000"
        },
        "ArnLike": {
            "aws:SourceArn": "arn:aws:ses:us-east-1:000000000000:mailmanager-ingress-
point/*"
        }
    }
}
```

SMTP 릴레이에 대한 권한 정책

SMTP 릴레이를 만들려면 이 섹션의 두 정책이 모두 필요합니다. SMTP 릴레이를 만드는 방법과 이러한 정책을 사용하는 경우를 알아보려면 the section called “SMTP 릴레이 만들기(콘솔)” 섹션을 참조하세요.

SMTP 릴레이에 대한 Secrets Manager 보안 암호 리소스 권한 정책

SES가 SMTP 릴레이 리소스를 사용하여 보안 암호에 액세스하도록 허용하려면 다음 Secrets Manager 보안 암호 리소스 권한 정책이 필요합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret"
            ],
            "Principal": {
                "Service": [
                    "ses.amazonaws.com"
                ]
            },
            "Resource": "*",
            "Condition": {
               "StringEquals": {
                   "aws:SourceAccount": "888888888888"
                },
               "ArnLike": {
                   "aws:SourceArn": "arn:aws:ses:us-
east-1:888888888888:mailmanager-smtp-relay/*"
               }
           }
        }
    ]
}
```

SMTP 릴레이에 대한 KMS 고객 관리형 키(CMK) 관련 키 정책

SES가 보안 암호를 사용하는 동안 키를 사용하도록 허용하려면 다음 KMS 고객 관리형 키(CMK) 관련 키 정책이 필요합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey"
            ],
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "secretsmanager.us-east-1.amazonaws.com",
                    "aws:SourceAccount": "000000000000"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:ses:us-
east-1:000000000000:mailmanager-smtp-relay/*"
                }
            }
        }
    ]
}
```

이메일 아카이빙에 대한 권한 정책

내보내기 아카이빙

IAM 자격 증명 호출은 다음 정책에 의해 구성된 대상 S3 버킷에 액세스할 수 있어야 `StartArchiveExport` 합니다.

JSON

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::MyDestinationBucketName"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:PutObjectTagging",
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::MyDestinationBucketName/*"
        }
    ]
}
```

이는 대상 버킷용 정책입니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::MyDestinationBucketName"
        },
        {
            "Effect": "Allow",
```

```
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:PutObjectTagging",
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::MyDestinationBucketName/*"
        }
    ]
}
```

> **Note**
>
> 아카이빙은 혼동된 대리 조건 키(aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, aws:SourceOrgPaths)를 지원하지 않습니다. 이는 Mail Manager의 이메일 아카이빙이 실제 내보내기를 시작하기 전에 먼저 직접 호출 ID가 전달 액세스 세션을 사용하여 내보내기 대상 버킷에 대한 쓰기 권한이 있는지 테스트하여 혼동된 대리자 문제를 방지하기 때문입니다.

KMS CMK로 저장 시 암호화 아카이빙

IAM 자격 증명 호출 `CreateArchive` 및는 다음 정책을 통해 KMS 키 ARN에 액세스할 수 있어야 `UpdateArchive` 합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "kms:DescribeKey",
            "kms:Decrypt",
            "kms:GenerateDataKey"
        ],
        "Resource": "arn:aws:kms:us-west-2:111122223333:key/MyKmsKeyArnID"
    }
```

```
}
```

이메일 아카이빙에 필요한 KMS 키 정책입니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:user/MyUserRoleOrGroupName"
            },
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": [
                        "ses.us-east-1.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

규칙 작업을 실행할 수 있는 권한 및 신뢰 정책

SES 규칙 실행 역할은 규칙 실행에 AWS 서비스 및 리소스에 액세스할 수 있는 권한을 부여하는 AWS Identity and Access Management (IAM) 역할입니다. 규칙 세트에서 규칙을 생성하기 전에 필요한 AWS 리소스에 대한 액세스를 허용하는 정책을 사용하여 IAM 역할을 생성해야 합니다. 규칙 작업을 실행할 때 SES가 이 역할을 맡습니다. 예를 들어 규칙 조건이 충족될 때 수행할 규칙 작업으로 S3 버킷에 이메일 메시지를 쓸 수 있는 권한이 있는 규칙 실행 역할을 만들 수 있습니다.

따라서 각 특정 규칙 작업을 실행하는 데 필요한이 섹션의 개별 권한 정책 외에도 다음 신뢰 정책이 필요합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Principal": {
          "Service": "ses.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
          "StringEquals": {
                   "aws:SourceAccount": "888888888888"
          },
          "ArnLike": {
             "aws:SourceArn": "arn:aws:ses:us-east-1:888888888888:mailmanager-
rule-set/*"
          }
        }
      }
    ]
  }
```

규칙 작업 정책

- S3에 쓰기 규칙 작업에 대한 권한 정책

- 메일박스로 전송 규칙 작업에 대한 권한 정책
- 인터넷으로 전송 규칙 작업에 대한 권한 정책
- Q Business로 전송 규칙 작업에 대한 권한 정책
- SNS에 게시 규칙 작업에 대한 권한 정책

S3에 쓰기 규칙 작업에 대한 권한 정책

수신된 이메일을 S3 버킷에 전달하는 S3에 쓰기 규칙 작업을 사용하려면 다음 정책이 필요합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutObject",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::MyDestinationBucketName/*"
            ]
        },
        {
            "Sid": "AllowListBucket",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::MyDestinationBucketName"
            ]
        }
    ]
}
```

서버 측 암호화가 활성화된 S3 버킷에 AWS KMS 고객 관리형 키를 사용하는 경우 IAM 역할 정책 작업을 추가해야 합니다`"kms:GenerateDataKey*"`. 앞의 예제를 다시 사용하여 이 작업을 역할 정책에 추가하면 다음과 같이 표시됩니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowKMSKeyAccess",
            "Effect": "Allow",
            "Action": "kms:GenerateDataKey*",
            "Resource": "arn:aws:kms:us-east-1:888888888888:key/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "kms:ResourceAliases": [
                        "alias/MyKeyAlias"
                    ]
                }
            }
        }
    ]
}
```

AWS KMS 키에 정책을 연결하는 방법에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 에서 키 정책 사용을 AWS KMS 참조하세요.

메일박스로 전송 규칙 작업에 대한 권한 정책

수신된 이메일을 Amazon WorkMail 계정으로 전달하는 메일박스로 전송 규칙 작업을 사용하려면 다음 정책이 필요합니다.

JSON

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

```
        "Action": ["workmail:DeliverToMailbox"],
        "Resource": "arn:aws:workmail:us-
east-1:888888888888:organization/MyWorkMailOrganizationID>"
      }
    ]
  }
```

인터넷으로 전송 규칙 작업에 대한 권한 정책

수신된 이메일을 외부 도메인으로 보내는 인터넷으로 전송 규칙 작업을 사용하려면 다음 정책이 필요합니다.

> **Note**
>
> SES 자격 증명이 기본 구성 세트를 사용하는 경우 다음 예제와 같이 구성 세트 리소스도 추가해야 합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": ["ses:SendEmail", "ses:SendRawEmail"],
        "Resource":[
        "arn:aws:ses:us-east-1:888888888888:identity/example.com",
        "arn:aws:ses:us-east-1:888888888888:configuration-set/my-configuration-
set"
        ]
      }
    ]
  }
```

Q Business로 전송 규칙 작업에 대한 권한 정책

수신된 이메일을 Amazon Q Business 인덱스로 전송하는 Q Business로 전송 규칙 작업을 사용하려면 다음 정책이 필요합니다.

Amazon Q Business 정책:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccessToQBusiness",
            "Effect": "Allow",
            "Action": [
                "qbusiness:BatchPutDocument"
            ],
            "Resource": [
                "arn:aws:qbusiness:us-
east-1:888888888888:application/ApplicationID/index/IndexID"
            ]
        }
    ]
}
```

Amazon Q Business에 대한 KMS 정책:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccessToKMSKeyForQbusiness",
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Encrypt",
                "kms:DescribeKey"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:888888888888:key/*"
            ],
            "Condition": {
                "StringEquals": {
```

```
                    "kms:ViaService": "qbusiness.us-east-1.amazonaws.com",
                    "kms:CallerAccount": "888888888888"
                },
                "ForAnyValue:StringEquals": {
                    "kms:ResourceAliases": [
                        "alias/MyKeyAlias"
                    ]
                }
            }
        }
    ]
}
```

AWS KMS 키에 정책을 연결하는 방법에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 에서 키 정책 사용을 AWS KMS 참조하세요.

SNS에 게시 규칙 작업에 대한 권한 정책

수신된 이메일을 Amazon SNS SNS 주제에 전달하는 SNS에 게시 규칙 작업을 사용하려면 다음 정책이 필요합니다.

Amazon SNS 정책:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccessToSNSTopic",
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": [
                "arn:aws:sns:us-east-1:888888888888:MySnsTopic"
            ]
        }
    ]
}
```

Amazon SNS에 대한 KMS 정책:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccessToKMSKeyForSNS",
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Encrypt",
                "kms:DescribeKey"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:888888888888:key/*"
            ],
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "qbusiness.us-east-1.amazonaws.com",
                    "kms:CallerAccount": "888888888888"
                 },
                 "ForAnyValue:StringEquals": {
                    "kms:ResourceAliases": [
                        "alias/MyKeyAlias"
                    ]
                }
            }
        }
    ]
}
```

AWS KMS 키에 정책을 연결하는 방법에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 에서 키 정책 사용을 AWS KMS 참조하세요.

Mail Manager 로깅

Mail Manager 로깅은 Mail Manager 작업에 대한 자세한 가시성을 제공합니다. 로깅 기능은 수신 엔드포인트의 초기 수신부터 구성된 규칙 세트 및 규칙에 따른 메시지 처리까지 메시지 흐름을 추적합니다.

Mail Manager는 다음 리소스에 대한 로깅을 제공합니다.

- 수신 엔드포인트
- 규칙 세트

Mail Manager는 Amazon CloudWatch Logs 서비스를 사용하여 로그를 전송하며 로그는 CloudWatch Logs, Amazon S3 또는 Amazon Data Firehose 대상 중 하나로 전송할 수 있습니다.

Mail Manager 로그 전송 설정

작동하는 로그 전송은 다음 세 가지 요소로 구성됩니다.

- DeliverySource - 수신 엔드포인트 또는 규칙 세트 등 로그를 전송하는 리소스를 나타내는 논리적 객체입니다.
- DeliveryDestination - 실제 전송 대상(CloudWatch Logs, S3 또는 Firehose)을 나타내는 논리적 객체입니다.
- 전송 - 전송 소스를 전송 대상에 연결합니다.

이 섹션에서는 Mail Manager 로깅을 사용하는 데 필요한 권한과 함께 이러한 객체를 생성하는 방법을 설명합니다.

사전 조건

Mail Manager 로깅을 설정하기 전에 다음을 확인합니다.

1. 수신 엔드포인트 또는 규칙 세트를 생성했습니다.
2. Mail Manager 리소스에서 전송 대상으로 로그를 보내는 데 필요한 CloudWatch Logs 및 SES Mail Manager 권한이 있습니다.

필수 권한

Amazon CloudWatch Logs 사용 설명서의 추가 권한 [V2]이 필요한 로깅 섹션에 설명된 대로 판매된 로그 권한을 설정하고 전송 대상에 해당하는 권한을 적용해야 합니다. Amazon CloudWatch

- CloudWatch Logs로 전송된 로그
- Amazon S3로 보낸 로그
- Firehose로 전송된 로그

또한 Mail Manager에서 로그 전송을 구성하려면 다음과 같은 사용자 권한이 필요합니다.

- `ses:AllowVendedLogDeliveryForResource` - Mail Manager가 예제와 같이 특정 리소스에 대해 사용자를 대신하여 CloudWatch Logs에 로그를 전송할 수 있도록 허용하는 데 필요합니다.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSesMailManagerLogDelivery",
            "Effect": "Allow",
            "Action": [
                "ses:AllowVendedLogDeliveryForResource"
            ],
            "Resource": [
            "arn:aws:ses:us-east-1:111122223333:mailmanager-ingress-point/inp-
xxxxx",
            "arn:aws:ses:us-east-1:111122223333:mailmanager-rule-set/rs-xxxx"
            ]
        }
    ]
}
```

SES 콘솔에서 로깅 활성화

콘솔을 사용하여 Mail Manager 리소스에 대한 로깅을 활성화하려면:

1. https://console.aws.amazon.com/ses/ SES 콘솔을 엽니다.
2. 탐색 창의 Mail Manager에서 수신 엔드포인트 또는 규칙 세트를 선택하고 로깅에 활성화하려는 특정 리소스를 선택합니다.
3. 리소스의 세부 정보 페이지에서 로그 전송 추가를 확장하고 CloudWatch Logs, S3 또는 Firehose로 전송을 선택합니다.
4. 선택한 대상에 대한 전송 추가 대화 상자에서 프롬프트에 따라 대상 유형에 맞는 로그 전송 옵션을 구성합니다.
5. (선택 사항) 추가 설정을 확장하여 레코드, 출력 형식, 필드 구분 기호 및 대상 유형과 관련된 기타 파라미터의 필드를 사용자 지정합니다.

CloudWatch Logs API를 사용하여 로깅 활성화

CloudWatch Logs API를 사용하여 Mail Manager 리소스에 대한 로깅을 활성화하려면 다음을 수행해야 합니다.

1. 를 사용하여 DeliverySource를 생성합니다`PutDeliverySource`.
2. 를 사용하여 DeliveryDestination을 생성합니다`PutDeliveryDestination`.
3. 를 사용하여 정확히 하나의 전송 소스와 하나의 전송 대상을 페어링하여 전송을 생성합니다`CreateDelivery`.

Amazon CloudWatch Logs 사용 설명서의 로깅[V2] 섹션에서 특정 로깅 대상에 필요한 모든 권한이 있는 IAM 역할 및 권한 정책의 예를 보고 CloudWatch Logs, S3 또는 Firehose와 같은 특정 로깅 대상 리소스에 대한 업데이트 허용을 포함하여 로깅 대상에 대한 IAM 역할 및 권한 정책 예제를 따를 수 있습니다.

> **Note**
>
> DeliverySource를 생성할 때는 수신 엔드포인트 ARN 또는 규칙 세트 ARN일 `resourceArn` 수 있습니다. DeliverySource에 따라는 다음과 같을 `logType` 수 있습니다.
>
> - 수신 엔드포인트 ARN – `APPLICATION_LOGS` 또는 `TRAFFIC_POLICY_DEBUG_LOGS`
> - 규칙 세트 ARN - `APPLICATION_LOGS`

로그 해석

로그는 Mail Manager에서 처리되는 수신된 메시지의 흐름에 대한 추가 인사이트를 얻는 데 사용할 수 있습니다.

다음 예제에서는 각 리소스 및 로그 유형에 대한 로그의 다양한 필드를 자세히 설명합니다.

로그 예제

- 수신 엔드포인트 로그 - APPLICATION_LOGS
- 수신 엔드포인트 로그 - TRAFFIC_POLICY_DEBUG_LOGS
- 규칙 세트 로그 - APPLICATION_LOGS

수신 엔드포인트 로그 - APPLICATION_LOGS

로그는 메시지별로 생성됩니다.

```
{
  "resource_arn": "arn:aws:ses:us-east-1:1234567890:mailmanager-ingress-point/inp-
xxxxx",
  "event_timestamp": 1728562395042,
  "ingress_point_type": "OPEN" | "AUTH",
  "ingress_point_name": "MyIngressPoint",
  "message_id": "0000llcki1jmushh817gr586f963a5inhkvnh81",
  "message_size_bytes": 100000,
  "rule_set_id": "rs-xxxx",
  "sender_ip_address": "1.2.3.4",
  "smtp_mail_from": "someone@domain.com",
  "smtp_helo": "domain.com",
  "tls_protocol": "TLSv1.2",
  "tls_cipher_suite": "TLS_AES_256_GCM_SHA384",
  "recipients": ["me@mydomain.com", "you@mydomain.com", "they@mydomain.com"],
  "ingress_point_metadata": { // only applies to AUTH Ingress Endpoint
       "password_version": "",
       "secrets_manager_arn": ""
  }
}
```

> **Note**
>
> 로그는 수신 엔드포인트에서 수락한 메시지에 대해서만 생성됩니다. 수신되는 모든 메시지를 거부하는 수신 엔드포인트는 애플리케이션 로그를 게시하지 않습니다.

CloudWatch Logs Insights 쿼리 예제

sender@domain.com에서 메시지 쿼리:

```
fields @timestamp, @message, @logStream, @log
| filter smtp_mail_from like /sender@domain.com/
| sort @timestamp desc
| limit 10000
```

크기가 5000바이트보다 큰 쿼리 메시지:

```
fields @timestamp, @message, @logStream, @log
| filter message_size_bytes > 5000
| sort @timestamp desc
| limit 10000
```

수신 엔드포인트 로그 - TRAFFIC_POLICY_DEBUG_LOGS

로그는 수신자별로 생성됩니다.

```
{
    "resource_arn": "arn:aws:ses:us-east-1:1234567890:mailmanager-ingress-point/inp-
xxxxx" CPY,
    "event_timestamp": 1728562395042,
    "ingress_point_type": "OPEN" | "AUTH",
    "ingress_point_id": "inp-xxxx",
    "ingress_point_session_id": "xxxx",
    "traffic_policy_id": "tp-xxxx",
    "traffic_policy_evaluation": [
           // Array of policy evaluations
           {
                "action": "ALLOW" | "DENY",
                "conditions": [
                // Array of conditions
                {
                    "expression": {
                        "attribute": "RECIPIENT",
                        "operator": "CONTAINS",
                        "value": ["@domain.com", "@mydomain.com"]
                    },
                    "expressionResult": true | false
                }],
                "policyStatementMatched": true | false
            },
            // If no policy statement match then default action will be applied
            {
                "action": "ALLOW" | "DENY",
                "policyStatementMatched": true,
                "type": "DefaultAction"
            }
    ],
    "traffic_policy_verdict": "REJECT" | "ACCEPT",
    "sender_ip_address": "1.2.3.4",
    "smtp_mail_from": "someone@domain.com",
```

```
    "smtp_helo": "domain.com",
    "tls_protocol": "TLSv1.2",
    "recipient": "me@mydomain.com",
    "tls_cipher_suite": "TLS_AES_256_GCM_SHA384"
}
```

> **Note**
>
> - 로그는 수락 또는 거부 여부에 관계없이 수신 엔드포인트에서 트래픽 정책에 의해 평가되는 모든 메시지에 대해 생성됩니다.
> - 동일한 메시지에 속하는 모든 수신자 트래픽 정책 평가(동일한 SMTP 대화 내)는 공통를 공유합니다`ingress_point_session_id`. 이 ID는 메시지 수락 후까지를 `message_id` 사용할 수 없으므로 상관관계 식별자 역할을 합니다.
> - `traffic_policy_evaluation` 콘텐츠는 구성에 따라 다르며 결과가 결정되면 조기에 종료될 수 있습니다.

CloudWatch Logs Insights 쿼리 예제

sender@domain.com에서 메시지 쿼리:

```
fields @timestamp, @message, @logStream, @log
| filter smtp_mail_from like /sender@domain.com/
| sort @timestamp desc
| limit 10000
```

특정에 속하는 쿼리 메시지`ingress_point_session_id`:

```
fields @timestamp, @message, @logStream, @log
| filter ingress_point_session_id = 'xxx'
| sort @timestamp desc
| limit 10000
```

거부된 메시지 쿼리:

```
fields @timestamp, @message, @logStream, @log
| filter traffic_policy_verdict = 'REJECT'
| sort @timestamp desc
```

```
| limit 10000
```

규칙 세트 로그 - APPLICATION_LOGS

로그는 작업당 메시지별로 생성됩니다. 즉, 규칙 세트의 규칙에서 작업이 메시지를 처리할 때마다 로그 레코드가 생성됩니다.

```
{
   "resource_arn": "arn:aws:ses:us-east-1:1234567890:mailmanager-rule-set/rs-xxxx",
   "event_timestamp": 1732298258254,
   "message_id": "0000llcki1jmushh817gr586f963a5inhkvnh81",
   "rule_set_name": "MyRuleSet",
   "rule_name": "MyRule",
   "rule_index": 1,
   "recipients_matched": ["recipient1@domain.com", "recipient2@domain.com"],
   "action_metadata": {
       "action_name": "WRITE_TO_S3" | "DROP" | "RELAY" | "DELIVER_TO_MAILBOX" | etc.,
       "action_index": 2,
       "action_status": "SUCCESS" | "FAILURE" | "IN_PROGRESS",
       "action_failure": "Access denied"
   }
}
```

- `recipients_matched` - 작업이 수행되는 규칙의 조건과 일치하는 수신자입니다.
- `rule_index` - 규칙 세트 내의 규칙 순서입니다.
- `action_index` - 규칙 내 작업의 순서입니다.
- `action_status` - 지정된 메시지에 대해 작업을 수행한 결과를 나타냅니다.
- `action_failure` - 작업의 실패 세부 정보를 나타냅니다(작업이 실패할 때만 적용됨). 예를 들어 제공된 역할에 작업을 수행할 수 있는 충분한 권한이 없는 경우입니다.

또한 규칙 조건이 메시지와 일치하지 않는 경우, 즉 메시지가 규칙에 의해 처리되지 않는 경우 단일 로그가 게시되어 메시지가 규칙 세트에 의해 처리되었지만 이에 대해 수행된 작업이 없음을 나타냅니다.

```
{
   "resource_arn": "arn:aws:ses:us-east-1:1234567890:mailmanager-rule-set/rs-xxxx",
   "event_timestamp": 1732298258254,
   "message_id": "0000llcki1jmushh817gr586f963a5inhkvnh81",
   "rule_set_name": "MyRuleSet",
   "rule_name": "MyRule",
```

```
    "rule_index": 1,
    "recipients_matched": [],
}
```

CloudWatch Logs Insights 쿼리 예제

특정 message-id에 대한 쿼리(규칙 세트를 통한 메시지 흐름 표시):

```
fields @timestamp, @message, @logStream, @log
| filter message_id = 'message-id-123'
| sort @timestamp desc
| limit 10000
```

실패한 WRITE_TO_S3 작업에 대한 쿼리:

```
fields @timestamp, @message, @logStream, @log
| filter action_metadata.action_name = 'WRITE_TO_S3'
    and action_metadata.action_status = 'FAILURE'
| sort @timestamp desc
| limit 10000
```

규칙 세트의 두 번째 규칙에 의해 처리되지 않은 메시지에 대한 쿼리(메시지가 규칙 조건을 충족하지 않음):

```
fields @timestamp, @message, @logStream, @log
| filter recipients_matched = '[]'
    and rule_index = 2
| sort @timestamp desc
| limit 10000
```

Amazon Simple Email Service에서 목록 및 구독 관리

Amazon SES에서 메일 발송 및 구독과 이메일 금지에 대한 자체 목록을 관리할 수 있습니다. 발신자 평판을 유지하기 위해 SES는 유효하지 않은 수신자에게 전송하여 발신자 평판을 떨어뜨리는 일이 없도록 하는 계정 수준 및 구성 세트 수준 금지 목록을 제공합니다. 반송 메일 및 수신 거부에 대한 또 다른 조치로 SES는 구독 관리를 통해 모든 발신 메일에 수신 거부 링크를 자동으로 추가할 수 있습니다.

이러한 각 목록 유형은 이 장의 주제에 나열된 섹션에서 자세히 설명합니다. 그러나 여기서는 전역 금지 목록 관리의 주요 변경 사항뿐만 아니라 차이점을 파악할 수 있도록 금지 목록의 개요를 소개합니다. 이 장에서 설명하는 목록으로 작업하기 전에 이 개요를 읽는 것이 좋습니다.

금지 목록 및 금지 재정의 메커니즘 개요

전역 금지 목록 제거 기능은 더 이상 고객에게 제공되지 않으며, 고객은 더 이상 이 기능과 상호 작용하여 금지를 관리할 수 없습니다. 전역 금지 목록은 SES에 의해 백그라운드에서 작동되고 관리됩니다. 고객은 이제 계정 수준 금지 목록 및 구성 세트 수준 금지 재정의를 사용할 수 있습니다. 이는 고객의 계정에 대한 이메일 금지를 처리하는 방법과 관련하여 사용자 지정된 제어 기능을 제공합니다.

다양한 유형의 금지 목록, 범위 및 이들이 제공하는 이점은 아래에 설명되어 있습니다.

- 전역 금지 목록 - SES 공유 IP 풀에서 주소의 평판을 보호하기 위해 SES가 소유 및 관리합니다.
- 계정 수준 금지 목록 - 계정 평판을 보호하기 위해 고객이 소유 및 관리하며, 전역 금지 목록을 재정의합니다.
  - 구성 세트 수준 금지 - 구성 세트에 지정된 재정의를 사용하여 계정 수준 금지 목록을 조건부 또는 세분화된 방식으로 제어할 수 있는 재정의 메커니즘입니다.

전역 금지 목록은 새로운 Amazon SES 콘솔 및 API v2에 계정 수준 및 구성 세트 수준 금지 기능이 도입될 때까지 금지 목록의 유일한 유형이었습니다. 전역 금지 목록은 SES의 평판을 보호하기 위해 SES가 소유 및 관리합니다. 전용 IP가 없는 한 모든 SES 고객이 동일한 IP 주소 풀을 공유하고 있는 만큼, 고객이 스팸을 보내지 않도록 하고 SES 공유 IP 풀에서 해당 IP 주소의 평판에 부정적인 영향을 줄 수 있는 모든 어떠한 행위도 하지 않도록 하는 것이 SES에게 중요하므로 이 기능이 필요한 것입니다. 고객은 더 이상 전역 금지 목록과 직접 상호 작용하지 않지만, 백그라운드에서 계속 작동하며, 전역 금지 목록 작동 방식에 일반적인 원칙을 적용하여 다른 유형의 금지 작동 방식에서 전체 보안 주체를 설명할 수도 있습니다. Amazon SES 전역 금지 목록을(를) 참조하세요.

> **Note**
>
> Amazon SES 콘솔에는 전역 금지 목록 제거 요청 양식이 더 이상 없습니다. 이 섹션에서 설명하는 모든 이점을 제공하는 계정 수준 금지 목록으로 대체되었기 때문입니다.

계정 수준 금지 목록이 도입되어 고객이 자체 금지 목록 및 평판을 만들고 제어할 수 있으므로, 계정 수준 금지 목록은 계정에만 적용됩니다. 새 콘솔의 계정 수준 금지 목록 인터페이스는 주소를 추가 또는 제거하기 위한 대량 작업을 비롯하여 계정 수준 금지 목록의 주소를 손쉽게 관리할 수 있는 방법을 제공합니다. 전역 금지 목록에 포함되어 있지만 계정 수준 금지 목록에 없는 주소(즉, 해당 주소로 전송되기를 원함)로 전송하면, Amazon SES가 배달을 시도하고 반송될 경우 반송 메일은 본인의 평판에 영향을 주지만, 본인의 계정 수준 금지 목록을 사용하지 않는 경우 해당 이메일 주소로 보낼 수 없기 때문에 반송 메일을 받을 수 없습니다. 따라서 계정 수준 금지 목록은 계정에 대한 전역 금지 목록만 재정의합니다. Amazon SES 계정 수준 금지 목록 사용을(를) 참조하세요.

구성 세트 수준 금지는 목록은 아니지만, 금지 사용자 지정을 구성하고, 각기 다른 이메일 전송 시나리오를 위해 특별히 만들어진 구성 세트를 사용함으로써 계정 수준 금지 목록에 재정의할 수 있는 메커니즘입니다. 예를 들어 이런 상황을 가정해 보겠습니다. 추가될 반송 메일 주소와 수신 거부 주소 모두에 대해 계정 수준 금지 목록이 구성되어 있습니다. 그러나 사용자의 구성 세트에 특정 인구 통계가 정의되어 있고, 사용자는 이 구성 세트에 수신 거부 주소만 추가되었으면 합니다. 그러기 위해서는 이 구성 세트를 사용하여 전송된 이메일에서 수신 거부에 대해서만(계정 수준 금지 목록에 설정된 반송 메일 및 수신 거부가 아님) 계정 수준 금지 목록에 이메일 주소가 추가되도록 하여 구성 세트의 금지 목록이 재정의하도록 하면 됩니다. 구성 세트 수준 금지 목록에는 금지를 전혀 사용하지 않는 것을 포함하여 계정 수준 금지를 재정의하는 다양한 수준이 있습니다. 구성 세트 수준 금지를 사용하여 계정 수준 금지 목록 재정의을(를) 참조하세요.

Amazon SES 전역 금지 목록

Amazon SES는 SES에 의해 백그라운드에서 작동되고 관리되는 내부 전역 금지 목록을 유지 관리합니다. SES 고객이 하드 바운스가 발생하는 이메일을 보내면 SES는 하드 바운스를 발생시킨 이메일 주소를 전역 금지 목록에 추가합니다. 전역 금지 목록은 모든 SES 고객에게 적용된다는 의미에서 전역적입니다. 즉, 다른 고객이 전역 금지 목록에 있는 주소로 이메일을 보내려고 시도하면 SES는 메시지를 수락하지만 이메일 주소가 표시되지 않으므로 메시지를 보내지 않습니다.

전역 금지 목록 이메일 주소 제거 요청 기능은 더 이상 고객에게 제공되지 않으며, 고객은 더 이상 이 기능과 상호 작용하여 금지를 관리할 수 없습니다. 이제 Amazon SES는 이 기능을 대체하기 위해, 사용 가능한 계정 수준 금지 목록 및 구성 세트 수준 금지 재정의를 만들어 고유한 계정에 대한 이메일 금지

를 처리하는 방법과 관련하여 사용자 지정된 제어를 제공함으로써, 금지를 관리할 수 있는 새로운 방법을 제공합니다. 자세한 내용은 Amazon SES 계정 수준 금지 목록 사용 및 구성 세트 수준 금지를 사용하여 계정 수준 금지 목록 재정의 섹션을 참조하세요.

> **Important**
>
> Amazon SES 콘솔에는 전역 금지 목록 이메일 주소 제거 요청 양식이 더 이상 없습니다. 계정 수준 금지 목록으로 대체되었기 때문입니다. 계정 수준 금지 목록을 사용하는 방법에 대한 자세한 내용은 Amazon SES 계정 수준 금지 목록 사용 섹션을 참조하세요.

전역 금지 목록 고려 사항

전역 금지 목록과 관련된 주요 요인:

- 전역 금지 목록은 SES에 의해 백그라운드에서 작동되고 관리되며 직접 상호 작용할 수 없으나, 고유한 계정 수준 금지 목록을 사용하여 이를 재정의할 수 있습니다.
- 전역 금지 목록은 모든 SES 계정에 기본적으로 활성화되어 있습니다. 전달을 비활성화할 수 없습니다.
- SES는 전역 금지 목록을 모든 고객에게 적용하므로, 전역 금지 목록을 쿼리하거나 주소를 전역 금지 목록에 수동으로 추가할 수 없습니다.
- 이메일 주소가 하드 바운스를 생성하면 SES가 단기 전역 금지 목록에 해당 주소를 추가합니다. 이 기간이 경과하면 SES가 목록에서 주소를 제거합니다. 주소에서 다른 하드 바운스가 생성되면 SES가 해당 주소를 장기 전역 금지 목록에 다시 추가하고 해당 기간이 끝날 때 제거합니다. 주소가 하드 바운스를 생성할 때마다 주소가 전역 금지 목록에 남아 있는 시간이 늘어납니다. 주소는 최대 14일까지 전역 금지 목록에 남아 있을 수 있습니다.
- 전역 금지 목록에 있는 주소로 메시지를 보내려고 시도하면 SES에서 메시지를 수락하지만 보내지는 않습니다. SES는 `Permanent`의 값 `bounceType`, `Suppressed`의 값 `bounceSubType`으로 반송 메일 알림을 생성합니다. 이러한 유형의 반송 알림을 수신하는 것이 특정 주소가 전역 금지 목록에 포함되어 있는지 알 수 있는 유일한 방법입니다. 전역 금지 목록을 쿼리할 수 없습니다.
- SES는 전역 금지 목록에 있는 주소로 보내는 메시지를 계정의 반송률과 일일 전송 할당량으로 계산합니다.
- 하드 바운스가 발생하는 모든 이메일 주소와 마찬가지로 해당 주소가 유효함을 확신하기 전에는 금지 목록 반송을 발생시키는 주소를 메일 그룹에서 제거해야 합니다.
- 금지 목록 반송 메일은 사용자 계정의 반송 메일 발생률에 포함됩니다. 반송률이 너무 높으면 계정이 검토 대상이 되거나 계정의 이메일 전송 기능이 일시 중지될 수 있습니다.

> **Note**
>
> SES 금지 목록과 계층 구조가 어떻게 상호 연관되어 있는지 이해하는 것이 중요합니다. 금지 목록 및 금지 재정의 메커니즘 개요를 참조하세요.

Amazon SES 계정 수준 금지 목록 사용

Amazon SES 계정 수준 금지 목록이 도입되어 고객이 자체 금지 목록을 생성하고 제어하며 평판을 관리할 수 있으므로, 계정 수준 금지 목록이 계정에만 적용됩니다. SES 콘솔의 계정 수준 금지 목록 인터페이스는 주소를 추가 또는 제거하기 위한 일괄 작업을 비롯하여 계정 수준 금지 목록의 주소를 손쉽게 관리할 수 있는 방법을 제공합니다.

SES 계정 수준 금지 목록은 현재 AWS 리전의 AWS 계정 에 적용됩니다. SES API v2 또는 콘솔을 사용하여 계정 수준 금지 목록에서 주소를 개별적으로 또는 일괄적으로 추가하거나 제거할 수 있습니다.

> **Note**
>
> 주소를 일괄 추가하거나 제거하려면 프로덕션 액세스 권한이 있어야 합니다. 샌드박스에 대한 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하세요.

> **Important**
>
> SES 계정 금지 목록은 대/소문자를 구분합니다. 이메일 주소가 금지 목록인 User@Example.com에 추가되면 수신된 그대로 저장되어 원래 사례를 보존합니다. 이메일 전송 기능은 예를 들어 User@Example.com은과 동일하지만, 금지 목록 관리를 위한 user@example.comAPI 호출에는 정확한 대/소문자 일치가 필요합니다. 따라서 금지 목록 관리와 관련된 APIs를 사용할 때는 금지 목록에 표시된 이메일 주소의 정확한 대/소문자를 사용해야 합니다.

Amazon SES 계정 수준 금지 목록 고려 사항

계정 수준 금지 목록을 사용하는 경우 다음 요소를 고려해야 합니다.

- 2019년 11월 25일 이후에 Amazon SES를 사용하기 시작한 경우 계정은 반송 및 수신 거부 모두에 기본적으로 계정 수준 금지 목록을 사용합니다. 이 날짜 이전에 SES를 사용하기 시작한 경우 SES API의 `PutAccountSuppressionAttributes` 작업을 사용하여 이 기능을 사용 설정해야 합니다.
- 계정 수준 금지 설정에 선택한 것과 금지 이유가 일치하는 계정 수준 금지 목록에 있는 주소로 메시지를 보내려고 하면 SES는 메시지를 수락하지만 보내지는 않습니다. 그러나 일치하지 않는 경우 SES가 이를 전송합니다. 명확한 이해를 돕기 위해 다음과 같은 예를 들 수 있습니다.
  - 금지 이유가 반송만 해당인 계정 수준 금지 설정을 지정한 경우, SES는 금지 이유가 반송인 계정 수준 금지 목록에 있는 주소로는 전송을 시도하지 않습니다. 그러나 SES는 금지 이유가 수신 거부인 계정 수준 금지 목록의 주소에 대한 전송을 시도합니다(이 경우 일치하지 않기 때문).
  - 금지 이유가 반송 및 수신 거부인 계정 수준 금지 설정을 지정한 경우, SES는 금지 이유가 반송 또는 수신 거부인 계정 수준 금지 목록에 있는 주소로는 전송을 시도하지 않습니다.
- SES는 계정 수준 금지 목록에 있는 주소로 보내는 메시지를 계정의 AWS/SES 네임스페이스에 있는 Reputation.BounceRate 또한 Reputation.ComplaintRate 지표에 포함하지 않습니다. 이러한 메시지는 AWS/SES 네임스페이스의 반송 메일 또는 수신 거부 지표에 따라 계산됩니다.
- 주소가 전역 금지 목록에 있지만 계정 수준 금지 목록에는 없는 경우(즉, 해당 주소로 전송하려는 경우) 사용자가 해당 주소로 메시지를 보내면 SES는 여전히 전송을 시도합니다. 하지만 반송되는 경우에는 계정의 반송 메일 발생률과 일일 전송 할당량에 반영됩니다.
- SES는 계정 수준 금지 목록에 있는 주소로 보내는 메시지를 일일 전송 할당량으로 계산합니다.
- 계정 수준 금지 목록에 있는 이메일 주소는 제거할 때까지 남아 있습니다.
- 계정의 이메일 전송 기능이 일시 중지되면 SES는 90일 후에 계정 수준 금지 목록에 있는 주소를 자동으로 삭제합니다. 이 90일 기간이 끝나기 전에 계정의 이메일 전송 기능이 복원되면 목록에 있는 주소가 삭제되지 않습니다.
- Gmail은 SES에 수신 거부 데이터를 제공하지 않습니다. 수신자가 Gmail 웹 클라이언트의 스팸(Spam) 버튼을 사용하여 사용자로부터 스팸으로 수신한 메시지를 보고하는 경우 계정 수준 금지 목록에 추가되지 않습니다.
- 계정이 SES 샌드박스에 있는 경우, 계정 수준 금지 목록을 사용 설정할 수 있습니다. 그러나 계정이 샌드박스에서 제거될 때까지 PutSuppressedDestination 또는 CreateImportJob 작업을 사용할 수 없습니다. 샌드박스에 대한 자세한 내용은 프로덕션 액세스 요청(Amazon SES 샌드박스에서 이동) 단원을 참조하세요.
- 계정 수준 금지 목록에는 하드 바운스만 추가됩니다. 소프트 바운스와 하드 바운스의 차이에 대한 자세한 내용은 the section called “Amazon SES가 이메일을 전송한 후” 섹션을 참조하세요.
- 계정 수준 금지 목록을 사용하는 경우 SES는 하드 바운스가 발생하는 주소도 전역 금지 목록에 추가합니다.

> **Note**
> 다음 단원의 절차에서는를 이미 설치했다고 가정합니다 AWS CLI. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

Amazon SES 계정 수준 금지 목록 사용 설정

Amazon SES API v2에서 PutAccountSuppressionAttributes 작업을 사용하여 계정 수준 금지 목록을 활성화하고 설정할 수 있습니다. AWS CLI을(를) 사용하여 이 설정을 쉽고 빠르게 구성할 수 있습니다. AWS CLI설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

를 사용하여 계정 수준 금지 목록을 구성하려면 AWS CLI

- 명령줄에 다음 명령을 입력합니다.

  Linux, macOS, or Unix

  ```
  aws sesv2 put-account-suppression-attributes \
  --suppressed-reasons BOUNCE COMPLAINT
  ```

  Windows

  ```
  aws sesv2 put-account-suppression-attributes `
  --suppressed-reasons BOUNCE COMPLAINT
  ```

  계정 수준 금지 목록을 활성화하려면 `suppressed-reasons` 파라미터에 대한 이유를 하나 이상 지정해야 합니다. 앞의 예와 같이 `BOUNCE` 또는 `COMPLAINT` 중 하나를 지정하거나 둘 모두 지정할 수 있습니다.

SES 콘솔을 사용하여 계정 수준 금지 목록을 구성하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성(Configuration)에서 금지 목록(Suppression list)을 선택합니다.
3. 계정 수준 설정(Account-level settings) 창에서 편집(Edit)을 선택합니다.

4. 금지 목록(Suppression list)에서 사용(Enabled) 상자를 선택합니다.
5. 금지 이유(Suppression reasons)에서 수신자 이메일 주소가 계정 수준 금지 목록에 자동으로 추가되어야 하는 이유 중 하나를 선택합니다.
6. 변경 사항 저장을 선택합니다.

구성 세트에 대한 Amazon SES 계정 수준 금지 목록 사용 설정

특정 구성 세트에만 적용되도록 Amazon SES 계정 수준 금지를 구성할 수도 있습니다. 이렇게 하면, 반송 또는 수신 거부 이벤트가 발생한 이메일을 보냈을 때 구성 세트를 지정한 경우에만 주소가 금지 목록에 추가됩니다.

를 사용하여 구성 세트에 대한 계정 수준 금지 목록을 구성하려면 AWS CLI

- 명령줄에 다음 명령을 입력합니다.

  Linux, macOS, or Unix

  ```
  aws sesv2 put-configuration-set-suppression-options \
  --configuration-set-name configSet \
  --suppressed-reasons BOUNCE COMPLAINT
  ```

  Windows

  ```
  aws sesv2 put-configuration-set-suppression-options `
  --configuration-set-name configSet `
  --suppressed-reasons BOUNCE COMPLAINT
  ```

  앞의 예에서 *configSet*을 계정 수준 금지 목록을 사용해야 하는 구성 세트의 이름으로 바꿉니다.

SES 콘솔을 사용하여 구성 세트에 대한 계정 수준 금지 목록을 구성하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Configuration sets(구성 세트)를 선택합니다.
3. 구성 세트(Configuration sets)에서 사용자 지정 금지를 사용하여 구성하려는 구성 세트의 이름을 선택합니다.

4. 금지 목록 옵션(Suppression list options) 창에서 편집(Edit)을 선택합니다.
5. 금지 목록 옵션(Suppression list options) 섹션은 이 구성 세트를 사용하여 계정 수준 금지를 재정의하는 옵션부터 사용자 지정 금지를 정의하기 위한 결정 세트를 제공합니다. 구성 세트 수준 금지 로직 맵을 사용하면 재정의 조합의 효과를 이해하는 데 도움이 됩니다. 이러한 다중 계층 재지정 선택을 결합하여 세 가지 다른 금지 수준을 구현할 수 있습니다.

   a. 계정 수준 금지 사용: 계정 수준 금지를 재정의하지 말고 구성 세트 수준 금지를 구현하지 마세요. 기본적으로, 이 구성 세트를 사용하여 보낸 이메일은 단순히 계정 수준 금지를 사용합니다. 방법:

      - 금지 목록 설정(Suppression list settings)에서 계정 수준 설정 재정의(Override account level settings) 상자를 선택 취소합니다.

   b. 금지 사용 안 함: 구성 세트 수준 금지를 사용하지 않고 계정 수준 금지를 재정의합니다. 즉, 이 구성 세트를 사용하여 전송된 이메일은 계정 수준 금지를 사용하지 않습니다. 다시 말해 모든 금지가 취소됩니다. 방법:

      i. 금지 목록 설정(Suppression list settings)에서 계정 수준 설정 재정의(Override account level settings) 상자를 선택합니다.

      ii. 금지 목록(Suppression list)에서 사용(Enabled) 상자를 선택 취소합니다.

   c. 구성 세트 수준 금지 사용: 계정 수준 금지를 이 구성 세트에 정의된 사용자 지정 금지 목록 설정으로 재정의합니다. 즉, 이 구성 세트를 사용하여 보낸 이메일은 자체 금지 설정만 사용하고 계정 수준 금지 설정은 무시합니다. 방법:

      i. 금지 목록 설정(Suppression list settings)에서 계정 수준 설정 재정의(Override account level settings) 상자를 선택합니다.

      ii. 금지 목록(Suppression list)에서 사용(Enabled)을 선택합니다.

      iii. 이유 지정...(Specify the reason(s)...)에서 사용할 구성 세트에 대해 금지 이유 중 하나를 선택합니다.
6. 변경 사항 저장을 선택합니다.

Amazon SES 계정 수준 금지 목록에 개별 이메일 주소 추가

SES API v2에서 PutSuppressedDestination 작업을 사용하여 Amazon SES 계정 수준 금지 목록에 개별 주소를 추가할 수 있습니다. 계정 수준 금지 목록에 추가할 수 있는 주소 수에는 제한이 없습니다.

AWS CLI를 사용하여 계정 수준 금지 목록에 개별 주소를 추가하려면

- 명령줄에 다음 명령을 입력합니다.

  Linux, macOS, or Unix

  ```
  aws sesv2 put-suppressed-destination \
  --email-address recipient@example.com \
  --reason BOUNCE
  ```

  Windows

  ```
  aws sesv2 put-suppressed-destination `
  --email-address recipient@example.com `
  --reason BOUNCE
  ```

  앞의 예에서 *recipient@example.com*을 계정 수준 금지 목록에 추가할 이메일 주소로 바꾸고 *BOUNCE*를 금지 목록에 주소를 추가하는 이유로 바꿉니다(허용되는 값: `BOUNCE` 및 `COMPLAINT`).

SES 콘솔을 사용하여 계정 수준 금지 목록에 개별 주소를 추가하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성(Configuration)에서 금지 목록(Suppression list)을 선택합니다.
3. 금지 목록(Suppression list) 창에서 이메일 주소 추가(Add email address)를 선택합니다.
4. 이메일 주소(Email address)에 이메일 주소를 입력하고 금지 이유(Suppression reason)에서 이유를 선택합니다. 주소를 더 입력해야 하는 경우 다른 주소 입력(Enter another address)을 선택하고 각 추가 항목에 대해 반복합니다.
5. 주소 입력이 완료되면 항목이 정확한지 검토합니다. 제출에 포함해서는 안 된다고 판단한 항목이 있는 경우 해당 항목의 제거(Remove) 버튼을 선택합니다.
6. 변경 사항 저장(Save changes)을 선택하여 입력한 이메일 주소를 계정 수준 금지 목록에 추가합니다.

Amazon SES 계정 수준 금지 목록에 이메일 주소 일괄 추가

Amazon SES API v2에서 CreateImportJob 작업을 사용하여 Amazon S3 객체에 연락처 목록을 먼저 업로드한 다음 일괄적으로 주소를 추가할 수 있습니다.

> **Note**
>
> - 계정 수준 금지 목록에 추가할 수 있는 주소 수에는 제한이 없지만 API 호출당 Amazon S3 객체에는 100,000개의 주소 일괄 추가 제한이 있습니다.
> - 동시 가져오기 작업은 20개 이상 실행할 수 없습니다.
> - 데이터 소스가 S3 버킷인 경우 데이터를 가져오는 대상 리전과 동일한 리전에 있어야 합니다.

계정 수준 금지 목록에 이메일 주소를 일괄적으로 추가하려면 다음 단계를 완료하십시오.

- CSV 또는 JSON 형식으로 Amazon S3 객체에 주소 목록을 업로드합니다.

  주소 추가를 위한 CSV 형식 예:

  ```
  recipient1@example.com,BOUNCE

  recipient2@example.com,COMPLAINT
  ```

  줄바꿈으로 구분된 JSON 파일만 지원됩니다. 이 형식에서 각 줄은 개별 주소 정의가 포함된 완전한 JSON 객체입니다.

  주소 추가를 위한 JSON 형식 예:

  ```
  {"emailAddress":"recipient1@example.com","reason":"BOUNCE"}

  {"emailAddress":"recipient2@example.com","reason":"COMPLAINT"}
  ```

  위의 예에서 *recipient1@example.com* 및 *recipient2@example.com*을 계정 수준 금지 목록에 추가할 이메일 주소로 바꿉니다. 금지 목록에 주소를 추가하는 허용 가능한 이유는 *BOUNCE* 및 *COMPLAINT*입니다.
- SES에 AWS KMS 키 사용 권한을 부여합니다.

Amazon S3 객체가 AWS KMS 키로 암호화된 경우 Amazon SES에 AWS KMS 키를 사용할 수 있는 권한을 부여해야 합니다. SES는 기본 KMS 키가 아닌 고객 관리형 키에서만 권한을 얻을 수 있습니다. 고객 관리형 키를 사용하려면 키의 정책에 설명을 추가하여 SES에 사용 권한을 부여해야 합니다.

다음 정책 설명을 키 정책에 붙여넣어 SES가 고객 관리형 키를 사용할 수 있도록 허가하세요.

```
{
   "Sid": "AllowSESToDecrypt",
   "Effect": "Allow",
   "Principal": {
       "Service":"ses.amazonaws.com"
   },
   "Action": [
       "kms:Decrypt",
   ],
   "Resource": "*"
}
```

- CreateImportJob 작업을 SES API v2에서 사용합니다.

> **Note**
> 다음 예제에서는 AWS CLI를 이미 설치했다고 가정합니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

명령줄에 다음 명령을 입력합니다. *s3bucket*을 Amazon S3 버킷의 이름으로 바꾸고 *s3object*를 Amazon S3 객체 이름으로 바꿉니다.

```
aws sesv2 create-import-job --import-destination
 SuppressionListDestination={SuppressionListImportAction=PUT} --import-data-source
 S3Url=s3://s3bucket/s3object,DataFormat=CSV
```

SES 콘솔을 사용하여 계정 수준 금지 목록에 이메일 주소 일괄 추가

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성(Configuration)에서 금지 목록(Suppression list)을 선택합니다.

3. 금지 목록(Suppression list) 테이블에서 대량 작업(Bulk actions) 버튼을 확장하고 이메일 주소를 대량으로 추가(Add email addresses in bulk)를 선택합니다.
4. 대량 작업 사양(Bulk action specifications)에서 (a) .S3 버킷에서 파일 선택(Choose file from S3 bucket) 또는 (b) 파일에서 가져오기(Import from file)를 선택합니다. 각 가져오기 방법에 대한 절차가 제공됩니다.

   a. S3 버킷에서 파일 선택(Choose file from S3 bucket) - 소스 파일이 이미 Amazon S3 버킷에 저장되어 있는 경우:

      i. 사용할 Amazon S3 버킷의 URI를 아는 경우 Amazon S3 URI 필드에 해당 URI를 입력하고 그렇지 않은 경우 S3 찾아보기(Browse S3)를 선택합니다.

         A. 버킷(Buckets)에서 S3 버킷의 이름을 선택합니다.

         B. 객체(Objects)에서 파일의 이름을 선택한 다음 선택(Choose)을 선택합니다. 대량 작업 사양(Bulk action specifications)으로 다시 돌아오게 됩니다.

         C. (선택 사항) Amazon S3 콘솔로 이동하여 S3 객체에 대한 세부 정보를 보려면 보기(View)를 선택합니다.

      ii. 파일 형식(File format)에서 Amazon S3 버킷에서 가져오도록 선택한 파일의 형식을 선택합니다.

      iii. 이메일 주소 추가(Add email addresses)를 선택하여 파일에서 주소 가져오기를 시작합니다. 대량 작업(Bulk actions) 탭 아래에 테이블이 표시됩니다.

   b. 파일에서 가져오기(Import from file) - 새 Amazon S3 버킷이나 기존 Amazon S3 버킷에 업로드할 로컬 소스 파일이 있는 경우:

      i. 소스 파일 가져오기(Import source file)에서 파일 선택(Choose file)을 선택합니다.

      ii. 파일 브라우저에서 JSON 또는 CSV 파일을 선택하고 열기(Open)를 선택합니다. 파일 이름, 크기 및 날짜가 파일 선택(Choose file) 버튼 아래에 표시됩니다.

      iii. Amazon S3 버킷(Amazon S3 bucket)을 확장하고 S3 버킷을 선택합니다.

         - 파일을 새 버킷에 업로드하려면 S3 버킷 생성(Create S3 bucket)을 선택하고 버킷 이름(Bucket name) 필드에 이름을 입력한 다음 버킷 생성(Create bucket)을 선택합니다.

      iv. 이메일 주소 추가(Add email addresses)를 선택하여 파일에서 주소 가져오기를 시작합니다. 대량 작업(Bulk actions) 탭 아래에 테이블이 표시됩니다.

5. 사용한 가져오기 방법에 관계없이 대량 작업(Bulk actions)에 작업 ID가 가져오기 유형, 상태 및 날짜와 함께 나열됩니다. 작업 세부 정보를 보려면 작업 ID를 선택합니다.

6. 금지 목록(Suppression list) 탭을 선택하면 성공적으로 가져온 모든 이메일 주소가 해당 금지 이유 및 추가된 날짜와 함께 표시됩니다. 다음 옵션을 사용할 수 있습니다.

   a. 이메일 주소를 선택하거나 해당 확인란을 선택하고 보고서 보기(View report)를 선택하여 세부 정보를 봅니다. (반송 메일 또는 수신 거부 때문에 금지 목록에 자동으로 추가된 주소인 경우, 트리거링 이벤트를 생성한 이메일 메시지에 대한 세부 정보를 포함하여 추가 사유가 된 피드백 이벤트에 대한 정보가 표시됩니다.)

   b. 계정 금지 목록에서 제거할 하나 이상의 이메일 주소에 해당하는 확인란을 선택하고 제거(Remove)를 선택합니다.

Amazon SES 계정 수준 금지 목록에 있는 주소 목록 보기

SES API v2에서 ListSuppressedDestinations 작업을 사용하여 계정에 대한 계정 수준 금지 목록에 있는 모든 이메일 주소 목록을 볼 수 있습니다.

계정 수준 금지 목록에 있는 모든 이메일 주소 목록을 보려면

- 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 list-suppressed-destinations
```

위의 명령은 계정의 계정 수준 금지 목록에 있는 모든 이메일 주소를 반환합니다. 출력은 다음 예제와 유사합니다.

```
{
    "SuppressedDestinationSummaries": [
        {
            "EmailAddress": "recipient2@example.com",
            "Reason": "COMPLAINT",
            "LastUpdateTime": "2020-04-10T21:03:05Z"
        },
        {
            "EmailAddress": "recipient0@example.com",
            "Reason": "COMPLAINT",
            "LastUpdateTime": "2020-04-10T21:04:26Z"
        },
        {
            "EmailAddress": "recipient1@example.com",
            "Reason": "BOUNCE",
```

```
            "LastUpdateTime": "2020-04-10T22:07:59Z"
        }
    ]
}
```

- 참고 - 출력에 문자열 값이 있는 'NextToken' 필드가 포함되어 있으면 계정의 금지 목록에 추가 이메일 주소가 있음을 나타냅니다. 금지 주소를 추가로 보려면 `ListSuppressedDestinations`로 다른 요청을 실행하고 다음과 같이 `--next-token` 파라미터에 반환된 문자열 값을 전달합니다.

  ```
  aws sesv2 list-suppressed-destinations --next-token string
  ```

  위의 명령에서 *###*을 반환된 NextToken 값으로 바꿉니다.

  자세한 내용은 계정 수준 금지 목록에서 1,000개가 넘는 이메일 주소를 나열하는 방법을 참조하세요.

`StartDate` 옵션을 사용하면 특정 날짜 이후 목록에 추가된 이메일 주소만 표시할 수 있습니다.

특정 날짜 이후 계정 수준 금지 목록에 추가된 주소 목록을 보려면

- 명령줄에 다음 명령을 입력합니다.

  ```
  aws sesv2 list-suppressed-destinations --start-date 1604394130
  ```

  위의 명령에서 *1604394130*을 시작 날짜의 Unix 타임스탬프로 바꿉니다.

`EndDate` 옵션을 사용하면 특정 날짜 이전 목록에 추가된 이메일 주소만 표시할 수도 있습니다.

특정 날짜 이전 계정 수준 금지 목록에 추가된 주소 목록을 보려면

- 명령줄에 다음 명령을 입력합니다.

  ```
  aws sesv2 list-suppressed-destinations --end-date 1611126000
  ```

  위의 명령에서 *1611126000*을 종료 날짜의 Unix 타임스탬프로 바꿉니다.

Linux, macOS 또는 Unix 명령줄에서 기본 제공 `grep` 유틸리티를 사용하여 특정 주소 또는 도메인을 검색할 수도 있습니다.

계정 수준 금지 목록에서 특정 주소를 검색하려면

- 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 list-suppressed-destinations | grep -A2 'example.com'
```

위의 명령에서 *example.com*을 검색할 텍스트 문자열(예: 주소 또는 도메인)로 바꿉니다.

SES 콘솔을 사용하여 계정 수준 금지 목록에 있는 모든 이메일 주소 목록을 보려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성(Configuration)에서 금지 목록(Suppression list)을 선택합니다.
3. 금지 목록(Suppression list) 창에 계정 수준 금지 목록의 모든 이메일 주소가 해당 금지 이유 및 추가된 날짜와 함께 표시됩니다. 다음 옵션을 사용할 수 있습니다.
   a. 이메일 주소를 선택하거나 해당 확인란을 선택하고 보고서 보기(View report)를 선택하여 세부 정보를 봅니다. (반송 메일 또는 수신 거부 때문에 금지 목록에 자동으로 추가된 주소인 경우, 트리거링 이벤트를 생성한 이메일 메시지에 대한 세부 정보를 포함하여 추가 사유가 된 피드백 이벤트에 대한 정보가 표시됩니다.)
   b. 톱니바퀴 아이콘을 선택하여 금지 목록 테이블을 사용자 지정할 수 있습니다. 페이지 크기, 줄 바꿈 및 표시할 열을 사용자 지정할 수 있는 모달이 표시됩니다. 선택을 마친 후 확인(Confirm)을 선택합니다. 금지 목록 테이블에는 보기 선택 항목이 반영됩니다.

Amazon SES 계정 수준 금지 목록에서 개별 이메일 주소 제거

주소가 계정의 금지 목록에 있지만 주소가 목록에 없어야 한다는 것을 알고 있는 경우 SES API v2에서 DeleteSuppressedDestination 작업을 사용하여 제거할 수 있습니다.

AWS CLI를 사용하여 계정 수준 금지 목록에서 개별 주소를 제거하려면

- 명령줄에 다음 명령을 입력합니다.

  Linux, macOS, or Unix

```
aws sesv2 delete-suppressed-destination \
--email-address recipient@example.com
```

Windows

```
aws sesv2 delete-suppressed-destination `
--email-address recipient@example.com
```

위의 예제에서 *recipient@example.com*을 계정 수준 금지 목록에서 제거할 이메일 주소로 바꿉니다.

SES 콘솔을 사용하여 계정 수준 금지 목록에서 개별 주소를 제거하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성(Configuration)에서 금지 목록(Suppression list)을 선택합니다.
3. (a) 테이블 선택 또는 (b) 입력 항목 중 하나를 사용하여 개별 이메일 주소를 제거합니다.
   a. 테이블 선택: 금지 목록(Suppression list) 테이블에서 하나 이상의 이메일 주소에 해당하는 확인란을 선택하고 제거(Remove)를 선택합니다.
   b. 필드 입력:
      i. 금지 목록(Suppression list) 테이블에서 이메일 주소 제거(Remove email address)를 선택합니다.
      ii. 이메일 주소(Email address) 필드에 이메일 주소를 입력합니다. 주소를 더 입력해야 하는 경우 다른 주소 입력(Enter another address)을 선택하고 각 추가 항목에 대해 반복합니다.
      iii. 주소 입력이 완료되면 항목이 정확한지 검토합니다. 제출에 포함해서는 안 된다고 판단한 항목이 있는 경우 해당 항목의 제거(Remove) 버튼을 선택합니다.
      iv. 변경 사항 저장(Save changes)을 선택하여 입력한 이메일 주소를 계정 수준 금지 목록에서 제거합니다.

Amazon SES 계정 수준 금지 목록에서 이메일 주소 일괄 제거

SES API v2에서 CreateImportJob 작업을 사용하여 Amazon S3 객체에 연락처 목록을 먼저 업로드한 다음 일괄적으로 주소를 제거할 수 있습니다.

> **Note**
>
> - 계정 수준 금지 목록에서 제거할 수 있는 주소 수에는 제한이 없지만 API 호출당 Amazon S3 객체에는 10,000개의 주소 일괄 제거 제한이 있습니다.
> - 데이터 소스가 S3 버킷인 경우 데이터를 가져오는 대상 리전과 동일한 리전에 있어야 합니다.

계정 수준 금지 목록에서 이메일 주소를 일괄적으로 제거하려면 다음 단계를 완료하세요.

- CSV 또는 JSON 형식으로 Amazon S3 객체에 주소 목록을 업로드합니다.

  주소를 제거하기 위한 CSV 형식 예:

  recipient3@example.com

  줄바꿈으로 구분된 JSON 파일만 지원됩니다. 이 형식에서 각 줄은 개별 주소 정의가 포함된 완전한 JSON 객체입니다.

  주소 추가를 위한 JSON 형식 예:

  {"emailAddress":"*recipient3@example.com*"}

  위의 예제에서 *recipient3@example.com*을 계정 수준 금지 목록에서 제거할 이메일 주소로 바꿉니다.

- SES에 S3 객체를 읽을 수 있는 권한을 부여합니다.

  Amazon S3 버킷에 적용하면 다음 정책은 SES에서 해당 버킷을 읽을 수 있는 권한을 부여합니다. Amazon S3의 버킷 정책에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 정책 및 사용자 정책 사용을 참조하십시오.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSESGet",
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "ses.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::BUCKET-NAME/OBJECT-NAME",
            "Condition": {
                "StringEquals": {
                    "aws:Referer": "AWSACCOUNTID"
                }
            }
        }
    ]
}
```

- SES에 AWS KMS 키를 사용할 수 있는 권한을 부여합니다.

  Amazon S3 객체가 AWS KMS 키로 암호화된 경우 Amazon SES에 AWS KMS 키를 사용할 수 있는 권한을 부여해야 합니다. SES는 기본 KMS 키가 아닌 고객 관리형 키에서만 권한을 얻을 수 있습니다. 고객 관리형 키를 사용하려면 키의 정책에 설명을 추가하여 SES에 사용 권한을 부여해야 합니다.

  다음 정책 설명을 키 정책에 붙여넣어 SES가 고객 관리형 키를 사용할 수 있도록 허가하세요.

```
{
   "Sid": "AllowSESToDecrypt",
   "Effect": "Allow",
   "Principal": {
       "Service":"ses.amazonaws.com"
   },
   "Action": [
       "kms:Decrypt",
   ],
   "Resource": "*"
}
```

- CreateImportJob 작업을 SES API v2에서 사용합니다.

> **Note**
>
> 다음 예제에서는 AWS CLI를 이미 설치했다고 가정합니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

명령줄에 다음 명령을 입력합니다. *s3bucket*을 Amazon S3 버킷의 이름으로 바꾸고 *s3object*를 Amazon S3 객체 이름으로 바꿉니다.

```
aws sesv2 create-import-job --import-destination
 SuppressionListDestination={SuppressionListImportAction=DELETE} --import-data-source
 S3Url="s3://s3bucket/s3object",DataFormat=CSV
```

SES 콘솔을 사용하여 계정 수준 금지 목록에서 이메일 주소 일괄 제거

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성(Configuration)에서 금지 목록(Suppression list)을 선택합니다.
3. 금지 목록(Suppression list) 테이블에서 대량 작업(Bulk actions) 버튼을 확장하고 이메일 주소를 대량으로 제거(Remove email addresses in bulk)를 선택합니다.
4. 일괄 작업 사양(Bulk action specifications)에서 (a) S3 버킷에서 파일 선택(Choose file from S3 bucket) 또는 (b) 파일에서 가져오기(Import from file)를 선택합니다. 각 가져오기 방법에 대한 절차가 제공됩니다.

   a. S3 버킷에서 파일 선택(Choose file from S3 bucket) - 소스 파일이 이미 Amazon S3 버킷에 저장되어 있는 경우:

      i. 사용할 Amazon S3 버킷의 URI를 아는 경우 Amazon S3 URI 필드에 해당 URI를 입력하고 그렇지 않은 경우 S3 찾아보기(Browse S3)를 선택합니다.

         A. 버킷(Buckets)에서 S3 버킷의 이름을 선택합니다.

         B. 객체(Objects)에서 파일의 이름을 선택한 다음 선택(Choose)을 선택합니다. 대량 작업 사양(Bulk action specifications)으로 다시 돌아오게 됩니다.

         C. (선택 사항) Amazon S3 콘솔로 이동하여 S3 객체에 대한 세부 정보를 보려면 보기(View)를 선택합니다.

      ii. 파일 형식(File format)에서 Amazon S3 버킷에서 가져오도록 선택한 파일의 형식을 선택합니다.

      iii. 이메일 주소 제거(Remove email addresses)를 선택하여 파일에서 주소 가져오기를 시작합니다. 대량 작업(Bulk actions) 탭 아래에 테이블이 표시됩니다.

   b. 파일에서 가져오기(Import from file) - 새 Amazon S3 버킷이나 기존 Amazon S3 버킷에 업로드할 로컬 소스 파일이 있는 경우:

      i. 소스 파일 가져오기(Import source file)에서 파일 선택(Choose file)을 선택합니다.

ii. 파일 브라우저에서 JSON 또는 CSV 파일을 선택하고 열기(Open)를 선택합니다. 파일 이름, 크기 및 날짜가 파일 선택(Choose file) 버튼 아래에 표시됩니다.

iii. Amazon S3 버킷(Amazon S3 bucket)을 확장하고 S3 버킷을 선택합니다.

- 파일을 새 버킷에 업로드하려면 S3 버킷 생성(Create S3 bucket)을 선택하고 버킷 이름(Bucket name) 필드에 이름을 입력한 다음 버킷 생성(Create bucket)을 선택합니다.

iv. 이메일 주소 제거(Remove email addresses)를 선택하여 파일에서 주소 가져오기를 시작합니다. 대량 작업(Bulk actions) 탭 아래에 테이블이 표시됩니다.

5. 사용한 가져오기 방법에 관계없이 대량 작업(Bulk actions)에 작업 ID가 가져오기 유형, 상태 및 날짜와 함께 나열됩니다. 작업 세부 정보를 보려면 작업 ID를 선택합니다.
6. 금지 목록(Suppression list) 탭을 선택하면 금지 목록에서 제거된 성공적으로 가져온 모든 이메일 주소가 더 이상 표시되지 않습니다.

계정 가져오기 작업 목록 보기

Amazon SES API v2에서 ListImportJobs 작업을 사용하여 계정에 대한 계정 수준 금지 목록에 있는 모든 이메일 주소 목록을 볼 수 있습니다.

계정 가져오기 작업 목록 보기

- 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 list-import-jobs
```

위의 명령은 계정의 모든 가져오기 작업을 반환합니다. 출력은 다음 예제와 유사합니다.

```
{
    "ImportJobs": [
        {
            "CreatedTimestamp": 2020-07-31T06:06:55Z",
            "ImportDestination": {
                "SuppressionListDestination": {
                    "SuppressionListImportAction": "PUT"
                }
            },
            "JobStatus": "COMPLETED",
            "JobId": "755380d7-fbdb-4ed2-a9a3-06866220f5b5"
```

```
        },
        {
            "CreatedTimestamp": "2020-07-30T18:45:32Z",
            "ImportDestination": {
                "SuppressionListDestination": {
                    "SuppressionListImportAction": "DELETE"
                }
            },
            "JobStatus": "COMPLETED",
            "JobId": "076683bd-a7ee-4a40-9754-4ad1161ba8b6"
        },
        {
            "CreatedTimestamp": "2020-08-05T16:45:18Z",
            "ImportDestination": {
                "SuppressionListDestination": {
                    "SuppressionListImportAction": "PUT"
                }
            },
            "JobStatus": "COMPLETED",
            "JobId": "6e261869-bd30-4b33-b1f2-9e035a83a395"
        }
    ]
}
```

SES 콘솔을 사용하여 계정 가져오기 작업 목록 보기

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성(Configuration)에서 금지 목록(Suppression list)을 선택합니다.
3. 금지 목록(Suppression list) 창에서 대량 작업(Bulk actions) 탭을 선택합니다.
4. 모든 가져오기 작업이 대량 작업(Bulk actions) 테이블에 가져오기 유형, 상태 및 날짜와 함께 나열됩니다.
5. 작업 세부 정보를 보려면 작업 ID를 선택합니다. 그러면 다음 창이 표시됩니다.

    a. 대량 작업 상태(Bulk action status): 전체 작업 상태, 완료된 시간 및 날짜, 가져온 레코드 수 및 성공적으로 가져오지 못한 레코드의 수를 표시합니다.

    b. 대량 작업 세부 정보(Bulk action details): 작업 ID, 주소 추가 또는 제거에 사용되었는지 여부, 파일 형식이 JSON인지 아니면 CSV인지 여부, 대량 파일이 저장된 Amazon S3 버킷의 URI, 대량 작업이 생성된 시간 및 날짜를 표시합니다.

계정 가져오기 작업에 대한 정보 가져오기

Amazon SES API v2에서 GetImportJob 작업을 사용하여 계정 가져오기 작업에 대한 정보를 얻을 수 있습니다.

계정 가져오기 작업에 대한 정보를 얻으려면

- 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 get-import-job --job-id JobId
```

위의 명령은 계정의 가져오기 작업에 대한 정보를 반환합니다. 출력은 다음 예제와 유사합니다.

```
{
    "ImportDataSource": {
        "S3Url": "s3://bucket/object",
        "DataFormat": "CSV"
    },
    "ProcessedRecordsCount": 2,
    "FailureInfo": {
        "FailedRecordsS3Url": "s3presignedurl"
    },
    "JobStatus": "COMPLETED",
    "JobId": "jobid",
    "CreatedTimestamp": "2020-08-12T17:05:15Z",
    "FailedRecordsCount": 1,
    "ImportDestination": {
        "SuppressionListDestination": {
            "SuppressionListImportAction": "PUT"
        }
    },
    "CompletedTimestamp": "2020-08-12T17:06:42Z"
}
```

SES 콘솔을 사용하여 계정 가져오기 작업에 대한 정보 얻기

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성(Configuration)에서 금지 목록(Suppression list)을 선택합니다.
3. 금지 목록(Suppression list) 창에서 대량 작업(Bulk actions) 탭을 선택합니다.

4. 모든 가져오기 작업이 대량 작업(Bulk actions) 테이블에 가져오기 유형, 상태 및 날짜와 함께 나열됩니다.
5. 작업 세부 정보를 보려면 작업 ID를 선택합니다. 그러면 다음 창이 표시됩니다.
   a. 대량 작업 상태(Bulk action status): 전체 작업 상태, 완료된 시간 및 날짜, 가져온 레코드 수 및 성공적으로 가져오지 못한 레코드의 수를 표시합니다.
   b. 대량 작업 세부 정보(Bulk action details): 작업 ID, 주소 추가 또는 제거에 사용되었는지 여부, 파일 형식이 JSON인지 아니면 CSV인지 여부, 대량 파일이 저장된 Amazon S3 버킷의 URI, 대량 작업이 생성된 시간 및 날짜를 표시합니다.

Amazon SES 계정 수준 금지 목록 사용 중지

SES API v2에서 PutAccountSuppressionAttributes 작업을 사용하여 `suppressed-reasons` 속성에서 값을 제거함으로써 계정 수준 금지 목록을 효과적으로 사용 중지합니다.

를 사용하여 계정 수준 금지 목록을 비활성화하려면 AWS CLI

- 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 put-account-suppression-attributes --suppressed-reasons
```

SES 콘솔을 사용하여 계정 수준 금지 목록 사용 중지하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성(Configuration)에서 금지 목록(Suppression list)을 선택합니다.
3. 계정 수준 설정(Account-level settings) 창에서 편집(Edit)을 선택합니다.
4. 금지 목록(Suppression list)에서 사용(Enabled) 상자를 선택 취소합니다.
5. 변경 사항 저장을 선택합니다.

구성 세트 수준 금지를 사용하여 계정 수준 금지 목록 재정의

전체 계정에 대해 계정 수준 금지 목록이 설정되어 있지만 구성 세트 수준 금지 목록으로 재정의하여 구성 세트마다 개별적으로 금지 목록을 사용자 정의할 수 있습니다. 이렇게 세분화하면 자체 구성 세트에 할당한 여러 이메일 전송 그룹에 대해 사용자 지정 금지 설정을 사용할 수 있습니다. 예를 들어 이런

상황을 가정해 보겠습니다. 추가될 반송 메일 주소와 수신 거부 주소 모두에 대해 계정 수준 금지 목록이 구성되어 있습니다. 그러나 사용자의 구성 세트에 특정 인구 통계가 정의되어 있고, 사용자는 이 구성 세트에 수신 거부 주소만 추가되었으면 합니다. 그러기 위해서는 이 구성 세트를 사용하여 전송된 이메일에서 수신 거부에 대해서만(계정 수준 금지 목록에 설정된 반송 메일 및 수신 거부가 아님) 계정 수준 금지 목록에 이메일 주소가 추가되도록 하여 구성 세트의 금지 목록이 재정의하도록 하면 됩니다.

구성 세트 수준 금지 목록에는 금지를 전혀 사용하지 않는 것을 포함하여 계정 수준 금지를 재정의하는 다양한 수준이 있습니다. 다음 콘솔 절차에서 설정할 수 있는 이러한 다양한 수준의 금지를 쉽게 이해할 수 있도록, 다음 관계 맵에서는 다양한 수준의 재정의를 사용 또는 사용 중지하기 위해 선택할 수 있는 결정 세트를 모델링하여 조합에 따라 세 가지 수준의 금지를 구현하는 데 사용할 수 있습니다.

- 재정의 없음(기본값) - 구성 세트에서 계정 수준 금지 목록 설정을 사용합니다.
- 계정 수준 설정 재정의 - 계정 수준 금지 목록 설정이 무효화되며, 이 구성 세트를 사용하여 전송된 이메일은 금지 설정을 전혀 사용하지 않습니다.
- 구성 세트 수준 금지를 사용 설정한 계정 수준 설정 - 이 구성 세트를 사용하여 전송된 이메일은 사용자가 사용 설정한 금지 조건 (반송 메일, 수신 거부 또는 반송 메일 및 수신 거부)만 사용합니다. 계정 수준 금지 목록 설정과 상관없이 해당 설정을 재정의합니다.

> **Note**
>
> 구성 세트 수준에서 지정되지 않은 금지 조건의 경우 계정 수준 설정이 재정의되었으므로 금지 동작이 전역 금지 목록으로 돌아갑니다.

참고로 구성 세트 수준 금지 목록은 실제로 금지 목록이 아니라, 구성 세트에 정의된 사용자 지정 금지 설정으로 계정 수준 금지 목록을 재정의하는 메커니즘일 뿐입니다. 즉, 이 구성 세트를 사용하여 보낸 이메일은 자체 금지 설정만 사용하고 계정 수준 금지 설정은 무시합니다. 즉, 구성 세트 수준 금지 목록은 계정 수준 금지 목록에 추가할 이메일 주소를 결정하는 금지 이유를 변경(재정의)하여 계정 수준 금지 목록과 상호 작용합니다.

구성 세트 레벨 금지 사용

Amazon SES 새 콘솔을 사용하여 구성 설정 수준 금지를 활성화하려면:

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창의 구성 아래에서 Configuration sets(구성 세트)를 선택합니다.
3. 구성 세트(Configuration sets)에서 사용자 지정 금지를 사용하여 구성하려는 구성 세트의 이름을 선택합니다.
4. 금지 목록 옵션(Suppression list options) 창에서 편집(Edit)을 선택합니다.
5. 금지 목록 옵션(Suppression list options) 섹션은 이 구성 세트를 사용하여 계정 수준 금지를 재정의하는 옵션부터 사용자 지정 금지를 정의하기 위한 결정 세트를 제공합니다. 구성 세트 수준 금지 로직 맵을 사용하면 재정의 조합의 효과를 이해하는 데 도움이 됩니다. 이러한 다중 계층 재지정 선택을 결합하여 세 가지 다른 금지 수준을 구현할 수 있습니다.

   a. 계정 수준 금지 사용: 계정 수준 금지를 재정의하지 말고 구성 세트 수준 금지를 구현하지 마세요. 기본적으로, 이 구성 세트를 사용하여 보낸 이메일은 단순히 계정 수준 금지를 사용합니다. 방법:

      - 금지 목록 설정(Suppression list settings)에서 계정 수준 설정 재정의(Override account level settings) 상자를 선택 취소합니다.

   b. 금지 사용 안 함: 구성 세트 수준 금지를 사용하지 않고 계정 수준 금지를 재정의합니다. 즉, 이 구성 세트를 사용하여 전송된 이메일은 계정 수준 금지를 사용하지 않습니다. 다시 말해 모든 금지가 취소됩니다. 방법:

      i. 금지 목록 설정(Suppression list settings)에서 계정 수준 설정 재정의(Override account level settings) 상자를 선택합니다.

      ii. 금지 목록(Suppression list)에서 사용(Enabled) 상자를 선택 취소합니다.

   c. 구성 세트 수준 금지 사용: 계정 수준 금지를 이 구성 세트에 정의된 사용자 지정 금지 설정으로 재정의합니다. 즉, 이 구성 세트를 사용하여 보낸 이메일은 자체 금지 설정만 사용하고 계정 수준 금지 설정은 무시합니다. 방법:

      i. 금지 목록 설정(Suppression list settings)에서 계정 수준 설정 재정의(Override account level settings) 상자를 선택합니다.

      ii. 금지 목록(Suppression list)에서 사용(Enabled)을 선택합니다.

iii. 이유 지정...(Specify the reason(s)...)에서 사용할 구성 세트에 대해 금지 이유 중 하나를 선택합니다.

6. 변경 사항 저장을 선택합니다.

목록 관리 사용

Amazon SES는 목록 관리 기능을 제공합니다. 즉, 고객이 연락처 목록이라고 하는 자체 메일링 목록을 관리할 수 있습니다. contact list(연락처 목록)는 특정 주제를 구독한 모든 연락처를 저장할 수 있는 목록입니다. contact(연락처)는 이메일을 수신하는 최종 사용자입니다. topic(주제)는 목록 내의 관심 그룹, 테마 또는 레이블입니다. 목록은 복수의 주제를 가질 수 있습니다.

Amazon SES API v2에서 `ListContacts` 작업을 수행하면 특정 주제를 구독한 모든 연락처의 목록을 검색할 수 있으며, `SendEmail` 작업을 사용하여 이들 연락처에 메일을 보낼 수 있습니다.

구독 관리에 대한 자세한 내용은 구독 관리 사용 단원을 참조하십시오.

목록 관리 개요

목록 관리를 사용하는 경우 다음 요소를 고려해야 합니다.

- 목록을 만드는 동안 목록 주제를 지정할 수 있습니다.
- 당 하나의 연락처 목록만 허용됩니다 AWS 계정.
- 목록은 최대 20개의 주제를 사용할 수 있습니다.
- 목록에 새 주제 추가, 목록에서 연락처 추가 또는 삭제, 목록이나 주제에 대한 연락처 기본 설정 업데이트 등 기존 연락처 목록을 업데이트할 수 있습니다.
- 주제 표시 이름 또는 설명과 같은 주제 메타데이터를 업데이트할 수 있습니다.
- 연락처 목록의 연락처 목록, 주제를 구독한 연락처, 주제 구독이 취소된 연락처 및 목록의 모든 주제에서 구독 취소된 연락처를 가져올 수 있습니다.
- `CreateImportJob` API를 사용하여 기존 연락처 목록을 Amazon SES로 가져올 수 있습니다.
- Amazon SES는 연락처 목록에 있는 구독 취소된 연락처로 이메일을 전송한 경우 이메일을 반송합니다. 자세한 내용은 구독 관리 사용 단원을 참조하세요
- 각 연락처에는 해당 연락처에 대한 정보를 저장하는 데 사용할 수 있는 연결된 속성이 있을 수 있습니다.

목록 관리 구성

다음 작업을 사용하여 목록 관리 기능을 구성할 수 있습니다. 연락처 목록 및 연락처 작업의 전체 목록은 Amazon SES API v2 참조를 참조하십시오.

연락처 목록 만들기

Amazon SES API v2에서 `CreateContactList` 작업을 사용하여 연락처 목록을 생성할 수 있습니다. AWS CLI을(를) 사용하여 이 설정을 쉽고 빠르게 구성할 수 있습니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

를 사용하여 연락처 목록을 생성하려면 AWS CLI

- 명령줄에 다음 명령을 입력합니다.

```
aws sesv2 create-contact-list --cli-input-json file://CONTACT-LIST-JSON
```

위의 명령에서 *CONTACT-LIST-JSON*을 `CreateContactList` 요청에 대한 JSON 파일 경로로 바꿉니다.

요청에 대한 `CreateContactList` 입력 JSON 파일의 예는 다음과 같습니다.

```
{
    "ContactListName": "ExampleContactListName",
    "Description": "Creating a contact list example",
    "Topics": [
     {
         "TopicName": "Sports",
         "DisplayName": "Sports Newsletter",
         "Description": "Sign up for our free newsletter to receive updates on all
 sports.",
         "DefaultSubscriptionStatus": "OPT_OUT"
     },
     {
         "TopicName": "Cycling",
         "DisplayName": "Cycling newsletter",
         "Description": "Never miss a cycling update by subscribing to our
 newsletter.",
         "DefaultSubscriptionStatus": "OPT_IN"
     },
     {
         "TopicName": "NewProducts",
```

```
        "DisplayName": "New products",
        "Description": "Hear about new products by subscribing to this mailing
 list.",
        "DefaultSubscriptionStatus": "OPT_IN"
    },
    {
        "TopicName": "DailyUpdates",
        "DisplayName": "Daily updates",
        "Description": "Start your day with sport updates, Monday through
 Friday.",
        "DefaultSubscriptionStatus": "OPT_OUT"
    }
   ]
}
```

연락처 만들기

Amazon SES API v2에서 `CreateContact` 작업을 사용하여 연락처를 생성할 수 있습니다. AWS CLI 을(를) 사용하여 이 설정을 쉽고 빠르게 구성할 수 있습니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

를 사용하여 연락처를 생성하려면 AWS CLI

- 명령줄에 다음 명령을 입력합니다.

  ```
  aws sesv2 create-contact --cli-input-json file://CONTACT-JSON
  ```

  위의 명령에서 *CONTACT-JSON*을 `CreateContact` 요청에 대한 JSON 파일 경로로 바꿉니다.

  요청에 대한 `CreateContact` 입력 JSON 파일의 예는 다음과 같습니다.

  ```
  {
      "ContactListName": "ExampleContactListName",
      "EmailAddress": "example@amazon.com",
      "UnsubscribeAll": false,
      "TopicPreferences": [
          {
              "TopicName": "Sports",
              "SubscriptionStatus": "OPT_IN"
          }
      ],
  ```

```
    "AttributesData": "{\"Name\": \"John\", \"Location\": \"Seattle\"}"
}
```

위 예제에서 `UnsubscribeAll` 값이 `false`이면 해당 연락처가 모든 주제의 구독을 취소하지 않았음을 나타냅니다. 여기서 값이 `true`이면 연락처가 모든 주제의 구독을 취소했음을 의미합니다.

`TopicPreferences`에는 연락처의 주제 구독 상태에 대한 정보가 포함되어 있습니다. 앞의 예에서 연락처는 'Sports' 주제를 선택했으며 'Sports' 주제에 대한 모든 이메일을 받게 됩니다.

`AttributesData`는 연락처에 대한 모든 메타데이터를 넣을 수 있는 JSON 필드입니다. 유효한 JSON 객체여야 합니다.

연락처 목록으로 연락처 일괄 가져오기

Amazon SES API v2 또는 SES 콘솔에서 `CreateImportJob` 작업을 사용하여 Amazon S3 객체에 연락처를 먼저 업로드한 다음 일괄적으로 주소를 수동으로 추가할 수 있습니다. 자세한 내용은 계정 수준 금지 목록에 이메일 주소 일괄 추가 단원을 참조하세요.

연락처를 가져오기 전에 연락처 목록을 만들어야 합니다.

> **Note**
>
> ImportJob당 연락처 목록에 최대 1백만 개의 연락처를 추가할 수 있습니다.

연락처 목록에 연락처를 일괄적으로 추가하려면 다음 단계를 완료하세요.

- CSV 또는 JSON 형식으로 Amazon S3 객체에 주소를 업로드합니다.

  CSV 형식

  Amazon S3에 업로드되는 파일의 첫 번째 줄은 헤더 행이어야 합니다.

  `topicPreferences` 객체를 CSV 형식으로 병합해야 합니다. `topicPreferences`의 모든 주제에는 별도의 헤더 필드가 있습니다.

  연락처 목록에 연락처를 일괄 추가하는 CSV 형식 예:

```
emailAddress,unsubscribeAll,attributesData,topicPreferences.Sports,topicPreferences.Cycling
```

```
example1@amazon.com,false,{"Name": "John"},OPT_IN,OPT_OUT
example2@amazon.com,true,,OPT_OUT,OPT_OUT
```

JSON 형식

줄바꿈으로 구분된 JSON 파일만 지원됩니다. 이 형식에서 각 줄은 하나의 연락처 정보가 포함된 완전한 JSON 객체입니다.

연락처 목록에 연락처를 일괄 추가하는 JSON 형식 예:

```
{
     "emailAddress": "example1@amazon.com",
     "unsubscribeAll": false,
     "attributesData": "{\"Name\":\"John\"}",
     "topicPreferences": [
      {
          "topicName": "Sports",
          "subscriptionStatus": "OPT_IN"
      },
      {
          "topicName": "Cycling",
          "subscriptionStatus": "OPT_OUT"
      }
     ]
}
{
     "emailAddress": "example2@amazon.com",
     "unsubscribeAll": true,
     "topicPreferences": [
      {
          "topicName": "Sports",
          "subscriptionStatus": "OPT_OUT"
      },
      {
          "topicName": "Cycling",
          "subscriptionStatus": "OPT_OUT"
      }
     ]
}
```

위의 예제에서 *example1@amazon.com* 및 *example2@amazon.com*을 연락처 목록에 추가하려는 이메일 주소로 바꿉니다. `attributesData` 값을 연락처와 관련된 값으로 바꿉니다. 또한 *Sports* 및 *Cycling*을 연락처에 적용되는 `topicName`으로 바꿉니다. 허용되는 `topicPreferences`는 *OPT_IN*과 *OPT_OUT*입니다.

연락처를 CSV 또는 JSON 형식으로 Amazon S3 객체에 업로드할 때 지원되는 속성은 다음과 같습니다.

속성	설명
`emailAddress`	연락처의 이메일 주소이며 필수 필드입니다.
`unsubscribeAll`	연락처가 모든 연락처 목록 주제에서 구독 취소되었는지 여부를 나타내는 부울 값 상태입니다.
`topicPreferences`	주제 수신 또는 수신 거부에 대한 연락처의 기본 설정입니다.
`attributesData`	연락처에 연결된 속성 데이터입니다.

- Amazon SES에 Amazon S3 객체를 읽을 수 있는 권한을 부여합니다.

  Amazon S3 버킷에 적용하면 다음 정책은 Amazon SES에서 해당 버킷을 읽을 수 있는 권한을 부여합니다. Amazon S3의 버킷 정책에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 정책 및 사용자 정책 사용을 참조하십시오.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSESGet",
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::BUCKET-NAME/OBJECT-NAME",
            "Condition": {
```

```
                "StringEquals": {
                    "aws:Referer": "AWSACCOUNTID"
                }
            }
        }
    ]
}
```

- Amazon SES에 AWS KMS 키를 사용할 수 있는 권한을 부여합니다.

  Amazon S3 객체가 AWS KMS 키로 암호화된 경우 Amazon SES에 KMS 키를 사용할 수 있는 권한을 부여해야 합니다. Amazon SES 기본 KMS 키가 아닌 고객 관리 키에서만 권한을 얻을 수 있습니다. 고객 관리 키를 사용하려면 키의 정책에 설명을 추가하여 Amazon SES에 사용 권한을 부여해야 합니다.

  다음 정책 설명을 키 정책에 붙여 넣어 Amazon SES가 고객 관리 키를 사용할 수 있도록 허가하십시오.

```
{
   "Sid": "AllowSESToDecrypt",
   "Effect": "Allow",
   "Principal": {
       "Service":"ses.amazonaws.com"
   },
   "Action": [
       "kms:Decrypt",
   ],
   "Resource": "*"
}
```

- `CreateImportJob` 작업을 Amazon SES API v2에서 사용합니다.

> **Note**
>
> 다음 예제에서는 AWS CLI를 이미 설치했다고 가정합니다. 설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

명령줄에 다음 명령을 입력합니다. *s3bucket*을 Amazon S3 버킷의 이름으로 바꾸고 *s3object*를 Amazon S3 객체 이름으로 바꿉니다.

```
aws sesv2 create-import-job --import-destination
 ContactListDestination={ContactListName=ExampleContactListName,ContactListImportAction=PUT}
 --import-data-source S3Url="s3://s3bucket/s3object",DataFormat=CSV
```

예제를 사용한 목록 관리 연습

다음 연습에서는 목록 관리를 사용하여 연락처를 나열하고, `ListManagementOptions`를 활용하여 이메일에 연락처 목록과 주제 이름을 지정하고, 구독 취소 링크를 삽입하는 방법에 대한 예제를 제공합니다.

1. 를 사용하여 연락처 나열 AWS CLI - `ListContacts` 작업을 사용하여 작업과 함께 특정 주제를 구독한 모든 연락처 목록을 검색할 수 있으며`SendEmail`, 이를 통해 이메일을 보낼 수 있습니다.

   명령줄에 다음 명령을 입력합니다.

   ```
   aws sesv2 list-contacts --cli-input-json file://LIST-CONTACTS-JSON
   ```

   위의 명령에서 *`LIST-CONTACT-JSON`*을 `ListContacts` 요청에 대한 JSON 파일 경로로 바꿉니다.

   요청에 대한 `ListContacts` 입력 JSON 파일의 예는 다음과 같습니다.

   ```
   {
       "ContactListName": "ExampleContactListName",
       "Filter": {
           "FilteredStatus": "OPT_IN",
           "TopicFilter": {
               "TopicName": "Cycling",
               "UseDefaultIfPreferenceUnavailable": true
           }
       },
       "PageSize": 50
   }
   ```

   `FilteredStatus`는 필터링하려는 구독 상태를 보여주며, 이 상태는 `OPT_IN` 또는 `OPT_OUT`입니다.

   `TopicFilter`는 결과로 나타나기를 원하는 주제를 지정하는 필터 옵션이며 위의 예에서는 'Cycling'입니다.

UseDefaultIfPreferenceUnavailable은 true 또는 false 값을 가질 수 있습니다. true의 경우, 연락처에 주제에 대한 명시적 기본 설정이 없는 경우 주제 기본 설정이 사용됩니다. false의 경우, 명시적으로 설정된 기본 설정이 있는 연락처만 필터링에 고려됩니다.

2. **ListManagementOptions**를 사용하여 메일 전송 – 위의 ListContacts 작업을 사용하여 목록에 연락처를 나열한 후 SendEmail 작업을 사용하여 ListManagementOptions 헤더를 통해 연락처 목록과 주제 이름을 지정함으로써 각 연락처에 이메일을 보낼 수 있습니다.

   SendEmail 작업에 ListManagementOptions를 사용하려면 이메일이 속한 contactListName 및 topicName을 포함합니다(topicName은 선택 사항임).

   ```
   ListManagementOptions:
       String contactListName
       String topicName
   ```

   연락처 목록에 없는 수신자 이메일 주소에 SendEmail 요청의 ListManagementOptions를 포함하면 목록에 자동으로 연락처가 만들어집니다.

   Amazon SES는 연락처 목록에 있는 구독 취소된 연락처로 이메일을 전송하면 반송합니다. 즉, 구독이 취소된 연락처로 전송되지 않도록 SendEmail 요청을 업데이트할 필요가 없습니다.

3. 구독 취소 링크의 위치 지정 – ListManagementOptions를 사용할 때 Amazon SES가 구독 취소 URL을 삽입해야 하는 위치를 지정하기 위해 {{amazonSESUnsubscribeUrl}} 자리 표시자를 사용하여 SES가 이메일에 구독 취소 바닥글 링크를 추가할 수 있도록 하는 옵션이 있습니다. HTML 및 TEXT 콘텐츠 형식에 대해서만 자리 표시자 교체가 지원됩니다. 자리 표시자를 최대 두 번 포함할 수 있습니다. 두 번 이상 사용하는 경우 처음 두 건만 교체됩니다. 자세한 내용은 구독 관리 사용 단원을 참조하십시오.

   또는 SMTP 인터페이스를 사용하여 이메일을 보내는 경우 X-SES-LIST-MANAGEMENT-OPTIONS 헤더를 사용하여 목록과 주제 이름을 지정할 수 있습니다.

   SMTP 인터페이스를 사용하여 이메일을 보내는 동안 목록과 주제 이름을 지정하려면 메시지에 다음 이메일 헤더를 추가합니다.

   ```
   X-SES-LIST-MANAGEMENT-OPTIONS: {contactListName}; topic={topicName}
   ```

구독 관리 사용

Amazon SES는 `SendEmail` 작업 요청의 `ListManagementOptions` 내에서 `contactListName` 및 `topicName`을 지정할 때 Amazon SES가 모든 발신 이메일의 구독 취소 링크를 자동으로 활성화하는 구독 관리 기능을 제공합니다.

연락처가 특정 주제 또는 목록에서 구독을 취소하는 경우 Amazon SES는 향후에 해당 주제 또는 목록의 연락처로 이메일을 보낼 수 없습니다.

> **Note**
>
> - Amazon SES 구독 관리는 많은 이메일 서비스 공급자가 시행하는 대량 발신자 요구 사항을 지원합니다. 자세한 내용은 대량 발신자 변경 사항 개요의 섹션 2를 참조하세요.
> - 구독 관리 기능은 Amazon SES에서 Easy DKIM을(를) 사용하는 사용자에게 제공됩니다. 하지만 Amazon SES에 전화하기 전에 직접 이메일에 서명하는 발신자의 이메일에 Amazon SES가 수신 거부 링크를 추가할 수는 없습니다.

특정 주제를 구독한 모든 연락처의 목록을 검색하는 등 목록 관리 및 사용 방법에 대한 자세한 내용은 목록 관리 사용 단원을 참조하세요.

구독 관리 개요

구독 관리를 사용하는 경우 다음 요소를 고려해야 합니다.

- 구독 관리는 Amazon SES에서 완전히 관리됩니다. 즉, Amazon SES가 구독 취소 웹 페이지에서 수신 거부 이메일과 요청을 수신하면 목록에서 연락처의 기본 설정을 업데이트합니다. 구성 세트 알림을 사용하여 구독 취소 알림을 받을 수 있습니다. 구성 세트에 대한 자세한 내용은 Amazon SES에서 구성 세트 사용 단원을 참조하세요.
- 이메일을 보내는 동안 연락처 목록을 지정해야 합니다. `List-Unsubscribe` 헤더와 `ListManagementOptions` 바닥글 링크를 통한 구독 관리는 그에 따라 처리됩니다.
- Amazon SES는 `List-Unsubscribe` 헤더 표준에 대한 지원을 추가하여 이메일 클라이언트와 받은 편지함 공급자가 지원하는 경우 이메일 상단에 구독 취소 링크를 표시할 수 있도록 합니다. 모든 이메일 서비스 공급자가 이러한 헤더를 지원하는 것은 아닙니다.
- `List-Unsubscribe` 헤더는 다음 동작을 따릅니다.
  - 연락처 목록과 주제가 모두 지정된 이메일에서 구독 취소 링크를 클릭하면 해당 연락처는 해당 특정 주제에서만 구독이 취소됩니다.

  - 주제를 지정하지 않으면 연락처가 목록의 모든 주제에서 구독 취소됩니다.
- 연락처가 이메일 바닥글에서 수신 거부 링크를 클릭하면 수신 거부 랜딩 페이지로 이동합니다.
- 구독 취소 랜딩 페이지는 연락처에 기본 설정을 업데이트할 수 있는 옵션을 제공합니다. 즉 특정 목록의 모든 주제에 대한 `OPT_IN` 또는 `OPT_OUT`입니다. 랜딩 페이지에는 목록의 모든 주제의 구독을 취소할 수 있는 옵션도 제공됩니다.
- `ListManagementOptions`를 사용하는 경우 Amazon SES가 구독 취소 URL을 삽입해야 하는 위치를 나타내려면 이메일에 `{{amazonSESUnsubscribeUrl}}` 자리 표시자를 포함해야 합니다. 자리 표시자를 최대 두 번 포함할 수 있습니다. 두 번 이상 사용하는 경우 처음 두 건만 교체됩니다.
- `List-Unsubscribe` 헤더 및 `ListManagementOptions` 바닥글 링크는 이메일이 단일 수신자에게 전송되는 경우에만 추가됩니다.
- 연락처가 구독을 취소할 수 없도록 하려는 트랜잭션 이메일의 경우 `SendEmail` 요청에서 `ListManagementOptions` 필드를 생략할 수 있습니다.

구독 취소 헤더 고려 사항

이메일에 다음 헤더가 포함된 경우 구독 취소 링크를 통한 구독 관리가 사용됩니다.

`List-Unsubscribe`

`List-Unsubscribe-Post`

Amazon SES의 구독 관리인 `ListManagementOptions`를 사용할 때 Amazon SES는 이러한 헤더가 이메일에 있는 경우 헤더를 재정의합니다.

이러한 헤더에 의해 생성된 링크를 클릭하여 구독을 취소하는 수신자는 이메일 클라이언트 또는 받은 편지함 공급자에 따라 다른 환경을 갖습니다. 일부 공급자는 `List-Unsubscribe` 및 `List-Unsubscribe-Post` 헤더를 인식하지 못하기 때문입니다. 이러한 공급자를 사용하여 수신자에게 보낸 이메일에는 구독 취소 링크가 표시되지 않습니다.

이메일 클라이언트에서 이러한 헤더를 인식하는 수신자는 구독 취소 링크를 보고 링크를 통해 구독을 취소할 수 있지만 구독을 취소할 주제를 선택할 수 없으며 단지 이메일이 전송된 주제의 구독이 취소됩니다.

`List-Unsubscribe` 헤더에 대한 자세한 내용은 RFC 2369를 참조하고, `List-Unsubscribe-Post` 헤더에 대한 자세한 내용은 RFC 8058을 참조하세요.

> **Note**
>
> Amazon SES는 많은 이메일 서비스 공급자가 시행하는 대량 발신자 요구 사항에 따라 원클릭 구독 취소를 지원합니다. 자세한 내용은 Using one-click unsubscribe with Amazon SES를 참조하세요.

구독 취소 바닥글 링크 추가

Amazon SES가 구독 취소 URL을 삽입해야 하는 위치를 지정하려면 템플릿이 있는 이메일과 템플릿이 없는 이메일에 `{{amazonSESUnsubscribeUrl}}` 자리 표시자를 사용해야 합니다.

HTML 및 TEXT 콘텐츠 형식에 대해서만 자리 표시자 교체가 지원됩니다.

자리 표시자를 최대 두 번 포함할 수 있습니다. 두 번 이상 사용하는 경우 처음 두 건만 교체됩니다.

> **Note**
>
> `{{amazonSESUnsubscribeUrl}}` 자리 표시자는 `SendEmail` 작업을 사용하는 동안 `ListManagementOptions`가 헤더로 지정되거나 SMTP 인터페이스를 사용하는 동안 X-SES-LIST-MANAGEMENT-OPTIONS가 헤더로 지정된 경우에만 사용할 수 있습니다. (`ListManagementOptions`에 종속되지 않고 단독으로 사용할 수 있는 `List-Unsubscribe` 또는 `List-Unsubscribe-Post` 헤더와 혼동하지 마세요.)

Amazon SES 전송 활동 모니터링

Amazon SES는 이벤트, 지표 및 통계를 사용하여 전송 활동을 모니터링할 수 있는 방법을 제공합니다. 이벤트는 지표로 추적하도록 지정한 전송 활동과 관련하여 발생합니다. 지표는 통계를 생성하는 모니터링된 이벤트 유형의 값을 나타내는 시간 순서별 데이터 포인트 세트를 나타냅니다. 통계는 현재까지의 지표를 포함하여 지정한 기간 동안의 지표 데이터 집계입니다.

이러한 모니터링 방법을 사용하여 계정의 반송, 수신 거부, 거부율 같은 주요 지표를 추적할 수 있습니다. 반송 및 수신 거부 발생률이 지나치게 높으면 SES를 사용하여 이메일을 전송하는 데 어려움을 겪을 수 있습니다. 또한 이러한 방법을 통해 이벤트 게시 및 구성 세트와 관련된 사용자 지정 도메인을 활용하여 전체 열림 및 클릭률을 식별할 수 있으므로 고객이 전송되는 이메일에 참여하는 비율을 측정할 수도 있습니다(확인 및 클릭 추적을 처리하기 위한 사용자 지정 도메인 구성 참조).

모니터링을 설정하는 첫 번째 단계는 SES를 사용하여 측정하고 모니터링하려는 전송 활동과 관련된 이메일 이벤트 유형을 식별하는 것입니다. SES에서 모니터링할 다음 이벤트 유형을 선택할 수 있습니다.

- 전송(Send) - 전송 요청이 성공했으며 Amazon SES는 메시지를 수신자의 메일 서버로 전송합니다. (계정 수준 또는 전역 금지를 사용하는 경우 SES가 여전히 전송으로 계산하지만 배달은 금지합니다.)
- 렌더링 오류 – 템플릿 렌더링 문제로 인해 이메일이 전송되지 않았습니다. 이 이벤트 유형은 템플릿 데이터가 누락되었을 때 또는 템플릿 파라미터와 데이터 사이에 불일치가 있을 때 발생할 수 있습니다. (이 이벤트 유형은 `SendTemplatedEmail` 또는 `SendBulkTemplatedEmail` API 작업을 사용하는 이메일을 전송할 때만 발생합니다.)
- 거부(Reject) - Amazon SES가 이메일을 수락했으나 이메일에 바이러스가 포함된 것으로 판단되어 수신자의 메일 서버로 전송하려고 시도하지 않았습니다.
- 배달(Delivery) - Amazon SES에서 이메일을 수신자의 메일 서버로 성공적으로 전송했습니다.
- 바운스 – 수신자의 메일 서버가 이메일을 영구적으로 거부하는 하드 바운스입니다. (소프트 바운스는 SES가 더 이상 이메일 전송을 재시도하지 않는 경우에만 포함됩니다. 일반적으로 이러한 소프트 바운스는 전송 실패를 나타내지만, 메일이 수신자의 받은 편지함에 도착하더라도 소프트 바운스가 반환되는 경우도 있습니다. 이러한 상황은 일반적으로 수신자가 부재 중 자동 회신을 보낼 때 발생합니다. AWS re:Post 문서에서 소프트 바운스에 대해 자세히 알아보세요.)
- 수신 거부(Complaint) - 이메일이 수신자의 메일 서버로 성공적으로 전송되었지만 수신자가 이를 스팸으로 표시했습니다.

- 배달 지연 – 일시적인 문제가 발생하여 수신자의 메일 서버에 이메일을 전송할 수 없습니다. 예를 들어 수신자의 받은 편지함이 가득 찼거나 이메일 수신 서버에 일시적인 문제가 발생했을 때 전송 지연이 발생할 수 있습니다.
- 구독(Subscription) - 이메일이 성공적으로 배달되었지만 수신자가 이메일 헤더에서 `List-Unsubscribe`를 클릭하거나, 바닥글에서 `Unsubscribe` 링크를 선택하여 구독 기본 설정을 업데이트했습니다.
- 열기(Open) – 수신자가 메시지를 수신하여 자신의 이메일 클라이언트에서 열었습니다.
- 클릭(Click) – 수신자가 이메일의 링크를 1개 이상 클릭했습니다.

여러 방법으로 이메일 전송 이벤트를 모니터링할 수 있습니다. 선택한 방법은 모니터링하려는 이벤트 유형, 모니터링하려는 세분화 및 세부 정보 수준, SES에서 데이터를 게시할 위치에 따라 달라집니다. 피드백 알림 또는 이벤트 게시를 사용하여 반송 메일 및 수신 거부 이벤트를 추적해야 합니다. 여러 가지 모니터링 방법을 사용하기로 선택할 수도 있습니다. 각 방법의 특성이 다음 표에 나와 있습니다.

모니터링 방법	모니터링할 수 있는 이벤트	데이터에 액세스하는 방법	세부 정보의 수준	세부 수준
SES 콘솔	계정 상태, 보낸 이메일, 사용된 할당량, 성공적인 전송 요청, 거부, 반송 및 수신 거부(현재 평판에 대한 최근 이력)	SES 콘솔의 계정 대시보드 페이지	카운트 및 백분율	전체 AWS 계정에서
SES 콘솔	계정 상태, 보낸 이메일, 반송 및 수신 거부(현재 평판)	SES 콘솔의 평판 지표 페이지	계산 비율만	전체 AWS 계정에서
가상 배달 가능성 관리자	계정 통계, ISP, 전송 ID, 구성 세트, 전송, 배달, 수신 거부, 일시적 및 영구 바운스, 열기 및 클릭, 전	SES 콘솔의 the section called “대시보드” SES 콘솔의 the section called “어드바이저”	카운트 및 백분율	전체 AWS 계정에서

모니터링 방법	모니터링할 수 있는 이벤트	데이터에 액세스하는 방법	세부 정보의 수준	세부 수준
	달 가능성 및 평판.			
SES API	전송 완료, 반송, 수신 거부, 거부	GetSendStatistics API 작업	개수만	전체 AWS 계정에서

모니터링 방법	모니터링할 수 있는 이벤트	데이터에 액세스하는 방법	세부 정보의 수준	세부 수준
Amazon CloudWatch 콘솔	발신, 전송, 조회, 클릭, 반송 메일, 반송 메일 발생률, 수신 거부, 수신 거부율, 거부, 렌더링 오류 및 차단 목록에 등록된 IP	CloudWatch 콘솔 **Note** 연관된 이벤트가 발생할 때까지 일부 지표가 CloudWatch에 표시되지 않습니다. 예를 들어 반송 메일을 하나 이상 전송하거나 메일박스 시뮬레이터를 사용하여 시뮬레이션된 반송 이벤트를 생성할 때까지 반송 지표는 CloudWatc	개수만	전체 AWS 계정에서

모니터링 방법	모니터링할 수 있는 이벤트	데이터에 액세스하는 방법	세부 정보의 수준	세부 수준
		h에 표시되지 않습니다.		
피드백 알림	전송 완료, 반송 및 수신 거부	Amazon SNS 알림(전달, 반송 메일 및 수신 거부) 또는 이메일(반송 메일 및 수신 거부만). 이메일 알림 설정을(를) 참조하세요.	각 이벤트의 세부 정보	전체 AWS 계정에서
이벤트 게시	전송, 배달, 열기, 클릭, 반송 메일, 수신 거부, 거부, 렌더링 실패.	Amazon CloudWatch 또는 Amazon Data Firehose 또는 Amazon SNS 알림 - 이벤트 게시를 사용하여 이메일 전송 모니터링의 내용을 참조하세요. (추가 요금이 적용되며, CloudWatch의 지표당 요금 참조)	각 이벤트의 세부 정보	자세히(사용자 정의 가능한 이메일 특성에 근거)

모니터링 방법	모니터링할 수 있는 이벤트	데이터에 액세스하는 방법	세부 정보의 수준	세부 수준
구성 세트와 관련된 사용자 지정 도메인을 활용한 이벤트 게시 - 추가 정보	추적을 열고 클릭합니다.	Amazon CloudWatch 또는 Amazon Data Firehose 또는 Amazon SNS 알림. (추가 요금이 적용되며, CloudWatch의 지표당 요금 참조)	각 이벤트의 세부 정보입니다.	자세히(사용자 정의 가능한 이메일 특성에 근거)

> **Note**
>
> 이메일 전송 이벤트에서 측정되는 지표는 발신 할당량과 정확히 일치하지 않을 수도 있습니다. 이러한 불일치는 이메일 바운스 및 수신 거부로 인해, 혹은 SES 받은 편지함 시뮬레이터를 사용하면서 발생할 수 있습니다. 발신 할당량에 얼마나 근접했는지 확인하는 방법에 대한 자세한 내용은 발신 할당량 모니터링 섹션을 참조하세요.

각 모니터링 방법의 사용법에 대한 정보는 다음 주제를 참조하세요.

- Amazon SES 콘솔을 사용하여 전송 통계 모니터링
- Amazon SES API를 사용하여 사용 통계 모니터링
- Amazon SES 이벤트 게시를 사용하여 이메일 전송 모니터링

Amazon SES 콘솔을 사용하여 전송 통계 모니터링

Amazon SES 콘솔의 계정 대시보드, 평판 지표 및 SMTP 설정 페이지에서 모든 이메일 전송, 사용량, 통계, SMTP 설정, 전체 계정 상태 및 평판 지표를 모니터링할 수 있습니다. 다음 단원에서는 이러한 각 콘솔 페이지에 표시되는 지표와 통계에 대해 설명합니다.

the section called “계정 대시보드” 및 the section called “평판 지표” 콘솔 페이지에 모두 반송 및 수신 거부 지표가 표시되지만 아래에 설명된 대로 이 두 페이지의 반송 비율 및 수신 거부 비율 사이에는 미세한 차이가 있습니다.

- 계정 대시보드 페이지 - 선택한 날짜 범위를 기준으로, 현재까지 이어지는 변화의 지표 진행 상황을 보여주는 과거의 반송 비율 및 수신 거부 비율을 확인할 수 있습니다.
- 평판 지표 페이지 - 전체 과거 평균을 개괄적인 수준으로 계산하여 얻은 최신 데이터 포인트를 기준으로 한 반송 및 수신 거부 비율. 계정 대시보드(Account dashboard) 페이지에 실시간으로 표시되는 정확한 반송/수신 거부 이벤트에 해당하는 정기적인 반송/수신 거부 비율과 혼동해서는 안 됨.

평판 지표(Reputation metrics) 페이지와 계정 대시보드(Account dashboard) 페이지의 반송 메일 비율 또는 수신 거부 비율 비교하기 위해 간단한 예로, 비율이 어제 2%였고 현재 1%라고 가정해 봅시다. 평판 지표(Reputation metrics) 페이지에는 1%의 현재 비율만 표시되지만, 계정 대시보드(Account dashboard) 페이지에는 어제 2%와 오늘 1%의 비율을 보여주는 진행률 차트로 그래프가 표시됩니다.

계정 대시보드

계정에서 전송한 이메일의 수와 사용된 전송 할당량의 비율을 SES 콘솔 계정 대시보드(Account dashboard) 페이지의 일일 이메일 사용량(Daily email usage) 창에서 직접 모니터링할 수 있습니다. 계정의 전송 비율 및 수신 거부율을 전송 통계(Sending Statistics) 창에서 모니터링할 수 있을 뿐만 아니라 다음 창에서 이메일 전송과 관련된 기타 주요 요소를 모니터링할 수 있습니다.

- 전송 제한 - SES를 통해 메일을 보내는 데 적용되는 다음 할당량을 포함합니다.
  - 일일 전송 할당량 - 24시간 내에 보낼 수 있는 최대 이메일 수입니다.
  - 최대 송신률 - 계정에서 초당 전송할 수 있는 최대 이메일 수입니다.
- 계정 상태 - SES 계정의 상태:
  - `Healthy` - 현재 계정에 영향을 미치는 평판 관련 문제는 없습니다.
  - `Under review` - SES 계정에 잠재적 문제가 확인되었습니다. 사용자가 문제를 해결하는 동안 계정이 검토 중입니다.
  - `Paused` - 사용자 계정에서 보낸 이메일 문제로 인해 사용자 계정의 이메일 전송 기능이 일시 중지되었습니다. 문제가 수정된 경우 사용자 계정의 이메일 전송 기능이 재개되도록 요청할 수 있습니다.
- 일일 이메일 사용량 - 전송 제한에 도달하지 않도록 일일 사용량을 확인하는 방법:
  - 전송된 이메일 - 24시간 동안 보낸 이메일 수입니다.
  - 남은 전송 - 24시간 동안 보낼 수 있는 남은 이메일의 총 개수입니다.

  - 사용된 전송 할당량 - 사용한 일일 전송 할당량의 백분율입니다.
- 전송 통계 - 네 가지 필수 지표의 진행률을 보여주는 그래프로 구성되어 있습니다. 모니터링된 이벤트 유형의 값을 나타내는 데이터 포인트 세트가 시간순으로 표시되며, 이 값이 1시간의 집계 기간을 사용하여 선택한 날짜 범위에 대한 통계를 생성합니다. 시작 값이 `Last 1 day`부터 `Last 14 days`까지인 데이터 범위를 선택하여 아래 차트를 필터링할 수 있습니다.
  - 전송 - 선택한 날짜 범위에서 성공한 이메일 전송 요청의 합계입니다.
  - 거부 - 선택한 날짜 범위에서 `Rejects/Sends * 100`을 기반으로 SES에 의해 거부된 전송 요청의 평균 비율입니다.
  - 반송 메일 - 선택한 날짜 범위의 진행률을 보여주는, 전체 과거 발신자 평판 지표에서 도출된 평균 비율입니다.
  - 수신 거부 - 선택한 날짜 범위의 진행률을 보여주는, 전체 과거 발신자 평판 지표에서 도출된 평균 비율입니다.

이러한 각 차트에 포함된 CloudWatch에서 보기 버튼을 사용하여 Amazon CloudWatch 콘솔에서 해당 지표를 열어 자세한 데이터를 보고, 사용자 지정된 지표 수식을 수행하며, CloudWatch에서 경보를 생성할 수 있습니다.

평판 지표

반송 메일 및 수신 거부 비율 외에도 평판 지표(Reputation metrics) 페이지에서는 다음 창을 통해 평판에 영향을 미치는 기타 주요 요소에 대해 개략적으로 보여줍니다.

- 요약 - 평판 상태에 대한 개요를 제공합니다.
  - 상태 - 과거 반송 및 수신 거부 비율을 기준으로 한 전반적인 평판 상태:
    - `Healthy` - 두 지표가 모두 정상 수준 내에 있습니다.
    - `Under review` - 지표 중 하나 또는 두 지표 모두로 인해 계정이 자동으로 검토 상태가 되었습니다.
    - `At risk` - 지표 중 하나 또는 두 지표 모두 좋지 않은 수준의 상태에 도달했으며 계정에서 이메일 전송 기능을 사용하지 못하게 될 수 있습니다.
  - 전송된 이메일 수(최근 24시간) - 최근 24시간 동안 보낸 이메일의 총 개수입니다.
  - 남은 전송 수 - 24시간 동안 보낼 수 있는 남은 이메일의 총 개수입니다.
  - 사용한 전송 할당량 - 사용한 일일 전송 할당량의 백분율입니다.
- 계정 수준 탭 내용:
  - 반송 비율(Bounce rate)

    - 상태 - 요약(Summary) 창에 설명된 것과 동일한 값을 사용하여 반송 비율의 상태를 나타냅니다.
    - 과거 반송 비율 - 계정에서 보낸 이메일 중 하드 바운스가 된 이메일의 백분율로, 일반적인 전송 관행을 나타내는 대표 볼륨을 기준으로 전체 과거 평균을 계산합니다.
  - 수신 거부율
    - 상태 - 요약(Summary) 창에 설명된 것과 동일한 값을 사용하여 수신 거부 비율의 상태를 나타냅니다.
    - 과거 반송 비율 - 계정에서 보낸 이메일 중 수신자가 스팸으로 신고한 이메일의 백분율로, 일반적인 전송 관행을 나타내는 대표 볼륨을 기준으로 전체 과거 평균을 계산합니다.
- 구성 세트 탭 내용:
  - 구성 세트별 평판
    - 구성 세트 - 평판 지표가 사용 설정된 구성 세트를 입력하거나 선택할 수 있으므로 선택한 구성 세트를 사용하여 전송된 이메일을 기반으로 요약, 반송 메일 및 수신 거부 데이터를 볼 수 있습니다. 구성 세트를 선택한 후 나타나는 결과 창은 위에서 설명한 평판 지표 페이지와 같지만, 다른 점은 전체 계정 수준 전송 지표가 아닌 선택한 구성 세트를 사용해서 보낸 이메일만을 기반으로 한다는 점입니다.

SMTP 설정

이 페이지에서는 SES API를 통해 또는 프로그래밍 방식으로 Amazon SES SMTP 인터페이스를 사용하는 데 필요한 SMTP 설정을 나열하고 SMTP 보안 인증 정보를 생성 및 관리하기 위한 링크를 제공합니다.

- SMTP 설정 - SMTP 지원 프로그래밍 언어, 이메일 서버 또는 애플리케이션을 사용하여 Amazon SES SMTP 인터페이스에 연결하려는 경우 제공되는 정보:
  - SMTP 엔드포인트
  - STARTTLS 포트
  - 전송 계층 보안(TLS)
  - TLS 래퍼 포트
  - SMTP 및 IAM 보안 인증 정보 생성 및 관리를 위해 제공되는 인증 링크

콘솔을 사용하여 전송 및 평판 지표 모니터링

다음 절차에서는 최근 기록(최대 14일)을 기반으로 한 지표에 대해 계정 대시보드(Account dashboard) 페이지를 사용하여 전송 및 평판 지표를 탐색하거나, 현재까지의 전체 내역을 기준으로 지표에 대해 평판 지표(Reputation metrics) 페이지를 사용하는 방법을 안내합니다.

보낸 이메일 및 전송 할당량을 보려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 탐색 창에서 Account dashboard(계정 대시보드)를 선택합니다. 사용 통계는 일별 이메일 사용량 섹션에 표시됩니다.

전송 수, 거부율, 반송 및 수신 거부 발생률 확인

1. 탐색 창에서 Account dashboard(계정 대시보드)를 선택합니다.
2. 전송 통계 섹션에서, 날짜 범위 드롭다운을 통해 날짜 범위의 시작 값을 선택하여 전송 통계 섹션 바로 아래에 있는 네 개의 차트를 필터링합니다.
3. 선택한 날짜 범위를 기준으로, 현재까지 이어지는 변화의 지표 진행 상황을 보여주는 과거의 수 및 비율을 확인할 수 있습니다.
4. 차트에 포함된 CloudWatch에서 보기 버튼을 사용하여 Amazon CloudWatch 콘솔에서 해당 지표를 열어 자세한 데이터를 보고, 사용자 지정된 지표 수식을 수행하며, CloudWatch에서 경보를 생성할 수 있습니다.

전체 과거 반송 메일 및 수신 거부 비율을 보는 방법

1. 탐색 창에서 평판 지표(Reputation metrics)를 선택합니다.
2. 반송 비율(Bounce rate) 창에서 사용자 계정에서 보낸 이메일 중 하드 바운스로 이어진 이메일의 비율을 볼 수 있습니다. 수신 거부 비율(Complaint rate) 창에서는 사용자 계정에서 보낸 이메일 중 수신자가 스팸으로 신고한 이메일의 비율을 볼 수 있습니다. 두 지표 모두 사용자의 일반적인 전송 관행을 기반으로 한 이메일의 대표 볼륨으로 계산한 수치입니다.
3. 두 창 중 하나에서 CloudWatch에서 보기(View in CloudWatch) 버튼을 사용하여 Amazon CloudWatch 콘솔에서 해당 지표를 열어 자세한 데이터를 보고, 사용자 지정된 지표 수식을 수행하며, CloudWatch에서 경보를 생성할 수 있습니다.

구성 세트별로 평판 지표를 보는 방법

1. 탐색 창에서 평판 지표(Reputation metrics)를 선택합니다.
2. 평판 지표(Reputation metrics) 페이지에서 구성 세트(Configuration set)탭을 선택합니다.
3. 구성 세트별 평판(Reputation by configuration set)창에서 구성 세트(Configuration set) 필드 내부를 클릭한 후, 평판 지표가 사용 설정된 구성 세트를 입력하기 시작하거나 선택합니다.
4. 구성 세트를 선택하면 선택한 구성 세트를 사용하여 전송된 이메일만을 기준으로 지표를 표시하는 요약(Summary), 반송 메일(Bounce), 수신 거부(Complaint) 창이 로드됩니다.

Amazon SES API를 사용하여 사용 통계 모니터링

Amazon SES API는 서비스 사용에 대한 정보를 반환하는 `GetSendStatistics` 작업을 제공합니다. 따라서 필요할 때는 조정할 수 있도록 전송 통계를 정기적으로 확인하는 것이 바람직합니다.

`GetSendStatistics` 작업을 호출하면 지난 2주 동안의 전송 활동을 나타내는 데이터 요소 목록이 전송됩니다. 이 목록의 각 데이터 요소는 15분 단위의 활동을 나타내며 해당 기간에 대한 다음과 같은 정보를 포함합니다.

- 하드 바운스 수
- 불만 제기 수
- 전송 완료 시도 수(전송한 이메일 수와 일치함)
- 거부된 전송 시도 수
- 분석 기간을 나타내는 타임스탬프

`GetSendStatistics` 작업에 대한 자세한 설명은 Amazon Simple Email Service API 참조를 참조하십시오.

이 단원에는 다음과 같은 주제가 포함되어 있습니다.

- the section called “를 사용하여 `GetSendStatistics` API 작업 호출 AWS CLI”
- the section called “프로그래밍 방식의 `GetSendStatistics` 작업 호출”

를 사용하여 **`GetSendStatistics`** API 작업 호출 AWS CLI

`GetSendStatistics` API 작업을 가장 쉽게 호출하려면 AWS Command Line Interface(AWS CLI)를 사용해야 합니다.

를 사용하여 **GetSendStatistics** API 작업을 호출하려면 AWS CLI

1. AWS CLI를 아직 설치하지 않았다면 설치합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서의 "설치" AWS Command Line Interface를 참조하세요.
2. 아직 구성하지 않은 경우 자격 증명을 사용하도록 AWS CLI AWS 를 구성합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서의 "Configuring the AWS CLI"를 참조하세요.
3. 명령줄 프롬프트에 다음 명령을 실행합니다.

   ```
   aws ses get-send-statistics
   ```

   AWS CLI 가 제대로 구성된 경우 전송 통계 목록이 JSON 형식으로 표시됩니다. JSON 객체에는 각각 15분 동안 수집된 전송 통계가 포함됩니다.

프로그래밍 방식의 **GetSendStatistics** 작업 호출

SDK를 사용하여 `GetSendStatistics` 작업을 호출할 수도 있습니다. AWS SDKs 이 섹션에는 Go, PHP, Python 및 Ruby용 AWS SDKs 대한 코드 예제가 포함되어 있습니다. 다음 링크 중 하나를 선택하면 해당 언어의 코드 예제를 볼 수 있습니다.

- AWS SDK for Go코드 예제
- AWS SDK for PHP코드 예제
- AWS SDK for Python (Boto)코드 예제
- AWS SDK for Ruby코드 예제

> **Note**
>
> 이 코드 예제에서는 액세스 키 ID, AWS 보안 AWS 액세스 키 및 선호하는 AWS 리전이 포함된 AWS 공유 자격 증명 파일을 생성했다고 가정합니다. 자세한 내용은 공유 자격 증명 및 구성 파일을 참조하십시오.

를 **GetSendStatistics** 사용하여 호출 AWS SDK for Go

```
package main

import (
```

```
    "fmt"

    //go get github.com/aws/aws-sdk-go/...
    "github.com/aws/aws-sdk-go/aws"
    "github.com/aws/aws-sdk-go/aws/session"
    "github.com/aws/aws-sdk-go/service/ses"
    "github.com/aws/aws-sdk-go/aws/awserr"
)

const (
    // Replace us-west-2 with the AWS Region you're using for Amazon SES.
    AwsRegion = "us-west-2"
)

func main() {

    // Create a new session and specify an AWS Region.
    sess, err := session.NewSession(&aws.Config{
        Region:aws.String(AwsRegion)},
    )

    // Create an SES client in the session.
    svc := ses.New(sess)
    input := &ses.GetSendStatisticsInput{}

    result, err := svc.GetSendStatistics(input)

    // Display error messages if they occur.
    if err != nil {
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            default:
                fmt.Println(aerr.Error())
            }
        } else {
            // Print the error, cast err to awserr.Error to get the Code and
            // Message from an error.
            fmt.Println(err.Error())
        }
        return
    }

    fmt.Println(result)
```

```
}
```

를 **GetSendStatistics** 사용하여 호출 AWS SDK for PHP

```
<?php

// Replace path_to_sdk_inclusion with the path to the SDK as described in
// http://docs.aws.amazon.com/aws-sdk-php/v3/guide/getting-started/basic-usage.html
define('REQUIRED_FILE','path_to_sdk_inclusion');

// Replace us-west-2 with the AWS Region you're using for Amazon SES.
define('REGION','us-west-2');

require REQUIRED_FILE;

use Aws\Ses\SesClient;

$client = SesClient::factory(array(
    'version'=> 'latest',
    'region' => REGION
));

try {
     $result = $client->getSendStatistics([]);
  echo($result);
} catch (Exception $e) {
     echo($e->getMessage()."\n");
}

?>
```

를 **GetSendStatistics** 사용하여 호출 AWS SDK for Python (Boto)

```
import boto3 #pip install boto3
import json
from botocore.exceptions import ClientError

client = boto3.client('ses')

try:
    response = client.get_send_statistics(
)
```

```
except ClientError as e:
    print(e.response['Error']['Message'])
else:
    print(json.dumps(response, indent=4, sort_keys=True, default=str))
```

를 **GetSendStatistics** 사용하여 호출 AWS SDK for Ruby

```
require 'aws-sdk' # gem install aws-sdk
require 'json'

# Replace us-west-2 with the AWS Region you're using for Amazon SES.
awsregion = "us-west-2"

# Create a new SES resource and specify a region
ses = Aws::SES::Client.new(region: awsregion)

begin

  resp = ses.get_send_statistics({
  })
  puts JSON.pretty_generate(resp.to_h)

# If something goes wrong, display an error message.
rescue Aws::SES::Errors::ServiceError => error
  puts error

end
```

Amazon SES 이벤트 게시를 사용하여 이메일 전송 모니터링

이메일 전송을 세부적으로 추적하기 위해 정의한 특성에 따라 Amazon CloudWatch, Amazon Data Firehose, Amazon Pinpoint, Amazon Simple Notification Service 또는 Amazon EventBridge에 이메일 전송 이벤트를 게시할 수 있도록 Amazon SES를 설정할 수 있습니다.

발신, 전송, 열기, 클릭, 반송, 수신 거부, 거부, 렌더링 실패 및 전송 지연을 포함하는 다양한 이메일 전송 이벤트 유형을 추적할 수 있습니다. 이 정보는 운영 및 분석에 유용할 수 있습니다. 예를 들어 이메일 전송 데이터를 CloudWatch에 게시하고 이메일 캠페인의 성과를 추적하는 대시보드를 만들거나 Amazon SNS를 사용해 특정 이벤트가 발생할 때 알림을 전송할 수 있습니다.

이벤트 게시가 구성 세트 및 메시지 태그와 작동하는 방법

이벤트 게시를 사용하려면 우선 하나 이상의 구성 세트를 설정해야 합니다. 구성 세트는 어떤 이벤트를 어디에 게시할지 지정합니다. 그런 다음 이메일을 보낼 때마다 구성 세트의 이름과 하나 이상의 메시지 태그를 이름/값 페어 형식으로 제공하여 이메일을 분류합니다. 예를 들어 책을 광고하는 경우, 메시지 태그에 genre라는 이름을 붙이고 sci-fi 또는 western 값을 할당할 수 있습니다.

사용 중인 이메일 전송 인터페이스에 따라 메시지 태그를 `SendEmail` API 작업 `EmailTags` 필드에 파라미터로 제공하거나 SES별 이메일 헤더 `X-SES-MESSAGE-TAGS`에 메시지 태그를 추가합니다. 구성 세트에 대한 자세한 내용은 Amazon SES에서 구성 세트 사용 단원을 참조하세요.

SES는 지정하는 메시지 태그 외에도 전송 메시지에 자동 태그도 추가합니다. 자동 태그를 사용하기 위해 추가로 실행해야 할 단계도 없습니다.

다음 테이블은 SES를 사용하여 전송하는 메시지에 자동으로 적용되는 자동 태그를 나열한 것입니다.

SES 자동 태그

자동 태그 이름	설명
`ses:caller-identity`	이메일을 전송한 SES 사용자의 IAM ID입니다.
`ses:configuration-set`	이메일에 연결된 구성 세트의 이름
`ses:from-domain`	"From" 주소의 도메인
`ses:outgoing-ip`	SES가 이메일 전송에 사용한 IP 주소입니다.
`ses:source-ip`	호출자가 이메일 전송에 사용한 IP 주소.
`ses:source-tls-version`	발신자가 이메일을 보내는 데 사용한 TLS 프로토콜 버전입니다.
`ses:outgoing-tls-version`	SES가 이메일 전송에 사용한 TLS 프로토콜 버전입니다.

이메일 캠페인에 대한 세분화된 피드백

`ses:feedback-id-<a or b>` 태그는 하이브리드 또는 반자동 태그로 생각할 수 있는 선택적 메시지 태그입니다. 이전 섹션에서 다룬 자동 태그와 비슷하지만, 차이점은 수동으로 추가하고 `ses:` 접두

사 키를 사용해야 한다는 것입니다. `ses:feedback-id-a` 및 `ses:feedback-id-b`로 정의된 이러한 태그 중 최대 2개를 사용할 수 있습니다.

이러한 태그를 지정하면 SES는 피드백 루프(FBL)의 일부로 수신 거부, 스팸 비율 등의 전송 통계를 제공하는 데 사용되는 표준 `Feedback-ID` 헤더에 태그를 자동으로 추가합니다. 피드백 루프의 내용을 참조하세요. `Feedback-ID` 헤더는 SES에서 수신 거부 정보 수집에 사용하는 SESInternalID 식별자와 SES를 다음과 같은 전송 플랫폼으로 식별하는 AmazonSES 정적 태그로 구성됩니다.

```
FeedBackId:feedback-id-a:feedback-id-b:((SESInternalID):(AmazonSES))
```

이러한 선택적 피드백 ID 태그는 이메일 캠페인의 일부로 보내는 메시지와 같이 세분화된 피드백을 생성하는 방법으로 제공됩니다. 다음 예제와 같이 `SendEmail` 작업 요청의 `EmailTags` 필드에 메시지 태그로 지정하여 `ses:feedback-id-<a or b>`를 사용할 수 있습니다.

```
{
  "FromEmailAddress": "noreply@example.com",
  "Destination": {
    "ToAddresses": [
      "customer@example.net"
    ]
  },
  "Content": {
    "Simple": {
      "Subject": {
        "Data": "Hello and welcome"
      },
      "Body": {
        "Text": {
          "Data": "Lorem ipsum dolor sit amet."
        },
        "Html": {
          "Data": "Lorem ipsum dolor sit amet."
        }
      }
    }
  },
  "EmailTags": [
    {
      "Name": "ses:feedback-id-a",
      "Value": "new-members-campaign"
    },
    {
      "Name": "ses:feedback-id-b",
```

```
      "Value": "football-campaign"
    }
  ],
  "ConfigurationSetName": "football-club"
}
```

원시 형식으로 전송하는 경우 SES별 헤더 X-SES-MESSAGE-TAGS에 메시지 태그로 ses:feedback-id-<*a* or *b*>를 추가하면 됩니다.

ses:feedback-id-<*a* or *b*> 메시지 태그는 다른 메시지 태그와 마찬가지로 CloudWatch 값 소스로 지정하여 Amazon CloudWatch에서 추적할 수도 있습니다. the section called "CloudWatch 이벤트 대상 세부 정보 추가"의 내용을 참조하세요(추가 요금이 적용되며, CloudWatch의 지표당 요금 참조).

이벤트 게시 사용 방법

다음 섹션에는 SES 이벤트 게시를 설정하고 사용하는 데 필요한 정보가 포함되어 있습니다.

- 이벤트 게시 설정
- 이벤트 데이터 작업

이벤트 게시 용어

다음 목록은 SES 이벤트 게시에 관련된 용어를 정의한 것입니다.

이메일 전송 이벤트

SES에 제출한 이메일의 결과와 관련된 정보입니다. 전송 이벤트에는 다음이 포함됩니다.

- 전송(Send) - 전송 요청이 성공했으며 Amazon SES는 메시지를 수신자의 메일 서버로 전송합니다. (계정 수준 또는 전역 금지를 사용하는 경우 SES가 여전히 전송으로 계산하지만 배달은 금지합니다.)
- 렌더링 오류 – 템플릿 렌더링 문제로 인해 이메일이 전송되지 않았습니다. 이 이벤트 유형은 템플릿 데이터가 누락되었을 때 또는 템플릿 파라미터와 데이터 사이에 불일치가 있을 때 발생할 수 있습니다. (이 이벤트 유형은 SendTemplatedEmail 또는 SendBulkTemplatedEmail API 작업을 사용하는 이메일을 전송할 때만 발생합니다.)
- 거부(Reject) - Amazon SES가 이메일을 수락했으나 이메일에 바이러스가 포함된 것으로 판단되어 수신자의 메일 서버로 전송하려고 시도하지 않았습니다.
- 배달(Delivery) - Amazon SES에서 이메일을 수신자의 메일 서버로 성공적으로 전송했습니다.

- 바운스 – 수신자의 메일 서버가 이메일을 영구적으로 거부하는 하드 바운스입니다. (소프트 바운스는 SES가 더 이상 이메일 전송을 재시도하지 않는 경우에만 포함됩니다. 일반적으로 이러한 소프트 바운스는 전송 실패를 나타내지만, 메일이 수신자의 받은 편지함에 도착하더라도 소프트 바운스가 반환되는 경우도 있습니다. 이러한 상황은 일반적으로 수신자가 부재 중 자동 회신을 보낼 때 발생합니다. AWS re:Post 문서에서 소프트 바운스에 대해 자세히 알아보세요.)
- 수신 거부(Complaint) - 이메일이 수신자의 메일 서버로 성공적으로 전송되었지만 수신자가 이를 스팸으로 표시했습니다.
- 배달 지연 – 일시적인 문제가 발생하여 수신자의 메일 서버에 이메일을 전송할 수 없습니다. 예를 들어 수신자의 받은 편지함이 가득 찼거나 이메일 수신 서버에 일시적인 문제가 발생했을 때 전송 지연이 발생할 수 있습니다.
- 구독(Subscription) - 이메일이 성공적으로 배달되었지만 수신자가 이메일 헤더에서 `List-Unsubscribe`를 클릭하거나, 바닥글에서 `Unsubscribe` 링크를 선택하여 구독 기본 설정을 업데이트했습니다.
- 열기(Open) – 수신자가 메시지를 수신하여 자신의 이메일 클라이언트에서 열었습니다.
- 클릭(Click) – 수신자가 이메일의 링크를 1개 이상 클릭했습니다.

구성 세트

SES에서 이메일 전송 이벤트를 게시할 대상과 게시하려는 이메일 전송 이벤트 유형을 정의하는 규칙 세트입니다. 이벤트 게시와 함께 사용하려는 이메일을 전송할 때 이메일에 연결할 구성 세트를 지정합니다.

이벤트 대상

SES 이메일 전송 이벤트를 게시하는 AWS 서비스입니다. 설정하는 각 이벤트 대상은 단 하나의 구성 세트에만 속합니다.

메시지 태그

이벤트 게시를 목적으로 이메일을 분류하는 데 사용하는 이름/값 페어입니다. 예를 들면 campaign/book, campaign/clothing의 형식입니다. 이메일을 전송할 때 API 직접 호출에 대한 파라미터 또는 SES별 이메일 헤더로 메시지 태그를 지정합니다.

자동 태그

이벤트 게시 보고서에 자동으로 포함되는 메시지 태그입니다. 구성 세트 이름, 'From' 주소의 도메인, 호출자의 발신 IP 주소, SES 발신 IP 주소, 호출자의 IAM ID를 표시하는 자동 태그가 있습니다.

Amazon SES를 사용하여 이벤트 게시 설정

이 섹션에서는 다음 AWS 서비스에 이메일 전송 이벤트를 게시하도록 Amazon SES를 구성하기 위해 수행해야 하는 작업에 대해 설명합니다.

- Amazon CloudWatch
- Amazon Data Firehose
- Amazon Pinpoint
- Amazon Simple Notification Service(Amazon SNS)

이벤트 게시 설정에 필요한 다음 단계는 아래 주제에서 다룹니다.

1. 먼저 Amazon SES 콘솔 또는 API를 사용하여 구성 세트를 생성해야 합니다.
2. 구성 세트에 하나 이상의 이벤트 대상(CloudWatch, Firehose, Pinpoint 또는 SNS)을 추가하고 이벤트 대상에 고유한 파라미터를 구성합니다.
3. 이메일을 전송할 때 이벤트 대상을 포함하여 사용할 구성 세트를 지정합니다.

이 섹션의 주제

- 1단계: 구성 세트 만들기
- 2단계: 이벤트 대상 추가
- 3단계: 이메일을 전송할 때 구성 세트 지정

1단계: 구성 세트 만들기

이벤트 게시를 설정하려면 먼저 구성 세트가 있어야 합니다. 아직 구성 세트가 없거나 새 구성 세트를 만들려는 경우SES에서 구성 세트 생성 단원을 참조하십시오.

또한 Amazon SES CLI v2를 사용하거나 Amazon SES API V2에서 CreateConfigurationSet 작업을 수행하여 구성 세트를 생성할 수 있습니다. 구성 세트를 만듭니다(AWS CLI). 단원을 참조하십시오.

2단계: 이벤트 대상 추가

이벤트 대상은 Amazon SES 이벤트를 게시하는 위치입니다. 설정하는 각 이벤트 대상은 단 하나의 구성 세트에만 속합니다. Amazon SES로 이벤트 대상을 설정할 때 AWS 서비스 대상을 선택하고 해당 대상과 연결된 파라미터를 지정합니다.

이벤트 대상을 설정할 때 다음 AWS 서비스 중 하나로 이벤트를 보내도록 선택할 수 있습니다.

- Amazon CloudWatch
- Amazon Data Firehose
- Amazon EventBridge
- Amazon Pinpoint
- Amazon Simple Notification Service(Amazon SNS)

여기서 선택하는 이벤트 대상은 이벤트에 대해 알고자 하는 세부 정보 수준과 이벤트 정보를 수신하는 방법에 따라 달라집니다. 각 이벤트 유형의 누계 정보만 필요한 경우에는(예: 총계가 너무 높을 경우 경보를 설정할 수 있도록) CloudWatch를 사용합니다.

Amazon OpenSearch Service 또는 Amazon Redshift 등의 다른 서비스에 출력할 수 있는 상세한 이벤트 레코드로 분석하려는 경우에는 Firehose를 선택합니다.

특정 이벤트 발생 시 알림을 수신하려면 Amazon SNS를 선택합니다.

이 섹션은 다음 주제를 포함합니다.

- 이벤트 게시에 필요한 CloudWatch 이벤트 대상 설정
- Amazon SES 이벤트 게시를 위한 Data Firehose 이벤트 대상 설정
- 이벤트 게시에 필요한 Amazon EventBridge 대상 설정
- 이벤트 게시에 필요한 Amazon Pinpoint 이벤트 대상 설정
- 이벤트 게시에 필요한 Amazon SNS 이벤트 대상 설정

이벤트 게시에 필요한 CloudWatch 이벤트 대상 설정

Amazon CloudWatch 지표를 통해 이벤트 대상을 사용하여 Amazon SES 이메일 전송 이벤트를 CloudWatch로 게시할 수 있습니다. CloudWatch 이벤트 대상은 구성 세트에서만 설정할 수 있으므로, 먼저 구성 세트를 만든 다음 이벤트 대상을 구성 세트에 추가해야 합니다.

구성 세트에 CloudWatch 이벤트 대상을 추가할 경우에는 이메일 전송 시 사용하는 메시지 태그와 일치하는 CloudWatch 차원을 하나 이상 선택해야 합니다. 메시지 태그와 마찬가지로 CloudWatch 차원은 지표를 고유하게 식별하는 데 도움을 주는 이름/값 페어입니다.

예컨대 `campaign`이라는 메시지 태그와 크기를 이메일 캠페인 식별에 사용할 수 있습니다. 이메일 전송 이벤트를 CloudWatch에 게시할 때 메시지 태그와 차원을 선택하는 것이 중요합니다. 이 선택이

CloudWatch 결제 금액에 영향을 미치고 CloudWatch 내에서 이메일 전송 이벤트 데이터를 필터링하는 방식을 결정하기 때문입니다.

이 섹션에서는 차원을 선택하는 데 도움이 되는 정보를 제공한 다음, CloudWatch 이벤트 대상을 구성 집합에 추가하는 방법을 보여줍니다.

이 섹션의 주제

- CloudWatch 이벤트 대상 추가
- CloudWatch 차원 선택

CloudWatch 이벤트 대상 추가

이 섹션의 절차에서는 구성 세트에 CloudWatch 이벤트 대상 세부 정보를 추가하는 방법을 보여주며 이벤트 대상 만들기 섹션의 1~6단계를 완료했다고 가정합니다.

뿐만 아니라 Amazon SES API V2에서 UpdateConfigurationSetEventDestination 작업을 사용하여 이벤트 대상을 만들고 수정할 수 있습니다.

콘솔을 사용하여 구성 세트에 CloudWatch 이벤트 대상을 추가하려면

1. 다음은 7단계에서 이벤트 대상 유형으로 CloudWatch를 선택하기 위한 자세한 지침입니다. 이벤트 대상 만들기의 이전 단계를 모두 완료했다고 가정합니다. CloudWatch 대상 유형을 선택하고 대상 이름을 입력한 후 이벤트 게시를 활성화하면 Amazon CloudWatch 차원 패널이 표시됩니다. 해당 필드는 다음 단계에서 처리됩니다. (추가 요금이 적용되며, CloudWatch의 지표당 요금 참조)
2. Value Source(값 소스)에 Amazon SES가 CloudWatch로 전달되는 데이터를 가져오는 방법을 지정합니다. 다음과 같은 값 원본을 사용할 수 있습니다.

   - Message Tag(메시지 태그) – Amazon SES가 `X-SES-MESSAGE-TAGS` 헤더 또는 `EmailTags` API 파라미터를 사용하여 지정한 태그에서 차원 이름과 값을 검색합니다. 메시지 태그 사용에 대한 자세한 내용은 the section called “3단계: 전송할 때 구성 세트 지정” 섹션을 참조하세요.

     > **Note**
     >
     > 메시지 태그에는 숫자 0-9, 문자 A-Z(대문자 및 소문자 모두 가능), 하이픈(-), 밑줄(_)이 포함될 수 있습니다.

     Message Tag(메시지 태그) 값 소스를 사용하여 Amazon SES 자동 태그에 따라 차원을 만들 수도 있습니다. 자동 태그를 사용하려면 자동 태그의 전체 이름을 Dimension Name(차원 이름)

으로 입력합니다. 예를 들어, 구성 집합 자동 태그를 기반으로 차원을 생성하려면 차원 이름에 `ses:configuration-set`, 기본값에 구성 세트의 이름을 지정합니다. 자동 태그의 전체 목록은 이벤트 게시가 구성 세트 및 메시지 태그와 작동하는 방법 섹션을 참조하세요.

- Email Header(이메일 헤더) – Amazon SES가 이메일 헤더에서 차원 이름과 값을 검색합니다.

> **Note**
>
> 다음 Dimension Name(차원 이름)을 이메일 제목으로 사용할 수 없습니다. `Received`, `To`, `From`, `DKIM-Signature`, `CC`, `message-id` 또는 `Return-Path`.

- Link Tag(링크 태그) – Amazon SES가 링크에 지정한 태그에서 차원 이름과 값을 검색합니다. 링크에 태그를 추가하는 방법에 대한 자세한 내용은 링크에 고유 식별자로 태그를 지정할 수 있습니까? 섹션을 참조하세요.

3. Dimension Name(차원 이름)에 CloudWatch로 전달하려는 차원 이름을 입력합니다.

> **Note**
>
> 차원 이름에는 ASCII 문자(a-z, A-Z), 숫자(0-9), 밑줄(_) 및 대시(-)만 포함할 수 있습니다. 공백, 억양 표시가 되어 있는 문자, 비라틴 문자 및 기타 특수 문자는 허용되지 않습니다.

4. Default Value(기본값)에 차원 값을 입력합니다.

> **Note**
>
> 차원 값에는 ASCII 문자(a-z, A-Z), 숫자(0-9), 밑줄(_), 대시(-), @ 기호 및 마침표(.)만 포함할 수 있습니다. 공백, 억양 표시가 되어 있는 문자, 비라틴 문자 및 기타 특수 문자는 허용되지 않습니다.

5. 차원을 더 추가하려면 차원 추가를 선택합니다. 그렇지 않은 경우 다음을 선택합니다.
6. 검토 화면에서 이벤트 대상 정의 방식에 만족하면 Add destination(대상 추가)를 선택합니다.

CloudWatch 차원 선택

CloudWatch 차원으로 사용할 이름 및 값을 선택할 때는 다음 요소를 고려해야 합니다.

- Price per metric(지표당 가격) – CloudWatch에서는 기본 Amazon SES 지표를 무료로 볼 수 있습니다. 그러나 이벤트 게시를 사용하여 지표를 수집할 경우 CloudWatch 세부 모니터링 비용이 발생합니다. 이벤트 유형, 차원 이름, 차원 값으로 구성되는 고유한 조합마다 CloudWatch 내에 서로 다른

지표를 만듭니다. CloudWatch, 세부 모니터링을 사용하는 경우 각 지표에 대해 요금이 부과됩니다. 이런 이유로, 다수의 서로 다른 값이 필요할 수 있는 크기를 선택하기 꺼려질 수 있습니다. 예를 들어 "발신" 도메인별로 이메일 전송 이벤트를 추적하는 데 큰 관심을 두지 않는 한, Amazon SES 자동 태그 `ses:from-domain`의 차원을 정의하지 않는 편이 낫습니다. 수많은 값이 필요할 수 있기 때문입니다. 자세한 내용은 CloudWatch 요금을 참조하세요.

- Metric filtering(지표 필터링) – 지표에 여러 개의 차원이 있으면 CloudWatch에서 각 차원에 따라 개별적으로 지표에 액세스할 수 없습니다. 따라서 하나의 CloudWatch 이벤트 대상에 두 개 이상의 대상을 추가하기 전에는 심사숙고해야 합니다. 예를 들어 `campaign` 지표와 `campaign` + `genre` 조합의 지표가 필요한 경우, 두 개의 이벤트 대상을 추가해야 합니다. 하나는 `campaign`만 차원으로 하고, 또 하나는 `campaign`과 `genre`를 차원으로 해야 합니다.
- Dimension value source(차원 값 소스) – Amazon SES지정 헤더 또는 API에 대한 파라미터를 사용하여 차원 값을 지정하는 대신 사용자 고유의 MIME 메시지 헤더에서 Amazon SES가 차원 값을 가져오도록 선택할 수도 있습니다. 사용자 지정 헤더를 이미 사용 중이고, 헤더 값을 기반으로 지표를 수집하기 위해 이메일이나 이메일 전송 API에 대한 호출을 변경하고 싶지 않은 경우에 이 옵션을 사용할 수 있습니다. Amazon SES 이벤트 게시에 사용자 고유의 MIME 메시지 헤더를 사용하는 경우 Amazon SES 이벤트 게시에 사용하는 헤더 이름과 값은 문자 A-Z, 숫자 0-9, 밑줄(_), @ 기호, 하이픈(-) 및 마침표(.)만 포함할 수 있습니다. 다른 문자가 들어 있는 이름이나 값을 지정하면 이메일 전송 호출은 성공하겠지만 이벤트 지표가 Amazon CloudWatch로 전송되지 않습니다.

CloudWatch 개념에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch 개념을 참조하세요.

Amazon SES 이벤트 게시를 위한 Data Firehose 이벤트 대상 설정

Amazon Data Firehose 이벤트 대상은 특정 Amazon SES 이메일 전송 이벤트를 Firehose로 게시하는 엔터티를 나타냅니다. Firehose 이벤트 대상은 구성 세트에서만 설정할 수 있으므로, 먼저 구성 세트를 만들어야 합니다. 그런 다음 구성 세트에 이벤트 대상을 추가합니다.

이 섹션의 절차에서는 구성 세트에 Firehose 이벤트 대상 세부 정보를 추가하는 방법을 보여주며 이벤트 대상 만들기 섹션의 1~6단계를 완료했다고 가정합니다.

뿐만 아니라 Amazon SES API V2 대상에서 UpdateConfigurationSetEventDestination 작업을 사용하여 이벤트 대상을 만들고 업데이트할 수 있습니다.

콘솔을 사용하여 구성 세트에 Firehose 이벤트 대상 세부 정보를 추가하려면

1. 다음은 7단계에서 이벤트 대상 유형으로 Firehose를 선택하기 위한 자세한 지침입니다. 이벤트 대상 만들기의 이전 단계를 모두 완료했다고 가정합니다. Firehose 대상 유형을 선택하고 대상 이

름을 입력한 후 이벤트 게시를 활성화하면 Amazon Data Firehose 전송 스트림 패널이 표시됩니다. 해당 필드는 다음 단계에서 처리됩니다.

2. 전송 스트림에서 기존 Firehose 전송 스트림을 선택하거나 새 스트림 만들기를 선택하여 Firehose 콘솔을 통해 새 스트림을 만듭니다.

   Firehose 콘솔을 사용하여 스트림을 만드는 방법에 대한 내용은 Amazon Data Firehose 개발자 안내서에서 Amazon Kinesis Firehose 전송 스트림 만들기를 참조하세요.

3. Identity and Access Management(IAM) 역할에 대해 Amazon SES가 사용자를 대신하여 Firehose에 게시할 수 있는 권한을 가진 IAM 역할을 선택합니다. 기존 역할을 선택하거나 Amazon SES가 역할을 대신 만들도록 설정하거나 직접 역할을 만들 수 있습니다.

   기존 역할을 선택하거나 직접 역할을 만드는 경우 역할의 정책을 수동으로 변경하여 Firehose 전송 스트림에 대한 액세스 권한을 역할에 부여하고 Amazon SES에 그 역할을 담당할 권한을 부여해야 합니다. 예시 정책은 Firehose 전송 스트림에 게시할 수 있는 권한을 Amazon SES에 부여하기 섹션을 참조하세요.

4. 다음을 선택합니다.
5. 검토 화면에서 이벤트 대상 정의 방식에 만족하면 Add destination(대상 추가)를 선택합니다.

`UpdateConfigurationSetEventDestination` API를 사용하여 Firehose 이벤트 대상을 추가하는 방법에 대한 내용은 Amazon Simple Email Service API 참조를 참조하세요.

Firehose 전송 스트림에 게시할 수 있는 권한을 Amazon SES에 부여하기

Amazon SES가 Firehose 전송 스트림에 레코드를 게시하도록 하려면 AWS Identity and Access Management (IAM) 역할을 이용하여 역할의 권한 정책과 신뢰 정책을 연결하거나 변경해야 합니다. 권한 정책은 역할이 Firehose 전송 스트림에 레코드를 게시하도록 허용하고 신뢰 정책은 Amazon SES가 그 역할을 담당하도록 허용합니다.

이 섹션에서는 두 정책의 예를 제공합니다. IAM 역할에 정책을 연결하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 역할 변경을 참조하세요.

권한 정책

다음 권한 정책은 역할이 Firehose 전송 스트림에 데이터 레코드를 게시하도록 허용합니다.

JSON

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Action": [
                "firehose:PutRecordBatch"
            ],
            "Resource": [
                "arn:aws:firehose:us-east-1:111122223333:deliverystream/delivery-
stream-name"
            ]
        }
    ]
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- *delivery-region*을 Firehose 전송 스트림을 생성한 AWS 리전으로 바꿉니다.
- *111122223333*을 AWS 계정 ID로 바꿉니다.
- *delivery-stream-name*을 Firehose 전송 스트림의 이름으로 바꿉니다.

신뢰 정책

다음 신뢰 정책은 Amazon SES가 역할을 담당하도록 허용합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ses.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
```

```
            "AWS:SourceAccount": "111122223333",
            "AWS:SourceArn": "arn:aws:ses:delivery-
region:111122223333:configuration-set/configuration-set-name"
          }
        }
      }
    ]
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- *delivery-region*을 Firehose 전송 스트림을 생성한 AWS 리전으로 바꿉니다.
- *111122223333*을 AWS 계정 ID로 바꿉니다.
- *configuration-set-name*을 Firehose 전송 스트림과 연결된 구성 세트 이름으로 바꿉니다.

이벤트 게시에 필요한 Amazon EventBridge 대상 설정

Amazon EventBridge 이벤트 대상은 구성 세트에서 지정한 이메일 전송 이벤트에 대해 알려줍니다. SES는 이벤트 대상을 만들 때 정의한 이메일 전송 이벤트를 생성하여 EventBridge 기본 이벤트 버스로 전송합니다. 이벤트 버스는 이벤트를 수신하여 여러 대상으로 전달할 수 있는 라우터입니다. EventBridge를 사용하여 모니터링에서 이메일 전송 이벤트를 Amazon EventBridge와 통합하는 방법에 대해 자세히 알아볼 수 있습니다. EventBridge 이벤트 대상은 구성 세트에서만 설정할 수 있으므로, 구성 세트에 이벤트 대상을 추가하기 전에 구성 세트를 만들어야 합니다.

이 섹션의 절차에서는 구성 세트에 EventBridge 이벤트 대상 세부 정보를 추가하는 방법을 보여주며 이벤트 대상 만들기 섹션의 1~6단계를 완료했다고 가정합니다.

뿐만 아니라 Amazon SES API V2에서 UpdateConfigurationSetEventDestination 작업을 사용하여 이벤트 대상을 만들고 수정할 수 있습니다.

콘솔을 사용하여 구성 세트에 EventBridge 이벤트 대상 세부 정보를 추가하려면

1. 다음은 7단계에서 이벤트 대상 유형으로 EventBridge를 선택하기 위한 자세한 지침입니다. 이벤트 대상 만들기의 이전 단계를 모두 완료했다고 가정합니다. Amazon EventBridge 대상 유형을 선택하고, 대상 이름을 입력하고, 이벤트 게시를 활성화하면 Amazon EventBridge 이벤트 버스 정보 패널이 표시됩니다.
2. 다음을 선택합니다.

3. 검토 화면에서 이벤트 대상 정의 방식에 만족하면 Add destination(대상 추가)를 선택합니다. 그러면 이벤트 대상의 요약 페이지가 열리고 거기서 성공 배너를 통해 이벤트 대상이 성공적으로 만들어지거나 수정되었음을 확인할 수 있습니다.

이벤트 게시에 필요한 Amazon Pinpoint 이벤트 대상 설정

Amazon Pinpoint 이벤트 대상은 구성 세트에서 지정한 이메일 전송 이벤트에 대해 알려줍니다. Amazon Pinpoint 이벤트 대상은 구성 세트에서만 설정할 수 있으므로, 구성 세트에 이벤트 대상을 추가하기 전에 구성 세트를 만들어야 합니다.

이 섹션의 절차에서는 구성 세트에 Amazon Pinpoint 이벤트 대상 세부 정보를 추가하는 방법을 보여주며 이벤트 대상 만들기 섹션의 1~6단계를 완료했다고 가정합니다.

뿐만 아니라 Amazon SES API V2에서 UpdateConfigurationSetEventDestination 작업을 사용하여 이벤트 대상을 만들고 수정할 수 있습니다.

Amazon Pinpoint 프로젝트에서 구성한 채널 유형에 대한 추가 요금이 있습니다. 자세한 내용은 Amazon Pinpoint 요금 섹션을 참조하세요.

콘솔을 사용하여 구성 집합에 Amazon Pinpoint 이벤트 대상을 추가하려면

1. 다음은 7단계에서 이벤트 대상 유형으로 Amazon Pinpoint를 선택하기 위한 자세한 지침입니다. 이벤트 대상 만들기의 이전 단계를 모두 완료했다고 가정합니다.

   > **Note**
   >
   > Amazon Pinpoint는 배달 지연 또는 구독 이벤트 유형은 지원하지 않습니다.

   Amazon Pinpoint 대상 유형을 선택하고 대상 이름을 입력한 후 이벤트 게시를 활성화하면 Amazon Pinpoint 프로젝트 세부 정보 패널이 표시됩니다. 해당 필드는 다음 단계에서 처리됩니다.

2. 프로젝트(Project)에서는 기존 Amazon Pinpoint 프로젝트를 선택하거나 Amazon Pinpoint에서 새 프로젝트 만들기(Create a new project in Amazon Pinpoin)를 선택하여 새 프로젝트를 생성합니다.

   프로젝트 생성에 대한 자세한 내용은 Amazon Pinpoint 사용 설명서의 프로젝트 만들기를 참조하세요.

3. 다음을 선택합니다.

4. 검토 화면에서 이벤트 대상 정의 방식에 만족하면 Add destination(대상 추가)를 선택합니다. 그러면 이벤트 대상의 요약 페이지가 열리고 거기서 성공 배너를 통해 이벤트 대상이 성공적으로 만들어지거나 수정되었음을 확인할 수 있습니다.

이벤트 게시에 필요한 Amazon SNS 이벤트 대상 설정

Amazon SNS 이벤트 대상은 구성 세트에서 지정한 이메일 전송 이벤트에 대해 알려줍니다. Amazon SNS 이벤트 대상은 구성 세트에서만 설정할 수 있으므로, 구성 세트에 이벤트 대상을 추가하기 전에 구성 세트를 만들어야 합니다.

이 섹션의 절차에서는 구성 세트에 Amazon SNS 이벤트 대상 세부 정보를 추가하는 방법을 보여주며 이벤트 대상 만들기 섹션의 1~6단계를 완료했다고 가정합니다.

뿐만 아니라 Amazon SES API V2에서 UpdateConfigurationSetEventDestination 작업을 사용하여 이벤트 대상을 만들고 수정할 수 있습니다.

> **Note**
>
> 확인된 모든 전송 보안 인증과 관련해 Amazon SNS에서 반송 메일, 수신 거부 및 전달에 대한 피드백 알림을 설정할 수도 있습니다. 자세한 내용은 the section called “Amazon SNS 알림 구성” 섹션을 참조하세요.

Amazon SNS 주제를 구독하는 엔드포인트로 메시지를 보내는 경우 추가 요금이 부과됩니다. 자세한 내용은 Amazon SNS 요금을 참조하세요.

콘솔을 사용하여 구성 집합에 Amazon SNS 이벤트 대상을 추가하려면

1. 다음은 7단계에서 이벤트 대상 유형으로 Amazon SNS를 선택하기 위한 자세한 지침입니다. 이벤트 대상 만들기의 이전 단계를 모두 완료했다고 가정합니다. Amazon SNS 대상 유형을 선택하고 대상 이름을 입력한 후 이벤트 게시를 활성화하면 Amazon Simple Notification Service(Amazon SNS) 주제 패널이 표시됩니다. 해당 필드는 다음 단계에서 처리됩니다.
2. SNS topic(SNS 주제)에서 기존 Amazon SNS 주제를 선택하거나 Create SNS topic(SNS 주제 만들기)을 선택하여 새로운 주제를 생성합니다.

   자세한 내용은 Amazon Simple Notification Service 개발자 가이드의 주제 만들기를 참조하세요.

> **Important**
>
> Amazon SNS를 사용하여 주제를 만들 때 유형으로 Standard(표준)만을 선택해야 합니다. (SES는 FIFO 유형 주제를 지원하지 않습니다.)

3. 다음을 선택합니다.
4. 검토 화면에서 이벤트 대상 정의 방식에 만족하면 Add destination(대상 추가)를 선택합니다. 그러면 이벤트 대상의 요약 페이지가 열리고 거기서 성공 배너를 통해 이벤트 대상이 성공적으로 만들어지거나 수정되었음을 확인할 수 있습니다.
5. 이제 새 SNS 주제를 만들든 기존 주제를 선택하든 관계없이 SES에 대한 액세스 권한을 부여하여 주제에 알림을 게시해야 합니다. 이전 단계의 이벤트 대상 요약 페이지에 있는 대상 유형 열에서 Amazon SNS를 선택합니다. 그러면 Amazon Simple Notification Service 콘솔의 주제(Topics) 목록이 나타납니다. Amazon SNS 콘솔에서 다음 단계를 수행합니다.
   a. 이전 단계에서 만들거나 수정한 SNS 주제의 이름을 선택합니다.
   b. 주제의 세부 정보 화면에서 편집(Edit)을 선택합니다.
   c. SES에 주제에 대한 알림을 게시할 수 있는 권한을 부여하려면 SNS 콘솔의 주제 편집 화면에서 액세스 정책을 확장하고 JSON 에디터에 다음 권한 정책을 추가합니다.

      JSON

```
{
    "Version": "2012-10-17",
    "Id": "notification-policy",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": "sns:Publish",
            "Resource": "arn:aws:sns:us-east-1:111122223333:topic_name",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "111122223333",
                    "AWS:SourceArn":
 "arn:aws:ses:topic_region:111122223333:configuration-set/configuration-
set-name"
```

```
                }
            }
        }
    ]
}
```

이전 정책 예제에서 다음과 같이 변경합니다.

- SNS 주제를 만드는 AWS 리전을 *topic_region*으로 교체합니다.
- *111122223333*을 AWS 계정 ID로 바꿉니다.
- *topic_name*을 SNS 주제의 이름으로 바꿉니다.
- *configuration-set-name*을 SNS 이벤트 대상과 연결된 구성 세트 이름으로 바꿉니다.

d. 변경 사항 저장을 선택합니다.

3단계: 이메일을 전송할 때 구성 세트 지정

구성 세트를 생성하고 이벤트 대상을 추가한 후 이벤트 게시의 마지막 단계는 이메일을 전송하는 것입니다.

이메일과 연결된 이벤트를 게시하려면 해당 이메일과 연결할 구성 세트 이름을 제공해야 합니다. 필요한 경우 메시지 태그를 제공하여 이메일을 분류할 수 있습니다.

이 정보를 이메일 전송 API에 대한 파라미터, Amazon SES 지정 이메일 헤더 또는 MIME 메시지 내 사용자 지정 헤더 중 하나로 Amazon SES에 제공합니다. 선택은 다음 표와 같이 사용 중인 이메일 전송 인터페이스에 따라 달라집니다.

이메일 전송 인터페이스	이벤트 게시 방법
`SendEmail`	API 파라미터
`SendTemplatedEmail`	API 파라미터
`SendBulkTemplatedEmail`	API 파라미터
`SendCustomVerificationEmail`	API 파라미터
`SendRawEmail`	API 파라미터, Amazon SES 지정 이메일 헤더 또는 사용자 지정 MIME 헤더

이메일 전송 인터페이스	이벤트 게시 방법
	Important 헤더와 API 파라미터를 모두 사용하여 메시지 태그를 지정하더라도 Amazon SES는 API 파라미터가 제공한 메시지 태그만 사용합니다. Amazon SES는 API 파라미터와 헤더로 지정된 메시지 태그를 조인하지 않습니다.
SMTP 인터페이스	Amazon SES 지정 이메일 헤더

다음 단원에서는 헤더 및 API 파라미터를 사용하여 구성 세트와 메시지 태그를 지정하는 방법에 대해 설명합니다.

- Amazon SES API 파라미터 사용
- Amazon SES 지정 이메일 헤더 사용
- 사용자 지정 이메일 헤더 사용

> **Note**
> 선택적으로 메시지 태그를 이메일의 헤더에 포함시킬 수 있습니다. 메시지 태그에는 숫자 0-9, 문자 A-Z(대문자 및 소문자 모두 가능), 하이픈(-), 밑줄(_)이 포함될 수 있습니다.

Amazon SES API 파라미터 사용

SendEmail, SendTemplatedEmail, SendBulkTemplatedEmail, SendCustomVerificationEmail 또는 SendRawEmail을 이벤트 게시와 함께 사용하려면 ConfigurationSet 및 MessageTag라는 데이터 구조를 API 호출로 전달하여 구성 세트와 메시지 태그를 지정합니다.

Amazon SES API 사용과 관련한 자세한 내용은 Amazon Simple Email Service API 참조를 참조하십시오.

Amazon SES 지정 이메일 헤더 사용

`SendRawEmail` 또는 SMTP 인터페이스를 사용하면 Amazon SES 지정 헤더를 이메일에 추가하여 구성 세트와 메시지 태그를 지정할 수 있습니다. Amazon SES는 이메일을 전송하기 전에 해당 이메일에서 헤더를 제거합니다. 다음 표에는 사용할 헤더의 이름이 나와 있습니다.

이벤트 게시 정보	헤더
구성 세트	`X-SES-CONFIGURATION-SET`
메시지 태그	`X-SES-MESSAGE-TAGS`

다음 예제는 Amazon SES에 제출하는 원시 이메일에서 헤더가 어떻게 표시되는지 보여줍니다.

```
X-SES-MESSAGE-TAGS: tagName1=tagValue1, tagName2=tagValue2
X-SES-CONFIGURATION-SET: myConfigurationSet
From: sender@example.com
To: recipient@example.com
Subject: Subject
Content-Type: multipart/alternative;
 boundary="----=_boundary"

------=_boundary
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 7bit

body
------=_boundary
Content-Type: text/html; charset=UTF-8
Content-Transfer-Encoding: 7bit

body
------=_boundary--
```

사용자 지정 이메일 헤더 사용

Amazon SES 지정 헤더 `X-SES-CONFIGURATION-SET`을(를) 사용하여 구성 세트 이름을 지정해야 하지만 사용자 고유의 MIME 헤더를 사용하여 메시지 태그를 지정할 수 있습니다.

> **Note**
>
> Amazon SES 이벤트 게시에 사용하는 헤더 이름과 값은 ASCII 형식이어야 합니다. Amazon SES 이벤트 게시에 ASCII가 아닌 헤더 이름이나 값을 지정하면 이메일 전송 호출은 성공하겠지만 이벤트 지표가 Amazon CloudWatch로 내보내지지 않습니다.

Amazon SES 이벤트 데이터 작업

이벤트 게시 설정을 완료하고 이메일 전송을 위한 구성 세트를 지정한 후, 이메일에 연결된 구성 세트를 설정할 때 지정한 이벤트 대상에서 이메일 전송 이벤트를 검색할 수 있습니다.

이 섹션에서는 Amazon CloudWatch 및 Amazon Data Firehose에서 이메일 전송 이벤트를 검색하는 방법과 Amazon SNS에서 제공한 이벤트 데이터를 해석하는 방법을 설명합니다.

- CloudWatch에서 Amazon SES 이벤트 데이터 가져오기
- Firehose에서 Amazon SES 이벤트 데이터 검색
- Amazon SNS의 Amazon SES 이벤트 데이터 해석

CloudWatch에서 Amazon SES 이벤트 데이터 가져오기

Amazon SES에서 이메일 전송 이벤트의 지표를 Amazon CloudWatch로 게시할 수 있습니다. CloudWatch에 이벤트 데이터를 게시하면 이러한 지표가 정렬된 시계열 데이터 세트로 제공됩니다. 이러한 지표를 사용해 이메일 전송 성능을 모니터링할 수 있습니다. 예를 들어 수신 거부 측정치를 모니터링하고, 측정치가 특정 값을 초과하면 CloudWatch 경보가 트리거되도록 설정할 수 있습니다.

Amazon SES는 다음 두 가지 세부 수준에서 이러한 이벤트를 CloudWatch에 게시할 수 있습니다.

- 전체 AWS 계정 - Amazon SES 콘솔 및 `GetSendStatistics` API를 사용하여 모니터링하는 지표에 해당하는 이러한 대략적인 지표는 전체의 합계입니다 AWS 계정. Amazon SES는 이러한 지표를 CloudWatch에 자동으로 게시합니다.
- Fine-grained(세부 지표) – 이 지표는 메시지 태그를 사용하여 정의하는 이메일 특성을 기준으로 분류됩니다. 이러한 지표를 CloudWatch에 게시하려면 CloudWatch 이벤트 대상과 함께 이벤트 게시를 설정하고 이메일을 전송할 때 구성 세트를 지정해야 합니다. 또한 메시지 태그를 지정하거나 Amazon SES가 자동으로 제공하는 자동 태그를 사용할 수 있습니다.

이 섹션에서는 CloudWatch에서 사용할 수 있는 지표와 이 지표를 보는 방법에 대해 설명합니다.

사용 가능한 지표

다음과 같은 Amazon SES 이메일 전송 지표를 CloudWatch에 게시할 수 있습니다.

- 전송(Send) - 전송 요청이 성공했으며 Amazon SES는 메시지를 수신자의 메일 서버로 전송합니다. (계정 수준 또는 전역 금지를 사용하는 경우 SES가 여전히 전송으로 계산하지만 배달은 금지합니다.)
- 렌더링 오류 – 템플릿 렌더링 문제로 인해 이메일이 전송되지 않았습니다. 이 이벤트 유형은 템플릿 데이터가 누락되었을 때 또는 템플릿 파라미터와 데이터 사이에 불일치가 있을 때 발생할 수 있습니다. (이 이벤트 유형은 `SendTemplatedEmail` 또는 `SendBulkTemplatedEmail` API 작업을 사용하는 이메일을 전송할 때만 발생합니다.)
- 거부(Reject) - Amazon SES가 이메일을 수락했으나 이메일에 바이러스가 포함된 것으로 판단되어 수신자의 메일 서버로 전송하려고 시도하지 않았습니다.
- 배달(Delivery) - Amazon SES에서 이메일을 수신자의 메일 서버로 성공적으로 전송했습니다.
- 바운스 – 수신자의 메일 서버가 이메일을 영구적으로 거부하는 하드 바운스입니다. (소프트 바운스는 SES가 더 이상 이메일 전송을 재시도하지 않는 경우에만 포함됩니다. 일반적으로 이러한 소프트 바운스는 전송 실패를 나타내지만, 메일이 수신자의 받은 편지함에 도착하더라도 소프트 바운스가 반환되는 경우도 있습니다. 이러한 상황은 일반적으로 수신자가 부재 중 자동 회신을 보낼 때 발생합니다. AWS re:Post 문서에서 소프트 바운스에 대해 자세히 알아보세요.)
- 수신 거부(Complaint) - 이메일이 수신자의 메일 서버로 성공적으로 전송되었지만 수신자가 이를 스팸으로 표시했습니다.
- 배달 지연 – 일시적인 문제가 발생하여 수신자의 메일 서버에 이메일을 전송할 수 없습니다. 예를 들어 수신자의 받은 편지함이 가득 찼거나 이메일 수신 서버에 일시적인 문제가 발생했을 때 전송 지연이 발생할 수 있습니다.
- 구독(Subscription) - 이메일이 성공적으로 배달되었지만 수신자가 이메일 헤더에서 `List-Unsubscribe`를 클릭하거나, 바닥글에서 `Unsubscribe` 링크를 선택하여 구독 기본 설정을 업데이트했습니다.
- 열기(Open) – 수신자가 메시지를 수신하여 자신의 이메일 클라이언트에서 열었습니다.
- 클릭(Click) – 수신자가 이메일의 링크를 1개 이상 클릭했습니다.

사용 가능한 크기

CloudWatch는 Amazon SES의 구성 세트에 CloudWatch 이벤트 대상을 추가할 때 지정하는 차원 이름을 사용합니다. 자세한 내용은 이벤트 게시에 필요한 CloudWatch 이벤트 대상 설정 섹션을 참조하세요.

CloudWatch 콘솔에서 Amazon SES 지표 보기

다음 절차에서는 CloudWatch 콘솔을 사용하여 Amazon SES 이벤트 게시 지표를 보는 방법을 설명합니다.

CloudWatch 콘솔을 사용하여 지표를 보려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/cloudwatch/ CloudWatch 콘솔을 엽니다.
2. 필요한 경우 리전을 변경합니다. 탐색 모음에서 AWS 리소스가 있는 리전을 선택합니다. 자세한 내용은 리전 및 엔드포인트 섹션을 참조하세요.
3. 탐색 창에서 모든 지표를 선택합니다.
4. 지표 창에서 SES를 선택합니다.
5. 확인할 지표를 선택합니다. 세부적인 이벤트 게시 지표를 보려면 CloudWatch 이벤트 대상을 설정할 때 지정한 차원 조합을 선택합니다. CloudWatch를 사용하여 지표를 보는 방법에 대해 자세히 알아보려면 Amazon CloudWatch 지표 사용을 참조하세요.

를 사용하여 지표를 보려면 AWS CLI

- 명령 프롬프트에서 다음 명령을 사용합니다.

```
aws cloudwatch list-metrics --namespace "AWS/SES"
```

Firehose에서 Amazon SES 이벤트 데이터 검색

Amazon SES에서 이메일 전송 이벤트를 Firehose에 JSON 레코드로 게시합니다. 그러면 Firehose는 Firehose에서 전송 스트림을 설정할 때 선택한 AWS 서비스 대상에 레코드를 게시합니다. Firehose 전송 스트림 설정에 대한 자세한 내용은 Amazon Data Firehose 개발자 가이드의 Firehose 전송 스트림 생성을 참조하세요.

이 섹션의 주제:

- Amazon SES에서 Firehose에 게시하는 이벤트 데이터 콘텐츠
- Amazon SES에서 Firehose에 게시하는 이벤트 데이터 예제

Amazon SES에서 Firehose에 게시하는 이벤트 데이터 콘텐츠

Amazon SES에서 이메일 전송 이벤트 레코드를 Amazon Data Firehose에 JSON 형식으로 게시합니다. Firehose에 이벤트를 게시할 때 Amazon SES는 줄바꿈 문자가 포함된 각 JSON 레코드를 따릅니다.

이러한 모든 알림 유형에 대한 레코드 예는 Amazon SES에서 Firehose에 게시하는 이벤트 데이터 예제에서 찾을 수 있습니다.

이 섹션의 주제

- 최상위 JSON 객체
- Mail 객체
- Bounce 객체
- Complaint 객체
- 전송 객체
- Send 객체
- Reject 객체
- Open 객체
- Click 객체
- Rendering Failure 객체
- DeliveryDelay 객체
- 구독 객체

최상위 JSON 객체

이메일 전송 이벤트 레코드의 최상위 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`eventType`	이벤트 유형을 설명하는 문자열입니다. 가능한 값: `Bounce`, `Complaint` , `Delivery`, `Send`, `Reject`, `Open`, `Click`, `Rendering Failure`, `DeliveryDelay` 또는 `Subscription` 이벤트 게시를 설정하지 않은 경우 이 필드의 이름은 `notificationType` 입니다.

필드 이름	설명
mail	이벤트를 만드는 이메일 관련 정보를 포함하는 JSON 객체입니다.
bounce	이 필드는 eventType 이 Bounce인 경우에만 존재합니다. 이 파일에는 반송 관련 정보가 포함되어 있습니다.
complaint	이 필드는 eventType 이 Complaint 인 경우에만 존재합니다. 이 파일에는 수신 거부 관련 정보가 포함되어 있습니다.
delivery	이 필드는 eventType 이 Delivery인 경우에만 존재합니다. 이 파일에는 전송 관련 정보가 포함되어 있습니다.
send	이 필드는 eventType 이 Send인 경우에만 존재합니다.
reject	이 필드는 eventType 이 Reject인 경우에만 존재합니다. 이 파일에는 거부 관련 정보가 포함되어 있습니다.
open	이 필드는 eventType 이 Open인 경우에만 존재합니다. 이 파일에는 열기 이벤트 관련 정보가 포함되어 있습니다.
click	이 필드는 eventType 이 Click인 경우에만 존재합니다. 이 파일에는 클릭 이벤트 관련 정보가 포함되어 있습니다.
failure	이 필드는 eventType 이 Rendering Failure인 경우에만 존재합니다. 이 파일에는 렌더링 오류 이벤트 관련 정보가 포함되어 있습니다.

필드 이름	설명
`deliveryDelay`	이 필드는 `eventType` 이 `DeliveryDelay` 인 경우에만 존재합니다. 이메일 전송 지연에 대한 정보가 포함되어 있습니다.
`subscription`	이 필드는 `eventType` 이 `Subscription` 인 경우에만 존재합니다. 여기에는 구독 기본 설정에 대한 정보가 포함되어 있습니다.

Mail 객체

각 이메일 전송 이벤트 레코드는 `mail` 객체에 원래 메일에 대한 정보를 포함하고 있습니다. `mail` 객체 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`timestamp`	메시지가 전송된 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`messageId`	Amazon SES가 메시지에 할당한 고유 ID입니다. 메시지를 보낼 때 Amazon SES에서 이 값을 반환했습니다. **Note** 이 메시지 ID는 Amazon SES에서 할당한 것입니다. `mail` 객체의 `headers` 및 `commonHeaders` 필드에서 원래 메시지의 메시지 ID를 찾을 수 있습니다.
`source`	메시지를 전송한 이메일 주소(엔벌로프 MAIL FROM 주소).
`sourceArn`	이메일을 전송하는 데 사용된 ID의 Amazon 리소스 이름(ARN). 권한 부여 전송의 경우

필드 이름	설명
	`sourceArn` 은 ID 소유자가 위임 발신자에게 메일 전송 시 사용하도록 권한을 부여한 ID의 ARN입니다. 권한 부여 전송에 대한 자세한 내용은 이메일 인증 방법 섹션을 참조하세요.
`sendingAccountId`	이메일을 보내는 데 사용된 AWS 계정의 계정 ID입니다. 권한 부여 전송의 경우 `sendingAccountId` 는 위임 발신자의 계정 ID입니다.
`destination`	원래 메일의 수신자인 이메일 주소의 목록.
`headersTruncated`	알림에서 헤더가 잘렸는지 여부를 나타내는 문자열입니다. 헤더의 용량이 10KB를 초과할 경우 헤더가 잘립니다. 가능한 값은 `true` 및 `false`입니다.
`headers`	이메일의 원래 헤더 목록입니다. 목록의 각 헤더에는 `name` 필드와 `value` 필드가 존재합니다. **Note** `headers` 필드의 모든 메시지 ID는 Amazon SES에 전달한 원본 메시지에서 가져온 것입니다. 이어서 Amazon SES가 메시지에 할당한 메시지 ID는 `mail` 객체의 `messageId` 필드에 들어 있습니다.

필드 이름	설명
`commonHeaders`	자주 사용되는 원래 이메일 헤더의 매핑입니다. **Note** `commonHeaders` 필드의 모든 메시지 ID는 Amazon SES가 `mail` 객체의 `messageId` 필드 내 메시지에 할당한 메시지 ID입니다.
`tags`	이메일과 연결된 태그 목록입니다.

Bounce 객체

`Bounce` 이벤트에 대한 정보를 포함하는 JSON 객체에는 항상 다음 필드가 포함됩니다.

필드 이름	설명
`bounceType`	Amazon SES가 결정한 반송 메일의 유형.
`bounceSubType`	Amazon SES가 결정한 반송 메일의 하위 유형.
`bouncedRecipients`	반송된 원래 메일의 수신자 정보를 포함하는 목록.
`timestamp`	ISP가 반송 메일 알림을 전송한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`feedbackId`	반송 메일의 고유 ID.
`reportingMTA`	DSN의 `Reporting-MTA` 필드의 값. DSN에서 설명하는 전송, 중계 또는 게이트웨이 작업을 시도하는 메시지 전송 권한(MTA)의 값입니다.

필드 이름	설명
	Note 이 필드는 반송 메일에 전송 상태 알림(DSN)이 첨부된 경우에만 표시됩니다.

반송 수신자

반송 이벤트는 단일 수신자 또는 여러 수신자와 관련이 있을 수 있습니다. `bouncedRecipients` 필드는 객체(반송 메일 이벤트가 관련된 수신자당 객체 1개)의 목록을 포함하고 있으며 항상 다음 필드로 구성됩니다.

필드 이름	설명
`emailAddress`	수신자의 이메일 주소. DSN이 사용 가능할 경우, DSN의 `Final-Recipient` 필드의 값입니다.

또는 반송 메일에 DSN이 첨부된 경우 다음 필드도 존재할 수 있습니다.

필드 이름	설명
`action`	DSN의 `Action` 필드의 값. 이 수신자에게 메시지를 전송하려는 시도의 결과로 보고-MTA가 수행하는 작업을 나타냅니다.
`status`	DSN의 `Status` 필드의 값. 메시지의 전송 상태를 나타내는 수신자별 전송 독립적 상태 코드입니다.
`diagnosticCode`	보고-MTA가 발행한 상태 코드. DSN의 `Diagnostic-Code` 필드의 값입니다. 이 필드가 DSN에는 없을 수 있습니다(따라서 JSON에도 없음).

반송 메일 유형

각 반송 메일 이벤트는 다음 표에 나와 있는 유형 중 하나가 됩니다.

이벤트 게시 시스템은 Amazon SES에서 더 이상 재시도하지 않을 하드 바운스 및 소프트 바운스만 게시합니다. `Permanent`로 표시된 반송 메일을 받으면 메일 그룹에서 해당 이메일 주소를 삭제해야 하며 다음부터는 그러한 이메일 주소로 전송할 수 없습니다. `Transient` 반송 메일은 메시지가 여러 번 소프트 바운스되어 Amazon SES에서 이러한 메일에 대해 재전송 시도를 중지한 경우에 전송됩니다. 처음에 `Transient` 반송 메일이 발생한 주소로 이후 전송 재시도가 성공할 수도 있습니다.

bounceType	bounceSubType	설명
`Undetermined`	`Undetermined`	Amazon SES가 특정 반송 사유를 결정하지 못했습니다.
`Permanent`	`General`	Amazon SES가 일반 하드 바운스를 수신했습니다. 이 유형의 반송 메일을 받은 경우, 이 수신자의 이메일 주소를 메일 발송 목록에서 삭제해야 합니다.
`Permanent`	`NoEmail`	대상 이메일 주소가 존재하지 않아 Amazon SES가 영구 하드 바운스를 수신했습니다. 이 유형의 반송 메일을 받은 경우, 이 수신자의 이메일 주소를 메일 발송 목록에서 삭제해야 합니다.
`Permanent`	`Suppressed`	최근의 잘못된 주소로 인한 반송 이력 때문에 Amazon SES가 이 주소로 메일 전송을 금지했습니다. 전역 금지 목록을 재정의하려면 Amazon SES 계정 수준 금지 목록 사용 섹션을 참조하세요.
`Permanent`	`OnAccountSuppressionList`	계정 수준 금지 목록에 있으므로 Amazon SES가 이 주소로 보내는 것을 금지했습니다. 이는 반송 비율 지표에 반영되지 않습니다.
`Transient`	`General`	Amazon SES가 일반 반송 메일을 수신했습니다. 이후에 이 수신자에게 전송이 성공할 수도 있습니다.

bounceType	bounceSubType	설명
`Transient`	`MailboxFull`	Amazon SES가 메일박스 가득 참 반송 메일을 수신했습니다. 이후에 이 수신자에게 전송이 성공할 수도 있습니다.
`Transient`	`MessageTooLarge`	Amazon SES가 메시지 너무 큼 반송 메일을 수신했습니다. 메시지 크기를 줄일 경우 이 수신자에게 전송이 성공할 수도 있습니다.
`Transient`	`CustomTimeoutExceeded`	Amazon SES가 이메일 발신자가 지정한 시간 내에 이메일을 성공적으로 전송하지 못했습니다. (반송 메시지는 정의된 TTL 내에서 가능한 전송 시도 실패의 이유를 지정합니다.)
`Transient`	`ContentRejected`	Amazon SES가 내용 거부 반송 메일을 수신했습니다. 메시지 내용을 변경할 경우 이 수신자에게 전송이 성공할 수도 있습니다.
`Transient`	`AttachmentRejected`	Amazon SES가 첨부 파일 거부 반송 메일을 수신했습니다. 첨부 파일을 제거하거나 변경할 경우 이 수신자에게 전송이 성공할 수도 있습니다.

Complaint 객체

`Complaint` 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`complainedRecipients`	수신 거부를 제출했을 수 있는 수신자에 대한 정보를 포함하는 목록.
`timestamp`	ISP가 수신 거부 알림을 전송한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`feedbackId`	수신 거부의 고유 ID.

필드 이름	설명
`complaintSubType`	Amazon SES가 결정한 수신 거부의 하위 유형.

또한, 수신 거부에 피드백 보고서가 첨부된 경우 다음 필드가 포함될 수 있습니다.

필드 이름	설명
`userAgent`	피드백 보고서의 `User-Agent` 필드의 값입니다. 보고서를 생성한 시스템의 이름 및 버전을 나타냅니다.
`complaintFeedbackType`	ISP로부터 수신된 피드백 보고서의 `Feedback-Type` 필드의 값. 이 값은 피드백의 유형을 포함합니다.
`arrivalDate`	피드백 보고서의 `Arrival-Date` 또는 `Received-Date` 필드 값으로 ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)입니다. 이 필드가 보고서에는 없을 수 있습니다(따라서 JSON에도 없음).

수신 거부한 수신자

`complainedRecipients` 필드는 수신 거부를 제출했을 수 있는 수신자의 목록을 포함합니다.

> **Important**
>
> 대부분의 ISP가 수신 거부를 제출한 수신자의 이메일 주소를 수신 거부 알림에서 삭제하므로, 이 목록에는 원래 메시지의 수신자 그리고 수신 거부를 보낸 ISP를 기준으로 수신 거부를 제출했을 수 있는 수신자의 목록이 포함되어 있습니다. Amazon SES는 원래 메시지를 조회하여 이 수신자 목록을 결정합니다.

이 목록의 JSON 객체는 다음 필드를 포함합니다.

필드 이름	설명
`emailAddress`	수신자의 이메일 주소.

수신 거부 유형

Internet Assigned Numbers Authority 웹사이트:에 따라 `complaintFeedbackType` 필드에 보고 ISP 가 할당한 다음과 같은 수신 거부 유형이 나타날 수 있습니다.

필드 이름	설명
`abuse`	원치 않는 이메일 또는 기타 유형의 이메일 침해를 나타냅니다.
`auth-failure`	이메일 인증 실패 보고서.
`fraud`	일종의 사기 또는 피싱 활동을 나타냅니다.
`not-spam`	보고서를 제공하는 엔터티가 메시지를 스팸으로 간주하지 않음을 나타냅니다. 이는 스팸으로 잘못 태그 지정 또는 분류된 메시지를 교정하기 위해 사용될 수 있습니다.
`other`	다른 등록된 유형에 들어맞지 않는 기타 피드백을 나타냅니다.
`virus`	발원 메시지에서 바이러스가 발견되었다는 보고서.

전송 객체

`Delivery` 이벤트에 대한 정보를 포함하는 JSON 객체에는 항상 다음 필드가 포함됩니다.

필드 이름	설명
`timestamp`	Amazon SES가 수신자의 메일 서버로 이메일을 전송한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`processingTimeMillis`	Amazon SES가 발신자로부터 요청을 수락한 때로부터 Amazon SES가 수신자의 메일 서버로 메시지를 전송한 때까지의 시간(단위: 밀리초).
`recipients`	전송 이벤트가 적용되는 의도한 수신자의 목록.
`smtpResponse`	Amazon SES로부터 이메일을 수락한 원격 ISP의 SMTP 응답 메시지. 이 메시지는 이메일, 수신 메일 서버, 수신 ISP마다 다릅니다.
`reportingMTA`	메일을 전송한 Amazon SES 메일 서버의 호스트 이름.
`remoteMtaIp`	Amazon SES에서 이메일을 전송한 MTA의 IP 주소입니다.

Send 객체

`send` 이벤트 정보를 포함하는 JSON 객체는 항상 비어 있습니다.

Reject 객체

`Reject` 이벤트에 대한 정보를 포함하는 JSON 객체에는 항상 다음 필드가 포함됩니다.

필드 이름	설명
`reason`	이메일이 거부된 이유입니다. 유일하게 가능한 값은 `Bad content`로, Amazon SES가 바이러스 포함 이메일을 감지했다는 뜻입니다. 메시지가 거부되면 Amazon SES는 메시지 처리를 중지하고 해당 메시지를 수신자의 메일 서버로 전송하지 않습니다.

Open 객체

`Open` 이벤트에 대한 정보를 포함하는 JSON 객체는 항상 다음 필드를 포함합니다.

필드 이름	설명
`ipAddress`	수신자의 IP 주소입니다.
`timestamp`	열기 이벤트가 발생한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`userAgent`	수신자가 이메일을 여는 데 사용한 이메일 클라이언트 또는 디바이스의 사용자 에이전트입니다.

Click 객체

`Click` 이벤트에 대한 정보를 포함하는 JSON 객체는 항상 다음 필드를 포함합니다.

필드 이름	설명
`ipAddress`	수신자의 IP 주소입니다.
`timestamp`	클릭 이벤트가 발생한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`userAgent`	수신자가 이메일의 링크를 클릭하는 데 사용한 클라이언트의 사용자 에이전트입니다.
`link`	수신자가 클릭한 링크의 URL입니다.
`linkTags`	`ses:tags` 속성을 사용하여 링크에 추가된 태그 목록입니다. 이메일의 링크에 태그를 추가하는 방법은 Q5. 링크에 고유 식별자로 태그를 지정할 수 있습니까?의 Amazon SES 이메일 전송 지표 FAQ 섹션을 참조하세요.

Rendering Failure 객체

`Rendering Failure` 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`templateName`	이메일을 전송하는 데 사용하는 템플릿의 이름입니다.
`errorMessage`	렌더링 오류에 관한 자세한 정보를 제공하는 메시지입니다.

DeliveryDelay 객체

`DeliveryDelay` 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`delayType`	지연 유형입니다. 가능한 값은 다음과 같습니다. • InternalFailure – 내부 Amazon SES 문제로 인해 메시지가 지연되었습니다. • General – SMTP 대화 중에 일반 오류가 발생했습니다. • MailboxFull – 수신자의 메일박스가 꽉 차서 추가 메시지를 받을 수 없습니다. • SpamDetected – 수신자의 메일 서버가 귀하의 계정에서 많은 양의 원치 않는 이메일을 감지했습니다. • RecipientServerError – 수신자의 이메일 서버에 일시적인 문제가 생겨 메시지를 전송할 수 없습니다. • IPFailure – 수신자의 이메일 공급자가 메시지를 보내는 IP 주소를 차단하거나 제한하고 있습니다.

필드 이름	설명
	• TransientCommunicationFailure – 수신자의 이메일 공급자와의 SMTP 대화 중에 일시적인 통신 오류가 발생했습니다. • BYOIPHostNameLookupUnavailable – Amazon SES에서 IP 주소의 DNS 호스트 이름을 찾을 수 없습니다. 이러한 지연 유형은 자체 IP 가져오기를 사용할 때만 발생합니다. • Undetermined – Amazon SES에서 전송 지연 사유를 확인할 수 없습니다. • SendingDeferral – Amazon SES가 메시지를 내부적으로 연기하는 것이 적절하다고 판단했습니다.
`delayedRecipients`	이메일 수신자에 대한 정보가 있는 객체입니다.
`expirationTime`	Amazon SES에서 메시지 전송 시도를 중지할 날짜 및 시간입니다. 이 값은 ISO 8601 형식으로 표시됩니다.
`reportingMTA`	지연을 보고한 MTA(메시지 전송 에이전트) 의 IP 주소입니다.
`timestamp`	지연이 발생한 날짜 및 시간으로 ISO 8601 형식으로 표시됩니다.

지연된 수신자

`delayedRecipients` 객체는 다음 값을 포함합니다.

필드 이름	설명
`emailAddress`	메시지 전송이 지연된 이메일 주소입니다.
`status`	전송 지연 관련된 SMTP 상태 코드입니다.

필드 이름	설명
diagnosticCode	수신 메시지 전송 에이전트(MTA)에서 제공하는 진단 코드입니다.

구독 객체

Subscription 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
contactList	연락처가 있는 목록의 이름입니다.
timestamp	ISP가 구독 알림을 전송한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
source	메시지를 전송한 이메일 주소(엔벌로프 MAIL FROM 주소).
newTopicPreferences	연락처 목록에 있는 모든 주제의 구독 상태를 지정하는 JSON 데이터 구조(맵)로, 변경 후 상태를 나타냅니다(연락처 구독함 또는 구독 취소함).
oldTopicPreferences	연락처 목록에 있는 모든 주제의 구독 상태를 지정하는 JSON 데이터 구조(맵)로, 변경 전 상태를 나타냅니다(연락처 구독함 또는 구독 취소함).

신규/이전 주제 기본 설정

newTopicPreferences 및 oldTopicPreferences 객체는 다음 값을 포함합니다.

필드 이름	설명
`unsubscribeAll`	연락처 목록의 모든 주제에서 연락처가 구독을 취소했는지를 지정합니다.
`topicSubscriptionStatus`	지정된 이벤트 유형에 대해 현재 SES로부터 알림을 수신하도록 구독하고 있는지를 나타내는 `topicName` 필드에서 주제의 구독 상태를 지정합니다. `subscriptionStatus` 필드에서 사용 가능한 값은 OptIn(구독) 또는 OptOut(구독 취소)입니다.
`topicDefaultSubscriptionStatus`	`topicName` 필드에서 주제의 기본 구독 상태를 지정하여 이벤트 대상에 추가된 새 주제를 기본적으로 구독할지 아니면 구독 취소할지 결정합니다. `subscriptionStatus` 필드에서 사용 가능한 값은 OptIn(기본적으로 구독) 또는 OptOut(기본적으로 구독 취소)입니다.

Amazon SES에서 Firehose에 게시하는 이벤트 데이터 예제

이 섹션에서는 Amazon SES가 Firehose에 게시하는 이메일 전송 이벤트 레코드의 유형별 예제를 제공합니다.

이 섹션의 주제:

- Bounce 레코드
- Complaint 레코드
- Delivery 레코드
- Send 레코드
- Reject 레코드
- Open 레코드
- Click 레코드
- Rendering Failure 레코드
- DeliveryDelay 레코드
- 구독 레코드

> **Note**
>
> 다음 예제에서는 `tag` 필드가 사용되며, SES가 모든 이벤트 유형에 대한 태그 게시를 지원하는 구성 집합을 통해 이벤트 게시를 사용하고 있습니다. ID에 직접 피드백 알림을 사용하는 경우 SES는 태그를 게시하지 않습니다. 구성 세트 생성 또는 구성 세트 수정 작업을 할 때 태그 추가하기에 대해 읽어보세요.

Bounce 레코드

다음은 Amazon SES가 Firehose에 게시하는 `Bounce` 이벤트 레코드의 예제입니다.

```
{
  "eventType":"Bounce",
  "bounce":{
    "bounceType":"Permanent",
    "bounceSubType":"General",
    "bouncedRecipients":[
      {
        "emailAddress":"recipient@example.com",
        "action":"failed",
        "status":"5.1.1",
        "diagnosticCode":"smtp; 550 5.1.1 user unknown"
      }
    ],
    "timestamp":"2017-08-05T00:41:02.669Z",
    "feedbackId":"01000157c44f053b-61b59c11-9236-11e6-8f96-7be8aexample-000000",
    "reportingMTA":"dsn; mta.example.com"
  },
  "mail":{
    "timestamp":"2017-08-05T00:40:02.012Z",
    "source":"Sender Name <sender@example.com>",
    "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId":"123456789012",
    "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination":[
      "recipient@example.com"
    ],
    "headersTruncated":false,
    "headers":[
      {
        "name":"From",
```

```
        "value":"Sender Name <sender@example.com>"
      },
      {
        "name":"To",
        "value":"recipient@example.com"
      },
      {
        "name":"Subject",
        "value":"Message sent from Amazon SES"
      },
      {
        "name":"MIME-Version",
        "value":"1.0"
      },
      {
        "name":"Content-Type",
        "value":"multipart/alternative; boundary=\"----
=_Part_7307378_1629847660.1516840721503\""
      }
    ],
    "commonHeaders":{
      "from":[
        "Sender Name <sender@example.com>"
      ],
      "to":[
        "recipient@example.com"
      ],
      "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject":"Message sent from Amazon SES"
    },
    "tags":{
      "ses:configuration-set":[
        "ConfigSet"
      ],
      "ses:source-ip":[
        "192.0.2.0"
      ],
      "ses:from-domain":[
        "example.com"
      ],
      "ses:caller-identity":[
        "ses_user"
      ]
    }
```

```
  }
}
```

Complaint 레코드

다음은 Amazon SES가 Firehose에 게시하는 `Complaint` 이벤트 레코드의 예제입니다.

```
{
  "eventType":"Complaint",
  "complaint": {
    "complainedRecipients":[
      {
        "emailAddress":"recipient@example.com"
      }
    ],
    "timestamp":"2017-08-05T00:41:02.669Z",
    "feedbackId":"01000157c44f053b-61b59c11-9236-11e6-8f96-7be8aexample-000000",
    "userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/60.0.3112.90 Safari/537.36",
    "complaintFeedbackType":"abuse",
    "arrivalDate":"2017-08-05T00:41:02.669Z"
  },
  "mail":{
    "timestamp":"2017-08-05T00:40:01.123Z",
    "source":"Sender Name <sender@example.com>",
    "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId":"123456789012",
    "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination":[
      "recipient@example.com"
    ],
    "headersTruncated":false,
    "headers":[
      {
        "name":"From",
        "value":"Sender Name <sender@example.com>"
      },
      {
        "name":"To",
        "value":"recipient@example.com"
      },
      {
        "name":"Subject",
        "value":"Message sent from Amazon SES"
```

```
      },
      {
        "name":"MIME-Version","value":"1.0"
      },
      {
        "name":"Content-Type",
        "value":"multipart/alternative; boundary=\"----
=_Part_7298998_679725522.1516840859643\""
      }
    ],
    "commonHeaders":{
      "from":[
        "Sender Name <sender@example.com>"
      ],
      "to":[
        "recipient@example.com"
      ],
      "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject":"Message sent from Amazon SES"
    },
    "tags":{
      "ses:configuration-set":[
        "ConfigSet"
      ],
      "ses:source-ip":[
        "192.0.2.0"
      ],
      "ses:from-domain":[
        "example.com"
      ],
      "ses:caller-identity":[
        "ses_user"
      ]
    }
  }
}
```

Delivery 레코드

다음은 Amazon SES가 Firehose에 게시하는 `Delivery` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Delivery",
  "mail": {
```

```
    "timestamp": "2016-10-19T23:20:52.240Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": [
      "recipient@example.com"
    ],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "text/html; charset=UTF-8"
      },
      {
        "name": "Content-Transfer-Encoding",
        "value": "7bit"
      }
    ],
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "to": [
        "recipient@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES"
```

```
    },
    "tags": {
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ],
      "ses:from-domain": [
        "example.com"
      ],
      "ses:caller-identity": [
        "ses_user"
      ],
      "ses:outgoing-ip": [
        "192.0.2.0"
      ],
      "myCustomTag1": [
        "myCustomTagValue1"
      ],
      "myCustomTag2": [
        "myCustomTagValue2"
      ]
    }
  },
  "delivery": {
    "timestamp": "2016-10-19T23:21:04.133Z",
    "processingTimeMillis": 11893,
    "recipients": [
      "recipient@example.com"
    ],
    "smtpResponse": "250 2.6.0 Message received",
    "remoteMtaIp": "123.456.789.012",
    "reportingMTA": "mta.example.com"
  }
}
```

Send 레코드

다음은 Amazon SES가 Firehose에 게시하는 `Send` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Send",
  "mail": {
```

```
    "timestamp": "2016-10-14T05:02:16.645Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": [
      "recipient@example.com"
    ],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "multipart/mixed;  boundary=\"----=_Part_0_716996660.1476421336341\""
      },
      {
        "name": "X-SES-MESSAGE-TAGS",
        "value": "myCustomTag1=myCustomTagValue1, myCustomTag2=myCustomTagValue2"
      }
    ],
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "to": [
        "recipient@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES"
```

```
    },
    "tags": {
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ],
      "ses:from-domain": [
        "example.com"
      ],
      "ses:caller-identity": [
        "ses_user"
      ],
      "myCustomTag1": [
        "myCustomTagValue1"
      ],
      "myCustomTag2": [
        "myCustomTagValue2"
      ]
    }
  },
  "send": {}
}
```

Reject 레코드

다음은 Amazon SES가 Firehose에 게시하는 `Reject` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Reject",
  "mail": {
    "timestamp": "2016-10-14T17:38:15.211Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": [
      "sender@example.com"
    ],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
```

```
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "multipart/mixed; boundary=\"qMm9M+Fa2AknHoGS\""
      },
      {
        "name": "X-SES-MESSAGE-TAGS",
        "value": "myCustomTag1=myCustomTagValue1, myCustomTag2=myCustomTagValue2"
      }
    ],
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "to": [
        "recipient@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES"
    },
    "tags": {
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ],
      "ses:from-domain": [
        "example.com"
      ],
      "ses:caller-identity": [
```

```
        "ses_user"
      ],
      "myCustomTag1": [
        "myCustomTagValue1"
      ],
      "myCustomTag2": [
        "myCustomTagValue2"
      ]
    }
  },
  "reject": {
    "reason": "Bad content"
  }
}
```

Open 레코드

다음은 Amazon SES가 Firehose에 게시하는 Open 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Open",
  "mail": {
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES",
      "to": [
        "recipient@example.com"
      ]
    },
    "destination": [
      "recipient@example.com"
    ],
    "headers": [
      {
        "name": "X-SES-CONFIGURATION-SET",
        "value": "ConfigSet"
      },
      {
        "name":"X-SES-MESSAGE-TAGS",
        "value":"myCustomTag1=myCustomValue1, myCustomTag2=myCustomValue2"
      },
```

```
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "multipart/alternative; boundary=\"XBoundary\""
      }
    ],
    "headersTruncated": false,
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "sendingAccountId": "123456789012",
    "source": "sender@example.com",
    "tags": {
      "myCustomTag1":[
        "myCustomValue1"
      ],
      "myCustomTag2":[
        "myCustomValue2"
      ],
      "ses:caller-identity": [
        "IAM_user_or_role_name"
      ],
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:from-domain": [
        "example.com"
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ]
```

```
    },
    "timestamp": "2017-08-09T21:59:49.927Z"
  },
  "open": {
    "ipAddress": "192.0.2.1",
    "timestamp": "2017-08-09T22:00:19.652Z",
    "userAgent": "Mozilla/5.0 (iPhone; CPU iPhone OS 10_3_3 like Mac OS X)
 AppleWebKit/603.3.8 (KHTML, like Gecko) Mobile/14G60"
  }
}
```

Click 레코드

다음은 Amazon SES가 Firehose에 게시하는 `Click` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Click",
  "click": {
    "ipAddress": "192.0.2.1",
    "link": "http://docs.aws.amazon.com/ses/latest/DeveloperGuide/send-email-
smtp.html",
    "linkTags": {
      "samplekey0": [
        "samplevalue0"
      ],
      "samplekey1": [
        "samplevalue1"
      ]
    },
    "timestamp": "2017-08-09T23:51:25.570Z",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/60.0.3112.90 Safari/537.36"
  },
  "mail": {
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES",
      "to": [
        "recipient@example.com"
      ]
    },
```

```
    "destination": [
      "recipient@example.com"
    ],
    "headers": [
      {
        "name": "X-SES-CONFIGURATION-SET",
        "value": "ConfigSet"
      },
      {
        "name":"X-SES-MESSAGE-TAGS",
        "value":"myCustomTag1=myCustomValue1, myCustomTag2=myCustomValue2"
      },
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "multipart/alternative; boundary=\"XBoundary\""
      },
      {
        "name": "Message-ID",
        "value": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000"
      }
    ],
    "headersTruncated": false,
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "sendingAccountId": "123456789012",
    "source": "sender@example.com",
    "tags": {
      "myCustomTag1":[
        "myCustomValue1"
```

```
      ],
      "myCustomTag2":[
        "myCustomValue2"
      ],
      "ses:caller-identity": [
        "ses_user"
      ],
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:from-domain": [
        "example.com"
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ]
    },
    "timestamp": "2017-08-09T23:50:05.795Z"
  }
}
```

Rendering Failure 레코드

다음은 Amazon SES가 Firehose에 게시하는 `Rendering Failure` 이벤트 레코드의 예제입니다.

```
{
  "eventType":"Rendering Failure",
  "mail":{
    "timestamp":"2018-01-22T18:43:06.197Z",
    "source":"sender@example.com",
    "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId":"123456789012",
    "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination":[
      "recipient@example.com"
    ],
    "headersTruncated":false,
    "tags":{
      "ses:configuration-set":[
        "ConfigSet"
      ]
    }
  },
  "failure":{
```

```
    "errorMessage":"Attribute 'attributeName' is not present in the rendering data.",
    "templateName":"MyTemplate"
  }
}
```

DeliveryDelay 레코드

다음은 Amazon SES가 Firehose에 게시하는 `DeliveryDelay` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "DeliveryDelay",
  "mail":{
    "timestamp":"2020-06-16T00:15:40.641Z",
    "source":"sender@example.com",
    "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId":"123456789012",
    "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination":[
      "recipient@example.com"
    ],
    "headersTruncated":false,
    "tags":{
      "ses:configuration-set":[
        "ConfigSet"
      ]
    }
  },
  "deliveryDelay": {
    "timestamp": "2020-06-16T00:25:40.095Z",
    "delayType": "TransientCommunicationFailure",
    "expirationTime": "2020-06-16T00:25:40.914Z",
    "delayedRecipients": [{
      "emailAddress": "recipient@example.com",
      "status": "4.4.1",
      "diagnosticCode": "smtp; 421 4.4.1 Unable to connect to remote host"
    }]
  }
}
```

구독 레코드

다음은 Amazon SES가 Firehose에 게시하는 `Subscription` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Subscription",
  "mail": {
    "timestamp": "2022-01-12T01:00:14.340Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLEe4bccb684-777bc8de-afa7-4970-92b0-f515137b1497-000000",
    "destination": ["recipient@example.com"],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "text/html; charset=UTF-8"
      },
      {
        "name": "Content-Transfer-Encoding",
        "value": "7bit"
      }
    ],
    "commonHeaders": {
      "from": ["sender@example.com"],
      "to": ["recipient@example.com"],
      "messageId": "EXAMPLEe4bccb684-777bc8de-afa7-4970-92b0-f515137b1497-000000",
      "subject": "Message sent from Amazon SES"
    },
    "tags": {
```

```
      "ses:operation": ["SendEmail"],
      "ses:configuration-set": ["ConfigSet"],
      "ses:source-ip": ["192.0.2.0"],
      "ses:from-domain": ["example.com"],
      "ses:caller-identity": ["ses_user"],
      "myCustomTag1": ["myCustomValue1"],
      "myCustomTag2": ["myCustomValue2"]
    }
  },
  "subscription": {
    "contactList": "ContactListName",
    "timestamp": "2022-01-12T01:00:17.910Z",
    "source": "UnsubscribeHeader",
    "newTopicPreferences": {
      "unsubscribeAll": true,
      "topicSubscriptionStatus": [
        {
          "topicName": "ExampleTopicName",
          "subscriptionStatus": "OptOut"
        }
      ]
    },
    "oldTopicPreferences": {
      "unsubscribeAll": false,
      "topicSubscriptionStatus": [
        {
          "topicName": "ExampleTopicName",
          "subscriptionStatus": "OptOut"
        }
      ]
    }
  }
}
```

Amazon SNS의 Amazon SES 이벤트 데이터 해석

Amazon SES는 이메일 전송 이벤트를 Amazon Simple Notification Service(Amazon SNS)에 JSON 레코드로 게시합니다. 그런 다음 Amazon SNS는 이벤트 대상과 연결된 Amazon SNS 주제를 구독하는 엔드포인트에 알림을 보냅니다. Amazon SNS 주제 설정 및 구독에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 가이드의 시작하기를 참조하십시오.

레코드 콘텐츠에 대한 설명과 레코드 예제는 다음 단원을 참조하세요.

- 이벤트 레코드 내용
- 이벤트 레코드 예제

Amazon SES가 Amazon SNS에 게시하는 이벤트 데이터 내용

Amazon SES는 이메일 전송 이벤트 레코드를 Amazon Simple Notification Service(Amazon SNS)에 JSON 형식으로 게시합니다.

이러한 모든 알림 유형에 대한 레코드 예는 Amazon SES가 Amazon SNS에 게시하는 이벤트 데이터 예제에서 찾을 수 있습니다.

이 섹션의 주제:

- 최상위 JSON 객체
- Mail 객체
- Bounce 객체
- Complaint 객체
- 전송 객체
- Send 객체
- Reject 객체
- Open 객체
- Click 객체
- Rendering Failure 객체
- DeliveryDelay 객체
- 구독 객체

최상위 JSON 객체

이메일 전송 이벤트 레코드의 최상위 JSON 객체는 다음 필드로 구성됩니다. 이벤트 유형에 따라 존재하는 다른 객체가 결정됩니다.

필드 이름	설명
`eventType`	이벤트 유형을 설명하는 문자열입니다. 가능한 값: `Bounce`, `Complaint` , `Delivery`, `Send`,

필드 이름	설명
	`Reject`, `Open`, `Click`, `Rendering Failure`, `DeliveryDelay` 또는 `Subscription` 이벤트 게시를 설정하지 않은 경우 이 필드의 이름은 `notificationType` 입니다.
`mail`	이벤트를 만드는 이메일 관련 정보를 포함하는 JSON 객체입니다.
`bounce`	이 필드는 `eventType` 이 `Bounce`인 경우에만 존재합니다. 이 파일에는 반송 관련 정보가 포함되어 있습니다.
`complaint`	이 필드는 `eventType` 이 `Complaint` 인 경우에만 존재합니다. 이 파일에는 수신 거부 관련 정보가 포함되어 있습니다.
`delivery`	이 필드는 `eventType` 이 `Delivery`인 경우에만 존재합니다. 이 파일에는 전송 관련 정보가 포함되어 있습니다.
`send`	이 필드는 `eventType` 이 `Send`인 경우에만 존재합니다.
`reject`	이 필드는 `eventType` 이 `Reject`인 경우에만 존재합니다. 이 파일에는 거부 관련 정보가 포함되어 있습니다.
`open`	이 필드는 `eventType` 이 `Open`인 경우에만 존재합니다. 이 파일에는 열기 이벤트 관련 정보가 포함되어 있습니다.
`click`	이 필드는 `eventType` 이 `Click`인 경우에만 존재합니다. 이 파일에는 클릭 이벤트 관련 정보가 포함되어 있습니다.

필드 이름	설명
`failure`	이 필드는 `eventType` 이 `Rendering Failure`인 경우에만 존재합니다. 이 파일에는 렌더링 오류 이벤트 관련 정보가 포함되어 있습니다.
`deliveryDelay`	이 필드는 `eventType` 이 `DeliveryDelay` 인 경우에만 존재합니다. 이메일 전송 지연에 대한 정보가 포함되어 있습니다.
`subscription`	이 필드는 `eventType` 이 `Subscription` 인 경우에만 존재합니다. 여기에는 구독 기본 설정에 대한 정보가 포함되어 있습니다.

Mail 객체

각 이메일 전송 이벤트 레코드는 `mail` 객체에 원래 메일에 대한 정보를 포함하고 있습니다. `mail` 객체 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`timestamp`	메시지가 전송된 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`messageId`	Amazon SES가 메시지에 할당한 고유 ID입니다. 메시지를 보낼 때 Amazon SES에서 이 값을 반환했습니다. **Note** 이 메시지 ID는 Amazon SES에서 할당한 것입니다. `mail` 객체의 `headers` 및 `commonHeaders` 필드에서 원래 메시지의 메시지 ID를 찾을 수 있습니다.

필드 이름	설명
`source`	메시지를 전송한 이메일 주소(엔벌로프 MAIL FROM 주소).
`sourceArn`	이메일을 전송하는 데 사용된 ID의 Amazon 리소스 이름(ARN). 권한 부여 전송의 경우 `sourceArn` 은 ID 소유자가 위임 발신자에게 메일 전송 시 사용하도록 권한을 부여한 ID의 ARN입니다. 권한 부여 전송에 대한 자세한 내용은 이메일 인증 방법 섹션을 참조하세요.
`sendingAccountId`	이메일을 보내는 데 사용된 AWS 계정의 계정 ID입니다. 권한 부여 전송의 경우 `sendingAccountId` 는 위임 발신자의 계정 ID입니다.
`destination`	원래 메일의 수신자인 이메일 주소의 목록.
`headersTruncated`	알림에서 헤더가 잘렸는지 여부를 나타내는 문자열입니다. 헤더의 용량이 10KB를 초과할 경우 헤더가 잘립니다. 가능한 값은 `true` 및 `false`입니다.
`headers`	이메일의 원래 헤더 목록입니다. 목록의 각 헤더에는 `name` 필드와 `value` 필드가 존재합니다. **Note** `headers` 필드의 모든 메시지 ID는 Amazon SES에 전달한 원본 메시지에서 가져온 것입니다. 이어서 Amazon SES가 메시지에 할당한 메시지 ID는 `mail` 객체의 `messageId` 필드에 들어 있습니다.

필드 이름	설명
commonHeaders	자주 사용되는 원래 이메일 헤더의 매핑입니다. **Note** commonHeaders 필드의 모든 메시지 ID는 Amazon SES가 mail 객체의 messageId 필드 내 메시지에 할당한 메시지 ID입니다.
tags	이메일과 연결된 태그 목록입니다.

Bounce 객체

Bounce 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
bounceType	Amazon SES가 결정한 반송 메일의 유형.
bounceSubType	Amazon SES가 결정한 반송 메일의 하위 유형.
bouncedRecipients	반송된 원래 메일의 수신자 정보를 포함하는 목록.
timestamp	ISP가 반송 메일 알림을 전송한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
feedbackId	반송 메일의 고유 ID.
reportingMTA	DSN의 Reporting-MTA 필드의 값. DSN에서 설명하는 전송, 중계 또는 게이트웨이 작업을 시도하는 메시지 전송 권한(MTA)의 값입니다.

필드 이름	설명
	Note 이 필드는 반송 메일에 전송 상태 알림(DSN)이 첨부된 경우에만 표시됩니다.

반송 수신자

반송 이벤트는 단일 수신자 또는 여러 수신자와 관련이 있을 수 있습니다. `bouncedRecipients` 필드는 객체(이메일 주소가 반송 메일을 생성한 수신자당 객체 1개)의 목록을 포함하고 있으며 다음 필드로 구성됩니다.

필드 이름	설명
`emailAddress`	수신자의 이메일 주소. DSN이 사용 가능할 경우, DSN의 `Final-Recipient` 필드의 값입니다.

또는 반송 메일에 DSN이 첨부된 경우 다음 필드도 존재할 수 있습니다.

필드 이름	설명
`action`	DSN의 `Action` 필드의 값. 이 수신자에게 메시지를 전송하려는 시도의 결과로 보고-MTA가 수행하는 작업을 나타냅니다.
`status`	DSN의 `Status` 필드의 값. 메시지의 전송 상태를 나타내는 수신자별 전송 독립적 상태 코드입니다.
`diagnosticCode`	보고-MTA가 발행한 상태 코드. DSN의 `Diagnostic-Code` 필드의 값입니다. 이 필드가 DSN에는 없을 수 있습니다(따라서 JSON에도 없음).

반송 메일 유형

각 반송 메일 이벤트는 다음 표에 나와 있는 유형 중 하나가 됩니다.

이벤트 게시 시스템은 Amazon SES에서 더 이상 재시도하지 않을 하드 바운스 및 소프트 바운스만 게시합니다. `Permanent`로 표시된 반송 메일을 받으면 메일 그룹에서 해당 이메일 주소를 삭제해야 하며 다음부터는 그러한 이메일 주소로 전송할 수 없습니다. `Transient` 반송 메일은 메시지가 여러 번 소프트 바운스되어 Amazon SES에서 이러한 메일에 대해 재전송 시도를 중지한 경우에 전송됩니다. 처음에 `Transient` 반송 메일이 발생한 주소로 이후 전송 재시도가 성공할 수도 있습니다.

bounceType	bounceSubType	설명
`Undetermined`	`Undetermined`	Amazon SES가 특정 반송 사유를 결정하지 못했습니다.
`Permanent`	`General`	Amazon SES가 일반 하드 바운스를 수신했습니다. 이 유형의 반송 메일을 받은 경우, 이 수신자의 이메일 주소를 메일 발송 목록에서 삭제해야 합니다.
`Permanent`	`NoEmail`	대상 이메일 주소가 존재하지 않아 Amazon SES가 영구 하드 바운스를 수신했습니다. 이 유형의 반송 메일을 받은 경우, 이 수신자의 이메일 주소를 메일 발송 목록에서 삭제해야 합니다.
`Permanent`	`Suppressed`	최근의 잘못된 주소로 인한 반송 이력 때문에 Amazon SES가 이 주소로 메일 전송을 금지했습니다. 전역 금지 목록을 재정의하려면 Amazon SES 계정 수준 금지 목록 사용 섹션을 참조하세요.
`Permanent`	`OnAccountSuppressionList`	계정 수준 금지 목록에 있으므로 Amazon SES가 이 주소로 보내는 것을 금지했습니다. 이는 반송 비율 지표에 반영되지 않습니다.
`Transient`	`General`	Amazon SES가 일반 반송 메일을 수신했습니다. 이후에 이 수신자에게 전송이 성공할 수도 있습니다.

bounceType	bounceSubType	설명
`Transient`	`MailboxFull`	Amazon SES가 메일박스 가득 참 반송 메일을 수신했습니다. 이후에 이 수신자에게 전송이 성공할 수도 있습니다.
`Transient`	`MessageTooLarge`	Amazon SES가 메시지 너무 큼 반송 메일을 수신했습니다. 메시지 크기를 줄일 경우 이 수신자에게 전송이 성공할 수도 있습니다.
`Transient`	`CustomTimeoutExceeded`	Amazon SES가 이메일 발신자가 지정한 시간 내에 이메일을 성공적으로 전송하지 못했습니다. (반송 메시지는 정의된 TTL 내에서 가능한 전송 시도 실패의 이유를 지정합니다.)
`Transient`	`ContentRejected`	Amazon SES가 내용 거부 반송 메일을 수신했습니다. 메시지 내용을 변경할 경우 이 수신자에게 전송이 성공할 수도 있습니다.
`Transient`	`AttachmentRejected`	Amazon SES가 첨부 파일 거부 반송 메일을 수신했습니다. 첨부 파일을 제거하거나 변경할 경우 이 수신자에게 전송이 성공할 수도 있습니다.

Complaint 객체

`Complaint` 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`complainedRecipients`	수신 거부를 제출했을 수 있는 수신자에 대한 정보를 포함하는 목록.
`timestamp`	ISP가 수신 거부 알림을 전송한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`feedbackId`	수신 거부의 고유 ID.

필드 이름	설명
`complaintSubType`	Amazon SES가 결정한 수신 거부의 하위 유형.

또한, 수신 거부에 피드백 보고서가 첨부된 경우 다음 필드가 포함될 수 있습니다.

필드 이름	설명
`userAgent`	피드백 보고서의 `User-Agent` 필드의 값입니다. 보고서를 생성한 시스템의 이름 및 버전을 나타냅니다.
`complaintFeedbackType`	ISP로부터 수신된 피드백 보고서의 `Feedback-Type` 필드의 값. 이 값은 피드백의 유형을 포함합니다.
`arrivalDate`	피드백 보고서의 `Arrival-Date` 또는 `Received-Date` 필드 값으로 ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)입니다. 이 필드가 보고서에는 없을 수 있습니다(따라서 JSON에도 없음).

수신 거부한 수신자

`complainedRecipients` 필드는 수신 거부를 제출했을 수 있는 수신자의 목록을 포함합니다.

> **Important**
> 대부분 ISP는 수신 거부를 제출한 수신자의 이메일 주소를 수정합니다. 이러한 이유로 `complainedRecipients` 필드에는 수신 거부 알림을 발행한 도메인의 주소로 이메일이 전송된 모든 사람의 목록이 있습니다.

이 목록의 JSON 객체는 다음 필드를 포함합니다.

필드 이름	설명
emailAddress	수신자의 이메일 주소.

수신 거부 유형

Internet Assigned Numbers Authority 웹사이트:에 따라 complaintFeedbackType 필드에 보고 ISP가 할당한 다음과 같은 수신 거부 유형이 나타날 수 있습니다.

필드 이름	설명
abuse	원치 않는 이메일 또는 기타 유형의 이메일 침해를 나타냅니다.
auth-failure	이메일 인증 실패 보고서.
fraud	일종의 사기 또는 피싱 활동을 나타냅니다.
not-spam	보고서를 제공하는 엔터티가 메시지를 스팸으로 간주하지 않음을 나타냅니다. 이는 스팸으로 잘못 태그 지정 또는 분류된 메시지를 교정하기 위해 사용될 수 있습니다.
other	다른 등록된 유형에 들어맞지 않는 기타 피드백을 나타냅니다.
virus	발원 메시지에서 바이러스가 발견되었다는 보고서.

수신 거부 하위 유형

complaintSubType 필드의 값은 null 또는 OnAccountSuppressionList일 수 있습니다. 값이 OnAccountSuppressionList인 경우 Amazon SES는 메시지를 수락했지만 계정 수준 금지 목록에 있었기 때문에 메시지를 보내려고 시도하지 않았습니다.

전송 객체

Delivery 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`timestamp`	Amazon SES가 수신자의 메일 서버로 이메일을 전송한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`processingTimeMillis`	Amazon SES가 발신자로부터 요청을 수락한 때로부터 Amazon SES가 수신자의 메일 서버로 메시지를 전송한 때까지의 시간(단위: 밀리초).
`recipients`	전송 이벤트가 적용되는 의도한 수신자의 목록.
`smtpResponse`	Amazon SES로부터 이메일을 수락한 원격 ISP의 SMTP 응답 메시지. 이 메시지는 이메일, 수신 메일 서버, 수신 ISP마다 다릅니다.
`reportingMTA`	메일을 전송한 Amazon SES 메일 서버의 호스트 이름.
`remoteMtaIp`	Amazon SES에서 이메일을 전송한 MTA의 IP 주소입니다.

Send 객체

`send` 이벤트 정보를 포함하는 JSON 객체는 항상 비어 있습니다.

Reject 객체

`Reject` 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`reason`	이메일이 거부된 이유입니다. 유일하게 가능한 값은 `Bad content`로, Amazon SES가 바이러스 포함 이메일을 감지했다는 뜻입니다. 메시지가 거부되면 Amazon SES는 메시지 처리를 중지하고 해당 메시지를 수신자의 메일 서버로 전송하지 않습니다.

Open 객체

`Open` 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`ipAddress`	수신자의 IP 주소입니다.
`timestamp`	열기 이벤트가 발생한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`userAgent`	수신자가 이메일을 여는 데 사용한 이메일 클라이언트 또는 디바이스의 사용자 에이전트입니다.

Click 객체

`Click` 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`ipAddress`	수신자의 IP 주소입니다.
`timestamp`	클릭 이벤트가 발생한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
`userAgent`	수신자가 이메일의 링크를 클릭하는 데 사용한 클라이언트의 사용자 에이전트입니다.
`link`	수신자가 클릭한 링크의 URL입니다.
`linkTags`	`ses:tags` 속성을 사용하여 링크에 추가된 태그 목록입니다. 이메일의 링크에 태그를 추가하는 방법은 Q5. 링크에 고유 식별자로 태그를 지정할 수 있습니까?의 Amazon SES 이메일 전송 지표 FAQ 섹션을 참조하세요.

Rendering Failure 객체

`Rendering Failure` 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`templateName`	이메일을 전송하는 데 사용하는 템플릿의 이름입니다.
`errorMessage`	렌더링 오류에 관한 자세한 정보를 제공하는 메시지입니다.

DeliveryDelay 객체

`DeliveryDelay` 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
`delayType`	지연 유형입니다. 가능한 값은 다음과 같습니다. • InternalFailure – 내부 Amazon SES 문제로 인해 메시지가 지연되었습니다. • General – SMTP 대화 중에 일반 오류가 발생했습니다. • MailboxFull – 수신자의 메일박스가 꽉 차서 추가 메시지를 받을 수 없습니다. • SpamDetected – 수신자의 메일 서버가 귀하의 계정에서 많은 양의 원치 않는 이메일을 감지했습니다. • RecipientServerError – 수신자의 이메일 서버에 일시적인 문제가 생겨 메시지를 전송할 수 없습니다. • IPFailure – 수신자의 이메일 공급자가 메시지를 보내는 IP 주소를 차단하거나 제한하고 있습니다.

필드 이름	설명
	• TransientCommunicationFailure – 수신자의 이메일 공급자와의 SMTP 대화 중에 일시적인 통신 오류가 발생했습니다. • BYOIPHostNameLookupUnavailable – Amazon SES에서 IP 주소의 DNS 호스트 이름을 찾을 수 없습니다. 이러한 지연 유형은 자체 IP 가져오기를 사용할 때만 발생합니다. • Undetermined – Amazon SES에서 전송 지연 사유를 확인할 수 없습니다. • SendingDeferral – Amazon SES가 메시지를 내부적으로 연기하는 것이 적절하다고 판단했습니다.
`delayedRecipients`	이메일 수신자에 대한 정보가 있는 객체입니다.
`expirationTime`	Amazon SES에서 메시지 전송 시도를 중지할 날짜 및 시간입니다. 이 값은 ISO 8601 형식으로 표시됩니다.
`reportingMTA`	지연을 보고한 MTA(메시지 전송 에이전트) 의 IP 주소입니다.
`timestamp`	지연이 발생한 날짜 및 시간으로 ISO 8601 형식으로 표시됩니다.

지연된 수신자

`delayedRecipients` 객체는 다음 값을 포함합니다.

필드 이름	설명
`emailAddress`	메시지 전송이 지연된 이메일 주소입니다.
`status`	전송 지연 관련된 SMTP 상태 코드입니다.

필드 이름	설명
diagnosticCode	수신 메시지 전송 에이전트(MTA)에서 제공하는 진단 코드입니다.

구독 객체

Subscription 이벤트 정보를 포함하는 JSON 객체는 다음 필드로 구성됩니다.

필드 이름	설명
contactList	연락처가 있는 목록의 이름입니다.
timestamp	ISP가 구독 알림을 전송한 날짜와 시간으로, ISO8601 형식(YYYY-MM-DDThh:mm:ss.sZ)으로 표시됩니다.
source	메시지를 전송한 이메일 주소(엔벌로프 MAIL FROM 주소).
newTopicPreferences	연락처 목록에 있는 모든 주제의 구독 상태를 지정하는 JSON 데이터 구조(맵)로, 변경 후 상태를 나타냅니다(연락처 구독함 또는 구독 취소함).
oldTopicPreferences	연락처 목록에 있는 모든 주제의 구독 상태를 지정하는 JSON 데이터 구조(맵)로, 변경 전 상태를 나타냅니다(연락처 구독함 또는 구독 취소함).

신규/이전 주제 기본 설정

newTopicPreferences 및 oldTopicPreferences 객체는 다음 값을 포함합니다.

필드 이름	설명
`unsubscribeAll`	연락처 목록의 모든 주제에서 연락처가 구독을 취소했는지를 지정합니다.
`topicSubscriptionStatus`	지정된 이벤트 유형에 대해 현재 SES로부터 알림을 수신하도록 구독하고 있는지를 나타내는 `topicName` 필드에서 주제의 구독 상태를 지정합니다. `subscriptionStatus` 필드에서 사용 가능한 값은 OptIn(구독) 또는 OptOut(구독 취소)입니다.
`topicDefaultSubscriptionStatus`	`topicName` 필드에서 주제의 기본 구독 상태를 지정하여 이벤트 대상에 추가된 새 주제를 기본적으로 구독할지 아니면 구독 취소할지 결정합니다. `subscriptionStatus` 필드에서 사용 가능한 값은 OptIn(기본적으로 구독) 또는 OptOut(기본적으로 구독 취소)입니다.

Amazon SES가 Amazon SNS에 게시하는 이벤트 데이터 예제

이 섹션에서는 Amazon SES가 Amazon SNS에 게시하는 이메일 전송 이벤트 레코드의 유형별 예제를 제공합니다.

이 섹션의 주제:

- Bounce 레코드
- Complaint 레코드
- Delivery 레코드
- Send 레코드
- Reject 레코드
- Open 레코드
- Click 레코드
- Rendering Failure 레코드
- DeliveryDelay 레코드
- 구독 레코드

> **Note**
>
> 다음 예제에서는 `tag` 필드가 사용되며, SES가 모든 이벤트 유형에 대한 태그 게시를 지원하는 구성 집합을 통해 이벤트 게시를 사용하고 있습니다. ID에 직접 피드백 알림을 사용하는 경우 SES는 태그를 게시하지 않습니다. 구성 세트 생성 또는 구성 세트 수정 작업을 할 때 태그 추가하기에 대해 읽어보세요.

Bounce 레코드

다음은 Amazon SES가 Amazon SNS에 게시하는 `Bounce` 이벤트 레코드의 예제입니다.

```
{
  "eventType":"Bounce",
  "bounce":{
    "bounceType":"Permanent",
    "bounceSubType":"General",
    "bouncedRecipients":[
      {
        "emailAddress":"recipient@example.com",
        "action":"failed",
        "status":"5.1.1",
        "diagnosticCode":"smtp; 550 5.1.1 user unknown"
      }
    ],
    "timestamp":"2017-08-05T00:41:02.669Z",
    "feedbackId":"01000157c44f053b-61b59c11-9236-11e6-8f96-7be8aexample-000000",
    "reportingMTA":"dsn; mta.example.com"
  },
  "mail":{
    "timestamp":"2017-08-05T00:40:02.012Z",
    "source":"Sender Name <sender@example.com>",
    "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId":"123456789012",
    "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination":[
      "recipient@example.com"
    ],
    "headersTruncated":false,
    "headers":[
      {
        "name":"From",
```

```
        "value":"Sender Name <sender@example.com>"
      },
      {
        "name":"To",
        "value":"recipient@example.com"
      },
      {
        "name":"Subject",
        "value":"Message sent from Amazon SES"
      },
      {
        "name":"MIME-Version",
        "value":"1.0"
      },
      {
        "name":"Content-Type",
        "value":"multipart/alternative; boundary=\"----
=_Part_7307378_1629847660.1516840721503\""
      }
    ],
    "commonHeaders":{
      "from":[
        "Sender Name <sender@example.com>"
      ],
      "to":[
        "recipient@example.com"
      ],
      "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject":"Message sent from Amazon SES"
    },
    "tags":{
      "ses:configuration-set":[
        "ConfigSet"
      ],
      "ses:source-ip":[
        "192.0.2.0"
      ],
      "ses:from-domain":[
        "example.com"
      ],
      "ses:caller-identity":[
        "ses_user"
      ]
    }
```

```
  }
}
```

Complaint 레코드

다음은 Amazon SES가 Amazon SNS에 게시하는 `Complaint` 이벤트 레코드의 예제입니다.

```
{
  "eventType":"Complaint",
  "complaint": {
    "complainedRecipients":[
      {
        "emailAddress":"recipient@example.com"
      }
    ],
    "timestamp":"2017-08-05T00:41:02.669Z",
    "feedbackId":"01000157c44f053b-61b59c11-9236-11e6-8f96-7be8aexample-000000",
    "userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/60.0.3112.90 Safari/537.36",
    "complaintFeedbackType":"abuse",
    "arrivalDate":"2017-08-05T00:41:02.669Z"
  },
  "mail":{
    "timestamp":"2017-08-05T00:40:01.123Z",
    "source":"Sender Name <sender@example.com>",
    "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId":"123456789012",
    "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination":[
      "recipient@example.com"
    ],
    "headersTruncated":false,
    "headers":[
      {
        "name":"From",
        "value":"Sender Name <sender@example.com>"
      },
      {
        "name":"To",
        "value":"recipient@example.com"
      },
      {
        "name":"Subject",
        "value":"Message sent from Amazon SES"
```

```
      },
      {
        "name":"MIME-Version","value":"1.0"
      },
      {
        "name":"Content-Type",
        "value":"multipart/alternative; boundary=\"----
=_Part_7298998_679725522.1516840859643\""
      }
    ],
    "commonHeaders":{
      "from":[
        "Sender Name <sender@example.com>"
      ],
      "to":[
        "recipient@example.com"
      ],
      "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject":"Message sent from Amazon SES"
    },
    "tags":{
      "ses:configuration-set":[
        "ConfigSet"
      ],
      "ses:source-ip":[
        "192.0.2.0"
      ],
      "ses:from-domain":[
        "example.com"
      ],
      "ses:caller-identity":[
        "ses_user"
      ]
    }
  }
}
```

Delivery 레코드

다음은 Amazon SES가 Amazon SNS에 게시하는 `Delivery` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Delivery",
  "mail": {
```

```
    "timestamp": "2016-10-19T23:20:52.240Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": [
      "recipient@example.com"
    ],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "text/html; charset=UTF-8"
      },
      {
        "name": "Content-Transfer-Encoding",
        "value": "7bit"
      }
    ],
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "to": [
        "recipient@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES"
```

```
    },
    "tags": {
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ],
      "ses:from-domain": [
        "example.com"
      ],
      "ses:caller-identity": [
        "ses_user"
      ],
      "ses:outgoing-ip": [
        "192.0.2.0"
      ],
      "myCustomTag1": [
        "myCustomTagValue1"
      ],
      "myCustomTag2": [
        "myCustomTagValue2"
      ]
    }
  },
  "delivery": {
    "timestamp": "2016-10-19T23:21:04.133Z",
    "processingTimeMillis": 11893,
    "recipients": [
      "recipient@example.com"
    ],
    "smtpResponse": "250 2.6.0 Message received",
    "remoteMtaIp": "123.456.789.012",
    "reportingMTA": "mta.example.com"
  }
}
```

Send 레코드

다음은 Amazon SES가 Amazon SNS에 게시하는 Send 이벤트 레코드의 예제입니다. 일부 필드는 표시되지 않을 수도 있습니다. 예를 들어 템플릿으로 작성된 이메일을 사용하면 제목이 나중에 렌더링되고 후속 이벤트에 포함됩니다.

```
{
  "eventType": "Send",
  "mail": {
    "timestamp": "2016-10-14T05:02:16.645Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": [
      "recipient@example.com"
    ],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "multipart/mixed;  boundary=\"----=_Part_0_716996660.1476421336341\""
      },
      {
        "name": "X-SES-MESSAGE-TAGS",
        "value": "myCustomTag1=myCustomTagValue1, myCustomTag2=myCustomTagValue2"
      }
    ],
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "to": [
```

```
        "recipient@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES"
    },
    "tags": {
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ],
      "ses:from-domain": [
        "example.com"
      ],
      "ses:caller-identity": [
        "ses_user"
      ],
      "myCustomTag1": [
        "myCustomTagValue1"
      ],
      "myCustomTag2": [
        "myCustomTagValue2"
      ]
    }
  },
  "send": {}
}
```

Reject 레코드

다음은 Amazon SES가 Amazon SNS에 게시하는 `Reject` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Reject",
  "mail": {
    "timestamp": "2016-10-14T17:38:15.211Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": [
      "sender@example.com"
    ],
```

```
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "multipart/mixed; boundary=\"qMm9M+Fa2AknHoGS\""
      },
      {
        "name": "X-SES-MESSAGE-TAGS",
        "value": "myCustomTag1=myCustomTagValue1, myCustomTag2=myCustomTagValue2"
      }
    ],
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "to": [
        "recipient@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES"
    },
    "tags": {
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ],
```

```
      "ses:from-domain": [
        "example.com"
      ],
      "ses:caller-identity": [
        "ses_user"
      ],
      "myCustomTag1": [
        "myCustomTagValue1"
      ],
      "myCustomTag2": [
        "myCustomTagValue2"
      ]
    }
  },
  "reject": {
    "reason": "Bad content"
  }
}
```

Open 레코드

다음은 Amazon SES가 Amazon SNS에 게시하는 Open 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Open",
  "mail": {
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES",
      "to": [
        "recipient@example.com"
      ]
    },
    "destination": [
      "recipient@example.com"
    ],
    "headers": [
      {
        "name": "X-SES-CONFIGURATION-SET",
        "value": "ConfigSet"
      },
```

```
      {
        "name":"X-SES-MESSAGE-TAGS",
        "value":"myCustomTag1=myCustomValue1, myCustomTag2=myCustomValue2"
      },
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "multipart/alternative; boundary=\"XBoundary\""
      }
    ],
    "headersTruncated": false,
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "sendingAccountId": "123456789012",
    "source": "sender@example.com",
    "tags": {
      "myCustomTag1":[
        "myCustomValue1"
      ],
      "myCustomTag2":[
        "myCustomValue2"
      ],
      "ses:caller-identity": [
        "IAM_user_or_role_name"
      ],
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:from-domain": [
        "example.com"
```

```
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ]
    },
    "timestamp": "2017-08-09T21:59:49.927Z"
  },
  "open": {
    "ipAddress": "192.0.2.1",
    "timestamp": "2017-08-09T22:00:19.652Z",
    "userAgent": "Mozilla/5.0 (iPhone; CPU iPhone OS 10_3_3 like Mac OS X)
 AppleWebKit/603.3.8 (KHTML, like Gecko) Mobile/14G60"
  }
}
```

Click 레코드

다음은 Amazon SES가 Amazon SNS에 게시하는 `Click` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Click",
  "click": {
    "ipAddress": "192.0.2.1",
    "link": "http://docs.aws.amazon.com/ses/latest/DeveloperGuide/send-email-
smtp.html",
    "linkTags": {
      "samplekey0": [
        "samplevalue0"
      ],
      "samplekey1": [
        "samplevalue1"
      ]
    },
    "timestamp": "2017-08-09T23:51:25.570Z",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/60.0.3112.90 Safari/537.36"
  },
  "mail": {
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES",
```

```
      "to": [
        "recipient@example.com"
      ]
    },
    "destination": [
      "recipient@example.com"
    ],
    "headers": [
      {
        "name": "X-SES-CONFIGURATION-SET",
        "value": "ConfigSet"
      },
      {
        "name":"X-SES-MESSAGE-TAGS",
        "value":"myCustomTag1=myCustomValue1, myCustomTag2=myCustomValue2"
      },
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "multipart/alternative; boundary=\"XBoundary\""
      },
      {
        "name": "Message-ID",
        "value": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000"
      }
    ],
    "headersTruncated": false,
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "sendingAccountId": "123456789012",
```

```
    "source": "sender@example.com",
    "tags": {
      "myCustomTag1":[
        "myCustomValue1"
      ],
      "myCustomTag2":[
        "myCustomValue2"
      ],
      "ses:caller-identity": [
        "ses_user"
      ],
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:from-domain": [
        "example.com"
      ],
      "ses:source-ip": [
        "192.0.2.0"
      ]
    },
    "timestamp": "2017-08-09T23:50:05.795Z"
  }
}
```

Rendering Failure 레코드

다음은 Amazon SES가 Amazon SNS에 게시하는 `Rendering Failure` 이벤트 레코드의 예제입니다.

```
{
  "eventType":"Rendering Failure",
  "mail":{
    "timestamp":"2018-01-22T18:43:06.197Z",
    "source":"sender@example.com",
    "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId":"123456789012",
    "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination":[
      "recipient@example.com"
    ],
    "headersTruncated":false,
    "tags":{
```

```
      "ses:configuration-set":[
        "ConfigSet"
      ]
    }
  },
  "failure":{
    "errorMessage":"Attribute 'attributeName' is not present in the rendering data.",
    "templateName":"MyTemplate"
  }
}
```

DeliveryDelay 레코드

다음은 Amazon SES가 Amazon SNS에 게시하는 `DeliveryDelay` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "DeliveryDelay",
  "mail":{
    "timestamp":"2020-06-16T00:15:40.641Z",
    "source":"sender@example.com",
    "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId":"123456789012",
    "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination":[
      "recipient@example.com"
    ],
    "headersTruncated":false,
    "tags":{
      "ses:configuration-set":[
        "ConfigSet"
      ]
    }
  },
  "deliveryDelay": {
    "timestamp": "2020-06-16T00:25:40.095Z",
    "delayType": "TransientCommunicationFailure",
    "expirationTime": "2020-06-16T00:25:40.914Z",
    "delayedRecipients": [{
      "emailAddress": "recipient@example.com",
      "status": "4.4.1",
      "diagnosticCode": "smtp; 421 4.4.1 Unable to connect to remote host"
    }]
  }
```

```
}
```

구독 레코드

다음은 Amazon SES가 Firehose에 게시하는 `Subscription` 이벤트 레코드의 예제입니다.

```
{
  "eventType": "Subscription",
  "mail": {
    "timestamp": "2022-01-12T01:00:14.340Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLEe4bccb684-777bc8de-afa7-4970-92b0-f515137b1497-000000",
    "destination": ["recipient@example.com"],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "text/html; charset=UTF-8"
      },
      {
        "name": "Content-Transfer-Encoding",
        "value": "7bit"
      }
    ],
    "commonHeaders": {
      "from": ["sender@example.com"],
```

```
      "to": ["recipient@example.com"],
      "messageId": "EXAMPLEe4bccb684-777bc8de-afa7-4970-92b0-f515137b1497-000000",
      "subject": "Message sent from Amazon SES"
    },
    "tags": {
      "ses:operation": ["SendEmail"],
      "ses:configuration-set": ["ConfigSet"],
      "ses:source-ip": ["192.0.2.0"],
      "ses:from-domain": ["example.com"],
      "ses:caller-identity": ["ses_user"],
      "myCustomTag1": ["myCustomValue1"],
      "myCustomTag2": ["myCustomValue2"]
    }
  },
  "subscription": {
    "contactList": "ContactListName",
    "timestamp": "2022-01-12T01:00:17.910Z",
    "source": "UnsubscribeHeader",
    "newTopicPreferences": {
      "unsubscribeAll": true,
      "topicSubscriptionStatus": [
        {
          "topicName": "ExampleTopicName",
          "subscriptionStatus": "OptOut"
        }
      ]
    },
    "oldTopicPreferences": {
      "unsubscribeAll": false,
      "topicSubscriptionStatus": [
        {
          "topicName": "ExampleTopicName",
          "subscriptionStatus": "OptOut"
        }
      ]
    }
  }
}
```

Amazon SES 발신자 평판 모니터링

Amazon SES는 발신자의 평판에 손상이 갈 수 있거나 이메일 전송 비율을 저하시킬 수 있는 여러 지표를 능동적으로 추적합니다. 이 프로세스에서 고려하는 두 가지 중요한 측정치는 해당 계정의 반송 비율과 불만 제기 비율입니다. 계정에 대한 반송 메일 또는 수신 거부 발생률이 너무 높으면 계정을 검토하거나 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

반송 메일 및 수신 거부 발생률은 계정 상태에 매우 중요하기 때문에 Amazon SES는 이러한 지표를 추적하는 데 사용할 수 있도록 Amazon SES 콘솔에 평판 지표 페이지를 제공합니다. 평판 지표는 반송 메일 또는 수신 거부와 관련이 없지만 발신자 평판을 손상할 수 있는 요인에 대한 정보도 표시될 수 있습니다. 예를 들어 확인된 spamtrap으로 이메일을 보내는 경우 이 대시보드에 메시지가 표시됩니다.

이 단원에서는 평판 지표에 액세스하고, 평판 지표의 정보를 해석하며, 발신자 평판에 영향을 줄 수 있는 요인을 능동적으로 알리도록 시스템을 설정하는 방법에 대한 정보를 설명합니다.

이 단원에는 다음과 같은 주제가 포함되어 있습니다.

- 평판 지표를 사용하여 반송 메일 및 수신 거부률 추적
- 평판 지표 메시지
- CloudWatch를 사용하여 평판 모니터링 경보 생성
- 전용 IP에 대한 SNDS 지표
- 이메일 전송 자동 일시 중지

평판 지표를 사용하여 반송 메일 및 수신 거부률 추적

평판 지표 콘솔 페이지에는 Amazon SES 팀에서 개별 계정의 상태를 확인할 때 보는 정보와 동일한 정보가 들어 있습니다.

평판 지표를 보려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 화면 왼쪽에 있는 탐색 창에서 Reputation metrics(평판 지표)를 클릭합니다.

   대시보드에 다음 정보가 표시됩니다.

   - 계정 상태 - 반송 메일 및 수신 거부률을 합친 상태 요약입니다. 가능한 값은 다음과 같습니다.
     - 정상 – 계정에 영향을 주는 문제가 현재로서는 없습니다.

- 검토 중 – 사용자 계정을 검토 중입니다. 계정 검토의 원인이 된 문제가 검토 기간이 종료될 때까지 해결되지 않으면 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.
- 검토 종료 결정 보류 중 – 사용자 계정을 검토 중입니다. 계정 검토의 원인이 된 문제의 특성으로 인해 추가 조치를 취하기 전에 사용자 계정을 수동으로 검토해야 합니다.
- 전송 일시 중지 – 사용자 계정의 이메일 전송 기능을 일시 중지했습니다. 사용자 계정의 이메일 전송 기능이 일시 중지된 동안에는 Amazon SES를 사용하여 이메일을 보낼 수 없습니다. 이 결정을 검토해 달라고 요청할 수 있습니다. 검토 요청에 대해 자세히 알아보려면 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하세요.
- 전송 일시 중지 보류 중 – 사용자 계정을 검토 중입니다. 계정 검토의 원인이 된 문제가 해결되지 않았습니다. 이 경우에는 일반적으로 사용자 계정의 이메일 전송 기능을 일시 중지합니다. 하지만 계정의 특성상 우선 해당 계정을 검토해야 추가 조치를 취할 수 있습니다.

- 반송 메일 발생률 – 사용자 계정에서 보낸 이메일 중 하드 바운스로 이어진 이메일의 비율입니다. 반송율 계산 방법을 참조하세요.
- 수신 거부 발생률 – 사용자 계정에서 보낸 이메일 중 수신자가 스팸으로 보고한 이메일의 비율입니다. 수신 거부 발생률 계산 방법을 참조하세요.

> **Note**
>
> [Bounce Rate] 및 [Complaint Rate] 섹션에는 해당 측정치에 대한 상태 메시지도 포함됩니다. 다음은 이러한 측정치에 대해 표시될 수 있는 상태 메시지의 목록입니다.
>
> - 정상 – 지표가 정상 수준 내에 있습니다.
> - 거의 복구 – 지표로 인해 사용자 계정이 검토 상태로 설정되었습니다. 검토 기간이 시작된 후 이 지표가 최대 비율 미만으로 유지되었습니다. 지표가 최대 비율 미만으로 유지된 경우 검토 기간이 끝나기 전에 이 지표의 상태가 정상으로 변경됩니다.
> - 검토 중 – 지표로 인해 사용자 계정이 검토 상태로 설정되었으며 지표가 최대 비율보다 높습니다. 지표가 최대 비율을 초과하는 문제가 검토 기간이 끝날 때까지 해결되지 않으면 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.
> - 전송 일시 중지 – 지표로 인해 사용자 계정의 이메일 전송 기능이 일시 중지되었습니다. 사용자 계정의 이메일 전송 기능이 일시 중지된 동안에는 Amazon SES를 사용하여 이메일을 보낼 수 없습니다. 이 결정을 검토해 달라고 요청할 수 있습니다. 검토 요청 제출에 대해 자세히 알아보려면 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하세요.
> - 전송 일시 중지 보류 중 – 이 지표로 인해 계정을 검토 중입니다. 이 검토 기간의 원인이 된 문제가 해결되지 않았습니다. 이러한 문제로 인해 사용자 계정의 이메일 전송

기능이 일시 중지될 수 있습니다. 추가 조치가 취해지기 전에 Amazon SES 팀의 담당자가 사용자 계정을 검토해야 합니다.

- 기타 알림 – 사용자 계정에 반송 메일 또는 수신 거부와 관련되지 않은 평판 관련 문제가 발생한 경우 여기에 간략한 메시지가 표시됩니다. 이 영역에 표시될 수 있는 알림에 대한 자세한 내용은 평판 지표 메시지 단원을 참조하세요.

평판 지표 메시지

Amazon SES 평판 지표 콘솔 페이지는 사용자 계정과 관련된 중요한 지표를 제공합니다. 다음 단원에서는 이 대시보드에 표시될 수 있는 메시지를 설명하고 발신자 평판과 관련된 문제를 해결하는 데 사용할 수 있는 도움말과 정보를 제공합니다.

이 단원에서는 다음 유형의 알림에 대해 설명합니다.

- 상태 메시지
- 반송 메일 발생률 알림
- 수신 거부 발생률 알림
- 스팸 방지 조직 알림
- listbombing 알림
- 직접 피드백 알림
- 도메인 차단 목록 알림
- 내부 검토 알림
- 메일박스 제공업체 알림
- 수신자 피드백 알림
- 관련 계정 알림
- 스팸 트랩 알림
- 취약 사이트 알림
- 손상된 보안 인증 정보 알림
- 기타 알림

상태 메시지

평판 지표 코놀 페이지를 사용하면 사용자의 Amazon SES 계정 상태를 나타내는 메시지가 표시됩니다. 다음은 계정 상태에 표시될 수 있는 값의 목록입니다.

- 정상 – 계정에 영향을 주는 문제가 현재로서는 없습니다.
- 검토 중 – 사용자 계정을 검토 중입니다. 계정 검토의 원인이 된 문제가 검토 기간이 종료될 때까지 해결되지 않으면 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.
- 검토 종료 결정 보류 중 – 사용자 계정을 검토 중입니다. 계정 검토의 원인이 된 문제의 특성으로 인해 추가 조치를 취하기 전에 사용자 계정을 수동으로 검토해야 합니다.
- 전송 일시 중지 – 사용자 계정의 이메일 전송 기능을 일시 중지했습니다. 사용자 계정의 이메일 전송 기능이 일시 중지된 동안에는 Amazon SES를 사용하여 이메일을 보낼 수 없습니다. 이 결정을 검토해 달라고 요청할 수 있습니다. 검토 요청에 대해 자세히 알아보려면 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하세요.
- 전송 일시 중지 보류 중 – 사용자 계정을 검토 중입니다. 계정 검토의 원인이 된 문제가 해결되지 않았습니다. 이 경우에는 일반적으로 사용자 계정의 이메일 전송 기능을 일시 중지합니다. 하지만 계정의 특성상 우선 해당 계정을 검토해야 추가 조치를 취할 수 있습니다.

또한 평판 지표 페이지의 Bounce Rate(반송 메일 발생률)와 Complaint Rate(수신 거부율) 섹션에는 해당 측정치에 대한 상태 요약 정보가 표시됩니다. 다음은 측정치 상태에 표시될 수 있는 값의 목록입니다.

- 정상 – 지표가 정상 수준 내에 있습니다.
- 거의 복구 – 지표로 인해 사용자 계정이 검토 상태로 설정되었습니다. 검토 기간이 시작된 후 이 지표가 최대 비율 미만으로 유지되었습니다. 지표가 최대 비율 미만으로 유지된 경우 검토 기간이 끝나기 전에 이 지표의 상태가 정상으로 변경됩니다.
- 검토 중 – 지표로 인해 사용자 계정이 검토 상태로 설정되었으며 지표가 최대 비율보다 높습니다. 지표가 최대 비율을 초과하는 문제가 검토 기간이 끝날 때까지 해결되지 않으면 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.
- 전송 일시 중지 – 지표로 인해 사용자 계정의 이메일 전송 기능이 일시 중지되었습니다. 사용자 계정의 이메일 전송 기능이 일시 중지된 동안에는 Amazon SES를 사용하여 이메일을 보낼 수 없습니다. 이 결정을 검토해 달라고 요청할 수 있습니다. 검토 요청 제출에 대해 자세히 알아보려면 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하세요.

- 전송 일시 중지 보류 중 – 이 지표로 인해 계정을 검토 중입니다. 이 검토 기간의 원인이 된 문제가 해결되지 않았습니다. 이러한 문제로 인해 사용자 계정의 이메일 전송 기능이 일시 중지될 수 있습니다. 추가 조치가 취해지기 전에 Amazon SES 팀의 담당자가 사용자 계정을 검토해야 합니다.

반송 메일 발생률 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 반송 메일 발생률 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

이 알림은 계정의 반송 메일 발생률이 너무 높은 경우 수신됩니다. 반송 메일 발생률은 Amazon SES 계정에서 생성된 하드 바운스 수를 기준으로 합니다. 반송 메일 발생률이 높은 경우 이메일 공급자는 발신자가 수신자 목록을 올바르게 관리하지 않거나 원치 않는 이메일을 전송한 것일 수도 있다는 신호로 해석합니다.

하드 바운스는 존재하지 않는 주소로 이메일이 전송되었을 때 발생합니다. 소프트 바운스는 수신자 주소가 일시적으로 메시지를 받을 수 없을 때 발생하며, Amazon SES는 이 계산에서 소프트 바운스를 고려하지 않습니다. 확인된 주소 및 도메인으로 전송되어 반송된 이메일과 Amazon SES 받은 편지함 시뮬레이터로 전송하는 이메일도 이 계산에서 고려되지 않습니다.

반송 메일 발생률은 이메일의 대표 볼륨을 기준으로 계산됩니다. 대표 볼륨은 일반적인 전송 실행을 나타내는 이메일의 양입니다. 대량 발신자와 소량 발신자 모두에게 공정을 기하기 위해 대표 볼륨은 계정마다 다르게 적용되며 계정의 전송 패턴 변화에 따라 변경됩니다.

최상의 결과를 얻으려면 반송 메일 발생률을 5% 미만으로 유지하세요. 반송 메일 발생률이 이것보다 높으면 이메일 배달에 영향을 미칠 수 있습니다. 반송 메일 발생률이 5% 이상이면 자동으로 사용자 계정을 검토합니다. 반송 메일 발생률이 10% 이상이면 높은 반송 메일 발생률의 원인이 된 문제를 해결할 때까지 사용자 계정의 추가 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결하기 위해 할 수 있는 조치

반송 메일 및 수신 거부를 확인하여 관리하는 프로세스가 아직 없으면 이러한 프로세스를 마련합니다. 모든 Amazon SES 계정에는 이러한 프로세스가 있어야 합니다. 자세한 내용은 이메일 프로그램을 위한 성공 지표 단원을 참조하십시오.

다음에는 어떤 이메일 주소가 반송되는지 확인하고, 그러한 반송을 최소화하거나 방지하는 계획을 만들어 시행합니다. 계정의 이메일 전송 기능이 이미 일시 중지된 경우에 로그인 AWS Management Console 하고 로 이동합니다 AWS Support. 사용자를 대신하여 개설된 사례에 회신합니다.

사용자 계정이 검토 중인 경우

검토 기간이 끝나도 사용자 계정의 반송 메일 발생률이 계속 10% 이상인 경우에는 문제를 해결할 때까지 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 사례에 대한 응답으로 구현한 변경 사항을 설명합니다. 그러한 조치가 반송 메일 발생률을 줄일 것으로 생각될 경우 해당 조치를 시행한 후에 받은 반송 메일만 계산에 포함시키도록 조정합니다.

사용자 계정의 이메일 전송 기능이 일시 중지된 경우

이 결정을 재고해 줄 것을 요청할 수 있습니다. 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하십시오.

문제를 해결할 것으로 생각되는 변경 사항을 구현할 때 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

수신 거부 발생률 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 수신 거부률 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

이 알림은 계정의 수신 거부 발생률이 너무 높은 경우 수신됩니다. 수신 거부 발생률은 Amazon SES 계정에서 생성된 수신 거부 수를 기준으로 합니다. 수신 거부 발생률이 높은 경우 이메일 공급자는 발신자가 수신자 목록을 올바르게 관리하지 않거나 원치 않는 이메일을 전송한 것일 수도 있다는 신호로 해석합니다.

수신 거부는 발신자가 전송한 이메일을 수신자가 스팸으로 분류할 때 발생합니다. 일반적으로 수신자가 이메일 클라이언트에서 Report Spam(스팸 신고) 버튼을 사용할 때 발생합니다. Amazon SES 받은 편지함 시뮬레이터로 전송되는 이메일에 의해 생성된 수신 거부는 이 계산에서 고려되지 않습니다.

수신 거부 발생률은 이메일의 대표 볼륨을 기준으로 계산됩니다. 대표 볼륨은 일반적인 전송 실행을 나타내는 이메일의 양입니다. 대량 발신자와 소량 발신자 모두에게 공정을 기하기 위해 대표 볼륨은 계정마다 다르게 적용되며 계정의 전송 패턴 변화에 따라 변경됩니다.

최상의 결과를 얻으려면 수신 거부 발생률을 0.1% 미만으로 유지해야 합니다. 수신 거부 발생률이 이것보다 높으면 이메일 배달에 영향을 미칠 수 있습니다. 수신 거부 발생률이 0.1% 이상이면 자동으로 사용자 계정을 검토합니다. 수신 거부 발생률이 0.5% 이상이면 높은 수신 거부 발생률의 원인이 된 문제를 해결할 때까지 사용자 계정의 추가 이메일 전송 기능이 일시 중지될 수 있습니다.

문제를 해결하기 위해 할 수 있는 조치

반송 메일 및 수신 거부를 확인하여 관리하는 프로세스가 아직 없으면 이러한 프로세스를 마련합니다. 모든 Amazon SES 계정에는 이러한 프로세스가 있어야 합니다. 자세한 내용은 이메일 프로그램을 위한 성공 지표 단원을 참조하십시오.

다음에는 어떤 메시지가 불만 제기되었는지 확인하여 이러한 불만 제기를 최소화하기 위한 계획을 시행합니다. 계정의 이메일 전송 기능이 이미 일시 중지된 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다.

불만 제기한 주소의 발송은 즉시 중지해야 하며, 수신자가 불만 제기를 제기하는 요인을 파악하는 것이 중요합니다. 이러한 요인을 파악한 후에는 이메일 발송 동작을 조정하여 문제를 해결하세요.

사용자 계정이 검토 중인 경우

검토 기간이 끝나도 사용자 계정의 수신 거부 발생률이 계속 0.5% 이상인 경우에는 문제를 해결할 때까지 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 사례에 대한 응답으로 구현한 변경 사항을 설명합니다. 그러한 조치가 수신 거부 발생률을 줄일 것으로 생각될 경우 해당 조치를 시행한 후에 받은 수신 거부만 계산에 포함시키도록 조정합니다.

사용자 계정의 이메일 전송 기능이 일시 중지된 경우

이 결정을 재고해 줄 것을 요청할 수 있습니다. 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하십시오.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

스팸 방지 조직 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 스팸 방지 조직 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

대표적인 스팸 방지 조직에서 사용자 Amazon SES 계정에서 전송되는 일부 내용이 원치 않는 내용이거나 문제가 있는 내용인 것으로 플래그가 지정되었음을 보고했습니다.

스팸 방지 조직에서는 사용자의 콘텐츠에 문제가 있다고 플래그를 표시하게 된 특정 메시지에 대한 정보를 제공할 수 없습니다. 이 보고서를 발행한 조직의 이름을 제공해 드릴 수 없습니다. 일반적으로 스팸 방지 기관에서는 수신자의 피드백, 메시지 관련 측정치, 유효하지 않은 주소로의 전송 시도, 스팸 필터를 통해 플래그가 지정된 내용 및 스팸 트랩 적중 수 등과 같은 요인을 조합하여 검토합니다. 이것이 전체 목록은 아니며 다른 요인으로 인해 이러한 조직에서 해당 내용에 플래그를 지정할 수 있습니다.

문제를 해결하기 위해 할 수 있는 조치

이 문제를 해결하려면 이메일 발송 프로그램의 어떤 부분으로 인해 스팸 방지 조직에서 해당 이메일을 문제가 있는 이메일로 플래그를 지정했는지 확인해야 합니다. 그런 다음 이러한 문제를 해결하도록 발송 프로그램을 변경해야 합니다.

사용자 계정이 검토 중인 경우

검토 기간이 끝나도 스팸 방지 조직이 사용자 계정에서 전송된 이메일을 문제가 있는 이메일로 계속 식별하는 경우에는 문제를 해결할 때까지 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 적용한 변경에 대한 세부 정보를 제공하세요. 이 정보를 받으면 검토 기간을 연장하여 사용자가 변경 사항을 구현한 후에 받은 스팸 방지 조직 알림만 분석합니다. 이 연장 검토 기간이 끝난 후에 해당 계정이 더 이상 스팸 방지 기관의 보고서에 포함되지 않으면 계정에 대한 검토 기간을 해제할 수 있습니다.

사용자 계정의 이메일 전송 기능이 일시 중지된 경우

이 결정을 재고해 줄 것을 요청할 수 있습니다. 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하십시오.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정

보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

listbombing 알림

이 섹션에는 Amazon SES 평판 지표 페이지에 표시되는 listbombing 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

스팸 방지 조직에서 이메일 전송 프로세스가 'listbombing'에 취약하다는 것을 확인했습니다. listbombing은 공격자가 웹 기반 양식에 매우 많은 수의 이메일 주소를 등록하는 일종의 침해입니다. listbombing은 영향을 받는 이메일 서비스 사용자에게 서비스 중단을 초래할 수 있습니다. 또한 이메일 공급자에 의한 이메일 차단을 초래할 수도 있습니다.

스팸 방지 조직은 독점 방법을 사용하여 listbombing에 취약한 사이트를 식별합니다. 이러한 이유로 스팸 방지 조직에서 이메일 전송 프로세스를 문제 있는 것으로 식별하게 된 이슈에 대한 추가 세부 정보를 제공할 수 없습니다. 해당 이슈를 식별한 조직의 이름도 공유할 수 없습니다.

문제를 해결하기 위해 할 수 있는 조치

모든 웹 기반 가입 양식을 검토하여 이러한 유형의 침해에 취약하지 않은지 확인해야 합니다. 모든 양식에는 자동화된 스크립트가 구독 요청을 제출하지 못하도록 CAPTCHA가 포함되어야 합니다. 또한 새 사용자가 제품이나 서비스에 가입할 때 실제로 가입하고자 하는지 확인하는 이메일을 보냅니다. 고객이 커뮤니케이션을 명시적으로 옵트인하지 않는 한 고객에게 추가 이메일을 보내지 마세요.

마지막으로 이메일 목록에 대한 '허용 확인(permission pass)'을 수행해야 합니다. 허용 확인을 통해, 모든 고객에게 여전히 이메일을 수신할지 묻는 이메일을 보냅니다. 이메일 수신을 계속 원한다고 확인한 고객에게만 이메일을 보내도록 합니다.

사용자 계정이 검토 중인 경우

검토 기간이 끝나도 스팸 방지 조직이 사용자 계정에서 전송된 이메일을 문제가 있는 이메일로 계속 식별하는 경우에는 문제를 해결할 때까지 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 적용한 변경에 대한 세부 정보를 제공하세요. 이 정보를 받으면 검토 기간을 연장하여 사용자가 변경 사항을 구현한 후에 받은 스팸 방지 조직 알림만 분석합니다. 이 연장 검토 기간이 끝난 후에 해당 계정이 더 이상 스팸 방지 기관의 보고서에 포함되지 않으면 계정에 대한 검토 기간을 해제할 수 있습니다.

사용자 계정의 이메일 전송 기능이 일시 중지된 경우

이 결정을 재고해 줄 것을 요청할 수 있습니다. 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하십시오.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

직접 피드백 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 직접 피드백 알림에 대한 추가 정보가 표시됩니다.

이 알림을 받는 이유

상당히 많은 수의 사용자가 Amazon SES에 직접 연락하여 Amazon SES 계정과 연결된 도메인 또는 주소로부터 받은 메시지를 신고했습니다. 이러한 피드백은 메일박스 제공업체에서 직접 보고한 수신 거부에 표시되지 않으며 평판 지표 페이지에 표시되는 반송 메일 및 수신 거부 지표에 포함되지 않습니다.

이러한 문제를 보고한 사용자의 개인 정보를 보호하기 위해 사용자의 이메일 주소를 제공하지 않습니다.

수신자는 자신이 수신 신청하지 않은 메시지를 수신한 경우, 예상했던 유형이 아닌 다른 메일을 수신한 경우, 수신한 이메일이 유용하지 않거나 관심 분야가 아닌 경우, 자신이 수신 신청한 메시지가 무엇인지 인식하지 못하는 경우, 너무 많은 메시지를 수신한 경우 등의 이유로 Amazon SES에 수신 거부할 수 있습니다. 이 외에도 특정 이메일 발송 프로그램에 따라 해당 사례와 관련된 요인이 있을 수 있습니다.

문제를 해결하기 위해 할 수 있는 조치

목록 작성 및 유지에서 설명한 대로 이중 옵트인 전략을 시행할 것을 권장하며, 새로운 주소를 획득하려는 경우 이중 옵트인 프로세스를 완료한 주소로만 이메일을 발송할 것을 권장합니다.

또한 최근 이메일에 상호 작용하지 않은 주소 목록을 삭제해야 합니다. Amazon SES 전송 활동 모니터링에서 설명한 대로 열기 및 클릭 추적을 사용하여 귀하가 발송한 콘텐츠를 보거나 상호 작용한 사용자를 확인할 수 있습니다.

사용자 계정이 검토 중인 경우

검토 기간이 끝나도 Amazon SES에서 사용자 계정에서 발송한 메시지에 대해 많은 수의 직접 수신 거부를 받을 경우에는 문제를 해결할 때까지 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다. 이러한 조치가 문제를 적절하게 해결할 것으로 생각되면 사용자 계정에 대한 검토 기간을 취소합니다.

사용자 계정의 이메일 전송 기능이 일시 중지된 경우

이 결정을 재고해 줄 것을 요청할 수 있습니다. 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하십시오.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

도메인 차단 목록 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 도메인 차단 목록 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

Amazon SES 계정에서 발송한 이메일에 신뢰할 수 있는 도메인 차단 목록에 추가된 도메인에 대한 참조가 포함되어 있습니다. 이러한 목록에 있는 도메인은 일반적으로 폭력적이거나 악의적인 행위와 관련이 있습니다. 해당 도메인에 사용자가 발송하는 이메일 도메인이 포함될 수도 있고 포함되지 않을 수도 있습니다. 차단 목록의 도메인에 대한 링크나 참조를 포함하는 메시지 또는 그러한 도메인에서 호스팅되는 이미지를 포함하는 메시지에도 플래그가 지정될 수 있습니다.

메시지에 플래그를 지정하게 만든 도메인 이름을 제공할 수 없으며 어떤 이메일이 이러한 방식으로 플래그가 지정되는지 식별할 수 없습니다.

문제를 해결하기 위해 할 수 있는 조치

먼저 Amazon SES를 통해 전송하는 이메일에 참조된 모든 도메인의 목록을 생성합니다. 그런 다음 Spamhaus 도메인 조회 도구를 사용하여 이메일의 도메인 차단 목록에 있는 도메인을 확인합니다. 발송한 이메일에 참조된 도메인 중 두 개 이상이 이 차단 목록에 있을 수 있습니다.

Spamhaus 도메인 차단 목록은 Amazon SES 또는와 연결되어 있지 않습니다 AWS. 이 목록에 있는 도메인의 정확성을 보장할 수 없습니다. 스팸하우스 도메인 차단 목록 및 도메인 조회 도구는 Spamhaus Project의 소유이며 여기서 운영하고 관리합니다.

사용자 계정이 검토 중인 경우

검토 기간 동안 받은 이메일에서 악의적인 목적으로 사용된 도메인에 대한 참조를 찾습니다. 이메일에 이러한 도메인에 대해 많은 수의 참조가 포함된 경우에는 문제를 해결할 때까지 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 적용한 변경에 대한 세부 정보를 제공하세요. 이 정보를 받으면 검토 기간을 연장하여 사용자가 조치를 시행한 후에 이메일에 포함된 차단 도메인 수만 분석합니다. 이 연장 검토 기간이 끝난 후에 도메인 차단 목록 알림 수가 줄거나 없어져서 향후 이 문제가 다시 발생하지 않도록 조치를 취했다고 판단되는 경우 계정에 대한 검토 기간을 취소합니다.

사용자 계정의 이메일 전송 기능이 일시 중지된 경우

이 결정을 재고해 줄 것을 요청할 수 있습니다. 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하십시오.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

내부 검토 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 내부 검토 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

귀하 계정의 포괄적인 검토 결과, 메일박스 제공업체나 수신자가 귀하의 메시지를 스팸으로 간주하게 만들 수 있는 몇 가지 특성이 식별되었습니다.

침해 감지 프로세스를 보호하기 위해 사용자 계정이 이와 같이 플래그가 지정된 특정 요인은 밝힐 수가 없습니다.

이러한 결정의 원인이 되는 몇 가지 요인은 다음과 같습니다.

- 상용 스팸 방지 시스템에서 플래그를 지정한 메시지
- 수신자가 명시적으로 이메일을 요청하지 않았음을 나타내는 메시지 콘텐츠
- 메시지 발신자와 이메일 본문 내 브랜드 간의 불일치
- 발신자가 누구인지 명백하지 않은 콘텐츠
- 원치 않는 이메일과 관련된 콘텐츠를 다루는 발신 메시지
- 원치 않는 이메일과 관련된 포맷 패턴
- 평판이 나쁜 도메인에서 발송하거나 그러한 도메인을 참조함

포괄적이지 않은 목록 이 외에도 이러한 요인이 조합되어 이 알림의 원인이 될 수도 있고 여기에 나와 있지 않은 요인도 있을 수 있습니다.

문제를 해결하기 위해 할 수 있는 조치

다음 제안은 이 문제의 심각성을 최소화하는 데 도움을 줍니다.

- 귀하가 발송하는 이메일을 받겠다고 명시적으로 요청한 사람에게만 이메일을 보내세요.
- 이메일 수신자 목록을 구입하거나 임차하거나 차용하지 마세요.
- 전송하는 메시지에 사용자의 신원이나 해당 커뮤니케이션의 목적을 숨기지 마세요.
- Amazon SES를 통해 발송한 이메일에 참조된 모든 도메인 목록을 생성한 다음 https://www.spamhaus.org/lookup/에서 스팸하우스 도메인 조회 도구를 사용하여 해당 도메인이 스팸하우스 도메인 차단 목록에 있는지 확인합니다.
- 이메일을 작성할 경우 다음과 같은 업계 모범 사례를 따르세요.

이 외에도 많은 모범 사례가 있지만 이메일에 플래그가 지정되는 가장 일반적인 몇 가지 요인을 식별하는 데 도움이 될 것입니다.

Spamhaus 도메인 차단 목록은 Amazon SES 또는와 연결되어 있지 않습니다 AWS. 이 목록에 있는 도메인의 정확성을 보장할 수 없습니다. 스팸하우스 도메인 차단 목록 및 도메인 조회 도구는 Spamhaus Project의 소유이며 여기서 운영하고 관리합니다.

사용자 계정이 검토 중이거나 사용자 계정의 이메일 전송 기능이 일시 중지된 경우

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다. 이러한 조치가 문제를 적절하게 해결할 것으로 생각되면 검토 기간을 취소하거나 사용자 계정에서 전송 일시 중지를 제거합니다.

검토 기간을 해제하거나 사용자 계정에서 전송 일시 중지를 제거하여 나중에 같은 문제가 발견되면 다시 사용자 계정을 검토하거나 계정의 이메일 전송 기능을 일시 중지할 수 있습니다. 극단적인 경우 또는 같은 문제가 반복적으로 발생하는 경우 사용자 계정의 이메일 전송 기능을 영구적으로 중지할 수 있습니다.

사용자 계정이 검토 중이거나 계정의 이메일 전송 기능이 일시 중지된 경우 수행할 작업에 대한 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하세요.

메일박스 제공업체 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 메일박스 제공업체 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

유명 메일박스 제공업체에서 사용자의 Amazon SES 계정과 관련된 도메인이나 주소로부터 원치 않는 이메일이나 악성 이메일이 발송되었다고 보고했습니다.

이 보고서를 발행한 조직의 신원은 제공해 드릴 수 없습니다. 또한 메일박스 제공업체에서 보고서를 발행하게 만든 특정 요인에 대한 정보를 가지고 있지 않습니다. 일반적으로 메일박스 제공업체에서는 고객 피드백, 고객 관련 측정치, 유효하지 않은 주소로의 전송 시도, 스팸 필터에서 플래그를 지정한 콘텐츠 등에 따라 이러한 결정을 내립니다. 이 외에도 메일박스 제공업체에서 사용자 콘텐츠에 플래그를 지정한 다른 원인이 있을 수 있습니다.

문제를 해결하기 위해 할 수 있는 조치

이 문제를 해결하려면 이메일 발송 프로그램의 어떤 부분으로 인해 메일박스 제공업체에서 해당 이메일을 문제가 있는 이메일로 플래그를 지정했는지 확인해야 합니다. 그런 다음 이러한 문제를 해결하도록 발송 프로그램을 변경해야 합니다.

사용자 계정이 검토 중인 경우

검토 기간이 끝나도 메일박스 제공업체가 사용자 계정에서 발송한 이메일을 문제가 있는 이메일로 계속 식별하는 경우에는 문제를 해결할 때까지 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 적용한 변경에 대한 세부 정보를 제공하세요. 이 정보를 받으면 검토 기간을 연장하여 사용자가 조치를 시행한 후에 받은 메일박스 제공업체 알림 수만 분석합니다. 이 연장 검토 기간이 끝난 후에 메일박스 제공업체에서 더 이상 사용자 계정을 문제 계정으로 보고하지 않을 경우 계정에 대한 검토를 해제할 수 있습니다.

사용자 계정의 이메일 전송 기능이 일시 중지된 경우

이 결정을 재고해 줄 것을 요청할 수 있습니다. 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하십시오.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

수신자 피드백 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 수신자 피드백 알림에 대한 추가 정보가 표시됩니다.

이 알림을 받는 이유

유명 메일박스 제공업체에서 많은 수의 사용자가 Amazon SES 계정에서 발송한 이메일을 원치 않는 것으로 보고했습니다. 이 유형의 피드백은 메일박스 제공업체에서 직접 보고한 수신 거부에 표시되지 않으며 Amazon SES 반송 메일 및 수신 거부 알림에 포함되지 않습니다.

수신 거부 수가 많으면 모든 Amazon SES 사용자에게 부정적인 영향을 줄 수 있습니다. 사용자 평판과 다른 Amazon SES 고객의 평판을 보호하기 위해 계정에서 특정 수의 수신 거부를 받는 경우 즉각적인 조치가 시행됩니다.

귀하의 이메일을 원치 않는 것으로 보고한 특정 이메일 주소 목록을 제공할 수 없습니다. 또한 이 문제를 당사에 보고한 메일박스 제공업체 이름을 알려드릴 수 없습니다.

문제를 해결하기 위해 할 수 있는 조치

이 문제를 해결하려면 이메일 발송 프로그램의 어떤 부분으로 인해 수신자가 사용자의 이메일 메시지에 대해 수신을 거부하는지 확인해야 합니다. 이러한 요인을 파악한 후에는 이메일 발송 체제를 변경하여 문제를 시정하세요.

새 주소를 획득하려면 목록 작성 및 유지에서 설명한 대로 이중 옵트인 전략을 시행할 것을 권장합니다. 이중 옵트인 프로세스를 완료한 주소로만 이메일을 발송할 것을 권장합니다.

또한 최근 이메일에 상호 작용하지 않은 주소 목록을 삭제해야 합니다. Amazon SES 전송 활동 모니터링에서 설명한 대로 열기 및 클릭 추적을 사용하여 귀하가 발송한 콘텐츠를 보거나 상호 작용한 사용자를 확인할 수 있습니다.

사용자 계정이 검토 중인 경우

검토 기간이 끝나도 메일박스 제공업체가 많은 수의 수신 거부를 계속 보고하는 경우에는 문제를 해결할 때까지 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 적용한 변경에 대한 세부 정보를 제공하세요. 이 정보를 받으면 검토 기간을 연장하여 사용자가 조치를 시행한 후에 받은 메일박스 제공업체 수신 거부의 수만 분석합니다. 이 연장 검토 기간이 끝나고 메일박스 제공업체 수신 거부 수가 줄거나 없어진 경우 사용자 계정에 대한 검토를 해제할 수 있습니다.

사용자 계정의 이메일 전송 기능이 일시 중지된 경우

이 결정을 재고해 줄 것을 요청할 수 있습니다. 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하십시오.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

관련 계정 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 간련 계정 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

다른 Amazon SES 계정에서 발송한 이메일과 관련하여 심각한 문제를 감지했습니다. 문제가 있는 계정은 사용자와 관련이 있다고 생각 AWS 계정하므로 유사한 문제를 방지하기 위한 조치를 취했습니다.

문제를 해결하기 위해 할 수 있는 조치

계정의 이메일 전송 기능이 일시 중지되면 전송 일시 중지의 이유에 대한 정보가 해당 계정의 소유자에게 항상 전송됩니다. 자세한 내용은 관련 계정의 소유자에게 전송된 이메일을 참조하세요.

먼저 관련 계정에 대한 문제를 해결해야 합니다. 문제를 해결할 것으로 생각되는 변경 사항을 구현한 후 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다. 이러한 조치가 문제를 적절하게 해결할 것으로 생각되면 검토 기간을 취소하거나 사용자 계정에서 전송 일시 중지를 제거합니다.

스팸 트랩 알림

이 단원에는 Amazon SES 평판 대시보드에 표시되는 스팸 트랩 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

타사 스팸 방지 조직에서 최근 스팸 트랩 주소에 사용자 Amazon SES 계정과 연결된 확인 주소 또는 도메인으로부터 이메일이 수신되었다고 보고했습니다.

스팸 트랩은 원치 않는 미끼 이메일(스팸)을 보내는 데 전용으로 사용되는 휴면 이메일 주소입니다. 스팸 트랩 보고 수가 많으면 모든 Amazon SES 사용자에게 부정적인 영향을 줄 수 있습니다. 사용자 평판과 다른 Amazon SES 고객의 평판을 보호하기 위해 계정에서 특정 수의 이메일을 스팸 트랩 주소로 보내는 경우 즉각적인 조치가 시행됩니다.

문제를 해결하기 위해 할 수 있는 조치

해당 스팸 트랩과 연결된 이메일 주소를 밝힐 수 없습니다. 이러한 주소는 해당 주소를 소유한 조직에서 철저히 감시하며 주소가 일단 확인되면 가치가 없어집니다.

스팸 트랩으로 이메일을 보내는 것은 일반적으로 귀하가 고객의 이메일 주소를 획득하는 방식에 문제가 있음을 나타냅니다. 예를 들어 구입한 이메일 주소 목록에 스팸 트랩 주소가 들어 있을 수 있으며 그래서 Amazon SES 서비스 약관에서는 구매하거나 임차한 목록으로 이메일을 보내는 것을 금지합니다. 새 주소를 획득하려면 목록 작성 및 유지에서 설명한 대로 이중 옵트인 전략을 시행할 것을 권장합니다. 이중 옵트인 프로세스를 완료한 주소로만 이메일을 발송할 것을 권장합니다.

또한 최근 이메일에 상호 작용하지 않은 주소 목록을 삭제해야 합니다. Amazon SES 전송 활동 모니터링에서 설명한 대로 열기 및 클릭 추적을 사용하여 귀하가 발송한 콘텐츠를 보거나 상호 작용한 사용자를 확인할 수 있습니다.

사용자 계정이 검토 중인 경우

검토 기간이 끝나도 메시지가 사용자 계정에서 스팸 트랩 주소로 계속 전송되는 경우에는 문제를 해결할 때까지 사용자 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 적용한 변경에 대한 세부 정보를 제공하세요. 이 정보를 받으면 검토 기간을 연장하여 사용자가 조치를 시행한 후에 받은 스팸 트랩 보고 수만 분석합니다. 이 연장 검토 기간이 끝나고 스팸 트랩 보고 수가 줄거나 없어진 경우 사용자 계정에 대한 검토를 해제할 수 있습니다.

사용자 계정의 이메일 전송 기능이 일시 중지된 경우

이 결정을 재고해 줄 것을 요청할 수 있습니다. 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하십시오.

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

취약 사이트 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 취약 사이트 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

포괄적인 검토 끝에 사용자의 계정에서 사용자가 전송하려고 한 것이 아닌 메시지가 전송된 것이 확인되었습니다. 이러한 메시지는 메일박스 제공업체 및 수신자가 스팸으로 플래그를 지정할 확률이 높습니다.

대부분의 이러한 상황은 제3자가 귀하의 웹 사이트 기능을 침해하여 원치 않는 이메일을 보내는 것입니다. 예를 들어 귀하의 웹 사이트에 "친구에게 이메일로 보내기", "문의하기", "친구 초대" 등의 기능이 있는 경우 제3자가 이 기능을 사용하여 원치 않는 이메일을 보낼 수 있습니다.

문제를 해결하기 위해 할 수 있는 조치

먼저 웹 사이트나 애플리케이션에서 타사가 사용자 승인 없이 Amazon SES를 사용하여 이메일을 보낼 수 있는 기능을 확인하십시오. 지원 센터의 경우 이러한 방식으로 전송된 것으로 판단되는 메시지의 샘플을 요청할 수 있습니다.

그런 다음 애플리케이션이나 웹 사이트를 수정하여 원치 않는 발송을 방지하세요. 예를 들어 CAPTCHA를 추가하고, 이메일을 발송할 수 있는 비율을 제한하고, 사용자가 사용자 지정 콘텐츠를 제출할 수 있는 기능을 제거하고, 사용자에게 로그인하여 이메일을 보내도록 요청하고, 동시에 여러 알림을 생성하는 애플리케이션 기능을 제거하세요.

사용자 계정이 검토 중이거나 사용자 계정의 이메일 전송 기능이 일시 중지된 경우

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

검토 기간을 해제하거나 사용자 계정에서 전송 일시 중지를 제거하여 나중에 같은 문제가 발견되면 다시 사용자 계정을 검토하거나 계정의 이메일 전송 기능을 일시 중지할 수 있습니다. 극단적인 문제가 나타나거나 같은 문제가 반복적으로 발생하는 경우 사용자 계정의 이메일 전송 기능을 영구적으로 중지할 수 있습니다.

사용자 계정이 검토 중이거나 계정의 이메일 전송 기능이 일시 중지된 경우 수행할 작업에 대한 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하세요.

손상된 보안 인증 정보 알림

이 섹션에는 Amazon SES 평판 지표 페이지에 표시되는 손상된 보안 인증 정보 사이트 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

포괄적인 검토 끝에 사용자의 계정에서 사용자가 전송하려고 한 것이 아닌 메시지가 전송된 것이 확인되었습니다. 이러한 메시지는 메일박스 제공업체 및 수신자가 스팸으로 플래그를 지정할 확률이 높습니다.

몇 가지 일반적인 원인은 손상된 IAM 액세스 키, 손상된 SMTP 암호 또는 기타 보안 취약점입니다.

문제를 해결하기 위해 할 수 있는 조치

SES 사용 메커니즘에 대한 포괄적인 보안 검토를 수행해야 합니다. 해당 암호 또는 SMTP 암호를 교체하고 계정에서 승인되지 않은 사용자 또는 리소스를 제거합니다. 서드 파티 웹 사이트 또는 리포지토리에 암호나 액세스 키와 같은 기밀 정보를 저장하고 있지 않은지 확인합니다. 이제 IAM 액세스 키를 사용자에게 사용하지 않고 루트 사용자에게는 절대 사용하지 않는 것이 좋습니다. 아직 사용 중이라면 AWS IAM Identity Center에서 사용자를 생성하는 등의 임시 보안 인증 정보를 제공하는 메커니즘으로 마이그레이션해야 합니다.

사용자 계정이 검토 중이거나 사용자 계정의 이메일 전송 기능이 일시 중지된 경우

문제를 해결할 것으로 생각되는 변경 사항을 구현한 경우 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 이 문제를 해결하기 위해 시행한 조치의 세부 정보뿐만 아니라 이 문제가 다시 발생하지 않도록 하기 위한 계획의 세부 정보를 포함합니다. 요청이 접수되면 제공된 정보를 검토하고 필요한 경우 계정 상태를 변경합니다.

검토 기간을 해제하거나 사용자 계정에서 전송 일시 중지를 제거하여 나중에 같은 문제가 발견되면 다시 사용자 계정을 검토하거나 계정의 이메일 전송 기능을 일시 중지할 수 있습니다. 극단적인 문제가 나타나거나 같은 문제가 반복적으로 발생하는 경우 사용자 계정의 이메일 전송 기능을 영구적으로 중지할 수 있습니다.

사용자 계정이 검토 중이거나 계정의 이메일 전송 기능이 일시 중지된 경우 수행할 작업에 대한 자세한 내용은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하세요.

기타 알림

이 단원에는 Amazon SES 평판 지표 페이지에 표시되는 기타 알림에 대한 추가 정보가 포함되어 있습니다.

이 알림을 받는 이유

자동 검토 또는 수동 검토를 통해 이 문서의 이전 단원에 나와 있지 않은 문제가 확인되었습니다.

문제를 해결하기 위해 할 수 있는 조치

특정 문제에 대한 자세한 내용은 사용자를 대신하여 개설된 지원 센터 사례를 참조하세요. 지원 센터에 액세스하려면에 로그인 AWS Management Console 한 다음 지원 센터를 선택합니다. 사례에 대한 응답으로 구현한 변경 사항을 설명합니다. 특정 상황 및 발견된 문제의 특성에 따라 검토 기간을 종료하거나 사용자 계정의 이메일 전송 기능을 복원할 수 있습니다.

CloudWatch를 사용하여 평판 모니터링 경보 생성

Amazon SES는 일련의 평판 관련 지표를 Amazon CloudWatch에 자동으로 게시합니다. 이러한 측정치를 사용하여 반송 메일이나 수신 거부율이 사용자 계정의 이메일 전송 기능에 영향을 줄 수 있는 수준에 도달할 때 알리는 경보를 만들 수 있습니다.

> **Note**
>
> 이 섹션 절차의 CloudWatch 부분은 SES 발신자 평판을 모니터링하기 위해 CloudWatch 경보를 설정하는 핵심 단계만 제시하기 위한 것입니다. CloudWatch 경보에 대한 설정 옵션과 관련된 고급 구성은 살펴보지 않습니다. CloudWatch 경보 구성에 대한 전체 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch 경보 사용을 참조하세요.

사전 조건

- Amazon SNS 주제를 생성한 다음 원하는 엔드포인트(예: 이메일 또는 SMS)를 사용하여 해당 주제를 구독합니다. 자세한 내용은 Amazon Simple Notification Service 개발자 가이드의 Amazon SNS 주제 생성 및 Amazon SNS 주제 구독을 참조하세요.
- 현재 리전에서 이메일을 전송한 적이 없는 경우 SES 네임스페이스가 표시되지 않을 수 있습니다. 지표가 있는지 확인하려면 메일박스 시뮬레이터로 테스트 이메일을 보냅니다.

전송 평판을 모니터링하는 CloudWatch 경보 생성

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/ses/ Amazon SES 콘솔을 엽니다.
2. 화면 왼쪽에 있는 탐색 창에서 Reputation metrics(평판 지표)를 클릭합니다.
3. 계정 수준 탭의 평판 지표 페이지에서, 반송 발생률 또는 수신 거부 발생률 창의 CloudWatch에서 보기를 선택하면 선택한 지표와 함께 CloudWatch 콘솔이 열립니다.
4. 그래프 지표(Graphed metrics) 탭 아래에 있는 선택한 지표의 라인(예: Reputation.BounceRate)에서, 작업(Actions)열(아래 이미지 참조)의 경보 벨(alarm bell) 아이콘을 선택하면 지표 및 조건 지정 페이지가 열립니다.

5. 조건(Conditions) 창까지 아래로 스크롤하여 임계값 유형(Threshold type) 필드에서 정적(Static)을 선택합니다.

   a. ##가 다음과 같을 때마다(Whenever metric is...) 필드에서 크거나 같음(Greater/Equal)을 선택합니다.

   b. 다음보다(than...) 필드에서 CloudWatch에 경보를 발생시켜야 하는 값을 지정합니다.

      - 반송 메일 발생률을 모니터링하기 위한 경보를 만들 경우, Amazon SES는 반송 메일 발생률을 5% 미만으로 유지할 것을 권장합니다. 해당 계정의 반송 메일 발생률이 10%가 넘으면 계정의 이메일 전송 기능이 일시 중지될 수 있습니다. 따라서 해당 계정의 반송 메일 발생률이 0.05(5%) 이상일 때 알림이 전송되도록 CloudWatch를 구성해야 합니다.
      - 수신 거부율을 모니터링하기 위한 경보를 만들 경우, Amazon SES는 수신 거부율을 0.1% 미만으로 유지할 것을 권장합니다. 해당 계정의 수신 거부율이 0.5%가 넘으면 계정의 이메일 전송 기능이 일시 중지될 수 있습니다. 따라서 해당 계정의 수신 거부율이 0.001(0.1%) 이상일 경우 알림이 전송되도록 CloudWatch를 구성해야 합니다.

   c. 추가 구성(Additional configuration)을 확장하고 누락된 데이터 처리(Missing data treatment) 필드에서 누락된 데이터를 무시로 처리(경보 상태 유지(Treat missing data as ignore(maintain the alarm state))를 선택합니다.

   d. 다음을 선택합니다.

6. 작업 구성(Configure actions) 창의 경보 상태 트리거(Alarm state trigger) 필드에서 경보 상태(In Alarm)를 선택합니다.

   a. SNS 주제 선택(Select an SNS topic)에서 기존 SNS 주제 선택(Select an existing SNS topic)을 선택합니다.
   b. 알림 전송 대상(Send notification to…) 검색 상자에서 사전 조건에서 만들어 구독한 주제를 선택합니다.
   c. 다음을 선택합니다.
7. 이름 및 설명 추가(Add name and description) 창에 경보의 이름과 설명을 입력하고 다음(Next)을 선택합니다.
8. 미리 보기 및 생성(Preview and create) 창에서 설정을 확인하고 만족스러운 경우 경보 생성(Create alarm)을 선택합니다. 변경할 항목이 있는 경우 돌아가서 편집할 각 섹션에 대해 이전(Previous) 버튼을 선택합니다.

전용 IP에 대한 SNDS 지표

Amazon SES를 사용하는 각에서 임대된 전용 IP 주소에 대한 스마트 네트워크 데이터 서비스(SNDS) 데이터를 볼 수 AWS 리전 있습니다. 이 SNDS 데이터는 Amazon CloudWatch 콘솔에서 확인할 수 있습니다.

SNDS는 IP 소유자가 IP 공간 내에서 스팸을 방지할 수 있도록 하는 Outlook 프로그램입니다. Amazon SES는 전용 IP를 임대하는 사용자를 위해 이 중요한 데이터를 제공합니다. SNDS 데이터는 IP의 메일 전송 동작에 대해 인사이트를 제공하고 발신자 평판에 대한 우려 사항을 불러옵니다.

> **Note**
>
> Outlook을 참조할 때 추적하는 모든 도메인을 다룹니다. 예를 들어, Hotmail.com, Outlook.com 및 Live.com이 포함될 수 있습니다.

전용 IP 주소에 대한 SNDS 데이터 보기

1. https://console.aws.amazon.com/cloudwatch/에서 Amazon CloudWatch 콘솔에 로그인합니다.
2. 탐색 창에서 지표(Metrics)를 확장한 다음 모든 지표(All metrics)를 선택합니다.

   (새로운 CloudWatch 콘솔 인터페이스에 대한 지침이 제공됩니다.)
3. 지표 컨테이너의 찾아보기 탭에서을 선택한 AWS 리전다음 SES를 선택합니다.
4. IP 지표(IP Metrics)를 선택하면 SNDS에서 추적한 모든 전용 IP가 표시됩니다.

(참고: 선택한 리전에서 계정과 연결된 전용 IP 주소가 없는 경우 IP 지표(IP Metrics)가 CloudWatch 콘솔에 표시되지 않습니다.)

5. 이 목록에서 SNDS가 추적한 전용 IP를 모두 보거나 개별 IP 주소를 선택하여 해당 지표만 볼 수 있습니다.

다음 지표는 각 전용 IP 주소에 대해 제공되며 Outlook에서 정의합니다. 자세한 내용은 Outlook의 SNDS FAQ를 참조하세요.

> **Note**
>
> 이러한 지표는 하루에 한 번 업데이트된 데이터를 제공하는 활동 기간을 나타냅니다. 또한 지표에는 24시간 기간을 반영하는 해당 타임스탬프가 있습니다.

- SNDS.RCPTCommands - 활동 기간 동안 SNDS에서 특정 IP 주소에 대해 인식하는 RCPT 명령 수입니다. RCPT 명령은 메일을 보내는 데 사용되는 SMTP 프로토콜의 일부로, 이메일을 전송하려는 수신자 주소를 지정합니다.
- SNDS.DATACommands - 활동 기간 동안 SNDS에서 특정 IP 주소에 대해 인식하는 DATA 명령 수입니다. DATA 명령은 메일을 보내는 데 사용되는 SMTP 프로토콜의 일부입니다. 특히 이전에 설정된 수신자에게 실제로 메시지를 전송하는 부분입니다.
- SNDS.MessageRecipients - 활동 기간 동안 특정 IP 주소에 대해 SNDS가 인식한 메시지의 수신자 수입니다.
- SNDS.SpamRate - 지정된 활동 기간 동안 IP 주소가 보낸 모든 메시지에 적용된 스팸 필터링의 집계 결과를 표시합니다.
  - 스팸률이 0이면 IP 주소의 스팸 발생률이 10% 미만임을 의미합니다.
  - 스팸률이 0.5이면 IP 주소에서 10% 에서 90% 사이의 스팸이 발생함을 의미합니다.
  - 스팸률이 1이면 IP 주소에서 90% 이상의 스팸이 발생함을 의미합니다.
- SNDS.ComplaintRate - 활동 기간 동안 Outlook 사용자가 IP에서 받은 메시지에 대해 수신 거부한 횟수의 비율입니다.
  - 수신 거부 1은 100% 수신 거부 발생률을 의미합니다.
  - 예를 들어, 0.05는 5%의 수신 거부 발생률을 의미합니다.
  - 0은 수신 거부 발생률이 0.1% 미만임을 의미합니다.

- SNDS.TrapHits - '트랩 계정'으로 전송된 메시지 수를 표시합니다. 트랩 계정은 메일을 요청하지 않는 Outlook에서 유지 관리하는 계정입니다. 따라서 트랩 계정으로 전송된 메시지는 스팸일 가능성이 큽니다.

문제 해결

Q1. 데이터가 매일 기록되지 않는 이유는 무엇인가요? 다음 중 하나의 경우일 수 있습니다.

- SNDS 데이터는 Outlook의 SNDS 프로그램에 따라 다릅니다.
- SNDS가 값을 계산하기 위해 수신해야 하는 이메일의 최소 임계값이 있습니다. IP의 이메일 볼륨이 부족한 경우에는 데이터가 없을 수 있습니다.

Q2. SNDS.SpamRate 및 SNDS.ComplaintRate 지표가 변경되는 이유는 무엇이며, 값이 1로 변경되면 어떻게 해야 하나요?

이것은 전송 동작의 무언가가 Outlook SNDS 프로그램에서 부정적인 응답을 트리거했음을 나타냅니다. 이 경우 전역적 문제가 아닌지 확인하기 위해 다른 인터넷 서비스 제공업체(ISP) 및 발생 횟수를 확인하려고 합니다. 전역적 문제인 경우 여러 ISP에서 문제를 발견하게 될 것이며, 이는 목록, 콘텐츠, 배포 또는 사용 권한 문제일 수 있습니다. Outlook과 관련이 있는 경우 Outlook으로 전송하는 최선의 방법을 확인해 보세요.

Q3. SNDS.SpamRate 값이 0(또는 0.5)에서 1로 변경되면 어떻게 AWS Support 되나요?

AWS 는 SNDS를 제어할 수 없으므로 SNDS에 영향을 미치지 않습니다. 모든 완화 조치 요청은 새 지원 요청 양식을 통해 Outlook에 직접 제출해야 합니다.

이메일 전송 자동 일시 중지

발신자 평판을 보호하기 위해 특정 구성 세트를 사용하여 전송된 메시지 또는 특정 AWS 리전의 Amazon SES 계정에서 전송된 모든 메시지에 대해 이메일 전송을 일시적으로 일시 중지할 수 있습니다.

Amazon CloudWatch 및 Lambda를 사용하면 평판 지표(예: 반송 메일 발생률 또는 수신 거부 발생률)가 특정 임계값을 초과할 때 이메일 전송을 자동으로 일시 중지하는 솔루션을 생성할 수 있습니다. 이 주제에는 이러한 솔루션을 설정하는 절차가 포함되어 있습니다.

이 섹션의 주제:

- 전체 Amazon SES 계정에 대한 이메일 전송 자동 일시 중지

- 구성 집합에 대한 이메일 전송 자동 일시 중지

전체 Amazon SES 계정에 대한 이메일 전송 자동 일시 중지

이 섹션의 절차에서는 Amazon SES, Amazon SNS, Amazon CloudWatch 및 AWS Lambda 을(를) 설정하여 단일 AWS 리전에 있는 Amazon SES 계정에 대한 이메일 전송을 자동으로 일시 중지하는 단계를 설명합니다. 여러 리전으로부터 이메일을 보내려면, 이 방법을 구현하고자 하는 리전마다 이 섹션의 절차를 반복하세요.

이 섹션의 주제:

- 1부 - IAM 역할 만들기
- 2부: Lambda 함수 만들기
- 3부: 계정에 대한 이메일 전송 재활성화
- 4부: Amazon SNS 주제 생성 및 구독
- 5부: CloudWatch 경보를 생성하려면
- 6부: 솔루션 테스트

1부 - IAM 역할 만들기

이메일 전송 자동 일시 중지를 구성하는 첫 단계는 `UpdateAccountSendingEnabled` API 작업을 실행할 수 있는 IAM 역할을 만드는 것입니다.

IAM 역할을 만들려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 역할 만들기를 선택합니다.
4. Select trusted entity(신뢰할 수 있는 엔터티 선택) 페이지의 Trusted entity type(신뢰할 수 있는 엔터티 유형) 아래에서 AWS service를 선택합니다.
5. 사용 사례(Use case)에서 Lambda를 선택한 후 다음(Next)을 선택합니다.
6. 권한 추가(Add permissions) 페이지에서 다음 정책을 선택합니다.
   - AWSLambdaBasicExecutionRole
   - AmazonSESFullAccess

> **Tip**
>
> 권한 정책(Permission policies) 아래의 검색 상자를 사용하여 이러한 정책을 빠르게 찾을 수 있지만 첫 번째 정책을 검색하고 선택한 후에는 두 번째 정책을 검색하고 선택하기 전에 필터 지우기(Clear filters)를 선택해야 합니다.

그런 다음 다음을 선택합니다.

7. 이름 지정, 검토 및 만들기(Name, review, and create) 페이지의 역할 세부 정보(Role details)에서 역할 이름(Role name) 필드에 정책의 의미 있는 이름을 입력합니다.
8. 선택한 두 정책이 권한 정책 요약(Permissions policy summary) 테이블에 나열되어 있는지 확인한 다음 역할 만들기(Create role)를 선택합니다.

2부: Lambda 함수 만들기

IAM 역할을 만든 후에는 계정에 대한 이메일 전송을 일시 중지시키는 Lambda 함수를 만들 수 있습니다.

Lambda 함수를 만들려면

1. https://console.aws.amazon.com/lambda/ AWS Lambda 콘솔을 엽니다.
2. 리전 선택기를 사용하여 이 Lambda 함수를 배포할 리전을 선택합니다.

> **Note**
>
> 이 함수는이 단계에서 선택한 AWS 리전에서만 이메일 전송을 일시 중지합니다. 둘 이상의 리전으로부터 이메일을 보내는 경우, 이메일 전송을 자동으로 일시 중지하려는 리전마다 이 섹션의 절차를 반복하세요.

3. 함수 만들기(Create function)를 선택합니다.
4. 함수 만들기에서 Author from scratch를 선택합니다.
5. 기본 정보(Basic information)에서 다음 단계를 완료합니다.
   - 함수 이름(Function name)에 Lambda 함수의 이름을 입력합니다.
   - 런타임에서 Node.js 18x(또는 선택 목록에 현재 제공된 버전)를 선택합니다.

- 아키텍처(Architecture)에서 미리 선택된 기본값 x86_64를 유지합니다.
- 권한(Permissions)에서 기본 실행 역할 변경(Change default execution role)을 확장하고 기존 역할 사용(Use an existing role)을 선택합니다.
- 기존 역할(Existing role) 목록 상자 내부를 클릭하고 the section called "1부 - IAM 역할 만들기"에서 생성한 IAM 역할을 선택합니다.

그 다음 함수 만들기(Create function)를 선택합니다.

6. 코드 편집기의 코드 소스(Code source)에 다음 코드를 붙여 넣습니다.

```
'use strict';


const { SES } = require("@aws-sdk/client-ses")

// Create a new SES object.

var ses = new SES({});

// Specify the parameters for this operation. In this case, there is only one
// parameter to pass: the Enabled parameter, with a value of false
// (Enabled = false disables email sending, Enabled = true enables it).
var params = {
    Enabled: false
};

exports.handler = (event, context, callback) => {
    // Pause sending for your entire SES account
    ses.updateAccountSendingEnabled(params, function(err, data) {
        if(err) {
            console.log(err.message);
        } else {
            console.log(data);
        }
    });
};
```

그런 다음 배포를 선택합니다.

7. 테스트를 선택합니다. 테스트 이벤트 구성(Configure test event) 창이 나타나면 이벤트 이름(Event name) 필드에 이름을 입력한 후 저장(Save)을 선택합니다.

8. 테스트(Test) 드롭 상자를 확장하고 방금 생성한 이벤트의 이름을 선택한 후 테스트(Test)를 선택합니다.
9. 실행 결과(Execution results) 탭이 나타납니다. 바로 아래 오른쪽에 `Status: Succeeded`가 표시되는지 확인합니다. 함수가 실행되지 못하면 다음을 수행합니다.
   - the section called "1부 - IAM 역할 만들기"에서 생성한 IAM 역할에 올바른 정책이 포함되는지 확인합니다.
   - Lambda 함수의 코드에 오류가 없는지 확인합니다. Lambda 코드 편집기가 구문 오류 및 다른 잠재적인 문제를 자동으로 강조 표시합니다.

3부: 계정에 대한 이메일 전송 재활성화

the section called "2부: Lambda 함수 만들기"에서 Lambda 함수를 테스트할 때의 부작용은 Amazon SES 계정에 대한 이메일 전송이 일시 중지되는 것입니다. 대부분의 경우 CloudWatch 경보가 트리거될 때까지 계정에 대한 전송을 일시 중지하지 않습니다.

이 섹션의 절차에서는 Amazon SES 계정에 대한 이메일 전송을 다시 활성화합니다. 이 절차를 완료하려면 AWS Command Line Interface을 설치하고 구성해야 합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

이메일 전송을 재활성화하려면

1. 명령줄에 다음 명령을 입력하여 계정에 대한 이메일 전송을 다시 활성화합니다. *sending_region*을 이메일 전송을 재활성화할 리전의 이름으로 바꿉니다.

```
aws ses update-account-sending-enabled --enabled --region sending_region
```

2. 명령줄에 다음 명령을 입력하여 계정에 대한 이메일 전송 상태를 확인합니다.

```
aws ses get-account-sending-enabled --region sending_region
```

   다음 결과가 표시되면 계정에 대한 이메일 전송이 성공적으로 재활성화된 것입니다.

```
{
    "Enabled": true
}
```

4부: Amazon SNS 주제 생성 및 구독

경보가 트리거될 때 CloudWatch가 Lambda 함수를 실행하려면 먼저 Amazon SNS 주제를 생성하고 해당 Lambda 함수를 구독해야 합니다.

Amazon SNS 주제를 생성하고 Lambda 함수로 구독

1. https://console.aws.amazon.com/sns/v3/home에서 Amazon SNS 콘솔을 엽니다.
2. Amazon Simple Notification Service 개발자 안내서의 단계에 따라 주제를 생성합니다.
   - Type(유형)은 Standard(표준)(FIFO 아님)여야 합니다.
3. Amazon Simple Notification Service 개발자 안내서의 단계에 따라 주제를 구독합니다.
   a. 프로토콜에서 AWS Lambda를 선택합니다.
   b. 엔드포인트에서 the section called “2부: Lambda 함수 만들기”에서 생성한 Lambda 함수를 선택합니다.

5부: CloudWatch 경보를 생성하려면

이 섹션에는 지표가 특정 임계값에 도달하면 트리거되는 CloudWatch에 경보를 생성하기 위한 절차가 포함되어 있습니다. 경보가 트리거되면 the section called “4부: Amazon SNS 주제 생성 및 구독”에서 생성한 Amazon SNS 주제에 알림을 전달한 후 the section called “2부: Lambda 함수 만들기”에서 생성한 Lambda 함수를 실행합니다.

CloudWatch 경보를 생성하려면

1. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 리전 선택기를 사용하여 이메일 전송을 자동으로 일시 중지하려는 리전을 선택합니다.
3. 탐색 창에서 경보(Alarms)를 선택하세요.
4. 경보 생성을 선택합니다.
5. [Create Alarm] 창의 [SES Metrics] 아래에서 [Account Metrics]를 선택합니다.
6. [Metric Name] 아래에서 다음 옵션 중 하나를 선택합니다.
   - Reputation.BounceRate – 계정의 전체 하드 바운스 발생률이 사용자가 정의하는 임계값을 초과할 때 계정에 대한 이메일 전송을 일시 중지하려면 이 지표를 선택합니다.
   - Reputation.ComplaintRate – 계정의 전체 수신 거부 발생률이 사용자가 정의하는 임계값을 초과할 때 계정에 대한 이메일 전송을 일시 중지하려면 이 지표를 선택합니다.

다음을 선택합니다.

7. 다음 단계를 완료합니다.

   - [Alarm Threshold]에서 [Name]에 경보 이름을 입력합니다.
   - [Whenever: Reputation.BounceRate] 또는 [Whenever: Reputation.ComplaintRate] 아래에서 경보를 트리거하게 하는 임계값을 지정합니다.

     > **Note**
     > 반송 메일 발생률이 5%를 초과하거나 수신 거부 발생률이 0.1%를 초과하는 경우 계정이 자동으로 검토 상태가 됩니다. CloudWatch 경보를 트리거하게 하는 반송 메일 또는 수신 거부 발생률을 지정하면 계정이 검토 상태가 되지 않도록 이러한 발생률보다 낮은 값을 사용하는 것이 좋습니다.

   - 작업의 이 경보가 발생할 경우 항상에서 상태가 ALARM입니다를 선택합니다. 알림 보내기에서 the section called “4부: Amazon SNS 주제 생성 및 구독”에서 생성한 Amazon SNS 주제를 선택합니다.

   경보 생성을 선택합니다.

6부: 솔루션 테스트

이제 경보를 테스트하여ALARM 상태로 전환될 때 Lambda 함수를 실행하는지 확인할 수 있습니다. `SetAlarmState` API 작업을 사용하여 경보의 상태를 일시적으로 변경할 수 있습니다.

이 섹션의 절차는 선택 사항이지만 전체 솔루션을 올바로 구성할 수 있도록 완료하는 것이 좋습니다.

1. 명령줄에 다음 명령을 입력하여 계정에 대한 이메일 전송 상태를 확인합니다. *region*을 리전의 이름으로 바꿉니다.

   ```
   aws ses get-account-sending-enabled --region region
   ```

   계정에 대해 전송이 활성화되어 있으면 다음 결과가 표시됩니다.

   ```
   {
       "Enabled": true
   ```

```
}
```

2. 명령줄에 다음 명령을 입력하여 경보 상태를 ALARM으로 임시 변경합니다. aws cloudwatch set-alarm-state --alarm-name ***MyAlarm*** --state-value ALARM --state-reason "Testing execution of Lambda function" --region ***region***

   위 명령에서 *MyAlarm*을 the section called “5부: CloudWatch 경보를 생성하려면” 섹션에서 생성한 경보의 이름으로 바꾸고, *region*을 이메일 전송을 자동으로 일시 중지하려는 리전으로 바꿉니다.

   > **Note**
   >
   > 이 명령을 실행하면 경보의 상태가 OK에서 ALARM으로 바뀌고 몇 초 이내에 다시 OK로 바뀝니다. 이러한 상태 변경 사항은 CloudWatch 콘솔 내 경보의 기록 탭을 사용하거나 DescribeAlarmHistory 작업을 사용합니다.

3. 명령줄에 다음 명령을 입력하여 계정에 대한 이메일 전송 상태를 확인합니다.

```
aws ses get-account-sending-enabled --region region
```

   Lambda 함수가 성공적으로 실행되면 다음 결과가 표시됩니다.

```
{
    "Enabled": false
}
```

4. the section called “3부: 계정에 대한 이메일 전송 재활성화”의 단계를 완료하여 계정에 대한 이메일 전송을 재활성화합니다.

구성 집합에 대한 이메일 전송 자동 일시 중지

Amazon CloudWatch로 설정된 특정 구성을 사용하여 전송된 이메일 전용 평판 지표를 내보내도록 Amazon SES를 구성할 수 있습니다. 그 다음 이러한 지표를 사용하여 이러한 구성 세트 전용 CloudWatch 경보를 생성할 수 있습니다. 이러한 경보가 특정 임계값을 초과하면 Amazon SES 계정의 전체 이메일 전송 기능에 영향을 주지 않고 지정된 구성 세트를 사용하는 이메일 전송을 자동으로 일시 중지할 수 있습니다.

> **Note**
>
> 이 섹션에 설명된 솔루션은 단일 AWS 리전의 특정 구성 세트에 대한 이메일 전송을 일시 중지합니다. 여러 리전으로부터 이메일을 보내려면, 이 방법을 구현하고자 하는 리전마다 이 섹션의 절차를 반복하세요.

이 섹션의 주제:

- 1부: 구성 세트에 대한 평판 지표 보고 활성화
- 2부 - IAM 역할 만들기
- 3부: Lambda 함수 생성
- 4부: 구성 세트에 대한 이메일 전송 재활성화
- 5부: Amazon SNS 주제 생성
- 6부: CloudWatch 경보를 생성하려면
- 7부: 솔루션 테스트

1부: 구성 세트에 대한 평판 지표 보고 활성화

구성 세트에 대해 이메일 전송을 자동으로 일시 중지하도록 Amazon SES를 구성하려면 먼저 구성 세트에 대한 평판 지표 내보내기를 활성화해야 합니다.

구성 세트에 대한 반송 메일 및 불만 지표 내보내기를 활성화하려면 the section called “평판 지표 보기 및 내보내기”의 단계를 완료해야 합니다.

2부 - IAM 역할 만들기

이메일 전송 자동 일시 중지를 구성하는 첫 단계는 `UpdateConfigurationSetSendingEnabled` API 작업을 실행할 수 있는 IAM 역할을 생성하는 것입니다.

IAM 역할을 만들려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 역할 생성(Create role)을 선택합니다.
4. 신뢰할 수 있는 엔티티 유형 선택 아래에서 AWS 서비스를 선택합니다.

5. Choose the service that will use this role(이 역할을 사용할 서비스 선택) 아래에서 Lambda를 선택합니다. Next: Permissions(다음: 권한)를 선택합니다.
6. [Attach permissions policies] 페이지에서 다음 정책을 선택합니다.
   - AWS Lambda BasicExecutionRole
   - AmazonSESFullAccess(를 호출할 수 있는 권한이 포함된 요구 사항에 맞는 사용자 지정 역할을 사용하는 것이 좋습니다`UpdateConfigurationSetSendingEnabled`.)

   > **Tip**
   > 정책 목록 상단의 검색 상자를 사용하여 이러한 정책을 빨리 찾습니다.

   다음: 검토를 선택합니다.
7. [Review] 페이지의 [Name]에서 역할 이름을 입력합니다. 역할 생성을 선택합니다.

3부: Lambda 함수 생성

IAM 역할을 생성한 후에는 구성 세트에 대한 이메일 전송을 일시 중지시키는 Lambda 함수를 생성할 수 있습니다.

Lambda 함수를 만들려면

1. https://console.aws.amazon.com/lambda/ AWS Lambda 콘솔을 엽니다.
2. 리전 선택기를 사용하여 이 Lambda 함수를 배포할 리전을 선택합니다.

   > **Note**
   > 이 함수는 이 단계에서 선택하는 AWS 리전에서만 구성 세트에 대한 이메일 전송을 일시 중지시킵니다. 둘 이상의 리전으로부터 이메일을 보내는 경우, 이메일 전송을 자동으로 일시 중지하려는 리전마다 이 섹션의 절차를 반복하세요.
3. 함수 만들기(Create function)를 선택합니다.
4. 함수 생성에서 Author from scratch를 선택합니다.
5. Author from scratch에서 다음 단계를 수행합니다.

- 이름에 Lambda 함수의 이름을 입력합니다.
- 런타임(Runtime)에서 Node.js 14x(또는 선택 목록에 현재 제공된 버전).
- 역할에서 기존 역할 선택을 선택합니다.
- 기존 역할에서 the section called “2부 - IAM 역할 만들기”에서 생성한 IAM 역할을 선택합니다.

함수 생성을 선택합니다.

6. 코드 편집기의 [Function code]에 다음 코드를 붙여 넣습니다.

```
'use strict';

import {
    SES
}
from 'aws-sdk';

const ses = new SES();
const configSet = 'CONFIG_SET_NAME_HERE';

const params = {
    ConfigurationSetName: configSet,
    Enabled: false
};

export const handler = async (event) => {
    try {
        const data = await
 ses.updateConfigurationSetSendingEnabled(params).promise();

        console.log('Configuration Set Update:', data);

        return {
            statusCode: 200,
            body: JSON.stringify({
                message: 'Successfully paused email sending for configuration
 set.',
                data
            }),
        };
    }
    catch (err) {
```

```
        console.error('Error:', err.message);
        return {
            statusCode: 500,
            body: JSON.stringify({
                message: 'Failed to pause email sending for configuration set.',
                error: err.message
            }),
        };
    }
};
```

위 코드의 *ConfigSet*을 구성 세트의 이름으로 바꿉니다. 저장을 선택합니다.

7. 테스트를 선택합니다. [Configure test event] 창이 나타나면 [Event name] 필드에 이름을 입력한 후 [Create]를 선택합니다.
8. 페이지 상단의 알림 표시줄에 `Execution result: succeeded`라고 표시되는지 확인합니다. 함수가 실행되지 못하면 다음을 수행합니다.
   - the section called "2부 - IAM 역할 만들기"에서 생성한 IAM 역할에 올바른 정책이 포함되는지 확인합니다.
   - Lambda 함수의 코드에 오류가 없는지 확인합니다. Lambda 코드 편집기가 구문 오류 및 다른 잠재적인 문제를 자동으로 강조 표시합니다.

4부: 구성 세트에 대한 이메일 전송 재활성화

the section called "3부: Lambda 함수 생성"에서 Lambda 함수를 테스트할 때의 부작용은 구성 세트에 대한 이메일 전송이 일시 중지되는 것입니다. 대부분의 경우 CloudWatch 경보가 트리거될 때까지 구성 세트에 대한 전송을 일시 중지하지 않습니다.

이 단원의 절차에서는 구성 세트에 대한 이메일 전송 재활성화를 설명합니다. 이 절차를 완료하려면 AWS Command Line Interface을 설치하고 구성해야 합니다. 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

이메일 전송을 재활성화하려면

1. 명령줄에 다음 명령을 입력하여 구성 세트에 대한 이메일 전송을 다시 활성화합니다.

```
aws ses update-configuration-set-sending-enabled \
--configuration-set-name ConfigSet \
--enabled
```

앞의 명령에서 *ConfigSet*를 이메일 전송을 일시 중지하려는 구성 세트의 이름으로 바꿉니다.

2. 명령줄에 다음 명령을 입력하여 이메일 전송이 활성화되었는지 확인합니다.

```
aws ses describe-configuration-set \
--configuration-set-name ConfigSet \
--configuration-set-attribute-names reputationOptions
```

이 명령은 다음 예제와 유사한 출력을 생성합니다.

```
{
    "ConfigurationSet": {
        "Name": "ConfigSet"
    },
    "ReputationOptions": {
        "ReputationMetricsEnabled": true,
        "SendingEnabled": true
    }
}
```

`SendingEnabled`의 값이 `true`이면 구성 세트에 대한 이메일 전송이 성공적으로 재활성화된 것입니다.

5부: Amazon SNS 주제 생성

경보가 트리거될 때 CloudWatch가 Lambda 함수를 실행하려면 먼저 Amazon SNS 주제를 생성하고 해당 Lambda 함수를 구독해야 합니다.

Amazon SNS 주제를 생성하려면

1. https://console.aws.amazon.com/sns/v3/home에서 Amazon SNS 콘솔을 엽니다.
2. 리전 선택기를 사용하여 이메일 전송을 자동으로 일시 중지하려는 리전을 선택합니다.
3. 탐색 창에서 주제를 선택합니다.
4. [Create new topic]을 선택합니다.
5. [Create new topic] 창의 [Topic name]에서 주제 이름을 입력합니다. 선택적으로 [Display name] 필드에 더 세부적인 이름을 입력할 수 있습니다.

   주제 생성을 선택합니다.

6. 주제 목록에서 이전 단계에서 생성한 주제 옆의 상자를 선택합니다. [Actions] 메뉴에서 [Subscribe to topic]을 선택합니다.
7. [Create subscription] 창에서 다음과 같이 선택합니다.
   - 프로토콜에서 AWS Lambda을 선택합니다.
   - 엔드포인트에서 the section called “3부: Lambda 함수 생성”에서 생성한 Lambda 함수를 선택합니다.
   - [Version or alias]에서 [default]를 선택합니다.
8. 구독 생성을 선택합니다.

6부: CloudWatch 경보를 생성하려면

이 단원에는 지표가 특정 임계값에 도달하면 트리거되는 CloudWatch에 경보를 생성하기 위한 절차가 포함되어 있습니다. 경보가 트리거되면 the section called “5부: Amazon SNS 주제 생성”에서 생성한 Amazon SNS 주제에 알림을 전달한 후 the section called “3부: Lambda 함수 생성”에서 생성한 Lambda 함수를 실행합니다.

CloudWatch 경보를 생성하려면

1. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 리전 선택기를 사용하여 이메일 전송을 자동으로 일시 중지하려는 리전을 선택합니다.
3. 왼쪽의 탐색 창에서 [Alarms]를 선택합니다.
4. 경보 생성을 선택합니다.
5. [Create Alarm] 창의 [SES Metrics] 아래에서 [Configuration Set Metrics]를 선택합니다.
6. [ses:configuration-set] 열에서 경보를 생성할 구성 세트를 찾습니다. [Metric Name] 아래에서 다음 옵션 중 하나를 선택합니다.
   - Reputation.BounceRate – 구성 세트의 전체 하드 반송 메일 발생률이 사용자가 정의하는 임계값을 교차할 때 구성 세트에 대한 이메일 전송을 일시 중지하려는 경우 이 지표를 선택합니다.
   - Reputation.ComplaintRate – 구성 세트의 전체 수신 거부 발생률이 사용자가 정의하는 임계값을 교차할 때 구성 세트에 대한 이메일 전송을 일시 중지하려는 경우 이 지표를 선택합니다.

   다음을 선택합니다.
7. 다음 단계를 완료합니다.
   - [Alarm Threshold]에서 [Name]에 경보 이름을 입력합니다.

- [Whenever: Reputation.BounceRate] 또는 [Whenever: Reputation.ComplaintRate] 아래에서 경보를 트리거하게 하는 임계값을 지정합니다.

> **Note**
>
> Amazon SES 계정에 대한 전체 반송 메일 발생률이 10%를 초과하거나 Amazon SES 계정에 대한 전체 수신 거부 발생률이 0.5%를 초과하면 Amazon SES 계정이 자동으로 검토 상태로 설정됩니다. CloudWatch 경보를 트리거하게 하는 반송 메일 또는 수신 거부 발생률을 지정하면 계정이 검토 상태가 되지 않도록 이러한 발생률보다 매우 낮은 값을 사용하는 것이 좋습니다.

- 작업의 이 경보가 발생할 경우 항상에서 상태가 ALARM입니다를 선택합니다. 알림 보내기에서 the section called "5부: Amazon SNS 주제 생성"에서 생성한 Amazon SNS 주제를 선택합니다.

경보 생성을 선택합니다.

7부: 솔루션 테스트

이제 경보를 테스트하여 ALARM 상태로 전환될 때 Lambda 함수를 실행하는지 확인할 수 있습니다. CloudWatch API에서 `SetAlarmState` 작업을 사용하여 경보의 상태를 일시적으로 변경할 수 있습니다.

이 단원의 절차는 선택 사항이지만 전체 솔루션을 올바로 구성할 수 있도록 완료하는 것이 좋습니다.

솔루션을 테스트하려면

1. 명령줄에 다음 명령을 입력하여 구성 세트에 대한 이메일 전송 상태를 확인합니다.

```
aws ses describe-configuration-set --configuration-set-name ConfigSet
```

구성 세트에 대해 전송이 활성화되어 있으면 다음 결과가 표시됩니다.

```
{
    "ConfigurationSet": {
        "Name": "ConfigSet"
    },
    "ReputationOptions": {
        "ReputationMetricsEnabled": true,
        "SendingEnabled": true
```

```
    }
}
```

SendingEnabled의 값이 true이면 구성 세트에 대한 이메일 전송이 현재 활성화된 것입니다.

2. 명령줄에 다음 명령을 입력하여 경보 상태를 ALARM으로 임시 변경합니다.

```
aws cloudwatch set-alarm-state \
--alarm-name MyAlarm \
--state-value ALARM \
--state-reason "Testing execution of Lambda function"
```

위 명령의 *MyAlarm*을 the section called “6부: CloudWatch 경보를 생성하려면”에서 생성한 경보의 이름으로 바꿉니다.

> **Note**
>
> 이 명령을 실행하면 경보의 상태가 OK에서 ALARM으로 바뀌고 몇 초 이내에 다시 OK로 바뀝니다. 이러한 상태 변경 사항은 CloudWatch 콘솔 내 경보의 기록 탭을 사용하거나 DescribeAlarmHistory 작업을 사용합니다.

3. 명령줄에 다음 명령을 입력하여 구성 세트에 대한 이메일 전송 상태를 확인합니다.

```
aws ses describe-configuration-set \
--configuration-set-name ConfigSet
```

Lambda 함수가 성공적으로 실행되면 다음 예제와 유사한 출력이 표시됩니다.

```
{
    "ConfigurationSet": {
        "Name": "ConfigSet"
    },
    "ReputationOptions": {
        "ReputationMetricsEnabled": true,
        "SendingEnabled": false
    }
}
```

SendingEnabled의 값이 false이면 구성 세트에 대한 이메일 전송이 비활성화된 것으로, Lambda 함수가 성공적으로 실행되었음을 나타냅니다.

4. the section called “4부: 구성 세트에 대한 이메일 전송 재활성화”의 단계를 완료하여 구성 세트에 대한 이메일 전송을 재활성화합니다.

Amazon EventBridge를 사용하여 SES 이벤트 모니터링

EventBridge는 이벤트를 사용하여 애플리케이션 구성 요소를 서로 연결하는 서버리스 서비스로, 확장 가능한 이벤트 기반 애플리케이션을 더 쉽게 구축할 수 있습니다. 이벤트 기반 아키텍처는 이벤트를 내보내고 이에 응답하여 함께 작동하는 느슨하게 결합된 소프트웨어 시스템을 구축하는 스타일입니다. 이벤트는 일반적으로 리소스나 환경의 변경 또는 기타 관리 이벤트를 나타내는 JSON 형식의 메시지입니다.

특정 SES 특성은 이벤트 대상을 만들 때 정의하는 이벤트를 생성하고 EventBridge 기본 이벤트 버스로 전송합니다. 이벤트 버스는 이벤트를 수신하여 0개 이상의 목적지 또는 대상에 전달하는 라우터입니다. 이벤트 버스와 연결한 규칙은 이벤트가 도착할 때 이벤트를 평가합니다. 각 규칙은 이벤트가 규칙의 패턴과 일치하는지 확인합니다. 이벤트가 일치하면 EventBridge는 이벤트를 지정된 대상으로 보냅니다.

특성에 상태 변경 또는 상태 업데이트가 있으면 SES가 EventBridge로 이벤트를 전송합니다. EventBridge 규칙을 사용하여 이벤트를 정의된 대상으로 라우팅할 수 있습니다. 이러한 이벤트는 최선의 방식으로 전달되며 순서가 맞지 않게 전달될 수 있습니다.

주제

- SES 이벤트
- SES 이벤트 스키마 참조
- SES 이벤트에 EventBridge 사용
- 추가 EventBridge 리소스

SES 이벤트

다음 이벤트는 SES 특성에 의해 생성되어 EventBridge의 기본 이벤트 버스로 전송됩니다. 각 이벤트 유형의 세부 정보 데이터를 포함한 자세한 내용은 ???에서 확인하세요.

Virtual Deliverability Manager 어드바이저 이벤트

이벤트 유형	설명
어드바이저 권장 사항 상태 개시	Virtual Deliverability Manager 어드바이저에서 새 권장 사항이 개시될 때마다 생성되는 이벤트입니다.

이벤트 유형	설명
어드바이저 권장 사항 상태 해결	Virtual Deliverability Manager 어드바이저에서 권장 사항이 해결될 때마다 생성되는 이벤트입니다.

SES 이메일 전송 이벤트

이벤트 유형	설명
이메일 반송됨	하드 바운스 – 수신자의 메일 서버가 이메일을 영구적으로 거부합니다. (소프트 바운스는 SES가 일정 시간 동안 재시도한 후 이메일을 전송하는 데 실패한 경우에만 포함됩니다.)
이메일 클릭함	수신자가 이메일에서 하나 이상의 링크를 클릭했습니다.
이메일 수신 거부 접수됨	이메일이 수신자의 메일 서버로 성공적으로 전송되었지만 수신자가 이를 스팸으로 표시했습니다.
이메일 배달됨	SES가 이메일을 수신자의 메일 서버로 성공적으로 전송했습니다.
이메일 전송 지연됨	일시적인 문제가 발생하여 수신자의 메일 서버로 이메일을 전송할 수 없습니다. 예를 들어 수신자의 받은 편지함이 가득 찼거나 이메일 수신 서버에 일시적인 문제가 발생했을 때 전송 지연이 발생할 수 있습니다.
이메일 열림	수신자가 메시지를 수신하여 자신의 이메일 클라이언트에서 열었습니다.
이메일 거부됨	SES가 이메일을 수락했으나 이메일에 바이러스가 포함된 것으로 판단되어 수신자의 메일 서버로 전송하려고 시도하지 않았습니다.
이메일 렌더링 실패	템플릿 렌더링 문제로 인해 이메일이 전송되지 않았습니다. 이 이벤트 유형은 템플릿 데이터가 누락되었을 때 또는 템플릿 파라미터와 데이터 사이에 불일치가 있을 때 발생할 수 있습니다. (이 이벤트 유형은 `SendTemplatedEmail` 또는 `SendBulkT`

이벤트 유형	설명
	emplatedEmail API 작업을 사용하는 이메일을 전송할 때만 발생합니다.)
이메일 전송됨	전송 요청이 성공했으며 SES는 메시지를 수신자의 메일 서버로 전송합니다. (계정 수준 또는 전역 금지를 사용하는 경우 SES가 여전히 전송으로 계산하지만 배달은 금지합니다.)
이메일 구독함	이메일이 성공적으로 전송되었지만 수신자가 이메일 헤더에서 `List-Unsubscribe` 를 클릭하거나, 바닥글에서 `Unsubscribe` 링크를 선택하여 구독 기본 설정을 업데이트했습니다.

SES 이벤트 스키마 참조

AWS 서비스의 모든 이벤트에는 이벤트의 소스인 AWS 서비스, 이벤트가 생성된 시간, 이벤트가 발생한 계정 및 리전 등 이벤트에 대한 메타데이터가 포함된 공통 필드 세트가 있습니다. 이러한 일반 필드에 대한 정의는 EventBridge 사용 설명서의 이벤트 구조 참조를 참조하세요.

또한 각 이벤트에는 해당 특정 이벤트와 관련된 데이터를 포함하는 `detail` 필드가 있습니다. 다음 참조는 다양한 SES 이벤트에 대한 세부 정보 필드를 정의합니다.

EventBridge를 사용하여 SES 이벤트를 선택하고 관리할 때는 다음 사항을 염두에 두는 것이 유용합니다.

- SES의 모든 이벤트 `source` 필드는 `aws.ses`로 설정됩니다.
- `detail-type` 필드는 이벤트 유형을 지정합니다. the section called “SES 이벤트”의 이벤트 유형 테이블을 참조하세요.
- `detail` 필드는 해당 특정 이벤트와 관련된 데이터를 포함합니다.

  Virtual Deliverability Manager와 같은 일부 이벤트 유형의 경우 세부 정보 필드는 한정된 정적 값 집합으로 채워지는 다소 단순한 데이터 문자열입니다. 반면에 이메일 전송 이벤트의 세부 정보 필드는 이메일이 전송된 시점의 타임스탬프, 수신자 주소 및 기타 여러 이메일 속성 등 정적 값과 동적 값이 모두 포함된 다수의 세부 정보 하위 필드로 구성될 수 있으므로 더 복잡합니다.

주제

- Virtual Deliverability Manager 어드바이저 상태 스키마

- SES 이메일 전송 상태 스키마

Virtual Deliverability Manager 어드바이저 상태 스키마

다음 스키마 참조는 Virtual Deliverability Manager 어드바이저 상태 이벤트와 관련된 필드를 정의합니다.

모든 이벤트 스키마(예:`version`, `id`, `account` 등)에 나타나는 일반 필드에 대한 정의는 EventBridge 사용 설명서의 이벤트 구조 참조에서 찾을 수 있습니다. `source` 및 `detail-type` 필드는 SES 이벤트에 대한 SES 관련 값을 포함하므로 다음 참조에 포함됩니다.

`source`

이벤트를 생성한 서비스를 식별합니다. SES 이벤트의 경우 이 값은 `aws.ses`입니다.

`detail-type`

이벤트의 유형을 식별합니다.

이 필드의 값은 the section called “SES 이벤트”의 Virtual Deliverability Manager 어드바이저 이벤트 테이블에 나열되어 있습니다.

`detail`

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

이 필드의 값은 다음과 같을 수 있습니다.

- `DKIM verification is not enabled.`
- `DKIM verification has failed.`
- `DKIM signing key length is below 2048 bits.`
- `DMARC configuration was not found.`
- `DMARC configuration could not be parsed.`
- `DKIM record was not found.`
- `DKIM record is not aligned.`
- `MAIL FROM record is not aligned.`
- `SPF record was not found.`

- `SPF record for Amazon SES was not found.`
- `SPF all qualifier is missing.`
- `An SPF configuration issue was found.`
- `BIMI record not found or configured without default selector.`
- `BIMI has malformed TXT record.`

Example 예: Virtual Deliverability Manager 어드바이저 상태 이벤트

다음은 이벤트 유형이 `Advisor Recommendation Status Open`인 경우 Virtual Deliverability Manager 어드바이저 상태 이벤트의 예입니다. 이 예의 세부 이벤트 값은 `SPF record was not found.`입니다.

```
{
 "version": "0",
  "id": "abcd9999-ef33-0123-90ab-abcdef666666",
  "detail-type": "Advisor Recommendation Status Open",
  "source": "aws.ses",
  "account": "012345678901",
  "time": "2023-11-15T17:00:59Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ses:us-east-1:012345678901:identity/vdm.events-publishing.cajun.syster-
games.example.com"
  ],
  "detail": { "version": "1.0.0", "data": "SPF record was not found." }
}
```

SES 이메일 전송 상태 스키마

다음 스키마 참조는 SES 이메일 전송 상태 이벤트와 관련된 필드를 정의합니다.

모든 이벤트 스키마(예:`version`, `id`, `account` 등)에 나타나는 일반 필드에 대한 정의는 EventBridge 사용 설명서의 이벤트 구조 참조에서 찾을 수 있습니다. `source` 및 `detail-type` 필드는 SES 이벤트에 대한 SES 관련 값을 포함하므로 다음 참조에 포함됩니다.

`source`

이벤트를 생성한 서비스를 식별합니다. SES 이벤트의 경우 이 값은 `aws.ses`입니다.

`detail-type`

이벤트의 유형을 식별합니다.

이 필드의 값은 the section called “SES 이벤트”의 SES 이메일 전송 이벤트 테이블에 나열되어 있습니다.

`detail`

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

이 필드에 들어갈 수 있는 모든 값은 특정 시점에 전송되는 각 고유 이메일에 의해 생성되는 정적 및 동적 값으로 구성되므로 여기에 나열되지 않습니다. 그러나 제공되는 예를 통해 이 필드에 포함할 수 있는 유형 데이터를 이해할 수 있습니다. EventBridge 샌드박스를 사용하여 모든 이메일 전송 이벤트 유형에 대한 세부 정보 데이터 예시를 찾아볼 수 있습니다. EventBridge에서 샘플 이벤트 지정의 내용을 참조하세요.

SES 이메일 전송 이벤트 `Email Rendering Failed`에 대해 생성된 세부 정보 데이터의 예:

```
...,
  "detail": {
    "eventType": "Rendering Failure",
    "mail": {
      "timestamp": "2018-01-22T18:43:06.197Z",
      "source": "sender@example.com",
      "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
      "sendingAccountId": "123456789012",
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "destination": ["recipient@example.com"],
      "headersTruncated": false,
      "tags": {
        "ses:configuration-set": ["ConfigSet"]
      }
    },
    "failure": {
      "errorMessage": "Attribute 'attributeName' is not present in the rendering
 data.",
      "templateName": "MyTemplate"
    }
  }
```

Example 예: 이메일 전송 상태 이벤트

다음은 이벤트 유형 `Email Rendering Failed`에 대한 전체 이메일 전송 상태 이벤트의 예입니다. 이 예의 세부 이벤트 값은 특정 이메일에 대한 이메일 전송 이벤트를 기반으로 하는 정적 및 동적 값의 조합입니다.

```
{
  "version": "0",
  "id": "12a18625-3328-fafd-2809-a5e16004f112",
  "detail-type": "Email Rendering Failed",
  "source": "aws.ses",
  "account": "123456789012",
  "time": "2023-07-17T16:48:05Z",
  "region": "us-east-1",
  "resources": ["arn:aws:ses:us-east-1:123456789012:identity/example.com"],
  "detail": {
    "eventType": "Rendering Failure",
    "mail": {
      "timestamp": "2018-01-22T18:43:06.197Z",
      "source": "sender@example.com",
      "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
      "sendingAccountId": "123456789012",
      "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "destination": ["recipient@example.com"],
      "headersTruncated": false,
      "tags": {
        "ses:configuration-set": ["ConfigSet"]
      }
    },
    "failure": {
      "errorMessage": "Attribute 'attributeName' is not present in the rendering
 data.",
      "templateName": "MyTemplate"
    }
  }
}
```

SES 이벤트에 EventBridge 사용

기본적으로 SES는 EventBridge 기본 이벤트 버스로 이벤트를 보냅니다. 기본 이벤트 버스에 규칙을 만들어 EventBridge가 하나 이상의 지정된 대상으로 보낼 특정 이벤트를 식별할 수 있습니다. 각 규칙에는 이벤트가 이벤트 버스에 도착할 때 EventBridge가 이벤트를 일치시키는 데 사용하는 이벤트 패턴

이 포함되어 있습니다. 이벤트가 지정된 규칙의 이벤트 패턴과 일치하면 EventBridge는 이벤트를 규칙에 지정된 대상으로 보냅니다.

EventBridge에서 이벤트 패턴을 정의하는 작업은 일반적으로 새 규칙을 만들거나 기존 규칙을 편집하는 대규모 프로세스의 일부입니다. EventBridge 규칙을 만드는 방법에 대한 자세한 정보는 EventBridge 사용 설명서의 이벤트에 대응하는 Amazon EventBridge 규칙 만들기를 참조하세요.

EventBridge의 샌드박스 특성을 사용하면 먼저 규칙을 만들거나 편집할 필요 없이 이벤트 패턴을 빠르게 정의하고 샘플 이벤트를 사용하여 패턴이 원하는 이벤트와 일치하는지 확인할 수 있습니다. 샌드박스 사용에 대한 자세한 지침은 EventBridge 사용 설명서의 EventBridge 샌드박스를 사용하여 이벤트 패턴 테스트를 참조하세요.

EventBridge 샌드박스에서 SES 샘플 이벤트 지정

SES 이벤트의 샘플 이벤트를 선택하여 생성한 이벤트 패턴을 테스트하는 데 사용할 수 있습니다.

EventBridge 샌드박스에서 SES 샘플 이벤트를 지정하려면

1. Amazon EventBridge 콘솔(https://console.aws.amazon.com/events/)을 엽니다.
2. 탐색 창에서 개발자 리소스를 선택한 다음, 샌드박스를 선택하고 샌드박스 페이지에서 이벤트 패턴 탭을 선택합니다.
3. 이벤트 소스에서 AWS 이벤트 또는 EventBridge 파트너 이벤트를 선택합니다.
4. 샘플 이벤트 섹션의 샘플 이벤트 유형에서 AWS 이벤트를 선택합니다.
5. 샘플 이벤트에서 SES까지 아래로 스크롤한 다음 원하는 SES 이벤트를 선택합니다.

   EventBridge는 이벤트 유형에 대한 모든 세부 정보 데이터와 함께 샘플 이벤트를 표시합니다.

   이 이벤트를 사용하여 이벤트 패턴 섹션에서 생성한 이벤트 패턴을 테스트하거나 패턴 테스트용 샘플 이벤트를 직접 생성하기 위한 기반으로 사용할 수 있습니다(이 내용은 다음 섹션에서 다룸).

SES 이벤트의 이벤트 패턴 생성 및 테스트

이전 섹션에서 설명했듯이 샘플 이벤트를 선택한 후에는 이벤트 패턴을 생성하고 샘플 이벤트를 사용하여 이벤트와 원하는 대로 일치하는지 확인할 수 있습니다.

EventBridge 샌드박스에서 SES 이벤트와 일치하는 이벤트 패턴을 생성하고 테스트하려면

1. Amazon EventBridge 콘솔(https://console.aws.amazon.com/events/)을 엽니다.

2. 탐색 창에서 개발자 리소스를 선택한 다음, 샌드박스를 선택하고 샌드박스 페이지에서 이벤트 패턴 탭을 선택합니다.
3. 이벤트 소스에 AWS 이벤트 또는 EventBridge 파트너 이벤트를 선택하고 이전 섹션에 설명된 대로 테스트할 샘플 이벤트를 선택합니다.
4. 생성 방법까지 아래로 스크롤하고 패턴 양식 사용을 선택합니다.
5. 이벤트 패턴 섹션의 이벤트 소스에서 AWS 서비스를 선택합니다.
6. AWS 서비스에서 SES를 선택합니다.
7. 이벤트 유형에서 일치시키려는 SES 이벤트 유형을 선택합니다.

   EventBridge는 선택한 SES 이벤트와 일치하는 `source` 및 `detail-type` 필드로 구성된 최소 이벤트 패턴을 표시합니다.

   두 예에서 첫 번째 이벤트 패턴은 모든 `Advisor Recommendation Status Resolved` 이벤트와 일치하고, 두 번째 이벤트 패턴은 모든 `Email Bounced` 이벤트와 일치합니다.

```
{
  "source": ["aws.ses"],
  "detail-type": ["Advisor Recommendation Status Resolved"]
}
```

```
{
  "source": ["aws.ses"],
  "detail-type": ["Email Bounced"]
}
```

8. 이벤트 패턴을 변경하려면 패턴 편집을 선택하고 JSON 편집기에서 변경합니다.

   하나 이상의 세부 정보 데이터 필드에 있는 값을 일치시킬 수도 있습니다. 여기에는 필드 값에 가능한 여러 값을 지정하는 것도 포함됩니다.

   다음 예에서는 동일한 세부 정보 값을 가진 모든 Virtual Deliverability Manager 어드바이저 이벤트를 찾기 위해 `data` 필드 값이 `DKIM record was not found`로 지정되어 있는 생성된 최소 이벤트 패턴에 세부 정보 필드가 추가되었습니다.

```
{
  "source": ["aws.ses"],
  "detail-type": ["Advisor Recommendation Status Resolved"],
  "detail": {
    "data": ["DKIM record was not found."]
```

```
  }
}
```

이 예에서는 2024-08-05에 noreply@example.com에서 전송되고 반송된 모든 이메일에서 생성된 이벤트를 보고하기 위해 세부 정보 하위 필드가 추가되었습니다. (여기에는 콘텐츠 필터링의 일부로 접두사 일치가 사용됩니다.)

```
{
  "source": ["aws.ses"],
  "detail-type": ["Email Bounced"],
  "detail": {
    "mail": {
      "timestamp": [{
        "prefix": "2024-08-05"
      }],
      "source": ["noreply@example.com"]
    }
  }
}
```

EventBridge 사용 설명서의 이벤트 패턴을 꼭 읽으시기 바랍니다. JSON 편집기에 입력하는 이벤트 패턴 값은 배열로 간주되므로 대괄호 `[...]`로 묶어야 한다는 내용이 설명되어 있습니다. 이 정보와 함께 고급 이벤트 패턴을 구성하는 방법에 관한 추가 정보도 제공됩니다.

9. 이벤트 패턴이 위의 샘플 이벤트 창에서 지정한 샘플 이벤트와 일치하는지 테스트하려면 패턴 테스트를 선택합니다. 일치하면 JSON 편집기 하단의 녹색 배너에 "이벤트 패턴과 일치하는 샘플 이벤트"라고 표시됩니다.
10. 패턴 테스트를 선택한 후 오류 해결하기:
    - JSON 관련 오류가 있는 경우 이유를 알려주는 메시지가 표시됩니다. 예: "이벤트 패턴이 유효하지 않습니다. 이유: "데이터"는 5행, 14열의 객체 또는 배열이어야 합니다. 이 문제를 해결하려면 5행의 값을 대괄호 `[...]`로 묶어야 합니다.
    - 샘플 이벤트의 값과 이벤트 패턴의 값 사이에 불일치가 있는 경우 "샘플 이벤트가 이벤트 패턴과 일치하지 않음" 메시지가 표시됩니다. 이는 테스트하려는 하나 이상의 값이 샘플 이벤트 생성기에서 생성된 예제 값과 다름을 의미합니다. 이 문제를 해결하려면 나머지 단계를 계속 진행합니다.
11. 이벤트 패턴을 성공적으로 테스트하기 위해 샘플 이벤트의 샘플 값을 변경하려면 샘플 이벤트 창에서 JSON 편집기 아래의 복사를 선택합니다.
12. 편집기 위의 샘플 이벤트 유형에서 내 이름 입력 옆의 라디오 버튼을 선택합니다.

13. 샘플 이벤트를 JSON 편집기에 붙여넣고 이벤트 패턴에 사용 중인 모든 필드에 대해 이벤트 패턴에 지정한 값과 일치하도록 동일한 필드의 값을 바꿉니다.
14. 아래로 스크롤하여 이벤트 패턴 창으로 이동하고 패턴 테스트를 다시 선택합니다. 모든 값이 올바르게 입력되고 일치하는 경우 JSON 편집기 하단의 녹색 배너에 "이벤트 패턴과 일치하는 샘플 이벤트"라고 표시됩니다.

추가 EventBridge 리소스

EventBridge를 사용하여 이벤트를 처리하고 관리하는 방법에 대한 자세한 내용은 Amazon EventBridge 사용 설명서의 다음 주제를 참조하세요.

- 이벤트 버스의 작동 방식에 대한 자세한 내용은 Amazon EventBridge 이벤트 버스를 참조하세요.
- 이벤트 구조에 대한 자세한 내용은 이벤트를 참조하세요.
- 이벤트를 규칙과 일치시킬 때 EventBridge가 사용할 이벤트 패턴을 구성하는 방법에 대한 자세한 내용은 이벤트 패턴을 참조하세요.
- EventBridge에서 처리하는 이벤트를 지정하는 규칙을 만드는 방법에 대한 자세한 내용은 규칙을 참조하세요.
- EventBridge가 일치하는 이벤트를 보낼 서비스나 기타 대상을 지정하는 방법에 대한 자세한 내용은 대상을 참조하세요.

AWS SDKs를 사용하는 Amazon SES의 코드 예제

다음 코드 예제에서는 Amazon SES를 AWS 소프트웨어 개발 키트(SDK)와 함께 사용하는 방법을 보여줍니다.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

코드 예시

- AWS SDKs를 사용하는 Amazon SES의 코드 예제
  - AWS SDKs 사용한 Amazon SES의 기본 예제
    - AWS SDKs를 사용하는 Amazon SES에 대한 작업
      - AWS SDK와 CreateReceiptFilter 함께 사용
      - AWS SDK와 CreateReceiptRule 함께 사용
      - AWS SDK와 CreateReceiptRuleSet 함께 사용
      - AWS SDK와 CreateTemplate 함께 사용
      - AWS SDK 또는 CLI와 DeleteIdentity 함께 사용
      - AWS SDK와 DeleteReceiptFilter 함께 사용
      - AWS SDK와 DeleteReceiptRule 함께 사용
      - AWS SDK와 DeleteReceiptRuleSet 함께 사용
      - AWS SDK와 DeleteTemplate 함께 사용
      - AWS SDK와 DescribeReceiptRuleSet 함께 사용
      - AWS SDK 또는 CLI와 GetIdentityVerificationAttributes 함께 사용
      - AWS SDK 또는 CLI와 GetSendQuota 함께 사용
      - CLI로 GetSendStatistics 사용
      - AWS SDK와 GetTemplate 함께 사용
      - AWS SDK 또는 CLI와 ListIdentities 함께 사용
      - AWS SDK와 ListReceiptFilters 함께 사용
      - AWS SDK와 ListTemplates 함께 사용
      - AWS SDK와 SendBulkTemplatedEmail 함께 사용
      - AWS SDK 또는 CLI와 SendEmail 함께 사용
      - AWS SDK 또는 CLI와 SendRawEmail 함께 사용

- AWS SDK와 SendTemplatedEmail 함께 사용
- AWS SDK와 UpdateTemplate 함께 사용
- AWS SDK 또는 CLI와 VerifyDomainIdentity 함께 사용
- AWS SDK 또는 CLI와 VerifyEmailIdentity 함께 사용

- AWS SDKs를 사용한 Amazon SES 시나리오
  - Amazon Transcribe 스트리밍 앱 구축
  - AWS SDK를 사용하여 한 AWS 리전에서 다른 리전으로 Amazon SES 이메일 및 도메인 자격 증명 복사
  - DynamoDB 데이터를 추적하는 웹 애플리케이션 만들기
  - Amazon Redshift 항목 추적기 만들기
  - Aurora 서버리스 작업 항목 트래커 만들기
  - AWS SDK를 사용하여 Amazon Rekognition으로 이미지에서 PPE 감지
  - AWS SDK를 사용하여 Amazon Rekognition으로 이미지의 객체 감지
  - AWS SDK를 사용하여 Amazon Rekognition을 사용하여 비디오에서 사람 및 객체 감지
  - Amazon SES SMTP 엔드포인트에 연결하는 자격 증명 생성
  - Step Functions를 사용하여 Lambda 함수 간접 호출
  - AWS SDK를 사용하여 Amazon SES로 이메일 자격 증명 확인 및 메시지 전송

- AWS SDKs를 사용한 Amazon SES API v2의 코드 예제
  - AWS SDKs를 사용한 Amazon SES API v2의 기본 예제
    - AWS SDKs를 사용하는 Amazon SES API v2에 대한 작업
      - AWS SDK와 CreateContact 함께 사용
      - AWS SDK와 CreateContactList 함께 사용
      - AWS SDK와 CreateEmailIdentity 함께 사용
      - AWS SDK와 CreateEmailTemplate 함께 사용
      - AWS SDK와 DeleteContactList 함께 사용
      - AWS SDK와 DeleteEmailIdentity 함께 사용
      - AWS SDK와 DeleteEmailTemplate 함께 사용
      - AWS SDK와 GetEmailIdentity 함께 사용
      - AWS SDK와 ListContactLists 함께 사용

      - AWS SDK와 ListContacts 함께 사용

    - AWS SDK와 SendEmail 함께 사용
- AWS SDKs를 사용한 Amazon SES API v2 시나리오
  - AWS SDK를 사용한 전체 Amazon SES API v2 뉴스레터 시나리오

AWS SDKs를 사용하는 Amazon SES의 코드 예제

다음 코드 예제에서는 Amazon SES를 AWS 소프트웨어 개발 키트(SDK)와 함께 사용하는 방법을 보여줍니다.

작업은 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 작업은 관련 시나리오의 컨텍스트에 따라 표시되며, 개별 서비스 함수를 직접적으로 호출하는 방법을 보여줍니다.

시나리오는 동일한 서비스 내에서 또는 다른 AWS 서비스와 결합된 상태에서 여러 함수를 호출하여 특정 태스크를 수행하는 방법을 보여주는 코드 예제입니다.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

코드 예시

- AWS SDKs 사용한 Amazon SES의 기본 예제
  - AWS SDKs를 사용하는 Amazon SES에 대한 작업
    - AWS SDK와 CreateReceiptFilter 함께 사용
    - AWS SDK와 CreateReceiptRule 함께 사용
    - AWS SDK와 CreateReceiptRuleSet 함께 사용
    - AWS SDK와 CreateTemplate 함께 사용
    - AWS SDK 또는 CLI와 DeleteIdentity 함께 사용
    - AWS SDK와 DeleteReceiptFilter 함께 사용
    - AWS SDK와 DeleteReceiptRule 함께 사용
    - AWS SDK와 DeleteReceiptRuleSet 함께 사용
    - AWS SDK와 DeleteTemplate 함께 사용
    - AWS SDK와 DescribeReceiptRuleSet 함께 사용
    - AWS SDK 또는 CLI와 GetIdentityVerificationAttributes 함께 사용
    - AWS SDK 또는 CLI와 GetSendQuota 함께 사용
    - CLI로 GetSendStatistics 사용

    - AWS SDK와 GetTemplate 함께 사용
    - AWS SDK 또는 CLI와 ListIdentities 함께 사용
    - AWS SDK와 ListReceiptFilters 함께 사용
    - AWS SDK와 ListTemplates 함께 사용
    - AWS SDK와 SendBulkTemplatedEmail 함께 사용
    - AWS SDK 또는 CLI와 SendEmail 함께 사용
    - AWS SDK 또는 CLI와 SendRawEmail 함께 사용
    - AWS SDK와 SendTemplatedEmail 함께 사용
    - AWS SDK와 UpdateTemplate 함께 사용
    - AWS SDK 또는 CLI와 VerifyDomainIdentity 함께 사용
    - AWS SDK 또는 CLI와 VerifyEmailIdentity 함께 사용
- AWS SDKs를 사용한 Amazon SES 시나리오
  - Amazon Transcribe 스트리밍 앱 구축
  - AWS SDK를 사용하여 한 AWS 리전에서 다른 리전으로 Amazon SES 이메일 및 도메인 자격 증명 복사
  - DynamoDB 데이터를 추적하는 웹 애플리케이션 만들기
  - Amazon Redshift 항목 추적기 만들기
  - Aurora 서버리스 작업 항목 트래커 만들기
  - AWS SDK를 사용하여 Amazon Rekognition으로 이미지에서 PPE 감지
  - AWS SDK를 사용하여 Amazon Rekognition으로 이미지의 객체 감지
  - AWS SDK를 사용하여 Amazon Rekognition을 사용하여 비디오에서 사람 및 객체 감지
  - Amazon SES SMTP 엔드포인트에 연결하는 자격 증명 생성
  - Step Functions를 사용하여 Lambda 함수 간접 호출
  - AWS SDK를 사용하여 Amazon SES로 이메일 자격 증명 확인 및 메시지 전송

AWS SDKs 사용한 Amazon SES의 기본 예제

다음 코드 예제에서는 AWS SDK와 함께 Amazon Simple Email Service의 기본 사항을 사용하는 방법을 보여줍니다.

예시

- AWS SDKs를 사용하는 Amazon SES에 대한 작업

- AWS SDK와 CreateReceiptFilter 함께 사용
- AWS SDK와 CreateReceiptRule 함께 사용
- AWS SDK와 CreateReceiptRuleSet 함께 사용
- AWS SDK와 CreateTemplate 함께 사용
- AWS SDK 또는 CLI와 DeleteIdentity 함께 사용
- AWS SDK와 DeleteReceiptFilter 함께 사용
- AWS SDK와 DeleteReceiptRule 함께 사용
- AWS SDK와 DeleteReceiptRuleSet 함께 사용
- AWS SDK와 DeleteTemplate 함께 사용
- AWS SDK와 DescribeReceiptRuleSet 함께 사용
- AWS SDK 또는 CLI와 GetIdentityVerificationAttributes 함께 사용
- AWS SDK 또는 CLI와 GetSendQuota 함께 사용
- CLI로 GetSendStatistics 사용
- AWS SDK와 GetTemplate 함께 사용
- AWS SDK 또는 CLI와 ListIdentities 함께 사용
- AWS SDK와 ListReceiptFilters 함께 사용
- AWS SDK와 ListTemplates 함께 사용
- AWS SDK와 SendBulkTemplatedEmail 함께 사용
- AWS SDK 또는 CLI와 SendEmail 함께 사용
- AWS SDK 또는 CLI와 SendRawEmail 함께 사용
- AWS SDK와 SendTemplatedEmail 함께 사용
- AWS SDK와 UpdateTemplate 함께 사용
- AWS SDK 또는 CLI와 VerifyDomainIdentity 함께 사용
- AWS SDK 또는 CLI와 VerifyEmailIdentity 함께 사용

AWS SDKs를 사용하는 Amazon SES에 대한 작업

다음 코드 예제에서는 AWS SDKs를 사용하여 개별 Amazon SES 작업을 수행하는 방법을 보여줍니다. 각 예제에는 GitHub에 대한 링크가 포함되어 있습니다. 여기에서 코드 설정 및 실행에 대한 지침을 찾을 수 있습니다.

이들 발췌문은 Amazon SES API를 호출하며, 컨텍스트에서 실행되어야 하는 더 큰 프로그램에서 발췌한 코드입니다. AWS SDKs를 사용한 Amazon SES 시나리오 에서 컨텍스트에 맞는 작업을 볼 수 있습니다.

다음 예제에는 가장 일반적으로 사용되는 작업만 포함되어 있습니다. 전체 목록은 Amazon Simple Email Service API 참조를 참조하세요.

예시

- AWS SDK와 CreateReceiptFilter 함께 사용
- AWS SDK와 CreateReceiptRule 함께 사용
- AWS SDK와 CreateReceiptRuleSet 함께 사용
- AWS SDK와 CreateTemplate 함께 사용
- AWS SDK 또는 CLI와 DeleteIdentity 함께 사용
- AWS SDK와 DeleteReceiptFilter 함께 사용
- AWS SDK와 DeleteReceiptRule 함께 사용
- AWS SDK와 DeleteReceiptRuleSet 함께 사용
- AWS SDK와 DeleteTemplate 함께 사용
- AWS SDK와 DescribeReceiptRuleSet 함께 사용
- AWS SDK 또는 CLI와 GetIdentityVerificationAttributes 함께 사용
- AWS SDK 또는 CLI와 GetSendQuota 함께 사용
- CLI로 GetSendStatistics 사용
- AWS SDK와 GetTemplate 함께 사용
- AWS SDK 또는 CLI와 ListIdentities 함께 사용
- AWS SDK와 ListReceiptFilters 함께 사용
- AWS SDK와 ListTemplates 함께 사용
- AWS SDK와 SendBulkTemplatedEmail 함께 사용
- AWS SDK 또는 CLI와 SendEmail 함께 사용
- AWS SDK 또는 CLI와 SendRawEmail 함께 사용
- AWS SDK와 SendTemplatedEmail 함께 사용
- AWS SDK와 UpdateTemplate 함께 사용
- AWS SDK 또는 CLI와 VerifyDomainIdentity 함께 사용
- AWS SDK 또는 CLI와 VerifyEmailIdentity 함께 사용

AWS SDK와 **CreateReceiptFilter** 함께 사용

다음 코드 예시는 CreateReceiptFilter의 사용 방법을 보여 줍니다.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Create an Amazon Simple Email Service (Amazon SES) receipt filter..
/*!
  \param receiptFilterName: The name for the receipt filter.
  \param cidr: IP address or IP address range in Classless Inter-Domain Routing
 (CIDR) notation.
  \param policy: Block or allow enum of type ReceiptFilterPolicy.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::createReceiptFilter(const Aws::String &receiptFilterName,
                                      const Aws::String &cidr,
                                      Aws::SES::Model::ReceiptFilterPolicy
 policy,
                                      const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);
    Aws::SES::Model::CreateReceiptFilterRequest createReceiptFilterRequest;
    Aws::SES::Model::ReceiptFilter receiptFilter;
    Aws::SES::Model::ReceiptIpFilter receiptIpFilter;
    receiptIpFilter.SetCidr(cidr);
    receiptIpFilter.SetPolicy(policy);
    receiptFilter.SetName(receiptFilterName);
    receiptFilter.SetIpFilter(receiptIpFilter);
    createReceiptFilterRequest.SetFilter(receiptFilter);
    Aws::SES::Model::CreateReceiptFilterOutcome createReceiptFilterOutcome =
 sesClient.CreateReceiptFilter(
            createReceiptFilterRequest);
    if (createReceiptFilterOutcome.IsSuccess()) {
        std::cout << "Successfully created receipt filter." << std::endl;
```

```
    }
    else {
        std::cerr << "Error creating receipt filter: " <<
                  createReceiptFilterOutcome.GetError().GetMessage() <<
 std::endl;
    }

    return createReceiptFilterOutcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 CreateReceiptFilter를 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import {
  CreateReceiptFilterCommand,
  ReceiptFilterPolicy,
} from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";

const createCreateReceiptFilterCommand = ({ policy, ipOrRange, name }) => {
  return new CreateReceiptFilterCommand({
    Filter: {
      IpFilter: {
        Cidr: ipOrRange, // string, either a single IP address (10.0.0.1) or an
 IP address range in CIDR notation (10.0.0.1/24)).
        Policy: policy, // enum ReceiptFilterPolicy, email traffic from the
 filtered addressesOptions.
      },
      /*
        The name of the IP address filter. Only ASCII letters, numbers,
 underscores, or dashes.
```

```
        Must be less than 64 characters and start and end with a letter or
 number.
       */
      Name: name,
    },
  });
};

const FILTER_NAME = getUniqueName("ReceiptFilter");

const run = async () => {
  const createReceiptFilterCommand = createCreateReceiptFilterCommand({
    policy: ReceiptFilterPolicy.Allow,
    ipOrRange: "10.0.0.1",
    name: FILTER_NAME,
  });

  try {
    return await sesClient.send(createReceiptFilterCommand);
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MessageRejected") {
      /** @type { import('@aws-sdk/client-ses').MessageRejected} */
      const messageRejectedError = caught;
      return messageRejectedError;
    }
    throw caught;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 CreateReceiptFilter를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesReceiptHandler:
    """Encapsulates Amazon SES receipt handling functions."""

    def __init__(self, ses_client, s3_resource):
        """
        :param ses_client: A Boto3 Amazon SES client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        """
        self.ses_client = ses_client
        self.s3_resource = s3_resource


    def create_receipt_filter(self, filter_name, ip_address_or_range, allow):
        """
        Creates a filter that allows or blocks incoming mail from an IP address
 or
        range.

        :param filter_name: The name to give the filter.
        :param ip_address_or_range: The IP address or range to block or allow.
        :param allow: When True, incoming mail is allowed from the specified IP
                      address or range; otherwise, it is blocked.
        """
        try:
            policy = "Allow" if allow else "Block"
            self.ses_client.create_receipt_filter(
                Filter={
                    "Name": filter_name,
                    "IpFilter": {"Cidr": ip_address_or_range, "Policy": policy},
                }
            )
            logger.info(
                "Created receipt filter %s to %s IP of %s.",
                filter_name,
                policy,
                ip_address_or_range,
            )
        except ClientError:
            logger.exception("Couldn't create receipt filter %s.", filter_name)
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 CreateReceiptFilter를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`CreateReceiptRule`** 함께 사용

다음 코드 예시는 `CreateReceiptRule`의 사용 방법을 보여 줍니다.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Create an Amazon Simple Email Service (Amazon SES) receipt rule.
/*!
  \param receiptRuleName: The name for the receipt rule.
  \param s3BucketName: The name of the S3 bucket for incoming mail.
  \param s3ObjectKeyPrefix: The prefix for the objects in the S3 bucket.
  \param ruleSetName: The name of the rule set where the receipt rule is added.
  \param recipients: Aws::Vector of recipients.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::createReceiptRule(const Aws::String &receiptRuleName,
                                    const Aws::String &s3BucketName,
                                    const Aws::String &s3ObjectKeyPrefix,
                                    const Aws::String &ruleSetName,
                                    const Aws::Vector<Aws::String> &recipients,
                                    const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::CreateReceiptRuleRequest createReceiptRuleRequest;
```

```
    Aws::SES::Model::S3Action s3Action;
    s3Action.SetBucketName(s3BucketName);
    s3Action.SetObjectKeyPrefix(s3ObjectKeyPrefix);

    Aws::SES::Model::ReceiptAction receiptAction;
    receiptAction.SetS3Action(s3Action);

    Aws::SES::Model::ReceiptRule receiptRule;
    receiptRule.SetName(receiptRuleName);
    receiptRule.WithRecipients(recipients);

    Aws::Vector<Aws::SES::Model::ReceiptAction> receiptActionList;
    receiptActionList.emplace_back(receiptAction);
    receiptRule.SetActions(receiptActionList);

    createReceiptRuleRequest.SetRuleSetName(ruleSetName);
    createReceiptRuleRequest.SetRule(receiptRule);

    auto outcome = sesClient.CreateReceiptRule(createReceiptRuleRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully created receipt rule." << std::endl;
    }
    else {
        std::cerr << "Error creating receipt rule. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 CreateReceiptRule을 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { CreateReceiptRuleCommand, TlsPolicy } from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";

const RULE_SET_NAME = getUniqueName("RuleSetName");
const RULE_NAME = getUniqueName("RuleName");
const S3_BUCKET_NAME = getUniqueName("S3BucketName");

const createS3ReceiptRuleCommand = ({
  bucketName,
  emailAddresses,
  name,
  ruleSet,
}) => {
  return new CreateReceiptRuleCommand({
    Rule: {
      Actions: [
        {
          S3Action: {
            BucketName: bucketName,
            ObjectKeyPrefix: "email",
          },
        },
      ],
      Recipients: emailAddresses,
      Enabled: true,
      Name: name,
      ScanEnabled: false,
      TlsPolicy: TlsPolicy.Optional,
    },
    RuleSetName: ruleSet, // Required
  });
```

```
};

const run = async () => {
  const s3ReceiptRuleCommand = createS3ReceiptRuleCommand({
    bucketName: S3_BUCKET_NAME,
    emailAddresses: ["email@example.com"],
    name: RULE_NAME,
    ruleSet: RULE_SET_NAME,
  });

  try {
    return await sesClient.send(s3ReceiptRuleCommand);
  } catch (err) {
    console.log("Failed to create S3 receipt rule.", err);
    throw err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 CreateReceiptRule을 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

Amazon SES가 수신 이메일의 사본을 넣을 수 있는 Amazon S3 버킷을 만들고 특정 수신자 목록에 대해 수신 이메일을 버킷에 복사하는 규칙을 만듭니다.

```
class SesReceiptHandler:
    """Encapsulates Amazon SES receipt handling functions."""

    def __init__(self, ses_client, s3_resource):
        """
        :param ses_client: A Boto3 Amazon SES client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        """
```

```
        self.ses_client = ses_client
        self.s3_resource = s3_resource


    def create_bucket_for_copy(self, bucket_name):
        """
        Creates a bucket that can receive copies of emails from Amazon SES. This
        includes adding a policy to the bucket that grants Amazon SES permission
        to put objects in the bucket.

        :param bucket_name: The name of the bucket to create.
        :return: The newly created bucket.
        """
        allow_ses_put_policy = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "AllowSESPut",
                    "Effect": "Allow",
                    "Principal": {"Service": "ses.amazonaws.com"},
                    "Action": "s3:PutObject",
                    "Resource": f"arn:aws:s3:::{bucket_name}/*",
                }
            ],
        }
        bucket = None
        try:
            bucket = self.s3_resource.create_bucket(
                Bucket=bucket_name,
                CreateBucketConfiguration={
                    "LocationConstraint":
self.s3_resource.meta.client.meta.region_name
                },
            )
            bucket.wait_until_exists()
            bucket.Policy().put(Policy=json.dumps(allow_ses_put_policy))
            logger.info("Created bucket %s to receive copies of emails.",
bucket_name)
        except ClientError:
            logger.exception("Couldn't create bucket to receive copies of
emails.")
            if bucket is not None:
                bucket.delete()
            raise
```

```
        else:
            return bucket


    def create_s3_copy_rule(
        self, rule_set_name, rule_name, recipients, bucket_name, prefix
    ):
        """
        Creates a rule so that all emails received by the specified recipients
are
        copied to an Amazon S3 bucket.

        :param rule_set_name: The name of a previously created rule set to
contain
                              this rule.
        :param rule_name: The name to give the rule.
        :param recipients: When an email is received by one of these recipients,
it
                           is copied to the Amazon S3 bucket.
        :param bucket_name: The name of the bucket to receive email copies. This
                            bucket must allow Amazon SES to put objects into it.
        :param prefix: An object key prefix to give the emails copied to the
bucket.
        """
        try:
            self.ses_client.create_receipt_rule(
                RuleSetName=rule_set_name,
                Rule={
                    "Name": rule_name,
                    "Enabled": True,
                    "Recipients": recipients,
                    "Actions": [
                        {
                            "S3Action": {
                                "BucketName": bucket_name,
                                "ObjectKeyPrefix": prefix,
                            }
                        }
                    ],
                },
            )
            logger.info(
                "Created rule %s to copy mail received by %s to bucket %s.",
                rule_name,
```

```
                recipients,
                bucket_name,
            )
        except ClientError:
            logger.exception("Couldn't create rule %s.", rule_name)
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 CreateReceiptRule를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`CreateReceiptRuleSet`** 함께 사용

다음 코드 예시는 `CreateReceiptRuleSet`의 사용 방법을 보여 줍니다.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Create an Amazon Simple Email Service (Amazon SES) receipt rule set.
/*!
  \param ruleSetName: The name of the rule set.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::createReceiptRuleSet(const Aws::String &ruleSetName,
                                       const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::CreateReceiptRuleSetRequest createReceiptRuleSetRequest;
```

```
    createReceiptRuleSetRequest.SetRuleSetName(ruleSetName);

    Aws::SES::Model::CreateReceiptRuleSetOutcome outcome =
 sesClient.CreateReceiptRuleSet(
            createReceiptRuleSetRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully created receipt rule set." << std::endl;
    }
    else {
        std::cerr << "Error creating receipt rule set. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 CreateReceiptRuleSet를 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { CreateReceiptRuleSetCommand } from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";

const RULE_SET_NAME = getUniqueName("RuleSetName");

const createCreateReceiptRuleSetCommand = (ruleSetName) => {
  return new CreateReceiptRuleSetCommand({ RuleSetName: ruleSetName });
};

const run = async () => {
```

```
  const createReceiptRuleSetCommand =
    createCreateReceiptRuleSetCommand(RULE_SET_NAME);

  try {
    return await sesClient.send(createReceiptRuleSetCommand);
  } catch (err) {
    console.log("Failed to create receipt rule set", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 CreateReceiptRuleSet를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesReceiptHandler:
    """Encapsulates Amazon SES receipt handling functions."""

    def __init__(self, ses_client, s3_resource):
        """
        :param ses_client: A Boto3 Amazon SES client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        """
        self.ses_client = ses_client
        self.s3_resource = s3_resource


    def create_receipt_rule_set(self, rule_set_name):
        """
        Creates an empty rule set. Rule sets contain individual rules and can be
        used to organize rules.

        :param rule_set_name: The name to give the rule set.
```

```
        """
        try:
            self.ses_client.create_receipt_rule_set(RuleSetName=rule_set_name)
            logger.info("Created receipt rule set %s.", rule_set_name)
        except ClientError:
            logger.exception("Couldn't create receipt rule set %s.",
rule_set_name)
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 CreateReceiptRuleSet를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`CreateTemplate`** 함께 사용

다음 코드 예시는 `CreateTemplate`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.

- 이메일 ID 확인 및 메시지 전송

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Create an email template.
    /// </summary>
```

```
    /// <param name="name">Name of the template.</param>
    /// <param name="subject">Email subject.</param>
    /// <param name="text">Email body text.</param>
    /// <param name="html">Email HTML body text.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> CreateEmailTemplateAsync(string name, string subject,
string text,
        string html)
    {
        var success = false;
        try
        {
            var response = await _amazonSimpleEmailService.CreateTemplateAsync(
                new CreateTemplateRequest
                {
                    Template = new Template
                    {
                        TemplateName = name,
                        SubjectPart = subject,
                        TextPart = text,
                        HtmlPart = html
                    }
                });
            success = response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (Exception ex)
        {
            Console.WriteLine("CreateEmailTemplateAsync failed with exception: "
+ ex.Message);
        }

        return success;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 CreateTemplate을 참조하십시오.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Create an Amazon Simple Email Service (Amazon SES) template.
/*!
  \param templateName: The name of the template.
  \param htmlPart: The HTML body of the email.
  \param subjectPart: The subject line of the email.
  \param textPart: The plain text version of the email.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::createTemplate(const Aws::String &templateName,
                                 const Aws::String &htmlPart,
                                 const Aws::String &subjectPart,
                                 const Aws::String &textPart,
                                 const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::CreateTemplateRequest createTemplateRequest;
    Aws::SES::Model::Template aTemplate;

    aTemplate.SetTemplateName(templateName);
    aTemplate.SetHtmlPart(htmlPart);
    aTemplate.SetSubjectPart(subjectPart);
    aTemplate.SetTextPart(textPart);

    createTemplateRequest.SetTemplate(aTemplate);

    Aws::SES::Model::CreateTemplateOutcome outcome = sesClient.CreateTemplate(
            createTemplateRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully created template." << templateName << "."
```

```
                    << std::endl;
    }
    else {
        std::cerr << "Error creating template. " <<
 outcome.GetError().GetMessage()
                    << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 CreateTemplate을 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { CreateTemplateCommand } from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";

const TEMPLATE_NAME = getUniqueName("TestTemplateName");

const createCreateTemplateCommand = () => {
  return new CreateTemplateCommand({
    /**
     * The template feature in Amazon SES is based on the Handlebars template
 system.
     */
    Template: {
      /**
       * The name of an existing template in Amazon SES.
       */
```

```
      TemplateName: TEMPLATE_NAME,
      HtmlPart: `
        <h1>Hello, {{contact.firstName}}!</h1>
        <p>
        Did you know Amazon has a mascot named Peccy?
        </p>
      `,
      SubjectPart: "Amazon Tip",
    },
  });
};

const run = async () => {
  const createTemplateCommand = createCreateTemplateCommand();

  try {
    return await sesClient.send(createTemplateCommand);
  } catch (err) {
    console.log("Failed to create template.", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 CreateTemplate을 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesTemplate:
    """Encapsulates Amazon SES template functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
```

```
        """
        self.ses_client = ses_client
        self.template = None
        self.template_tags = set()

    def _extract_tags(self, subject, text, html):
        """
        Extracts tags from a template as a set of unique values.

        :param subject: The subject of the email.
        :param text: The text version of the email.
        :param html: The html version of the email.
        """
        self.template_tags = set(re.findall(TEMPLATE_REGEX, subject + text +
html))
        logger.info("Extracted template tags: %s", self.template_tags)


    def create_template(self, name, subject, text, html):
        """
        Creates an email template.

        :param name: The name of the template.
        :param subject: The subject of the email.
        :param text: The plain text version of the email.
        :param html: The HTML version of the email.
        """
        try:
            template = {
                "TemplateName": name,
                "SubjectPart": subject,
                "TextPart": text,
                "HtmlPart": html,
            }
            self.ses_client.create_template(Template=template)
            logger.info("Created template %s.", name)
            self.template = template
            self._extract_tags(subject, text, html)
        except ClientError:
            logger.exception("Couldn't create template %s.", name)
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 CreateTemplate를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 **`DeleteIdentity`** 함께 사용

다음 코드 예시는 `DeleteIdentity`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.

- 이메일 ID 확인 및 메시지 전송

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Delete an email identity.
/// </summary>
/// <param name="identityEmail">The identity email to delete.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteIdentityAsync(string identityEmail)
{
    var success = false;
    try
    {
        var response = await _amazonSimpleEmailService.DeleteIdentityAsync(
            new DeleteIdentityRequest
            {
                Identity = identityEmail
            });
        success = response.HttpStatusCode == HttpStatusCode.OK;
```

```
        }
        catch (Exception ex)
        {
            Console.WriteLine("DeleteIdentityAsync failed with exception: " +
 ex.Message);
        }

        return success;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 DeleteIdentity를 참조하십시오.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Delete the specified identity (an email address or a domain).
/*!
  \param identity: The identity to delete.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::deleteIdentity(const Aws::String &identity,
                                 const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::DeleteIdentityRequest deleteIdentityRequest;

    deleteIdentityRequest.SetIdentity(identity);

    Aws::SES::Model::DeleteIdentityOutcome outcome = sesClient.DeleteIdentity(
            deleteIdentityRequest);
```

```
    if (outcome.IsSuccess()) {
        std::cout << "Successfully deleted identity." << std::endl;
    }
    else {
        std::cerr << "Error deleting identity. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 DeleteIdentity를 참조하세요.

CLI

AWS CLI

ID를 삭제하려면

다음 예시에서는 `delete-identity` 명령을 사용하여 Amazon SES로 확인된 ID 목록에서 ID를 삭제합니다.

```
aws ses delete-identity --identity user@example.com
```

확인된 ID에 대한 자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES에서 이메일 주소 및 도메인 확인을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeleteIdentity를 참조하세요.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DeleteIdentityCommand } from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";

const IDENTITY_EMAIL = "fake@example.com";

const createDeleteIdentityCommand = (identityName) => {
  return new DeleteIdentityCommand({
    Identity: identityName,
  });
};

const run = async () => {
  const deleteIdentityCommand = createDeleteIdentityCommand(IDENTITY_EMAIL);

  try {
    return await sesClient.send(deleteIdentityCommand);
  } catch (err) {
    console.log("Failed to delete identity.", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 DeleteIdentity를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesIdentity:
    """Encapsulates Amazon SES identity functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
```

```
        """
        self.ses_client = ses_client


    def delete_identity(self, identity):
        """
        Deletes an identity.

        :param identity: The identity to remove.
        """
        try:
            self.ses_client.delete_identity(Identity=identity)
            logger.info("Deleted identity %s.", identity)
        except ClientError:
            logger.exception("Couldn't delete identity %s.", identity)
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 DeleteIdentity를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`DeleteReceiptFilter`** 함께 사용

다음 코드 예시는 `DeleteReceiptFilter`의 사용 방법을 보여 줍니다.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Delete an Amazon Simple Email Service (Amazon SES) receipt filter.
/*!
  \param receiptFilterName: The name for the receipt filter.
```

```
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::deleteReceiptFilter(const Aws::String &receiptFilterName,
                                      const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::DeleteReceiptFilterRequest deleteReceiptFilterRequest;

    deleteReceiptFilterRequest.SetFilterName(receiptFilterName);

    Aws::SES::Model::DeleteReceiptFilterOutcome outcome =
 sesClient.DeleteReceiptFilter(
            deleteReceiptFilterRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully deleted receipt filter." << std::endl;
    }
    else {
        std::cerr << "Error deleting receipt filter. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 DeleteReceiptFilter를 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DeleteReceiptFilterCommand } from "@aws-sdk/client-ses";
```

```
import { sesClient } from "./libs/sesClient.js";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";

const RECEIPT_FILTER_NAME = getUniqueName("ReceiptFilterName");

const createDeleteReceiptFilterCommand = (filterName) => {
  return new DeleteReceiptFilterCommand({ FilterName: filterName });
};

const run = async () => {
  const deleteReceiptFilterCommand =
    createDeleteReceiptFilterCommand(RECEIPT_FILTER_NAME);

  try {
    return await sesClient.send(deleteReceiptFilterCommand);
  } catch (err) {
    console.log("Error deleting receipt filter.", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 DeleteReceiptFilter를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesReceiptHandler:
    """Encapsulates Amazon SES receipt handling functions."""

    def __init__(self, ses_client, s3_resource):
        """
        :param ses_client: A Boto3 Amazon SES client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        """
```

```
        self.ses_client = ses_client
        self.s3_resource = s3_resource


    def delete_receipt_filter(self, filter_name):
        """
        Deletes a receipt filter.

        :param filter_name: The name of the filter to delete.
        """
        try:
            self.ses_client.delete_receipt_filter(FilterName=filter_name)
            logger.info("Deleted receipt filter %s.", filter_name)
        except ClientError:
            logger.exception("Couldn't delete receipt filter %s.", filter_name)
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 DeleteReceiptFilter를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`DeleteReceiptRule`** 함께 사용

다음 코드 예시는 `DeleteReceiptRule`의 사용 방법을 보여 줍니다.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Delete an Amazon Simple Email Service (Amazon SES) receipt rule.
/*!
  \param receiptRuleName: The name for the receipt rule.
  \param receiptRuleSetName: The name for the receipt rule set.
```

```
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::deleteReceiptRule(const Aws::String &receiptRuleName,
                                    const Aws::String &receiptRuleSetName,
                                    const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::DeleteReceiptRuleRequest deleteReceiptRuleRequest;

    deleteReceiptRuleRequest.SetRuleName(receiptRuleName);
    deleteReceiptRuleRequest.SetRuleSetName(receiptRuleSetName);

    Aws::SES::Model::DeleteReceiptRuleOutcome outcome =
 sesClient.DeleteReceiptRule(
            deleteReceiptRuleRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully deleted receipt rule." << std::endl;
    }
    else {
        std::cout << "Error deleting receipt rule. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 DeleteReceiptRule을 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DeleteReceiptRuleCommand } from "@aws-sdk/client-ses";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { sesClient } from "./libs/sesClient.js";

const RULE_NAME = getUniqueName("RuleName");
const RULE_SET_NAME = getUniqueName("RuleSetName");

const createDeleteReceiptRuleCommand = () => {
  return new DeleteReceiptRuleCommand({
    RuleName: RULE_NAME,
    RuleSetName: RULE_SET_NAME,
  });
};

const run = async () => {
  const deleteReceiptRuleCommand = createDeleteReceiptRuleCommand();
  try {
    return await sesClient.send(deleteReceiptRuleCommand);
  } catch (err) {
    console.log("Failed to delete receipt rule.", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 DeleteReceiptRule을 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesReceiptHandler:
    """Encapsulates Amazon SES receipt handling functions."""

    def __init__(self, ses_client, s3_resource):
```

```
        """
        :param ses_client: A Boto3 Amazon SES client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        """
        self.ses_client = ses_client
        self.s3_resource = s3_resource


    def delete_receipt_rule(self, rule_set_name, rule_name):
        """
        Deletes a rule.

        :param rule_set_name: The rule set that contains the rule to delete.
        :param rule_name: The rule to delete.
        """
        try:
            self.ses_client.delete_receipt_rule(
                RuleSetName=rule_set_name, RuleName=rule_name
            )
            logger.info("Removed rule %s from rule set %s.", rule_name,
rule_set_name)
        except ClientError:
            logger.exception(
                "Couldn't remove rule %s from rule set %s.", rule_name,
rule_set_name
            )
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 DeleteReceiptRule를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`DeleteReceiptRuleSet`** 함께 사용

다음 코드 예시는 `DeleteReceiptRuleSet`의 사용 방법을 보여 줍니다.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Delete an Amazon Simple Email Service (Amazon SES) receipt rule set.
/*!
  \param receiptRuleSetName: The name for the receipt rule set.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::deleteReceiptRuleSet(const Aws::String &receiptRuleSetName,
                                       const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::DeleteReceiptRuleSetRequest deleteReceiptRuleSetRequest;

    deleteReceiptRuleSetRequest.SetRuleSetName(receiptRuleSetName);

    Aws::SES::Model::DeleteReceiptRuleSetOutcome outcome =
 sesClient.DeleteReceiptRuleSet(
            deleteReceiptRuleSetRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully deleted receipt rule set." << std::endl;
    }

    else {
        std::cerr << "Error deleting receipt rule set. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 DeleteReceiptRuleSet를 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DeleteReceiptRuleSetCommand } from "@aws-sdk/client-ses";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { sesClient } from "./libs/sesClient.js";

const RULE_SET_NAME = getUniqueName("RuleSetName");

const createDeleteReceiptRuleSetCommand = () => {
  return new DeleteReceiptRuleSetCommand({ RuleSetName: RULE_SET_NAME });
};

const run = async () => {
  const deleteReceiptRuleSetCommand = createDeleteReceiptRuleSetCommand();

  try {
    return await sesClient.send(deleteReceiptRuleSetCommand);
  } catch (err) {
    console.log("Failed to delete receipt rule set.", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 DeleteReceiptRuleSet를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesReceiptHandler:
    """Encapsulates Amazon SES receipt handling functions."""

    def __init__(self, ses_client, s3_resource):
        """
        :param ses_client: A Boto3 Amazon SES client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        """
        self.ses_client = ses_client
        self.s3_resource = s3_resource


    def delete_receipt_rule_set(self, rule_set_name):
        """
        Deletes a rule set. When a rule set is deleted, all of the rules it
 contains
        are also deleted.

        :param rule_set_name: The name of the rule set to delete.
        """
        try:
            self.ses_client.delete_receipt_rule_set(RuleSetName=rule_set_name)
            logger.info("Deleted rule set %s.", rule_set_name)
        except ClientError:
            logger.exception("Couldn't delete rule set %s.", rule_set_name)
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 DeleteReceiptRuleSet를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`DeleteTemplate`** 함께 사용

다음 코드 예시는 `DeleteTemplate`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.

- 이메일 ID 확인 및 메시지 전송

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Delete an email template.
/// </summary>
/// <param name="templateName">Name of the template.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteEmailTemplateAsync(string templateName)
{
    var success = false;
    try
    {
        var response = await _amazonSimpleEmailService.DeleteTemplateAsync(
            new DeleteTemplateRequest
            {
                TemplateName = templateName
            });
        success = response.HttpStatusCode == HttpStatusCode.OK;
    }
    catch (Exception ex)
```

```
        {
            Console.WriteLine("DeleteEmailTemplateAsync failed with exception: "
 + ex.Message);
        }

        return success;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 DeleteTemplate을 참조하십시오.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Delete an Amazon Simple Email Service (Amazon SES) template.
/*!
  \param templateName: The name for the template.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::deleteTemplate(const Aws::String &templateName,
                                 const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::DeleteTemplateRequest deleteTemplateRequest;

    deleteTemplateRequest.SetTemplateName(templateName);

    Aws::SES::Model::DeleteTemplateOutcome outcome = sesClient.DeleteTemplate(
            deleteTemplateRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully deleted template." << std::endl;
```

```
    }
    else {
        std::cerr << "Error deleting template. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 DeleteTemplate을 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DeleteTemplateCommand } from "@aws-sdk/client-ses";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { sesClient } from "./libs/sesClient.js";

const TEMPLATE_NAME = getUniqueName("TemplateName");

const createDeleteTemplateCommand = (templateName) =>
  new DeleteTemplateCommand({ TemplateName: templateName });

const run = async () => {
  const deleteTemplateCommand = createDeleteTemplateCommand(TEMPLATE_NAME);

  try {
    return await sesClient.send(deleteTemplateCommand);
  } catch (err) {
    console.log("Failed to delete template.", err);
    return err;
  }
```

```
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 DeleteTemplate을 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesTemplate:
    """Encapsulates Amazon SES template functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client
        self.template = None
        self.template_tags = set()

    def _extract_tags(self, subject, text, html):
        """
        Extracts tags from a template as a set of unique values.

        :param subject: The subject of the email.
        :param text: The text version of the email.
        :param html: The html version of the email.
        """
        self.template_tags = set(re.findall(TEMPLATE_REGEX, subject + text +
 html))
        logger.info("Extracted template tags: %s", self.template_tags)


    def delete_template(self):
        """
        Deletes an email template.
```

```
        """
        try:

 self.ses_client.delete_template(TemplateName=self.template["TemplateName"])
            logger.info("Deleted template %s.", self.template["TemplateName"])
            self.template = None
            self.template_tags = None
        except ClientError:
            logger.exception(
                "Couldn't delete template %s.", self.template["TemplateName"]
            )
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 DeleteTemplate를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`DescribeReceiptRuleSet`** 함께 사용

다음 코드 예시는 `DescribeReceiptRuleSet`의 사용 방법을 보여 줍니다.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesReceiptHandler:
    """Encapsulates Amazon SES receipt handling functions."""

    def __init__(self, ses_client, s3_resource):
        """
        :param ses_client: A Boto3 Amazon SES client.
        :param s3_resource: A Boto3 Amazon S3 resource.
```

```
        """
        self.ses_client = ses_client
        self.s3_resource = s3_resource


    def describe_receipt_rule_set(self, rule_set_name):
        """
        Gets data about a rule set.

        :param rule_set_name: The name of the rule set to retrieve.
        :return: Data about the rule set.
        """
        try:
            response = self.ses_client.describe_receipt_rule_set(
                RuleSetName=rule_set_name
            )
            logger.info("Got data for rule set %s.", rule_set_name)
        except ClientError:
            logger.exception("Couldn't get data for rule set %s.", rule_set_name)
            raise
        else:
            return response
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 DescribeReceiptRuleSet를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 **`GetIdentityVerificationAttributes`** 함께 사용

다음 코드 예시는 `GetIdentityVerificationAttributes`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.

- 이메일 ID 확인 및 메시지 전송

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Get identity verification status for an email.
    /// </summary>
    /// <returns>The verification status of the email.</returns>
    public async Task<VerificationStatus> GetIdentityStatusAsync(string email)
    {
        var result = VerificationStatus.TemporaryFailure;
        try
        {
            var response =
                await
_amazonSimpleEmailService.GetIdentityVerificationAttributesAsync(
                    new GetIdentityVerificationAttributesRequest
                    {
                        Identities = new List<string> { email }
                    });

            if (response.VerificationAttributes.ContainsKey(email))
                result =
response.VerificationAttributes[email].VerificationStatus;
        }
        catch (Exception ex)
        {
            Console.WriteLine("GetIdentityStatusAsync failed with exception: " +
ex.Message);
        }

        return result;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 GetIdentityVerificationAttributes를 참조하세요.

CLI

AWS CLI

ID 목록에 대한 Amazon SES 확인 상태를 가져오려면

다음 예시에서는 `get-identity-verification-attributes` 명령을 사용하여 ID 목록에 대한 Amazon SES 확인 상태를 가져옵니다.

```
aws ses get-identity-verification-attributes --
identities "user1@example.com" "user2@example.com"
```

출력:

```
{
   "VerificationAttributes": {
       "user1@example.com": {
           "VerificationStatus": "Success"
       },
       "user2@example.com": {
           "VerificationStatus": "Pending"
       }
   }
}
```

확인을 위해 제출한 적이 없는 ID를 사용하여 이 명령을 직접적으로 호출하는 경우 해당 ID는 출력에 표시되지 않습니다.

확인된 ID에 대한 자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES에서 이메일 주소 및 도메인 확인을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetIdentityVerificationAttributes를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesIdentity:
    """Encapsulates Amazon SES identity functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client


    def get_identity_status(self, identity):
        """
        Gets the status of an identity. This can be used to discover whether
        an identity has been successfully verified.

        :param identity: The identity to query.
        :return: The status of the identity.
        """
        try:
            response = self.ses_client.get_identity_verification_attributes(
                Identities=[identity]
            )
            status = response["VerificationAttributes"].get(
                identity, {"VerificationStatus": "NotFound"}
            )["VerificationStatus"]
            logger.info("Got status of %s for %s.", status, identity)
        except ClientError:
            logger.exception("Couldn't get status for %s.", identity)
            raise
        else:
            return status
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 GetIdentityVerificationAttributes를 참조하세요.

Ruby

SDK for Ruby

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require 'aws-sdk-ses' # v2: require 'aws-sdk'

# Create client in us-west-2 region
# Replace us-west-2 with the AWS Region you're using for Amazon SES.
client = Aws::SES::Client.new(region: 'us-west-2')

# Get up to 1000 identities
ids = client.list_identities({
                              identity_type: 'EmailAddress'
                            })

ids.identities.each do |email|
  attrs = client.get_identity_verification_attributes({
                                                        identities: [email]
                                                      })

  status = attrs.verification_attributes[email].verification_status

  # Display email addresses that have been verified
  puts email if status == 'Success'
end
```

- API 세부 정보는 AWS SDK for Ruby API 참조의 GetIdentityVerificationAttributes를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 **GetSendQuota** 함께 사용

다음 코드 예시는 GetSendQuota의 사용 방법을 보여 줍니다.

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Get information on the current account's send quota.
    /// </summary>
    /// <returns>The send quota response data.</returns>
    public async Task<GetSendQuotaResponse> GetSendQuotaAsync()
    {
        var result = new GetSendQuotaResponse();
        try
        {
            var response = await _amazonSimpleEmailService.GetSendQuotaAsync(
                new GetSendQuotaRequest());
            result = response;
        }
        catch (Exception ex)
        {
            Console.WriteLine("GetSendQuotaAsync failed with exception: " +
ex.Message);
        }

        return result;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 GetSendQuota를 참조하세요.

CLI

AWS CLI

Amazon SES 발신 한도 가져오기

다음 예시에서는 `get-send-quota` 명령을 사용하여 Amazon SES 발신 한도를 반환합니다.

```
aws ses get-send-quota
```

출력:

```
{
   "Max24HourSend": 200.0,
   "SentLast24Hours": 1.0,
   "MaxSendRate": 1.0
}
```

Max24HourSend는 발신 할당량으로, 24시간 내에 보낼 수 있는 최대 이메일 수입니다. 발신 할당량은 롤링 기간을 반영합니다. 이메일 전송을 시도할 때마다 Amazon SES에서 지난 24시간 내에 보낸 이메일 수를 확인합니다. 보낸 이메일의 총 수가 할당량보다 적으면 전송 요청이 수락되고 이메일이 전송됩니다.

SentLast24Hours는 지난 24시간 내에 보낸 이메일 수입니다.

MaxSendRate는 초당 보낼 수 있는 최대 이메일 수입니다.

발신 한도는 메시지 수가 아닌 수신자 수를 기준으로 한다는 점에 유의하세요. 예를 들어, 수신자가 10명인 이메일은 전송 할당량에서 10개로 간주됩니다.

자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES 발신 한도 관리를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetSendQuota를 참조하세요.

PowerShell

Tools for PowerShell V4

예제 1: 이 명령은 사용자의 현재 전송 한도를 반환합니다.

```
Get-SESSendQuota
```

- API 세부 정보는 Cmdlet 참조(V4)의 GetSendQuota를 참조하세요. AWS Tools for PowerShell

Tools for PowerShell V5

예제 1: 이 명령은 사용자의 현재 전송 한도를 반환합니다.

```
Get-SESSendQuota
```

- API 세부 정보는 Cmdlet 참조(V5)의 GetSendQuota를 참조하세요. AWS Tools for PowerShell

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

CLI로 **`GetSendStatistics`** 사용

다음 코드 예시는 `GetSendStatistics`의 사용 방법을 보여 줍니다.

CLI

AWS CLI

Amazon SES 전송 한도를 확인하는 방법

다음 예시에서는 `get-send-statistics` 명령을 사용하여 Amazon SES 발신 통계를 반환합니다.

```
aws ses get-send-statistics
```

출력:

```
{
   "SendDataPoints": [
       {
           "Complaints": 0,
           "Timestamp": "2013-06-12T19:32:00Z",
           "DeliveryAttempts": 2,
           "Bounces": 0,
```

```
            "Rejects": 0
        },
        {
            "Complaints": 0,
            "Timestamp": "2013-06-12T00:47:00Z",
            "DeliveryAttempts": 1,
            "Bounces": 0,
            "Rejects": 0
        }
    ]
}
```

결과는 지난 2주 동안의 전송 활동을 나타내는 데이터 포인트 목록입니다. 목록의 각 데이터 포인트에는 15분 간격의 통계가 포함됩니다.

이 예시에서는 지난 2주 동안 사용자가 보낸 유일한 이메일이 두 번의 15분 간격 내에 속했기 때문에 두 개의 데이터 포인트만 있습니다.

자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES 사용 통계 모니터링을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetSendStatistics 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예제 1: 이 명령은 사용자의 전송 통계를 반환합니다. 결과는 지난 2주 동안의 전송 활동을 나타내는 데이터 요소 목록입니다. 목록의 각 데이터 요소에는 15분 간격의 통계가 포함됩니다.

```
Get-SESSendStatistic
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 GetSendStatistics를 참조하세요.

Tools for PowerShell V5

예제 1: 이 명령은 사용자의 전송 통계를 반환합니다. 결과는 지난 2주 동안의 전송 활동을 나타내는 데이터 요소 목록입니다. 목록의 각 데이터 요소에는 15분 간격의 통계가 포함됩니다.

```
Get-SESSendStatistic
```

- API 세부 정보는 Cmdlet 참조(V5)의 GetSendStatistics를 참조하세요. AWS Tools for PowerShell

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`GetTemplate`** 함께 사용

다음 코드 예시는 `GetTemplate`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.

- 이메일 ID 확인 및 메시지 전송

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Get a template's attributes.
/*!
  \param templateName: The name for the template.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::getTemplate(const Aws::String &templateName,
                              const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::GetTemplateRequest getTemplateRequest;

    getTemplateRequest.SetTemplateName(templateName);

    Aws::SES::Model::GetTemplateOutcome outcome = sesClient.GetTemplate(
```

```
            getTemplateRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully got template." << std::endl;
    }

    else {
        std::cerr << "Error getting template. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 GetTemplate을 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { GetTemplateCommand } from "@aws-sdk/client-ses";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { sesClient } from "./libs/sesClient.js";

const TEMPLATE_NAME = getUniqueName("TemplateName");

const createGetTemplateCommand = (templateName) =>
  new GetTemplateCommand({ TemplateName: templateName });

const run = async () => {
  const getTemplateCommand = createGetTemplateCommand(TEMPLATE_NAME);

  try {
```

```
    return await sesClient.send(getTemplateCommand);
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MessageRejected") {
      /** @type { import('@aws-sdk/client-ses').MessageRejected} */
      const messageRejectedError = caught;
      return messageRejectedError;
    }
    throw caught;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 GetTemplate을 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesTemplate:
    """Encapsulates Amazon SES template functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client
        self.template = None
        self.template_tags = set()

    def _extract_tags(self, subject, text, html):
        """
        Extracts tags from a template as a set of unique values.

        :param subject: The subject of the email.
        :param text: The text version of the email.
        :param html: The html version of the email.
```

```
        """
        self.template_tags = set(re.findall(TEMPLATE_REGEX, subject + text +
html))
        logger.info("Extracted template tags: %s", self.template_tags)


    def get_template(self, name):
        """
        Gets a previously created email template.

        :param name: The name of the template to retrieve.
        :return: The retrieved email template.
        """
        try:
            response = self.ses_client.get_template(TemplateName=name)
            self.template = response["Template"]
            logger.info("Got template %s.", name)
            self._extract_tags(
                self.template["SubjectPart"],
                self.template["TextPart"],
                self.template["HtmlPart"],
            )
        except ClientError:
            logger.exception("Couldn't get template %s.", name)
            raise
        else:
            return self.template
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 GetTemplate를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 **`ListIdentities`** 함께 사용

다음 코드 예시는 `ListIdentities`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.

- 리전 전체에서 이메일 및 도메인 ID 복사

- 이메일 ID 확인 및 메시지 전송

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   /// <summary>
   /// Get the identities of a specified type for the current account.
   /// </summary>
   /// <param name="identityType">IdentityType to list.</param>
   /// <returns>The list of identities.</returns>
   public async Task<List<string>> ListIdentitiesAsync(IdentityType
identityType)
   {
       var result = new List<string>();
       try
       {
           var response = await _amazonSimpleEmailService.ListIdentitiesAsync(
               new ListIdentitiesRequest
               {
                   IdentityType = identityType
               });
           result = response.Identities;
       }
       catch (Exception ex)
       {
           Console.WriteLine("ListIdentitiesAsync failed with exception: " +
ex.Message);
       }

       return result;
   }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 ListIdentities를 참조하십시오.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! List the identities associated with this account.
/*!
  \param identityType: The identity type enum. "NOT_SET" is a valid option.
  \param identities; A vector to receive the retrieved identities.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::listIdentities(Aws::SES::Model::IdentityType identityType,
                                 Aws::Vector<Aws::String> &identities,
                                 const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::ListIdentitiesRequest listIdentitiesRequest;

    if (identityType != Aws::SES::Model::IdentityType::NOT_SET) {
        listIdentitiesRequest.SetIdentityType(identityType);
    }

    Aws::String nextToken; // Used for paginated results.
    do {
        if (!nextToken.empty()) {
            listIdentitiesRequest.SetNextToken(nextToken);
        }
        Aws::SES::Model::ListIdentitiesOutcome outcome =
 sesClient.ListIdentities(
                listIdentitiesRequest);

        if (outcome.IsSuccess()) {
```

```
            const auto &retrievedIdentities =
 outcome.GetResult().GetIdentities();
            if (!retrievedIdentities.empty()) {
                identities.insert(identities.cend(),
 retrievedIdentities.cbegin(),
                                  retrievedIdentities.cend());
            }
            nextToken = outcome.GetResult().GetNextToken();
        }
        else {
            std::cout << "Error listing identities. " <<
 outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    } while (!nextToken.empty());

    return true;
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 ListIdentities를 참조하세요.

CLI

AWS CLI

특정 AWS 계정의 모든 자격 증명(이메일 주소 및 도메인)을 나열하려면

다음 예시에서는 `list-identities` 명령을 사용하여 Amazon SES로 확인하기 위해 제출된 모든 ID를 나열합니다.

```
aws ses list-identities
```

출력:

```
{
    "Identities": [
      "user@example.com",
      "example.com"
    ]
```

```
}
```

반환되는 목록에는 확인 상태(확인됨, 확인 보류 중, 실패 등)와 상관없이 모든 ID가 포함됩니다.

이 예시에서는 ID 유형 파라미터를 지정하지 않았으므로 이메일 주소와 도메인이 모두 반환됩니다.

확인에 대한 자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES에서 이메일 주소 및 도메인 확인을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 ListIdentities를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ses.SesClient;
import software.amazon.awssdk.services.ses.model.ListIdentitiesResponse;
import software.amazon.awssdk.services.ses.model.SesException;
import java.io.IOException;
import java.util.List;

/**
 * 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 ListIdentities {

    public static void main(String[] args) throws IOException {
```

```
        Region region = Region.US_WEST_2;
        SesClient client = SesClient.builder()
                .region(region)
                .build();

        listSESIdentities(client);
    }

    public static void listSESIdentities(SesClient client) {
        try {
            ListIdentitiesResponse identitiesResponse = client.listIdentities();
            List<String> identities = identitiesResponse.identities();
            for (String identity : identities) {
                System.out.println("The identity is " + identity);
            }

        } catch (SesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 ListIdentities를 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { ListIdentitiesCommand } from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";

const createListIdentitiesCommand = () =>
  new ListIdentitiesCommand({ IdentityType: "EmailAddress", MaxItems: 10 });
```

```
const run = async () => {
  const listIdentitiesCommand = createListIdentitiesCommand();

  try {
    return await sesClient.send(listIdentitiesCommand);
  } catch (err) {
    console.log("Failed to list identities.", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 ListIdentities를 참조하세요.

PowerShell

Tools for PowerShell V4

예제 1:이 명령은 확인 상태에 관계없이 특정 AWS 계정의 모든 자격 증명(이메일 주소 및 도메인)이 포함된 목록을 반환합니다.

```
Get-SESIdentity
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 ListIdentities를 참조하세요.

Tools for PowerShell V5

예제 1:이 명령은 확인 상태에 관계없이 특정 AWS 계정의 모든 자격 증명(이메일 주소 및 도메인)이 포함된 목록을 반환합니다.

```
Get-SESIdentity
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 ListIdentities를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesIdentity:
    """Encapsulates Amazon SES identity functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client


    def list_identities(self, identity_type, max_items):
        """
        Gets the identities of the specified type for the current account.

        :param identity_type: The type of identity to retrieve, such as
 EmailAddress.
        :param max_items: The maximum number of identities to retrieve.
        :return: The list of retrieved identities.
        """
        try:
            response = self.ses_client.list_identities(
                IdentityType=identity_type, MaxItems=max_items
            )
            identities = response["Identities"]
            logger.info("Got %s identities for the current account.",
 len(identities))
        except ClientError:
            logger.exception("Couldn't list identities for the current account.")
            raise
        else:
            return identities
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 ListIdentities를 참조하세요.

Ruby

SDK for Ruby

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require 'aws-sdk-ses' # v2: require 'aws-sdk'

# Create client in us-west-2 region
# Replace us-west-2 with the AWS Region you're using for Amazon SES.
client = Aws::SES::Client.new(region: 'us-west-2')

# Get up to 1000 identities
ids = client.list_identities({
                              identity_type: 'EmailAddress'
                            })

ids.identities.each do |email|
  attrs = client.get_identity_verification_attributes({
                                                        identities: [email]
                                                      })

  status = attrs.verification_attributes[email].verification_status

  # Display email addresses that have been verified
  puts email if status == 'Success'
end
```

- API 세부 정보는 AWS SDK for Ruby API 참조의 ListIdentities를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **ListReceiptFilters** 함께 사용

다음 코드 예시는 `ListReceiptFilters`의 사용 방법을 보여 줍니다.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! List the receipt filters associated with this account.
/*!
  \param filters; A vector of "ReceiptFilter" to receive the retrieved filters.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool
AwsDoc::SES::listReceiptFilters(Aws::Vector<Aws::SES::Model::ReceiptFilter>
 &filters,
                                const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);
    Aws::SES::Model::ListReceiptFiltersRequest listReceiptFiltersRequest;

    Aws::SES::Model::ListReceiptFiltersOutcome outcome =
 sesClient.ListReceiptFilters(
            listReceiptFiltersRequest);
    if (outcome.IsSuccess()) {
        auto &retrievedFilters = outcome.GetResult().GetFilters();
        if (!retrievedFilters.empty()) {
            filters.insert(filters.cend(), retrievedFilters.cbegin(),
                           retrievedFilters.cend());
        }
    }
    else {
```

```
        std::cerr << "Error retrieving IP address filters: "
                  << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 ListReceiptFilters를 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { ListReceiptFiltersCommand } from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";

const createListReceiptFiltersCommand = () => new ListReceiptFiltersCommand({});

const run = async () => {
  const listReceiptFiltersCommand = createListReceiptFiltersCommand();

  return await sesClient.send(listReceiptFiltersCommand);
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 ListReceiptFilters를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesReceiptHandler:
    """Encapsulates Amazon SES receipt handling functions."""

    def __init__(self, ses_client, s3_resource):
        """
        :param ses_client: A Boto3 Amazon SES client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        """
        self.ses_client = ses_client
        self.s3_resource = s3_resource


    def list_receipt_filters(self):
        """
        Gets the list of receipt filters for the current account.

        :return: The list of receipt filters.
        """
        try:
            response = self.ses_client.list_receipt_filters()
            filters = response["Filters"]
            logger.info("Got %s receipt filters.", len(filters))
        except ClientError:
            logger.exception("Couldn't get receipt filters.")
            raise
        else:
            return filters
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 ListReceiptFilters를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`ListTemplates`** 함께 사용

다음 코드 예시는 `ListTemplates`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.

- 이메일 ID 확인 및 메시지 전송

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// List email templates for the current account.
    /// </summary>
    /// <returns>A list of template metadata.</returns>
    public async Task<List<TemplateMetadata>> ListEmailTemplatesAsync()
    {
        var result = new List<TemplateMetadata>();
        try
        {
            var response = await _amazonSimpleEmailService.ListTemplatesAsync(
                new ListTemplatesRequest());
            result = response.TemplatesMetadata;
        }
        catch (Exception ex)
        {
            Console.WriteLine("ListEmailTemplatesAsync failed with exception: " +
ex.Message);
        }
```

```
        return result;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 ListTemplates를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sesv2.SesV2Client;
import software.amazon.awssdk.services.sesv2.model.ListEmailTemplatesRequest;
import software.amazon.awssdk.services.sesv2.model.ListEmailTemplatesResponse;
import software.amazon.awssdk.services.sesv2.model.SesV2Exception;

public class ListTemplates {

    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        SesV2Client sesv2Client = SesV2Client.builder()
                .region(region)
                .build();

        listAllTemplates(sesv2Client);
    }

    public static void listAllTemplates(SesV2Client sesv2Client) {
        try {
            ListEmailTemplatesRequest templatesRequest =
 ListEmailTemplatesRequest.builder()
                    .pageSize(1)
                    .build();
```

```
            ListEmailTemplatesResponse response =
 sesv2Client.listEmailTemplates(templatesRequest);
            response.templatesMetadata()
                    .forEach(template -> System.out.println("Template name: " +
 template.templateName()));

        } catch (SesV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 ListTemplates를 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { ListTemplatesCommand } from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";

const createListTemplatesCommand = (maxItems) =>
  new ListTemplatesCommand({ MaxItems: maxItems });

const run = async () => {
  const listTemplatesCommand = createListTemplatesCommand(10);

  try {
    return await sesClient.send(listTemplatesCommand);
  } catch (err) {
    console.log("Failed to list templates.", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 ListTemplates를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesTemplate:
    """Encapsulates Amazon SES template functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client
        self.template = None
        self.template_tags = set()

    def _extract_tags(self, subject, text, html):
        """
        Extracts tags from a template as a set of unique values.

        :param subject: The subject of the email.
        :param text: The text version of the email.
        :param html: The html version of the email.
        """
        self.template_tags = set(re.findall(TEMPLATE_REGEX, subject + text +
 html))
        logger.info("Extracted template tags: %s", self.template_tags)


    def list_templates(self):
        """
        Gets a list of all email templates for the current account.
```

```
        :return: The list of retrieved email templates.
        """
        try:
            response = self.ses_client.list_templates()
            templates = response["TemplatesMetadata"]
            logger.info("Got %s templates.", len(templates))
        except ClientError:
            logger.exception("Couldn't get templates.")
            raise
        else:
            return templates
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 ListTemplates를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **SendBulkTemplatedEmail** 함께 사용

다음 코드 예시는 SendBulkTemplatedEmail의 사용 방법을 보여 줍니다.

JavaScript

JavaScript용 SDK(v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { SendBulkTemplatedEmailCommand } from "@aws-sdk/client-ses";
import {
  getUniqueName,
  postfix,
} from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { sesClient } from "./libs/sesClient.js";

/**
 * Replace this with the name of an existing template.
```

```
 */
const TEMPLATE_NAME = getUniqueName("ReminderTemplate");

/**
 * Replace these with existing verified emails.
 */
const VERIFIED_EMAIL_1 = postfix(getUniqueName("Bilbo"), "@example.com");
const VERIFIED_EMAIL_2 = postfix(getUniqueName("Frodo"), "@example.com");

const USERS = [
  { firstName: "Bilbo", emailAddress: VERIFIED_EMAIL_1 },
  { firstName: "Frodo", emailAddress: VERIFIED_EMAIL_2 },
];

/**
 *
 * @param { { emailAddress: string, firstName: string }[] } users
 * @param { string } templateName the name of an existing template in SES
 * @returns { SendBulkTemplatedEmailCommand }
 */
const createBulkReminderEmailCommand = (users, templateName) => {
  return new SendBulkTemplatedEmailCommand({
    /**
     * Each 'Destination' uses a corresponding set of replacement data. We can
 map each user
     * to a 'Destination' and provide user specific replacement data to create
 personalized emails.
     *
     * Here's an example of how a template would be replaced with user data:
     * Template: <h1>Hello {{name}},</h1><p>Don't forget about the party gifts!</
p>
     * Destination 1: <h1>Hello Bilbo,</h1><p>Don't forget about the party gifts!
</p>
     * Destination 2: <h1>Hello Frodo,</h1><p>Don't forget about the party gifts!
</p>
     */
    Destinations: users.map((user) => ({
      Destination: { ToAddresses: [user.emailAddress] },
      ReplacementTemplateData: JSON.stringify({ name: user.firstName }),
    })),
    DefaultTemplateData: JSON.stringify({ name: "Shireling" }),
    Source: VERIFIED_EMAIL_1,
    Template: templateName,
  });
```

```
};

const run = async () => {
  const sendBulkTemplateEmailCommand = createBulkReminderEmailCommand(
    USERS,
    TEMPLATE_NAME,
  );
  try {
    return await sesClient.send(sendBulkTemplateEmailCommand);
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MessageRejected") {
      /** @type { import('@aws-sdk/client-ses').MessageRejected} */
      const messageRejectedError = caught;
      return messageRejectedError;
    }
    throw caught;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 SendBulkTemplatedEmail을 참조하십시오.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 **`SendEmail`** 함께 사용

다음 코드 예시는 `SendEmail`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.

- 이메일 ID 확인 및 메시지 전송

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
///  Send an email by using Amazon SES.
/// </summary>
/// <param name="toAddresses">List of recipients.</param>
/// <param name="ccAddresses">List of cc recipients.</param>
/// <param name="bccAddresses">List of bcc recipients.</param>
/// <param name="bodyHtml">Body of the email in HTML.</param>
/// <param name="bodyText">Body of the email in plain text.</param>
/// <param name="subject">Subject line of the email.</param>
/// <param name="senderAddress">From address.</param>
/// <returns>The messageId of the email.</returns>
public async Task<string> SendEmailAsync(List<string> toAddresses,
    List<string> ccAddresses, List<string> bccAddresses,
    string bodyHtml, string bodyText, string subject, string senderAddress)
{
    var messageId = "";
    try
    {
        var response = await _amazonSimpleEmailService.SendEmailAsync(
            new SendEmailRequest
            {
                Destination = new Destination
                {
                    BccAddresses = bccAddresses,
                    CcAddresses = ccAddresses,
                    ToAddresses = toAddresses
                },
                Message = new Message
                {
                    Body = new Body
                    {
```

```
                            Html = new Content
                            {
                                Charset = "UTF-8",
                                Data = bodyHtml
                            },
                            Text = new Content
                            {
                                Charset = "UTF-8",
                                Data = bodyText
                            }
                        },
                        Subject = new Content
                        {
                            Charset = "UTF-8",
                            Data = subject
                        }
                    },
                    Source = senderAddress
                });
            messageId = response.MessageId;
        }
        catch (Exception ex)
        {
            Console.WriteLine("SendEmailAsync failed with exception: " +
ex.Message);
        }

        return messageId;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 SendEmail을 참조하십시오.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Send an email to a list of recipients.
/*!
  \param recipients; Vector of recipient email addresses.
  \param subject: Email subject.
  \param htmlBody: Email body as HTML. At least one body data is required.
  \param textBody: Email body as plain text. At least one body data is required.
  \param senderEmailAddress: Email address of sender. Ignored if empty string.
  \param ccAddresses: Vector of cc addresses. Ignored if empty.
  \param replyToAddress: Reply to email address. Ignored if empty string.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::sendEmail(const Aws::Vector<Aws::String> &recipients,
                            const Aws::String &subject,
                            const Aws::String &htmlBody,
                            const Aws::String &textBody,
                            const Aws::String &senderEmailAddress,
                            const Aws::Vector<Aws::String> &ccAddresses,
                            const Aws::String &replyToAddress,
                            const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::Destination destination;
    if (!ccAddresses.empty()) {
        destination.WithCcAddresses(ccAddresses);
    }
    if (!recipients.empty()) {
        destination.WithToAddresses(recipients);
    }

    Aws::SES::Model::Body message_body;
    if (!htmlBody.empty()) {
        message_body.SetHtml(

 Aws::SES::Model::Content().WithCharset("UTF-8").WithData(htmlBody));
    }

    if (!textBody.empty()) {
        message_body.SetText(

 Aws::SES::Model::Content().WithCharset("UTF-8").WithData(textBody));
    }
```

```
    Aws::SES::Model::Message message;
    message.SetBody(message_body);
    message.SetSubject(
            Aws::SES::Model::Content().WithCharset("UTF-8").WithData(subject));

    Aws::SES::Model::SendEmailRequest sendEmailRequest;
    sendEmailRequest.SetDestination(destination);
    sendEmailRequest.SetMessage(message);
    if (!senderEmailAddress.empty()) {
        sendEmailRequest.SetSource(senderEmailAddress);
    }
    if (!replyToAddress.empty()) {
        sendEmailRequest.AddReplyToAddresses(replyToAddress);
    }

    auto outcome = sesClient.SendEmail(sendEmailRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully sent message with ID "
                  << outcome.GetResult().GetMessageId()
                  << "." << std::endl;
    }
    else {
        std::cerr << "Error sending message. " << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 SendEmail을 참조하세요.

CLI

AWS CLI

Amazon SES를 사용하여 서식 지정된 이메일 전송

다음 예시에서는 `send-email` 명령을 사용하여 서식이 지정된 이메일을 보냅니다.

```
aws ses send-email --from sender@example.com --destination file:// destination.json --message file://message.json
```

출력:

```
{
   "MessageId": "EXAMPLEf3a5efcd1-51adec81-d2a4-4e3f-9fe2-5d85c1b23783-000000"
}
```

대상 및 메시지는 현재 디렉터리의 .json 파일에 저장된 JSON 데이터 구조입니다. 이러한 파일은 다음과 같습니다.

`destination.json`:

```
{
  "ToAddresses":  ["recipient1@example.com", "recipient2@example.com"],
  "CcAddresses":  ["recipient3@example.com"],
  "BccAddresses": []
}
```

`message.json`:

```
{
   "Subject": {
       "Data": "Test email sent using the AWS CLI",
       "Charset": "UTF-8"
   },
   "Body": {
       "Text": {
           "Data": "This is the message body in text format.",
           "Charset": "UTF-8"
       },
       "Html": {
           "Data": "This message body contains HTML formatting. It can, for
 example, contain links like this one: <a class=\"ulink\" href=\"http://
docs.aws.amazon.com/ses/latest/DeveloperGuide\" target=\"_blank\">Amazon SES
 Developer Guide</a>.",
           "Charset": "UTF-8"
       }
   }
}
```

발신자 및 수신자 이메일 주소를 사용하려는 주소로 바꿉니다. 단, 발신자의 이메일 주소는 Amazon SES로 확인해야 합니다. Amazon SES에 대한 프로덕션 액세스 권한을 부여받기 전까지는 각 수신자의 이메일 주소도 확인해야 합니다. 단, 수신자가 Amazon SES 메일박스 시뮬레이터인 경우는 예외입니다. 확인에 대한 자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES에서 이메일 주소 및 도메인 확인을 참조하세요.

출력의 Message ID는 직접적인 send-email 호출이 성공했음을 나타냅니다.

이메일을 받지 못한 경우 정크 박스를 확인해 보세요.

서식 지정된 이메일을 보내는 방법에 대한 자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES API를 사용하여 서식 지정된 이메일 전송을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 SendEmail을 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ses.SesClient;
import software.amazon.awssdk.services.ses.model.Content;
import software.amazon.awssdk.services.ses.model.Destination;
import software.amazon.awssdk.services.ses.model.Message;
import software.amazon.awssdk.services.ses.model.Body;
import software.amazon.awssdk.services.ses.model.SendEmailRequest;
import software.amazon.awssdk.services.ses.model.SesException;

import javax.mail.MessagingException;

/**
 * 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 SendMessageEmailRequest {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <sender> <recipient> <subject>\s

                Where:
                    sender - An email address that represents the sender.\s
                    recipient -  An email address that represents the recipient.
\s
                    subject - The  subject line.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String sender = args[0];
        String recipient = args[1];
        String subject = args[2];

        Region region = Region.US_EAST_1;
        SesClient client = SesClient.builder()
                .region(region)
                .build();

        // The HTML body of the email.
        String bodyHTML = "<html>" + "<head></head>" + "<body>" + "<h1>Hello!</
h1>"
                + "<p> See the list of customers.</p>" + "</body>" + "</html>";

        try {
            send(client, sender, recipient, subject, bodyHTML);
            client.close();
            System.out.println("Done");

        } catch (MessagingException e) {
            e.printStackTrace();
        }
```

```
    }

    public static void send(SesClient client,
            String sender,
            String recipient,
            String subject,
            String bodyHTML) throws MessagingException {

        Destination destination = Destination.builder()
                .toAddresses(recipient)
                .build();

        Content content = Content.builder()
                .data(bodyHTML)
                .build();

        Content sub = Content.builder()
                .data(subject)
                .build();

        Body body = Body.builder()
                .html(content)
                .build();

        Message msg = Message.builder()
                .subject(sub)
                .body(body)
                .build();

        SendEmailRequest emailRequest = SendEmailRequest.builder()
                .destination(destination)
                .message(msg)
                .source(sender)
                .build();

        try {
            System.out.println("Attempting to send an email through Amazon SES "
+ "using the AWS SDK for Java...");
            client.sendEmail(emailRequest);

        } catch (SesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
```

```
    }
}

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ses.SesClient;
import javax.activation.DataHandler;
import javax.activation.DataSource;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.Session;
import javax.mail.internet.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMultipart;
import javax.mail.internet.MimeBodyPart;
import javax.mail.util.ByteArrayDataSource;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.nio.ByteBuffer;
import java.nio.file.Files;
import java.util.Properties;
import software.amazon.awssdk.core.SdkBytes;
import software.amazon.awssdk.services.ses.model.SendRawEmailRequest;
import software.amazon.awssdk.services.ses.model.RawMessage;
import software.amazon.awssdk.services.ses.model.SesException;

/**
 * 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 SendMessageAttachment {
    public static void main(String[] args) throws IOException {
        final String usage = """

                Usage:
                    <sender> <recipient> <subject> <fileLocation>\s

                Where:
```

```
                    sender - An email address that represents the sender.\s
                    recipient -  An email address that represents the recipient.
\s
                    subject - The  subject line.\s
                    fileLocation - The location of a Microsoft Excel file to use
 as an attachment (C:/AWS/customers.xls).\s
                """;

        if (args.length != 4) {
            System.out.println(usage);
            System.exit(1);
        }

        String sender = args[0];
        String recipient = args[1];
        String subject = args[2];
        String fileLocation = args[3];

        // The email body for recipients with non-HTML email clients.
        String bodyText = "Hello,\r\n" + "Please see the attached file for a list
 "
                + "of customers to contact.";

        // The HTML body of the email.
        String bodyHTML = "<html>" + "<head></head>" + "<body>" + "<h1>Hello!</
h1>"
                + "<p>Please see the attached file for a " + "list of customers
 to contact.</p>" + "</body>"
                + "</html>";

        Region region = Region.US_WEST_2;
        SesClient client = SesClient.builder()
                .region(region)
                .build();

        try {
            sendemailAttachment(client, sender, recipient, subject, bodyText,
 bodyHTML, fileLocation);
            client.close();
            System.out.println("Done");

        } catch (IOException | MessagingException e) {
            e.printStackTrace();
        }
```

```
    }

    public static void sendemailAttachment(SesClient client,
            String sender,
            String recipient,
            String subject,
            String bodyText,
            String bodyHTML,
            String fileLocation) throws AddressException, MessagingException,
IOException {

        java.io.File theFile = new java.io.File(fileLocation);
        byte[] fileContent = Files.readAllBytes(theFile.toPath());

        Session session = Session.getDefaultInstance(new Properties());

        // Create a new MimeMessage object.
        MimeMessage message = new MimeMessage(session);

        // Add subject, from and to lines.
        message.setSubject(subject, "UTF-8");
        message.setFrom(new InternetAddress(sender));
        message.setRecipients(Message.RecipientType.TO,
InternetAddress.parse(recipient));

        // Create a multipart/alternative child container.
        MimeMultipart msgBody = new MimeMultipart("alternative");

        // Create a wrapper for the HTML and text parts.
        MimeBodyPart wrap = new MimeBodyPart();

        // Define the text part.
        MimeBodyPart textPart = new MimeBodyPart();
        textPart.setContent(bodyText, "text/plain; charset=UTF-8");

        // Define the HTML part.
        MimeBodyPart htmlPart = new MimeBodyPart();
        htmlPart.setContent(bodyHTML, "text/html; charset=UTF-8");

        // Add the text and HTML parts to the child container.
        msgBody.addBodyPart(textPart);
        msgBody.addBodyPart(htmlPart);

        // Add the child container to the wrapper object.
```

```
        wrap.setContent(msgBody);

        // Create a multipart/mixed parent container.
        MimeMultipart msg = new MimeMultipart("mixed");

        // Add the parent container to the message.
        message.setContent(msg);
        msg.addBodyPart(wrap);

        // Define the attachment.
        MimeBodyPart att = new MimeBodyPart();
        DataSource fds = new ByteArrayDataSource(fileContent,
                "application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet");
        att.setDataHandler(new DataHandler(fds));

        String reportName = "WorkReport.xls";
        att.setFileName(reportName);

        // Add the attachment to the message.
        msg.addBodyPart(att);

        try {
            System.out.println("Attempting to send an email through Amazon SES "
 + "using the AWS SDK for Java...");

            ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
            message.writeTo(outputStream);

            ByteBuffer buf = ByteBuffer.wrap(outputStream.toByteArray());

            byte[] arr = new byte[buf.remaining()];
            buf.get(arr);

            SdkBytes data = SdkBytes.fromByteArray(arr);
            RawMessage rawMessage = RawMessage.builder()
                    .data(data)
                    .build();

            SendRawEmailRequest rawEmailRequest = SendRawEmailRequest.builder()
                    .rawMessage(rawMessage)
                    .build();

            client.sendRawEmail(rawEmailRequest);
```

```
        } catch (SesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Email sent using SesClient with attachment");
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 SendEmail을 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { SendEmailCommand } from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";

const createSendEmailCommand = (toAddress, fromAddress) => {
  return new SendEmailCommand({
    Destination: {
      /* required */
      CcAddresses: [
        /* more items */
      ],
      ToAddresses: [
        toAddress,
        /* more To-email addresses */
      ],
    },
    Message: {
      /* required */
      Body: {
        /* required */
        Html: {
```

```
          Charset: "UTF-8",
          Data: "HTML_FORMAT_BODY",
        },
        Text: {
          Charset: "UTF-8",
          Data: "TEXT_FORMAT_BODY",
        },
      },
      Subject: {
        Charset: "UTF-8",
        Data: "EMAIL_SUBJECT",
      },
    },
    Source: fromAddress,
    ReplyToAddresses: [
      /* more items */
    ],
  });
};

const run = async () => {
  const sendEmailCommand = createSendEmailCommand(
    "recipient@example.com",
    "sender@example.com",
  );

  try {
    return await sesClient.send(sendEmailCommand);
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MessageRejected") {
      /** @type { import('@aws-sdk/client-ses').MessageRejected} */
      const messageRejectedError = caught;
      return messageRejectedError;
    }
    throw caught;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 SendEmail을 참조하십시오.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesMailSender:
    """Encapsulates functions to send emails with Amazon SES."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client


    def send_email(self, source, destination, subject, text, html,
 reply_tos=None):
        """
        Sends an email.

        Note: If your account is in the Amazon SES  sandbox, the source and
        destination email accounts must both be verified.

        :param source: The source email account.
        :param destination: The destination email account.
        :param subject: The subject of the email.
        :param text: The plain text version of the body of the email.
        :param html: The HTML version of the body of the email.
        :param reply_tos: Email accounts that will receive a reply if the
 recipient
                          replies to the message.
        :return: The ID of the message, assigned by Amazon SES.
        """
        send_args = {
            "Source": source,
            "Destination": destination.to_service_format(),
            "Message": {
```

```
                "Subject": {"Data": subject},
                "Body": {"Text": {"Data": text}, "Html": {"Data": html}},
            },
        }
        if reply_tos is not None:
            send_args["ReplyToAddresses"] = reply_tos
        try:
            response = self.ses_client.send_email(**send_args)
            message_id = response["MessageId"]
            logger.info(
                "Sent mail %s from %s to %s.", message_id, source,
 destination.tos
            )
        except ClientError:
            logger.exception(
                "Couldn't send mail from %s to %s.", source, destination.tos
            )
            raise
        else:
            return message_id
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 SendEmail를 참조하세요.

Ruby

SDK for Ruby

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require 'aws-sdk-ses' # v2: require 'aws-sdk'

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
sender = 'sender@example.com'
```

```
# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
recipient = 'recipient@example.com'

# Specify a configuration set. To use a configuration
# set, uncomment the next line and line 74.
#   configsetname = "ConfigSet"

# The subject line for the email.
subject = 'Amazon SES test (AWS SDK for Ruby)'

# The HTML body of the email.
htmlbody =
  '<h1>Amazon SES test (AWS SDK for Ruby)</h1>'\
  '<p>This email was sent with <a href="https://aws.amazon.com/ses/">'\
  'Amazon SES</a> using the <a href="https://aws.amazon.com/sdk-for-ruby/">'\
  'AWS SDK for Ruby</a>.'

# The email body for recipients with non-HTML email clients.
textbody = 'This email was sent with Amazon SES using the AWS SDK for Ruby.'

# Specify the text encoding scheme.
encoding = 'UTF-8'

# Create a new SES client in the us-west-2 region.
# Replace us-west-2 with the AWS Region you're using for Amazon SES.
ses = Aws::SES::Client.new(region: 'us-west-2')

# Try to send the email.
begin
  # Provide the contents of the email.
  ses.send_email(
    destination: {
      to_addresses: [
        recipient
      ]
    },
    message: {
      body: {
        html: {
          charset: encoding,
          data: htmlbody
        },
        text: {
```

```
          charset: encoding,
          data: textbody
        }
      },
      subject: {
        charset: encoding,
        data: subject
      }
    },
    source: sender
    # Uncomment the following line to use a configuration set.
    # configuration_set_name: configsetname,
  )

  puts "Email sent to #{recipient}"

# If something goes wrong, display an error message.
rescue Aws::SES::Errors::ServiceError => e
  puts "Email not sent. Error message: #{e}"
end
```

- API 세부 정보는 AWS SDK for Ruby API 참조의 SendEmail을 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 **SendRawEmail** 함께 사용

다음 코드 예시는 SendRawEmail의 사용 방법을 보여 줍니다.

CLI

AWS CLI

Amazon SES를 사용하여 원시 이메일 전송

다음 예시에서는 send-raw-email 명령을 사용하여 TXT 첨부 파일이 있는 이메일을 보냅니다.

```
aws ses send-raw-email --raw-message file://message.json
```

출력:

```
{
   "MessageId": "EXAMPLEf3f73d99b-c63fb06f-d263-41f8-a0fb-d0dc67d56c07-000000"
}
```

원시 메시지는 현재 디렉터리의 `message.json` 파일에 저장된 JSON 데이터 구조입니다. 이는 다음을 포함합니다.

```
{
   "Data": "From: sender@example.com\nTo: recipient@example.com\nSubject:
 Test email sent using the AWS CLI (contains an attachment)\nMIME-Version:
 1.0\nContent-type: Multipart/Mixed; boundary=\"NextPart\"\n\n--NextPart
\nContent-Type: text/plain\n\nThis is the message body.\n\n--NextPart\nContent-
Type: text/plain;\nContent-Disposition: attachment; filename=\"attachment.txt\"\n
\nThis is the text in the attachment.\n\n--NextPart--"
}
```

보시다시피 'Data'는 attachment.txt 첨부 파일을 포함하여 MIME 형식의 원시 이메일 콘텐츠 전체를 포함하는 하나의 긴 문자열입니다.

sender@example.com 및 recipient@example.com을 사용하려는 주소로 바꿉니다. 단, 발신자의 이메일 주소는 Amazon SES로 확인해야 합니다. Amazon SES에 대한 프로덕션 액세스 권한을 부여받기 전까지는 수신자의 이메일 주소도 확인해야 합니다. 단, 수신자가 Amazon SES 메일박스 시뮬레이터가 아닌 경우는 예외입니다. 확인에 대한 자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES에서 이메일 주소 및 도메인 확인을 참조하세요.

출력의 Message ID는 직접적인 send-raw-email 호출이 성공했음을 나타냅니다.

이메일을 받지 못한 경우 정크 박스를 확인해 보세요.

원시 이메일을 보내는 방법에 대한 자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES API를 사용하여 원시 이메일 전송을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 SendRawEmail을 참조하세요.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

nodemailer를 사용하여 첨부 파일이 있는 이메일을 보냅니다.

```
import sesClientModule from "@aws-sdk/client-ses";
/**
 * nodemailer wraps the SES SDK and calls SendRawEmail. Use this for more
 advanced
 * functionality like adding attachments to your email.
 *
 * https://nodemailer.com/transports/ses/
 */
import nodemailer from "nodemailer";

/**
 * @param {string} from An Amazon SES verified email address.
 * @param {*} to An Amazon SES verified email address.
 */
export const sendEmailWithAttachments = (
  from = "from@example.com",
  to = "to@example.com",
) => {
  const ses = new sesClientModule.SESClient({});
  const transporter = nodemailer.createTransport({
    SES: { ses, aws: sesClientModule },
  });

  return new Promise((resolve, reject) => {
    transporter.sendMail(
      {
        from,
        to,
        subject: "Hello World",
        text: "Greetings from Amazon SES!",
        attachments: [{ content: "Hello World!", filename: "hello.txt" }],
```

```
      },
      (err, info) => {
        if (err) {
          reject(err);
        } else {
          resolve(info);
        }
      },
    );
  });
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 SendRawEmail을 참조하십시오.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`SendTemplatedEmail`** 함께 사용

다음 코드 예시는 `SendTemplatedEmail`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.

- 이메일 ID 확인 및 메시지 전송

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Send an email using a template.
```

```
    /// </summary>
    /// <param name="sender">Address of the sender.</param>
    /// <param name="recipients">Addresses of the recipients.</param>
    /// <param name="templateName">Name of the email template.</param>
    /// <param name="templateDataObject">Data for the email template.</param>
    /// <returns>The messageId of the email.</returns>
    public async Task<string> SendTemplateEmailAsync(string sender, List<string>
recipients,
        string templateName, object templateDataObject)
    {
        var messageId = "";
        try
        {
            // Template data should be serialized JSON from either a class or a
dynamic object.
            var templateData = JsonSerializer.Serialize(templateDataObject);

            var response = await
_amazonSimpleEmailService.SendTemplatedEmailAsync(
                new SendTemplatedEmailRequest
                {
                    Source = sender,
                    Destination = new Destination
                    {
                        ToAddresses = recipients
                    },
                    Template = templateName,
                    TemplateData = templateData
                });
            messageId = response.MessageId;
        }
        catch (Exception ex)
        {
            Console.WriteLine("SendTemplateEmailAsync failed with exception: " +
ex.Message);
        }

        return messageId;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 SendTemplatedEmail을 참조하십시오.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Send a templated email to a list of recipients.
/*!
  \param recipients; Vector of recipient email addresses.
  \param templateName: The name of the template to use.
  \param templateData: Map of key-value pairs for replacing text in template.
  \param senderEmailAddress: Email address of sender. Ignored if empty string.
  \param ccAddresses: Vector of cc addresses. Ignored if empty.
  \param replyToAddress: Reply to email address. Ignored if empty string.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::sendTemplatedEmail(const Aws::Vector<Aws::String> &recipients,
                                     const Aws::String &templateName,
                                     const Aws::Map<Aws::String, Aws::String>
 &templateData,
                                     const Aws::String &senderEmailAddress,
                                     const Aws::Vector<Aws::String> &ccAddresses,
                                     const Aws::String &replyToAddress,
                                     const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::Destination destination;
    if (!ccAddresses.empty()) {
        destination.WithCcAddresses(ccAddresses);
    }
    if (!recipients.empty()) {
        destination.WithToAddresses(recipients);
    }

    Aws::SES::Model::SendTemplatedEmailRequest sendTemplatedEmailRequest;
    sendTemplatedEmailRequest.SetDestination(destination);
```

```
    sendTemplatedEmailRequest.SetTemplate(templateName);

    std::ostringstream templateDataStream;
    templateDataStream << "{";
    size_t dataCount = 0;
    for (auto &pair: templateData) {
        templateDataStream << "\"" << pair.first << "\":\"" << pair.second <<
 "\"";
        dataCount++;
        if (dataCount < templateData.size()) {
            templateDataStream << ",";
        }
    }
    templateDataStream << "}";

    sendTemplatedEmailRequest.SetTemplateData(templateDataStream.str());

    if (!senderEmailAddress.empty()) {
        sendTemplatedEmailRequest.SetSource(senderEmailAddress);
    }
    if (!replyToAddress.empty()) {
        sendTemplatedEmailRequest.AddReplyToAddresses(replyToAddress);
    }

    auto outcome = sesClient.SendTemplatedEmail(sendTemplatedEmailRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully sent templated message with ID "
                  << outcome.GetResult().GetMessageId()
                  << "." << std::endl;
    }
    else {
        std::cerr << "Error sending templated message. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 SendTemplatedEmail을 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sesv2.model.Destination;
import software.amazon.awssdk.services.sesv2.model.EmailContent;
import software.amazon.awssdk.services.sesv2.model.SendEmailRequest;
import software.amazon.awssdk.services.sesv2.model.SesV2Exception;
import software.amazon.awssdk.services.sesv2.SesV2Client;
import software.amazon.awssdk.services.sesv2.model.Template;

/**
 * Before running this AWS SDK for Java (v2) 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
 *
 * Also, make sure that you create a template. See the following documentation
 * topic:
 *
 * https://docs.aws.amazon.com/ses/latest/dg/send-personalized-email-api.html
 */

public class SendEmailTemplate {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <template> <sender> <recipient>\s

                Where:
                    template - The name of the email template.
```

```
                    sender - An email address that represents the sender.\s
                    recipient - An email address that represents the recipient.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String templateName = args[0];
        String sender = args[1];
        String recipient = args[2];
        Region region = Region.US_EAST_1;
        SesV2Client sesv2Client = SesV2Client.builder()
                .region(region)
                .build();

        send(sesv2Client, sender, recipient, templateName);
    }

    public static void send(SesV2Client client, String sender, String recipient,
String templateName) {
        Destination destination = Destination.builder()
                .toAddresses(recipient)
                .build();

        /*
         * Specify both name and favorite animal (favoriteanimal) in your code
when
         * defining the Template object.
         * If you don't specify all the variables in the template, Amazon SES
doesn't
         * send the email.
         */
        Template myTemplate = Template.builder()
                .templateName(templateName)
                .templateData("{\n" +
                        "  \"name\": \"Jason\"\n," +
                        "  \"favoriteanimal\": \"Cat\"\n" +
                        "}")
                .build();

        EmailContent emailContent = EmailContent.builder()
                .template(myTemplate)
```

```
                .build();

        SendEmailRequest emailRequest = SendEmailRequest.builder()
                .destination(destination)
                .content(emailContent)
                .fromEmailAddress(sender)
                .build();

        try {
            System.out.println("Attempting to send an email based on a template
 using the AWS SDK for Java (v2)...");
            client.sendEmail(emailRequest);
            System.out.println("email based on a template was sent");

        } catch (SesV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 SendTemplatedEmail을 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { SendTemplatedEmailCommand } from "@aws-sdk/client-ses";
import {
  getUniqueName,
  postfix,
} from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { sesClient } from "./libs/sesClient.js";

/**
```

```
 * Replace this with the name of an existing template.
 */
const TEMPLATE_NAME = getUniqueName("ReminderTemplate");

/**
 * Replace these with existing verified emails.
 */
const VERIFIED_EMAIL = postfix(getUniqueName("Bilbo"), "@example.com");

const USER = { firstName: "Bilbo", emailAddress: VERIFIED_EMAIL };

/**
 *
 * @param { { emailAddress: string, firstName: string } } user
 * @param { string } templateName - The name of an existing template in Amazon
 SES.
 * @returns { SendTemplatedEmailCommand }
 */
const createReminderEmailCommand = (user, templateName) => {
  return new SendTemplatedEmailCommand({
    /**
     * Here's an example of how a template would be replaced with user data:
     * Template: <h1>Hello {{contact.firstName}},</h1><p>Don't forget about the
 party gifts!</p>
     * Destination: <h1>Hello Bilbo,</h1><p>Don't forget about the party gifts!</
p>
     */
    Destination: { ToAddresses: [user.emailAddress] },
    TemplateData: JSON.stringify({ contact: { firstName: user.firstName } }),
    Source: VERIFIED_EMAIL,
    Template: templateName,
  });
};

const run = async () => {
  const sendReminderEmailCommand = createReminderEmailCommand(
    USER,
    TEMPLATE_NAME,
  );
  try {
    return await sesClient.send(sendReminderEmailCommand);
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MessageRejected") {
      /** @type { import('@aws-sdk/client-ses').MessageRejected} */
```

```
      const messageRejectedError = caught;
      return messageRejectedError;
    }
    throw caught;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 SendTemplatedEmail을 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesMailSender:
    """Encapsulates functions to send emails with Amazon SES."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client


    def send_templated_email(
        self, source, destination, template_name, template_data, reply_tos=None
    ):
        """
        Sends an email based on a template. A template contains replaceable tags
        each enclosed in two curly braces, such as {{name}}. The template data
 passed
        in this function contains key-value pairs that define the values to
 insert
        in place of the template tags.

        Note: If your account is in the Amazon SES  sandbox, the source and
```

```
        destination email accounts must both be verified.

        :param source: The source email account.
        :param destination: The destination email account.
        :param template_name: The name of a previously created template.
        :param template_data: JSON-formatted key-value pairs of replacement
values
                              that are inserted in the template before it is
sent.
        :return: The ID of the message, assigned by Amazon SES.
        """
        send_args = {
            "Source": source,
            "Destination": destination.to_service_format(),
            "Template": template_name,
            "TemplateData": json.dumps(template_data),
        }
        if reply_tos is not None:
            send_args["ReplyToAddresses"] = reply_tos
        try:
            response = self.ses_client.send_templated_email(**send_args)
            message_id = response["MessageId"]
            logger.info(
                "Sent templated mail %s from %s to %s.",
                message_id,
                source,
                destination.tos,
            )
        except ClientError:
            logger.exception(
                "Couldn't send templated mail from %s to %s.", source,
destination.tos
            )
            raise
        else:
            return message_id
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 SendTemplatedEmail를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **UpdateTemplate** 함께 사용

다음 코드 예시는 UpdateTemplate의 사용 방법을 보여 줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 이메일 ID 확인 및 메시지 전송

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Update an Amazon Simple Email Service (Amazon SES) template.
/*!
  \param templateName: The name of the template.
  \param htmlPart: The HTML body of the email.
  \param subjectPart: The subject line of the email.
  \param textPart: The plain text version of the email.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::updateTemplate(const Aws::String &templateName,
                                 const Aws::String &htmlPart,
                                 const Aws::String &subjectPart,
                                 const Aws::String &textPart,
                                 const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::Template templateValues;

    templateValues.SetTemplateName(templateName);
```

```
    templateValues.SetSubjectPart(subjectPart);
    templateValues.SetHtmlPart(htmlPart);
    templateValues.SetTextPart(textPart);

    Aws::SES::Model::UpdateTemplateRequest updateTemplateRequest;
    updateTemplateRequest.SetTemplate(templateValues);

    Aws::SES::Model::UpdateTemplateOutcome outcome =
 sesClient.UpdateTemplate(updateTemplateRequest);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully updated template." << std::endl;
    } else {
        std::cerr << "Error updating template. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ API 참조의 UpdateTemplate을 참조하십시오.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { UpdateTemplateCommand } from "@aws-sdk/client-ses";
import { getUniqueName } from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { sesClient } from "./libs/sesClient.js";

const TEMPLATE_NAME = getUniqueName("TemplateName");
const HTML_PART = "<h1>Hello, World!</h1>";

const createUpdateTemplateCommand = () => {
```

```
  return new UpdateTemplateCommand({
    Template: {
      TemplateName: TEMPLATE_NAME,
      HtmlPart: HTML_PART,
      SubjectPart: "Example",
      TextPart: "Updated template text.",
    },
  });
};

const run = async () => {
  const updateTemplateCommand = createUpdateTemplateCommand();

  try {
    return await sesClient.send(updateTemplateCommand);
  } catch (err) {
    console.log("Failed to update template.", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 UpdateTemplate을 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesTemplate:
    """Encapsulates Amazon SES template functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client
```

```
        self.template = None
        self.template_tags = set()

    def _extract_tags(self, subject, text, html):
        """
        Extracts tags from a template as a set of unique values.

        :param subject: The subject of the email.
        :param text: The text version of the email.
        :param html: The html version of the email.
        """
        self.template_tags = set(re.findall(TEMPLATE_REGEX, subject + text +
html))
        logger.info("Extracted template tags: %s", self.template_tags)


    def update_template(self, name, subject, text, html):
        """
        Updates a previously created email template.

        :param name: The name of the template.
        :param subject: The subject of the email.
        :param text: The plain text version of the email.
        :param html: The HTML version of the email.
        """
        try:
            template = {
                "TemplateName": name,
                "SubjectPart": subject,
                "TextPart": text,
                "HtmlPart": html,
            }
            self.ses_client.update_template(Template=template)
            logger.info("Updated template %s.", name)
            self.template = template
            self._extract_tags(subject, text, html)
        except ClientError:
            logger.exception("Couldn't update template %s.", name)
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 UpdateTemplate를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 **`VerifyDomainIdentity`** 함께 사용

다음 코드 예시는 `VerifyDomainIdentity`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.

- 리전 전체에서 이메일 및 도메인 ID 복사
- 이메일 ID 확인 및 메시지 전송

CLI

AWS CLI

Amazon SES를 사용하여 도메인 확인

다음 예시에서는 `verify-domain-identity` 명령을 사용하여 도메인을 확인합니다.

```
aws ses verify-domain-identity --domain example.com
```

출력:

```
{
   "VerificationToken": "eoEmxw+YaYhb3h3iVJHuXMJXqeu1q1/wwmvjuEXAMPLE"
}
```

도메인 확인을 완료하려면 반환된 확인 토큰이 포함된 TXT 레코드를 도메인의 DNS 설정에 추가해야 합니다. 자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES에서 도메인 확인을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 VerifyDomainIdentity를 참조하세요.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { VerifyDomainIdentityCommand } from "@aws-sdk/client-ses";
import {
  getUniqueName,
  postfix,
} from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { sesClient } from "./libs/sesClient.js";

/**
 * You must have access to the domain's DNS settings to complete the
 * domain verification process.
 */
const DOMAIN_NAME = postfix(getUniqueName("Domain"), ".example.com");

const createVerifyDomainIdentityCommand = () => {
  return new VerifyDomainIdentityCommand({ Domain: DOMAIN_NAME });
};

const run = async () => {
  const VerifyDomainIdentityCommand = createVerifyDomainIdentityCommand();

  try {
    return await sesClient.send(VerifyDomainIdentityCommand);
  } catch (err) {
    console.log("Failed to verify domain.", err);
    return err;
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 VerifyDomainIdentity를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesIdentity:
    """Encapsulates Amazon SES identity functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client


    def verify_domain_identity(self, domain_name):
        """
        Starts verification of a domain identity. To complete verification, you
 must
        create a TXT record with a specific format through your DNS provider.

        For more information, see *Verifying a domain with Amazon SES* in the
        Amazon SES documentation:
            https://docs.aws.amazon.com/ses/latest/DeveloperGuide/verify-domain-
procedure.html

        :param domain_name: The name of the domain to verify.
        :return: The token to include in the TXT record with your DNS provider.
        """
        try:
            response = self.ses_client.verify_domain_identity(Domain=domain_name)
            token = response["VerificationToken"]
            logger.info("Got domain verification token for %s.", domain_name)
        except ClientError:
            logger.exception("Couldn't verify domain %s.", domain_name)
            raise
        else:
```

```
            return token
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 VerifyDomainIdentity를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 **`VerifyEmailIdentity`** 함께 사용

다음 코드 예시는 `VerifyEmailIdentity`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.

- 리전 전체에서 이메일 및 도메인 ID 복사
- 이메일 ID 확인 및 메시지 전송

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Starts verification of an email identity. This request sends an email
    /// from Amazon SES to the specified email address. To complete
    /// verification, follow the instructions in the email.
    /// </summary>
    /// <param name="recipientEmailAddress">Email address to verify.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyEmailIdentityAsync(string
recipientEmailAddress)
```

```
    {
        var success = false;
        try
        {
            var response = await
 _amazonSimpleEmailService.VerifyEmailIdentityAsync(
                new VerifyEmailIdentityRequest
                {
                    EmailAddress = recipientEmailAddress
                });

            success = response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (Exception ex)
        {
            Console.WriteLine("VerifyEmailIdentityAsync failed with exception: "
 + ex.Message);
        }

        return success;
    }
```

- API 세부 정보는 AWS SDK for .NET SDK API 참조의 VerifyEmailIdentity를 참조하십시오.

C++

SDK for C++

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
//! Add an email address to the list of identities associated with this account
 and
//! initiate verification.
/*!
  \param emailAddress; The email address to add.
  \param clientConfiguration: AWS client configuration.
```

```
  \return bool: Function succeeded.
 */
bool AwsDoc::SES::verifyEmailIdentity(const Aws::String &emailAddress,
                          const Aws::Client::ClientConfiguration
 &clientConfiguration)
{
    Aws::SES::SESClient sesClient(clientConfiguration);

    Aws::SES::Model::VerifyEmailIdentityRequest verifyEmailIdentityRequest;

    verifyEmailIdentityRequest.SetEmailAddress(emailAddress);

    Aws::SES::Model::VerifyEmailIdentityOutcome outcome =
 sesClient.VerifyEmailIdentity(verifyEmailIdentityRequest);

    if (outcome.IsSuccess())
    {
        std::cout << "Email verification initiated." << std::endl;
    }

    else
    {
        std::cerr << "Error initiating email verification. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API 세부 정보는 AWS SDK for C++ SDK API 참조의 VerifyEmailIdentity를 참조하세요.

CLI

AWS CLI

Amazon SES로 이메일 주소 확인

다음 예시에서는 `verify-email-identity` 명령을 사용하여 이메일 주소를 확인합니다.

```
aws ses verify-email-identity --email-address user@example.com
```

Amazon SES를 사용하여 이메일을 보내려면 발신 이메일 주소 또는 도메인이 사용자 본인의 소유인지 확인해야 합니다. 프로덕션 액세스 권한이 아직 없는 경우Amazon SES 메일박스 시뮬레이터에서 제공하는 이메일 주소를 제외하고 이메일을 전송하는 모든 이메일 주소도 확인해야 합니다.

verify-email-identity가 직접적으로 호출되면 해당 이메일 주소에 확인 이메일이 전송됩니다. 확인 프로세스를 완료하려면 이메일에 포함된 링크를 클릭해야 합니다.

자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon SES에서 이메일 주소 확인을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 VerifyEmailIdentity를 참조하세요.

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
// Import required AWS SDK clients and commands for Node.js
import { VerifyEmailIdentityCommand } from "@aws-sdk/client-ses";
import { sesClient } from "./libs/sesClient.js";

const EMAIL_ADDRESS = "name@example.com";

const createVerifyEmailIdentityCommand = (emailAddress) => {
  return new VerifyEmailIdentityCommand({ EmailAddress: emailAddress });
};

const run = async () => {
  const verifyEmailIdentityCommand =
    createVerifyEmailIdentityCommand(EMAIL_ADDRESS);
  try {
    return await sesClient.send(verifyEmailIdentityCommand);
  } catch (err) {
    console.log("Failed to verify email identity.", err);
    return err;
  }
```

```
};
```

- API 세부 정보는 AWS SDK for JavaScript SDK API 참조의 VerifyEmailIdentity를 참조하십시오.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SesIdentity:
    """Encapsulates Amazon SES identity functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client


    def verify_email_identity(self, email_address):
        """
        Starts verification of an email identity. This function causes an email
        to be sent to the specified email address from Amazon SES. To complete
        verification, follow the instructions in the email.

        :param email_address: The email address to verify.
        """
        try:
            self.ses_client.verify_email_identity(EmailAddress=email_address)
            logger.info("Started verification of %s.", email_address)
        except ClientError:
            logger.exception("Couldn't start verification of %s.", email_address)
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 VerifyEmailIdentity를 참조하세요.

Ruby

SDK for Ruby

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require 'aws-sdk-ses' # v2: require 'aws-sdk'

# Replace recipient@example.com with a "To" address.
recipient = 'recipient@example.com'

# Create a new SES resource in the us-west-2 region.
# Replace us-west-2 with the AWS Region you're using for Amazon SES.
ses = Aws::SES::Client.new(region: 'us-west-2')

# Try to verify email address.
begin
  ses.verify_email_identity({
                              email_address: recipient
                            })

  puts "Email sent to #{recipient}"

# If something goes wrong, display an error message.
rescue Aws::SES::Errors::ServiceError => e
  puts "Email not sent. Error message: #{e}"
end
```

- API 세부 정보는 AWS SDK for Ruby SDK API 참조의 VerifyEmailIdentity를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDKs를 사용한 Amazon SES 시나리오

다음 코드 예제에서는 Amazon SES AWS SDKs에서 일반적인 시나리오를 구현하는 방법을 보여줍니다. 이러한 시나리오에서는 Amazon SES 내에서 또는 다른 AWS 서비스와 결합된 상태에서 여러 함수를 직접적으로 호출하여 특정 태스크를 수행하는 방법을 보여줍니다. 각 시나리오에는 전체 소스 코드에 대한 링크가 포함되어 있습니다. 여기에서 코드를 설정하고 실행하는 방법에 대한 지침을 찾을 수 있습니다.

시나리오는 컨텍스트에 맞는 서비스 작업을 이해하는 데 도움이 되도록 중급 수준의 경험을 대상으로 합니다.

예시

- Amazon Transcribe 스트리밍 앱 구축
- AWS SDK를 사용하여 한 AWS 리전에서 다른 리전으로 Amazon SES 이메일 및 도메인 자격 증명 복사
- DynamoDB 데이터를 추적하는 웹 애플리케이션 만들기
- Amazon Redshift 항목 추적기 만들기
- Aurora 서버리스 작업 항목 트래커 만들기
- AWS SDK를 사용하여 Amazon Rekognition으로 이미지에서 PPE 감지
- AWS SDK를 사용하여 Amazon Rekognition으로 이미지의 객체 감지
- AWS SDK를 사용하여 Amazon Rekognition을 사용하여 비디오에서 사람 및 객체 감지
- Amazon SES SMTP 엔드포인트에 연결하는 자격 증명 생성
- Step Functions를 사용하여 Lambda 함수 간접 호출
- AWS SDK를 사용하여 Amazon SES로 이메일 자격 증명 확인 및 메시지 전송

Amazon Transcribe 스트리밍 앱 구축

다음 코드 예시는 라이브 오디오를 실시간으로 기록, 변환 및 번역하고 결과를 이메일로 보내는 앱을 구축하는 방법을 보여줍니다.

JavaScript

SDK for JavaScript (v3)

Amazon Transcribe를 사용하여 라이브 오디오를 실시간으로 기록, 변환 및 번역하고 Amazon Simple Email Service(Amazon SES)를 사용하여 결과를 이메일로 전송하는 앱을 구축하는 방법을 보여줍니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예시를 참조하세요.

이 예시에서 사용되는 서비스

- Amazon Comprehend
- Amazon SES
- Amazon Transcribe
- Amazon Translate

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK를 사용하여 한 AWS 리전에서 다른 리전으로 Amazon SES 이메일 및 도메인 자격 증명 복사

다음 코드 예제에서는 Amazon SES 이메일 및 도메인 자격 증명을 한 AWS 리전에서 다른 리전으로 복사하는 방법을 보여줍니다. Route 53에서 도메인 ID를 관리하는 경우 확인 레코드가 대상 리전의 도메인에 복사됩니다.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import argparse
import json
import logging
```

```
from pprint import pprint
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def get_identities(ses_client):
    """
    Gets the identities for the current Region. The Region is specified in the
    Boto3 Amazon SES client object.

    :param ses_client: A Boto3 Amazon SES client.
    :return: The list of email identities and the list of domain identities.
    """
    email_identities = []
    domain_identities = []
    try:
        identity_paginator = ses_client.get_paginator("list_identities")
        identity_iterator = identity_paginator.paginate(
            PaginationConfig={"PageSize": 20}
        )
        for identity_page in identity_iterator:
            for identity in identity_page["Identities"]:
                if "@" in identity:
                    email_identities.append(identity)
                else:
                    domain_identities.append(identity)
        logger.info(
            "Found %s email and %s domain identities.",
            len(email_identities),
            len(domain_identities),
        )
    except ClientError:
        logger.exception("Couldn't get identities.")
        raise
    else:
        return email_identities, domain_identities


def verify_emails(email_list, ses_client):
    """
    Starts verification of a list of email addresses. Verification causes an
 email
```

```
    to be sent to each address. To complete verification, the recipient must
 follow
    the instructions in the email.

    :param email_list: The list of email addresses to verify.
    :param ses_client: A Boto3 Amazon SES client.
    :return: The list of emails that were successfully submitted for
 verification.
    """
    verified_emails = []
    for email in email_list:
        try:
            ses_client.verify_email_identity(EmailAddress=email)
            verified_emails.append(email)
            logger.info("Started verification of %s.", email)
        except ClientError:
            logger.warning("Couldn't start verification of %s.", email)
    return verified_emails


def verify_domains(domain_list, ses_client):
    """
    Starts verification for a list of domain identities. This returns a token for
    each domain, which must be registered as a TXT record with the DNS provider
 for
    the domain.

    :param domain_list: The list of domains to verify.
    :param ses_client: A Boto3 Amazon SES client.
    :return: The generated domain tokens to use to completed verification.
    """
    domain_tokens = {}
    for domain in domain_list:
        try:
            response = ses_client.verify_domain_identity(Domain=domain)
            token = response["VerificationToken"]
            domain_tokens[domain] = token
            logger.info("Got verification token %s for domain %s.", token,
 domain)
        except ClientError:
            logger.warning("Couldn't get verification token for domain %s.",
 domain)
    return domain_tokens
```

```
def get_hosted_zones(route53_client):
    """
    Gets the Amazon Route 53 hosted zones for the current account.

    :param route53_client: A Boto3 Route 53 client.
    :return: The list of hosted zones.
    """
    zones = []
    try:
        zone_paginator = route53_client.get_paginator("list_hosted_zones")
        zone_iterator = zone_paginator.paginate(PaginationConfig={"PageSize":
 20})
        zones = [
            zone for zone_page in zone_iterator for zone in
 zone_page["HostedZones"]
        ]
        logger.info("Found %s hosted zones.", len(zones))
    except ClientError:
        logger.warning("Couldn't get hosted zones.")
    return zones


def find_domain_zone_matches(domains, zones):
    """
    Finds matches between Amazon SES verified domains and Route 53 hosted zones.
    Subdomain matches are taken when found, otherwise root domain matches are
 taken.

    :param domains: The list of domains to match.
    :param zones: The list of hosted zones to match.
    :return: The set of matched domain-zone pairs. When a match is not found, the
             domain is included in the set with a zone value of None.
    """
    domain_zones = {}
    for domain in domains:
        domain_zones[domain] = None
        # Start at the most specific sub-domain and walk up to the root domain
 until a
        # zone match is found.
        domain_split = domain.split(".")
        for index in range(0, len(domain_split) - 1):
            sub_domain = ".".join(domain_split[index:])
            for zone in zones:
```

```
                # Normalize the zone name from Route 53 by removing the trailing
 '.'.
                zone_name = zone["Name"][:-1]
                if sub_domain == zone_name:
                    domain_zones[domain] = zone
                    break
            if domain_zones[domain] is not None:
                break
    return domain_zones


def add_route53_verification_record(domain, token, zone, route53_client):
    """
    Adds a domain verification TXT record to the specified Route 53 hosted zone.
    When a TXT record already exists in the hosted zone for the specified domain,
    the existing values are preserved and the new token is added to the list.

    :param domain: The domain to add.
    :param token: The verification token for the domain.
    :param zone: The hosted zone where the domain verification record is added.
    :param route53_client: A Boto3 Route 53 client.
    """
    domain_token_record_set_name = f"_amazonses.{domain}"
    record_set_paginator =
 route53_client.get_paginator("list_resource_record_sets")
    record_set_iterator = record_set_paginator.paginate(
        HostedZoneId=zone["Id"], PaginationConfig={"PageSize": 20}
    )
    records = []
    for record_set_page in record_set_iterator:
        try:
            txt_record_set = next(
                record_set
                for record_set in record_set_page["ResourceRecordSets"]
                if record_set["Name"][:-1] == domain_token_record_set_name
                and record_set["Type"] == "TXT"
            )
            records = txt_record_set["ResourceRecords"]
            logger.info(
                "Existing TXT record found in set %s for zone %s.",
                domain_token_record_set_name,
                zone["Name"],
            )
            break
```

```
        except StopIteration:
            pass
    records.append({"Value": json.dumps(token)})
    changes = [
        {
            "Action": "UPSERT",
            "ResourceRecordSet": {
                "Name": domain_token_record_set_name,
                "Type": "TXT",
                "TTL": 1800,
                "ResourceRecords": records,
            },
        }
    ]
    try:
        route53_client.change_resource_record_sets(
            HostedZoneId=zone["Id"], ChangeBatch={"Changes": changes}
        )
        logger.info(
            "Created or updated the TXT record in set %s for zone %s.",
            domain_token_record_set_name,
            zone["Name"],
        )
    except ClientError as err:
        logger.warning(
            "Got error %s. Couldn't create or update the TXT record for zone
 %s.",
            err.response["Error"]["Code"],
            zone["Name"],
        )


def generate_dkim_tokens(domain, ses_client):
    """
    Generates DKIM tokens for a domain. These must be added as CNAME records to
 the
    DNS provider for the domain.

    :param domain: The domain to generate tokens for.
    :param ses_client: A Boto3 Amazon SES client.
    :return: The list of generated DKIM tokens.
    """
    dkim_tokens = []
    try:
```

```
        dkim_tokens = ses_client.verify_domain_dkim(Domain=domain)["DkimTokens"]
        logger.info("Generated %s DKIM tokens for domain %s.", len(dkim_tokens),
 domain)
    except ClientError:
        logger.warning("Couldn't generate DKIM tokens for domain %s.", domain)
    return dkim_tokens


def add_dkim_domain_tokens(hosted_zone, domain, tokens, route53_client):
    """
    Adds DKIM domain token CNAME records to a Route 53 hosted zone.

    :param hosted_zone: The hosted zone where the records are added.
    :param domain: The domain to add.
    :param tokens: The DKIM tokens for the domain to add.
    :param route53_client: A Boto3 Route 53 client.
    """
    try:
        changes = [
            {
                "Action": "UPSERT",
                "ResourceRecordSet": {
                    "Name": f"{token}._domainkey.{domain}",
                    "Type": "CNAME",
                    "TTL": 1800,
                    "ResourceRecords": [{"Value":
 f"{token}.dkim.amazonses.com"}],
                },
            }
            for token in tokens
        ]
        route53_client.change_resource_record_sets(
            HostedZoneId=hosted_zone["Id"], ChangeBatch={"Changes": changes}
        )
        logger.info(
            "Added %s DKIM CNAME records to %s in zone %s.",
            len(tokens),
            domain,
            hosted_zone["Name"],
        )
    except ClientError:
        logger.warning(
            "Couldn't add DKIM CNAME records for %s to zone %s.",
            domain,
```

```
            hosted_zone["Name"],
        )


def configure_sns_topics(identity, topics, ses_client):
    """
    Configures Amazon Simple Notification Service (Amazon SNS) notifications for
    an identity. The Amazon SNS topics must already exist.

    :param identity: The identity to configure.
    :param topics: The list of topics to configure. The choices are Bounce,
 Delivery,
                   or Complaint.
    :param ses_client: A Boto3 Amazon SES client.
    """
    for topic in topics:
        topic_arn = input(
            f"Enter the Amazon Resource Name (ARN) of the {topic} topic or press
 "
            f"Enter to skip: "
        )
        if topic_arn != "":
            try:
                ses_client.set_identity_notification_topic(
                    Identity=identity, NotificationType=topic, SnsTopic=topic_arn
                )
                logger.info("Configured %s for %s notifications.", identity,
 topic)
            except ClientError:
                logger.warning(
                    "Couldn't configure %s for %s notifications.", identity,
 topic
                )


def replicate(source_client, destination_client, route53_client):
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print(
        f"Replicating Amazon SES identities and other configuration from "
        f"{source_client.meta.region_name} to
 {destination_client.meta.region_name}."
    )
```

```
    print("-" * 88)

    print(f"Retrieving identities from {source_client.meta.region_name}.")
    source_emails, source_domains = get_identities(source_client)
    print("Email addresses found:")
    print(*source_emails)
    print("Domains found:")
    print(*source_domains)

    print("Starting verification for email identities.")
    dest_emails = verify_emails(source_emails, destination_client)
    print("Getting domain tokens for domain identities.")
    dest_domain_tokens = verify_domains(source_domains, destination_client)

    # Get Route 53 hosted zones and match them with Amazon SES domains.
    answer = input(
        "Is the DNS configuration for your domains managed by Amazon Route 53 (y/
n)? "
    )
    use_route53 = answer.lower() == "y"
    hosted_zones = get_hosted_zones(route53_client) if use_route53 else []
    if use_route53:
        print("Adding or updating Route 53 TXT records for your domains.")
        domain_zones = find_domain_zone_matches(dest_domain_tokens.keys(),
 hosted_zones)
        for domain in domain_zones:
            add_route53_verification_record(
                domain, dest_domain_tokens[domain], domain_zones[domain],
 route53_client
            )
    else:
        print(
            "Use these verification tokens to create TXT records through your DNS
 "
            "provider:"
        )
        pprint(dest_domain_tokens)

    answer = input("Do you want to configure DKIM signing for your identities (y/
n)? ")
    if answer.lower() == "y":
        # Build a set of unique domains from email and domain identities.
        domains = {email.split("@")[1] for email in dest_emails}
        domains.update(dest_domain_tokens)
```

```
        domain_zones = find_domain_zone_matches(domains, hosted_zones)
        for domain, zone in domain_zones.items():
            answer = input(
                f"Do you want to configure DKIM signing for {domain} (y/n)? "
            )
            if answer.lower() == "y":
                dkim_tokens = generate_dkim_tokens(domain, destination_client)
                if use_route53 and zone is not None:
                    add_dkim_domain_tokens(zone, domain, dkim_tokens,
 route53_client)
                else:
                    print(
                        "Add the following DKIM tokens as CNAME records through
 your "
                        "DNS provider:"
                    )
                    print(*dkim_tokens, sep="\n")

    answer = input(
        "Do you want to configure Amazon SNS notifications for your identities
 (y/n)? "
    )
    if answer.lower() == "y":
        for identity in dest_emails + list(dest_domain_tokens.keys()):
            answer = input(
                f"Do you want to configure Amazon SNS topics for {identity} (y/
n)? "
            )
            if answer.lower() == "y":
                configure_sns_topics(
                    identity, ["Bounce", "Delivery", "Complaint"],
 destination_client
                )

    print(f"Replication complete for {destination_client.meta.region_name}.")
    print("-" * 88)


def main():
    boto3_session = boto3.Session()
    ses_regions = boto3_session.get_available_regions("ses")
    parser = argparse.ArgumentParser(
        description="Copies email address and domain identities from one AWS
 Region to "
```

```
        "another. Optionally adds records for domain verification and DKIM "
        "signing to domains that are managed by Amazon Route 53, "
        "and sets up Amazon SNS notifications for events of interest."
    )
    parser.add_argument(
        "source_region", choices=ses_regions, help="The region to copy from."
    )
    parser.add_argument(
        "destination_region", choices=ses_regions, help="The region to copy to."
    )
    args = parser.parse_args()
    source_client = boto3.client("ses", region_name=args.source_region)
    destination_client = boto3.client("ses", region_name=args.destination_region)
    route53_client = boto3.client("route53")
    replicate(source_client, destination_client, route53_client)


if __name__ == "__main__":
    main()
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 다음 주제를 참조하십시오.
  - ListIdentities
  - SetIdentityNotificationTopic
  - VerifyDomainDkim
  - VerifyDomainIdentity
  - VerifyEmailIdentity

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

DynamoDB 데이터를 추적하는 웹 애플리케이션 만들기

다음 코드 예제는 Amazon DynamoDB 테이블에서 작업 항목을 추적하고 Amazon Simple Email Service(Amazon SES)를 사용하여 보고서를 보내는 웹 애플리케이션을 만드는 방법을 보여 줍니다.

.NET

SDK for .NET

Amazon DynamoDB .NET API를 사용하여 DynamoDB 작업 데이터를 추적하는 동적 웹 애플리케이션을 만드는 방법을 보여줍니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하세요.

이 예제에서 사용되는 서비스

- DynamoDB
- Amazon SES

Java

SDK for Java 2.x

Amazon DynamoDB API를 사용하여 DynamoDB 작업 데이터를 추적하는 동적 웹 애플리케이션을 만드는 방법을 보여줍니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하세요.

이 예제에서 사용되는 서비스

- DynamoDB
- Amazon SES

Kotlin

SDK for Kotlin

Amazon DynamoDB API를 사용하여 DynamoDB 작업 데이터를 추적하는 동적 웹 애플리케이션을 만드는 방법을 보여줍니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하세요.

이 예제에서 사용되는 서비스

- DynamoDB
- Amazon SES

Python

SDK for Python(Boto3)

AWS SDK for Python (Boto3) 를 사용하여 Amazon DynamoDB의 작업 항목을 추적하고 Amazon Simple Email Service(Amazon SES)를 사용하여 보고서를 이메일로 보내는 REST 서비스를 생성하는 방법을 보여줍니다. 이 예제는 Flask 웹 프레임워크를 사용하여 HTTP 라우팅을 처리하고 React 웹 페이지와 통합하여 완전한 기능을 갖춘 웹 애플리케이션을 제공합니다.

- 와 통합되는 Flask REST 서비스를 빌드합니다 AWS 서비스.
- DynamoDB 테이블에 저장된 작업 항목을 읽고, 쓰고, 업데이트합니다.
- Amazon SES를 사용하여 작업 항목에 대한 이메일 보고서를 보냅니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub의 AWS 코드 예제 리포지토리에서 전체 예제를 참조하세요.

이 예제에서 사용되는 서비스

- DynamoDB
- Amazon SES

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

Amazon Redshift 항목 추적기 만들기

다음 코드 예제에서는 Amazon Redshift 데이터베이스를 사용하는 작업 항목을 추적하고 보고하는 웹 애플리케이션을 만드는 방법을 보여줍니다.

Java

SDK for Java 2.x

Amazon Redshift 데이터베이스에 저장된 작업 항목을 추적하고 보고하는 웹 애플리케이션을 만드는 방법을 보여줍니다.

Amazon Redshift 데이터를 쿼리하고 React 애플리케이션에서 사용하도록 Spring REST API를 설정하는 방법에 대한 지침과 전체 소스 코드는 GitHub에서 전체 예제를 참조하세요.

이 예시에서 사용되는 서비스

- Amazon Redshift

- Amazon SES

Kotlin

SDK for Kotlin

Amazon Redshift 데이터베이스에 저장된 작업 항목을 추적하고 보고하는 웹 애플리케이션을 만드는 방법을 보여줍니다.

Amazon Redshift 데이터를 쿼리하고 React 애플리케이션에서 사용하도록 Spring REST API를 설정하는 방법에 대한 지침과 전체 소스 코드는 GitHub에서 전체 예제를 참조하세요.

이 예시에서 사용되는 서비스

- Amazon Redshift
- Amazon SES

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

Aurora 서버리스 작업 항목 트래커 만들기

다음 코드 예제는 Amazon Aurora Serverless 데이터베이스에서 작업 항목을 추적하고 Amazon Simple Email Service(Amazon SES)를 사용하여 보고서를 보내는 웹 애플리케이션을 만드는 방법을 보여 줍니다.

.NET

SDK for .NET

AWS SDK for .NET 를 사용하여 Amazon Aurora 데이터베이스의 작업 항목을 추적하고 Amazon Simple Email Service(Amazon SES)를 사용하여 보고서를 이메일로 보내는 웹 애플리케이션을 생성하는 방법을 보여줍니다. 이 예제에서는 RESTful .NET 백엔드와의 상호 작용을 위해 React.js로 빌드된 프런트엔드를 사용합니다.

- React 웹 애플리케이션을 AWS 서비스와 통합합니다.
- Aurora 테이블의 항목을 나열, 추가, 업데이트 및 삭제합니다.
- Amazon SES를 사용하여 필터링된 작업 항목에 대한 이메일 보고서를 보냅니다.
- 포함된 AWS CloudFormation 스크립트를 사용하여 예제 리소스를 배포하고 관리합니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하세요.

이 예시에서 사용되는 서비스

- Aurora
- Amazon RDS
- Amazon RDS 데이터 서비스
- Amazon SES

C++

SDK for C++

Amazon Aurora Serverless 데이터베이스에 저장된 작업 항목을 추적하고 보고하는 웹 애플리케이션을 만드는 방법을 보여줍니다.

Amazon Aurora Serverless 데이터를 쿼리하고 React 애플리케이션에서 사용하도록 C++ REST API를 설정하는 방법에 대한 지침과 전체 소스 코드는 GitHub에서 전체 예제를 참조하세요.

이 예시에서 사용되는 서비스

- Aurora
- Amazon RDS
- Amazon RDS 데이터 서비스
- Amazon SES

Java

SDK for Java 2.x

Amazon RDS 데이터베이스에 저장된 작업 항목을 추적하고 보고하는 웹 애플리케이션을 만드는 방법을 보여줍니다.

Amazon Aurora Serverless 데이터를 쿼리하고 React 애플리케이션에서 사용하도록 Spring REST API를 설정하는 방법에 대한 지침과 전체 소스 코드는 GitHub에서 전체 예제를 참조하십시오.

JDBC API를 사용한 예제를 설정하고 실행하는 방법에 대한 전체 소스 코드와 지침은 GitHub에서 전체 예제를 참조하세요.

이 예시에서 사용되는 서비스

- Aurora
- Amazon RDS
- Amazon RDS 데이터 서비스
- Amazon SES

JavaScript

SDK for JavaScript (v3)

AWS SDK for JavaScript (v3)를 사용하여 Amazon Aurora 데이터베이스의 작업 항목을 추적하고 Amazon Simple Email Service(Amazon SES)를 사용하여 보고서를 이메일로 보내는 웹 애플리케이션을 생성하는 방법을 보여줍니다. 이 예제에서는 Express Node.js 백엔드와의 상호 작용을 위해 React.js로 빌드된 프런트엔드를 사용합니다.

- React.js 웹 애플리케이션을와 통합합니다 AWS 서비스.
- Aurora 테이블의 항목을 나열, 추가 및 업데이트합니다.
- Amazon SES를 사용하여 필터링된 작업 항목에 대한 이메일 보고서를 보냅니다.
- 포함된 AWS CloudFormation 스크립트를 사용하여 예제 리소스를 배포하고 관리합니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하세요.

이 예시에서 사용되는 서비스

- Aurora
- Amazon RDS
- Amazon RDS 데이터 서비스
- Amazon SES

Kotlin

SDK for Kotlin

Amazon RDS 데이터베이스에 저장된 작업 항목을 추적하고 보고하는 웹 애플리케이션을 만드는 방법을 보여줍니다.

Amazon Aurora Serverless 데이터를 쿼리하고 React 애플리케이션에서 사용하도록 Spring REST API를 설정하는 방법에 대한 지침과 전체 소스 코드는 GitHub에서 전체 예제를 참조하세요.

이 예시에서 사용되는 서비스

- Aurora
- Amazon RDS
- Amazon RDS 데이터 서비스
- Amazon SES

PHP

SDK for PHP

AWS SDK for PHP 를 사용하여 Amazon RDS 데이터베이스의 작업 항목을 추적하고 Amazon Simple Email Service(Amazon SES)를 사용하여 보고서를 이메일로 보내는 웹 애플리케이션을 생성하는 방법을 보여줍니다. 이 예제에서는 RESTful PHP 백엔드와의 상호 작용을 위해 React.js로 빌드된 프런트엔드를 사용합니다.

- React.js 웹 애플리케이션을 AWS 서비스와 통합합니다.
- Amazon RDS 테이블의 항목을 나열, 추가, 업데이트 및 삭제합니다.
- Amazon SES를 사용하여 필터링된 작업 항목에 대한 이메일 보고서를 보냅니다.
- 포함된 AWS CloudFormation 스크립트를 사용하여 예제 리소스를 배포하고 관리합니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하세요.

이 예시에서 사용되는 서비스

- Aurora
- Amazon RDS
- Amazon RDS 데이터 서비스
- Amazon SES

Python

SDK for Python(Boto3)

AWS SDK for Python (Boto3) 를 사용하여 Amazon Aurora Serverless 데이터베이스의 작업 항목을 추적하고 Amazon Simple Email Service(Amazon SES)를 사용하여 보고서를 이메일로 보내는 REST 서비스를 생성하는 방법을 보여줍니다. 이 예제는 Flask 웹 프레임워크를 사용하여 HTTP 라우팅을 처리하고 React 웹 페이지와 통합하여 완전한 기능을 갖춘 웹 애플리케이션을 제공합니다.

- 와 통합되는 Flask REST 서비스를 빌드합니다 AWS 서비스.
- Aurora Serverless 데이터베이스에 저장된 작업 항목을 읽고, 쓰고, 업데이트합니다.
- 데이터베이스 자격 증명이 포함된 AWS Secrets Manager 보안 암호를 생성하고 이를 사용하여 데이터베이스에 대한 호출을 인증합니다.
- Amazon SES를 사용하여 작업 항목에 대한 이메일 보고서를 보냅니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하십시오.

이 예시에서 사용되는 서비스

- Aurora
- Amazon RDS
- Amazon RDS 데이터 서비스
- Amazon SES

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK를 사용하여 Amazon Rekognition으로 이미지에서 PPE 감지

다음 코드 예제에서는 Amazon Rekognition을 사용하여 이미지에서 개인 보호 장비(PPE)를 감지하는 앱을 구축하는 방법을 보여줍니다.

Java

SDK for Java 2.x

개인 보호 장비로 이미지를 감지하는 AWS Lambda 함수를 생성하는 방법을 보여줍니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하세요.

이 예제에서 사용되는 서비스

- DynamoDB
- Amazon Rekognition
- Amazon S3
- Amazon SES

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK를 사용하여 Amazon Rekognition으로 이미지의 객체 감지

다음 코드 예제는 Amazon Rekognition을 사용하여 이미지에서 범주별 객체를 감지하는 앱을 구축하는 방법을 보여줍니다.

.NET

SDK for .NET

Amazon Rekognition .NET을 사용하여 Amazon Simple Storage Service(Amazon S3) 버킷에 있는 이미지에서 범주별로 객체를 식별하기 위해 Amazon Rekognition을 사용하여 앱을 만드는 방법을 보여줍니다. 이 앱은 Amazon Simple Email Service(Amazon SES)를 사용하여 결과와 함께 이메일 알림을 관리자에게 보냅니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하십시오.

이 예제에서 사용되는 서비스

- Amazon Rekognition
- Amazon S3
- Amazon SES

Java

SDK for Java 2.x

Amazon Rekognition을 사용하여 Amazon Simple Storage Service (Amazon S3) 버킷에 있는 이미지에서 범주별로 객체를 식별하기 위해 Amazon Rekognition을 사용하여 앱을 만드는 방법을 보여줍니다. 이 앱은 Amazon Simple Email Service(Amazon SES)를 사용하여 결과와 함께 이메일 알림을 관리자에게 보냅니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하십시오.

이 예제에서 사용되는 서비스

- Amazon Rekognition
- Amazon S3
- Amazon SES

JavaScript

SDK for JavaScript (v3)

에서 Amazon Rekognition AWS SDK for JavaScript 을 사용하여 Amazon Rekognition을 사용하여 Amazon Simple Storage Service(Amazon S3) 버킷에 있는 이미지에서 범주별로 객체를 식별하는 앱을 생성하는 방법을 보여줍니다. 이 앱은 Amazon Simple Email Service(Amazon SES)를 사용하여 결과와 함께 이메일 알림을 관리자에게 보냅니다.

다음 작업을 수행하는 방법에 대해 알아보세요.

- Amazon Cognito를 사용하여 인증되지 않은 사용자를 만듭니다.
- Amazon Rekognition을 사용하여 객체용 이미지를 분석합니다.
- Amazon SES 이메일 주소를 확인합니다.
- Amazon SES를 사용하여 이메일 알림을 전송합니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하십시오.

이 예제에서 사용되는 서비스

- Amazon Rekognition
- Amazon S3
- Amazon SES

Kotlin

SDK for Kotlin

Amazon Rekognition Kotlin API를 사용하여 Amazon Simple Storage Service (Amazon S3) 버킷에 있는 이미지에서 범주별로 객체를 식별하기 위해 Amazon Rekognition을 사용하여 앱을 만드는 방법을 보여줍니다. 이 앱은 Amazon Simple Email Service(Amazon SES)를 사용하여 결과와 함께 이메일 알림을 관리자에게 보냅니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하십시오.

이 예제에서 사용되는 서비스

- Amazon Rekognition
- Amazon S3
- Amazon SES

Python

SDK for Python(Boto3)

를 사용하여 다음을 수행할 수 있는 웹 애플리케이션을 AWS SDK for Python (Boto3) 생성하는 방법을 보여줍니다.

- 사진을 Amazon Simple Storage Service (Amazon S3) 버킷에 업로드합니다.
- Amazon Rekognition을 사용하여 사진을 분석하고 레이블을 지정합니다.
- Amazon Simple Email Service(Amazon SES)를 사용하여 이미지 분석에 대한 이메일 보고서를 보냅니다.

이 예제에는 두 가지 주요 구성 요소가 포함되어 있습니다. 바로 JavaScript로 작성되고 React로 빌드된 웹 페이지와 Python으로 작성되고 Flask-RESTful로 빌드된 REST 서비스입니다.

React 웹 페이지를 사용하여 다음을 수행할 수 있습니다.

- S3 버킷에 저장된 이미지 목록을 표시합니다.
- 컴퓨터에서 S3 버킷에 이미지를 업로드합니다.
- 이미지에서 감지된 항목을 식별하는 이미지와 레이블을 표시합니다.
- S3 버킷의 모든 이미지에 대한 보고서를 받고 보고서의 이메일을 보냅니다.

웹 페이지가 REST 서비스를 호출합니다. 서비스가 다음 작업을 수행하기 위해 AWS 에 요청을 전송합니다.

- S3 버킷의 이미지 목록을 가져오고 필터링합니다.
- S3 버킷에 사진을 업로드합니다.
- Amazon Rekognition을 사용하여 개별 사진을 분석하고 사진에서 감지된 항목을 식별하는 레이블 목록을 가져옵니다.
- S3 버킷의 모든 사진을 분석하고 Amazon SES를 사용하여 보고서를 이메일로 보냅니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하십시오.

이 예제에서 사용되는 서비스

- Amazon Rekognition
- Amazon S3
- Amazon SES

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK를 사용하여 Amazon Rekognition을 사용하여 비디오에서 사람 및 객체 감지

다음 코드 예제에서는 Amazon Rekognition을 사용하여 동영상에서 사람과 객체를 감지하는 방법을 보여줍니다.

Java

SDK for Java 2.x

Amazon Rekognition Java API를 사용하여 Amazon Simple Storage Service (Amazon S3) 버킷에 있는 동영상에서 얼굴과 객체를 감지하기 위한 앱을 만드는 방법을 보여줍니다. 이 앱은 Amazon Simple Email Service(Amazon SES)를 사용하여 결과와 함께 이메일 알림을 관리자에게 보냅니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하십시오.

이 예제에서 사용되는 서비스

- Amazon Rekognition
- Amazon S3
- Amazon SES
- Amazon SNS
- Amazon SQS

Python

SDK for Python(Boto3)

Amazon Rekognition을 사용하여 비동기식 감지 작업을 시작해 동영상의 얼굴, 객체 및 사람을 감지할 수 있습니다. 또한 이 예제에서는 작업이 완료되고 주제에 대한 Amazon Simple Queue Service(Amazon SQS) 대기열을 구독할 때 Amazon Simple Notification Service(Amazon SNS)

주제를 알리도록 Amazon Rekognition을 구성합니다. 대기열이 작업에 대한 메시지를 받으면 작업이 검색되고 결과가 출력됩니다.

이 예제는 GitHub에서 가장 잘 볼 수 있습니다. 전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하세요.

이 예제에서 사용되는 서비스

- Amazon Rekognition
- Amazon S3
- Amazon SES
- Amazon SNS
- Amazon SQS

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

Amazon SES SMTP 엔드포인트에 연결하는 자격 증명 생성

다음 코드 예제에서는 Amazon SES SMTP 엔드포인트에 연결하는 자격 증명을 생성하는 방법을 보여줍니다.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
#!/usr/bin/env python3

import hmac
import hashlib
import base64
import argparse

SMTP_REGIONS = [
```

```
    "us-east-2",  # US East (Ohio)
    "us-east-1",  # US East (N. Virginia)
    "us-west-2",  # US West (Oregon)
    "ap-south-1",  # Asia Pacific (Mumbai)
    "ap-northeast-2",  # Asia Pacific (Seoul)
    "ap-southeast-1",  # Asia Pacific (Singapore)
    "ap-southeast-2",  # Asia Pacific (Sydney)
    "ap-northeast-1",  # Asia Pacific (Tokyo)
    "ca-central-1",  # Canada (Central)
    "eu-central-1",  # Europe (Frankfurt)
    "eu-west-1",  # Europe (Ireland)
    "eu-west-2",  # Europe (London)
    "eu-south-1",  # Europe (Milan)
    "eu-north-1",  # Europe (Stockholm)
    "sa-east-1",  # South America (Sao Paulo)
    "us-gov-west-1",  # AWS GovCloud (US)
    "us-gov-east-1",  # AWS GovCloud (US)
]

# These values are required to calculate the signature. Do not change them.
DATE = "11111111"
SERVICE = "ses"
MESSAGE = "SendRawEmail"
TERMINAL = "aws4_request"
VERSION = 0x04


def sign(key, msg):
    return hmac.new(key, msg.encode("utf-8"), hashlib.sha256).digest()


def calculate_key(secret_access_key, region):
    if region not in SMTP_REGIONS:
        raise ValueError(f"The {region} Region doesn't have an SMTP endpoint.")

    signature = sign(("AWS4" + secret_access_key).encode("utf-8"), DATE)
    signature = sign(signature, region)
    signature = sign(signature, SERVICE)
    signature = sign(signature, TERMINAL)
    signature = sign(signature, MESSAGE)
    signature_and_version = bytes([VERSION]) + signature
    smtp_password = base64.b64encode(signature_and_version)
    return smtp_password.decode("utf-8")
```

```
def main():
    parser = argparse.ArgumentParser(
        description="Convert a Secret Access Key to an SMTP password."
    )
    parser.add_argument("secret", help="The Secret Access Key to convert.")
    parser.add_argument(
        "region",
        help="The AWS Region where the SMTP password will be used.",
        choices=SMTP_REGIONS,
    )
    args = parser.parse_args()
    print(calculate_key(args.secret, args.region))


if __name__ == "__main__":
    main()
```

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

Step Functions를 사용하여 Lambda 함수 간접 호출

다음 코드 예제에서는 AWS Lambda 함수를 순차적으로 호출하는 AWS Step Functions 상태 시스템을 생성하는 방법을 보여줍니다.

Java

SDK for Java 2.x

AWS Step Functions 및를 사용하여 AWS 서버리스 워크플로를 생성하는 방법을 보여줍니다 AWS SDK for Java 2.x. 각 워크플로 단계는 AWS Lambda 함수를 사용하여 구현됩니다.

전체 소스 코드와 설정 및 실행 방법에 대한 지침은 GitHub에서 전체 예제를 참조하세요.

이 예제에서 사용되는 서비스

- DynamoDB
- Lambda
- Amazon SES
- Step Functions

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK를 사용하여 Amazon SES로 이메일 자격 증명 확인 및 메시지 전송

다음 코드 예제는 다음과 같은 작업을 수행하는 방법을 보여줍니다.

- Amazon SES로 이메일 주소를 추가하고 확인합니다.
- 표준 이메일 메시지를 보냅니다.
- 템플릿을 만들고 템플릿 이메일 메시지를 보냅니다.
- Amazon SES SMTP 서버를 사용하여 메시지를 보냅니다.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

Amazon SES로 이메일 주소를 확인하고 메시지를 보냅니다.

```
def usage_demo():
    print("-" * 88)
    print("Welcome to the Amazon Simple Email Service (Amazon SES) email demo!")
    print("-" * 88)

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    ses_client = boto3.client("ses")
    ses_identity = SesIdentity(ses_client)
    ses_mail_sender = SesMailSender(ses_client)
    ses_template = SesTemplate(ses_client)
    email = input("Enter an email address to send mail with Amazon SES: ")
    status = ses_identity.get_identity_status(email)
    verified = status == "Success"
    if not verified:
        answer = input(
```

```
            f"The address '{email}' is not verified with Amazon SES. Unless your
"
            f"Amazon SES account is out of sandbox, you can send mail only from "
            f"and to verified accounts. Do you want to verify this account for
use "
            f"with Amazon SES? If yes, the address will receive a verification "
            f"email (y/n): "
        )
        if answer.lower() == "y":
            ses_identity.verify_email_identity(email)
            print(f"Follow the steps in the email to {email} to complete
verification.")
            print("Waiting for verification...")
            try:
                ses_identity.wait_until_identity_exists(email)
                print(f"Identity verified for {email}.")
                verified = True
            except WaiterError:
                print(
                    f"Verification timeout exceeded. You must complete the "
                    f"steps in the email sent to {email} to verify the address."
                )

    if verified:
        test_message_text = "Hello from the Amazon SES mail demo!"
        test_message_html = "<p>Hello!</p><p>From the <b>Amazon SES</b> mail
demo!</p>"

        print(f"Sending mail from {email} to {email}.")
        ses_mail_sender.send_email(
            email,
            SesDestination([email]),
            "Amazon SES demo",
            test_message_text,
            test_message_html,
        )
        input("Mail sent. Check your inbox and press Enter to continue.")

        template = {
            "name": "doc-example-template",
            "subject": "Example of an email template.",
            "text": "This is what {{name}} will {{action}} if {{name}} can't
display "
            "HTML.",
```

```
            "html": "<p><i>This</i> is what {{name}} will {{action}} if {{name}}
 "
            "<b>can</b> display HTML.</p>",
        }
        print("Creating a template and sending a templated email.")
        ses_template.create_template(**template)
        template_data = {"name": email.split("@")[0], "action": "read"}
        if ses_template.verify_tags(template_data):
            ses_mail_sender.send_templated_email(
                email, SesDestination([email]), ses_template.name(),
 template_data
            )
            input("Mail sent. Check your inbox and press Enter to continue.")

        print("Sending mail through the Amazon SES SMTP server.")
        boto3_session = boto3.Session()
        region = boto3_session.region_name
        credentials = boto3_session.get_credentials()
        port = 587
        smtp_server = f"email-smtp.{region}.amazonaws.com"
        password = calculate_key(credentials.secret_key, region)
        message = """
Subject: Hi there

This message is sent from the Amazon SES SMTP mail demo."""
        context = ssl.create_default_context()
        with smtplib.SMTP(smtp_server, port) as server:
            server.starttls(context=context)
            server.login(credentials.access_key, password)
            server.sendmail(email, email, message)
        print("Mail sent. Check your inbox!")

    if ses_template.template is not None:
        print("Deleting demo template.")
        ses_template.delete_template()
    if verified:
        answer = input(f"Do you want to remove {email} from Amazon SES (y/n)? ")
        if answer.lower() == "y":
            ses_identity.delete_identity(email)
    print("Thanks for watching!")
    print("-" * 88)
```

Amazon SES ID 작업을 래핑하는 함수를 만듭니다.

```
class SesIdentity:
    """Encapsulates Amazon SES identity functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client


    def verify_domain_identity(self, domain_name):
        """
        Starts verification of a domain identity. To complete verification, you
 must
        create a TXT record with a specific format through your DNS provider.

        For more information, see *Verifying a domain with Amazon SES* in the
        Amazon SES documentation:
            https://docs.aws.amazon.com/ses/latest/DeveloperGuide/verify-domain-
procedure.html

        :param domain_name: The name of the domain to verify.
        :return: The token to include in the TXT record with your DNS provider.
        """
        try:
            response = self.ses_client.verify_domain_identity(Domain=domain_name)
            token = response["VerificationToken"]
            logger.info("Got domain verification token for %s.", domain_name)
        except ClientError:
            logger.exception("Couldn't verify domain %s.", domain_name)
            raise
        else:
            return token


    def verify_email_identity(self, email_address):
        """
        Starts verification of an email identity. This function causes an email
        to be sent to the specified email address from Amazon SES. To complete
```

```
        verification, follow the instructions in the email.

        :param email_address: The email address to verify.
        """
        try:
            self.ses_client.verify_email_identity(EmailAddress=email_address)
            logger.info("Started verification of %s.", email_address)
        except ClientError:
            logger.exception("Couldn't start verification of %s.", email_address)
            raise


    def wait_until_identity_exists(self, identity):
        """
        Waits until an identity exists. The waiter polls Amazon SES until the
        identity has been successfully verified or until it exceeds its maximum
time.

        :param identity: The identity to wait for.
        """
        try:
            waiter = self.ses_client.get_waiter("identity_exists")
            logger.info("Waiting until %s exists.", identity)
            waiter.wait(Identities=[identity])
        except WaiterError:
            logger.error("Waiting for identity %s failed or timed out.",
identity)
            raise


    def get_identity_status(self, identity):
        """
        Gets the status of an identity. This can be used to discover whether
        an identity has been successfully verified.

        :param identity: The identity to query.
        :return: The status of the identity.
        """
        try:
            response = self.ses_client.get_identity_verification_attributes(
                Identities=[identity]
            )
            status = response["VerificationAttributes"].get(
                identity, {"VerificationStatus": "NotFound"}
```

```
            )["VerificationStatus"]
            logger.info("Got status of %s for %s.", status, identity)
        except ClientError:
            logger.exception("Couldn't get status for %s.", identity)
            raise
        else:
            return status


    def delete_identity(self, identity):
        """
        Deletes an identity.

        :param identity: The identity to remove.
        """
        try:
            self.ses_client.delete_identity(Identity=identity)
            logger.info("Deleted identity %s.", identity)
        except ClientError:
            logger.exception("Couldn't delete identity %s.", identity)
            raise


    def list_identities(self, identity_type, max_items):
        """
        Gets the identities of the specified type for the current account.

        :param identity_type: The type of identity to retrieve, such as
EmailAddress.
        :param max_items: The maximum number of identities to retrieve.
        :return: The list of retrieved identities.
        """
        try:
            response = self.ses_client.list_identities(
                IdentityType=identity_type, MaxItems=max_items
            )
            identities = response["Identities"]
            logger.info("Got %s identities for the current account.",
len(identities))
        except ClientError:
            logger.exception("Couldn't list identities for the current account.")
            raise
        else:
            return identities
```

Amazon SES 템플릿 작업을 래핑하는 함수를 만듭니다.

```
class SesTemplate:
    """Encapsulates Amazon SES template functions."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
        self.ses_client = ses_client
        self.template = None
        self.template_tags = set()

    def _extract_tags(self, subject, text, html):
        """
        Extracts tags from a template as a set of unique values.

        :param subject: The subject of the email.
        :param text: The text version of the email.
        :param html: The html version of the email.
        """
        self.template_tags = set(re.findall(TEMPLATE_REGEX, subject + text +
 html))
        logger.info("Extracted template tags: %s", self.template_tags)


    def create_template(self, name, subject, text, html):
        """
        Creates an email template.

        :param name: The name of the template.
        :param subject: The subject of the email.
        :param text: The plain text version of the email.
        :param html: The HTML version of the email.
        """
        try:
            template = {
                "TemplateName": name,
                "SubjectPart": subject,
```

```
                "TextPart": text,
                "HtmlPart": html,
            }
            self.ses_client.create_template(Template=template)
            logger.info("Created template %s.", name)
            self.template = template
            self._extract_tags(subject, text, html)
        except ClientError:
            logger.exception("Couldn't create template %s.", name)
            raise


    def delete_template(self):
        """
        Deletes an email template.
        """
        try:

self.ses_client.delete_template(TemplateName=self.template["TemplateName"])
            logger.info("Deleted template %s.", self.template["TemplateName"])
            self.template = None
            self.template_tags = None
        except ClientError:
            logger.exception(
                "Couldn't delete template %s.", self.template["TemplateName"]
            )
            raise


    def get_template(self, name):
        """
        Gets a previously created email template.

        :param name: The name of the template to retrieve.
        :return: The retrieved email template.
        """
        try:
            response = self.ses_client.get_template(TemplateName=name)
            self.template = response["Template"]
            logger.info("Got template %s.", name)
            self._extract_tags(
                self.template["SubjectPart"],
                self.template["TextPart"],
                self.template["HtmlPart"],
```

```
            )
        except ClientError:
            logger.exception("Couldn't get template %s.", name)
            raise
        else:
            return self.template


    def list_templates(self):
        """
        Gets a list of all email templates for the current account.

        :return: The list of retrieved email templates.
        """
        try:
            response = self.ses_client.list_templates()
            templates = response["TemplatesMetadata"]
            logger.info("Got %s templates.", len(templates))
        except ClientError:
            logger.exception("Couldn't get templates.")
            raise
        else:
            return templates


    def update_template(self, name, subject, text, html):
        """
        Updates a previously created email template.

        :param name: The name of the template.
        :param subject: The subject of the email.
        :param text: The plain text version of the email.
        :param html: The HTML version of the email.
        """
        try:
            template = {
                "TemplateName": name,
                "SubjectPart": subject,
                "TextPart": text,
                "HtmlPart": html,
            }
            self.ses_client.update_template(Template=template)
            logger.info("Updated template %s.", name)
            self.template = template
```

```
            self._extract_tags(subject, text, html)
        except ClientError:
            logger.exception("Couldn't update template %s.", name)
            raise
```

Amazon SES 이메일 작업을 래핑하는 함수를 만듭니다.

```
class SesDestination:
    """Contains data about an email destination."""

    def __init__(self, tos, ccs=None, bccs=None):
        """
        :param tos: The list of recipients on the 'To:' line.
        :param ccs: The list of recipients on the 'CC:' line.
        :param bccs: The list of recipients on the 'BCC:' line.
        """
        self.tos = tos
        self.ccs = ccs
        self.bccs = bccs

    def to_service_format(self):
        """
        :return: The destination data in the format expected by Amazon SES.
        """
        svc_format = {"ToAddresses": self.tos}
        if self.ccs is not None:
            svc_format["CcAddresses"] = self.ccs
        if self.bccs is not None:
            svc_format["BccAddresses"] = self.bccs
        return svc_format



class SesMailSender:
    """Encapsulates functions to send emails with Amazon SES."""

    def __init__(self, ses_client):
        """
        :param ses_client: A Boto3 Amazon SES client.
        """
```

```
        self.ses_client = ses_client


    def send_email(self, source, destination, subject, text, html,
reply_tos=None):
        """
        Sends an email.

        Note: If your account is in the Amazon SES  sandbox, the source and
        destination email accounts must both be verified.

        :param source: The source email account.
        :param destination: The destination email account.
        :param subject: The subject of the email.
        :param text: The plain text version of the body of the email.
        :param html: The HTML version of the body of the email.
        :param reply_tos: Email accounts that will receive a reply if the
recipient
                              replies to the message.
        :return: The ID of the message, assigned by Amazon SES.
        """
        send_args = {
            "Source": source,
            "Destination": destination.to_service_format(),
            "Message": {
                "Subject": {"Data": subject},
                "Body": {"Text": {"Data": text}, "Html": {"Data": html}},
            },
        }
        if reply_tos is not None:
            send_args["ReplyToAddresses"] = reply_tos
        try:
            response = self.ses_client.send_email(**send_args)
            message_id = response["MessageId"]
            logger.info(
                "Sent mail %s from %s to %s.", message_id, source,
destination.tos
            )
        except ClientError:
            logger.exception(
                "Couldn't send mail from %s to %s.", source, destination.tos
            )
            raise
        else:
```

```
            return message_id


    def send_templated_email(
        self, source, destination, template_name, template_data, reply_tos=None
    ):
        """
        Sends an email based on a template. A template contains replaceable tags
        each enclosed in two curly braces, such as {{name}}. The template data
passed
        in this function contains key-value pairs that define the values to
insert
        in place of the template tags.

        Note: If your account is in the Amazon SES  sandbox, the source and
        destination email accounts must both be verified.

        :param source: The source email account.
        :param destination: The destination email account.
        :param template_name: The name of a previously created template.
        :param template_data: JSON-formatted key-value pairs of replacement
values
                              that are inserted in the template before it is
sent.
        :return: The ID of the message, assigned by Amazon SES.
        """
        send_args = {
            "Source": source,
            "Destination": destination.to_service_format(),
            "Template": template_name,
            "TemplateData": json.dumps(template_data),
        }
        if reply_tos is not None:
            send_args["ReplyToAddresses"] = reply_tos
        try:
            response = self.ses_client.send_templated_email(**send_args)
            message_id = response["MessageId"]
            logger.info(
                "Sent templated mail %s from %s to %s.",
                message_id,
                source,
                destination.tos,
            )
        except ClientError:
```

```
            logger.exception(
                "Couldn't send templated mail from %s to %s.", source,
destination.tos
            )
            raise
        else:
            return message_id
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 다음 주제를 참조하세요.
  - CreateTemplate
  - DeleteIdentity
  - DeleteTemplate
  - GetIdentityVerificationAttributes
  - GetTemplate
  - ListIdentities
  - ListTemplates
  - SendEmail
  - SendTemplatedEmail
  - UpdateTemplate
  - VerifyDomainIdentity
  - VerifyEmailIdentity

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDKs를 사용한 Amazon SES API v2의 코드 예제

다음 코드 예제에서는 Amazon SES API v2를 AWS 소프트웨어 개발 키트(SDK)와 함께 사용하는 방법을 보여줍니다.

작업은 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 작업은 관련 시나리오의 컨텍스트에 따라 표시되며, 개별 서비스 함수를 직접적으로 호출하는 방법을 보여줍니다.

시나리오는 동일한 서비스 내에서 또는 다른 AWS 서비스와 결합된 상태에서 여러 함수를 호출하여 특정 태스크를 수행하는 방법을 보여주는 코드 예제입니다.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

코드 예시

- AWS SDKs를 사용한 Amazon SES API v2의 기본 예제
  - AWS SDKs를 사용하는 Amazon SES API v2에 대한 작업
    - AWS SDK와 CreateContact 함께 사용
    - AWS SDK와 CreateContactList 함께 사용
    - AWS SDK와 CreateEmailIdentity 함께 사용
    - AWS SDK와 CreateEmailTemplate 함께 사용
    - AWS SDK와 DeleteContactList 함께 사용
    - AWS SDK와 DeleteEmailIdentity 함께 사용
    - AWS SDK와 DeleteEmailTemplate 함께 사용
    - AWS SDK와 GetEmailIdentity 함께 사용
    - AWS SDK와 ListContactLists 함께 사용
    - AWS SDK와 ListContacts 함께 사용
    - AWS SDK와 SendEmail 함께 사용
- AWS SDKs를 사용한 Amazon SES API v2 시나리오
  - AWS SDK를 사용한 전체 Amazon SES API v2 뉴스레터 시나리오

AWS SDKs를 사용한 Amazon SES API v2의 기본 예제

다음 코드 예제에서는 AWS SDK와 함께 Amazon Simple Email Service API v2의 기본 사항을 사용하는 방법을 보여줍니다.

예시

- AWS SDKs를 사용하는 Amazon SES API v2에 대한 작업
  - AWS SDK와 CreateContact 함께 사용
  - AWS SDK와 CreateContactList 함께 사용
  - AWS SDK와 CreateEmailIdentity 함께 사용

- AWS SDK와 CreateEmailTemplate 함께 사용
- AWS SDK와 DeleteContactList 함께 사용
- AWS SDK와 DeleteEmailIdentity 함께 사용
- AWS SDK와 DeleteEmailTemplate 함께 사용
- AWS SDK와 GetEmailIdentity 함께 사용
- AWS SDK와 ListContactLists 함께 사용
- AWS SDK와 ListContacts 함께 사용
- AWS SDK와 SendEmail 함께 사용

AWS SDKs를 사용하는 Amazon SES API v2에 대한 작업

다음 코드 예제에서는 AWS SDKs를 사용하여 개별 Amazon SES API v2 작업을 수행하는 방법을 보여줍니다. 각 예제에는 GitHub에 대한 링크가 포함되어 있습니다. 여기에서 코드 설정 및 실행에 대한 지침을 찾을 수 있습니다.

이들 발췌문은 Amazon SES API v2 API를 호출하며, 컨텍스트에서 실행되어야 하는 더 큰 프로그램에서 발췌한 코드입니다. AWS SDKs를 사용한 Amazon SES API v2 시나리오 에서 컨텍스트에 맞는 작업을 볼 수 있습니다.

다음 예제에는 가장 일반적으로 사용되는 작업만 포함되어 있습니다. 전체 목록은 Amazon Simple Email Service API v2 API 참조를 참조하세요.

예시

- AWS SDK와 CreateContact 함께 사용
- AWS SDK와 CreateContactList 함께 사용
- AWS SDK와 CreateEmailIdentity 함께 사용
- AWS SDK와 CreateEmailTemplate 함께 사용
- AWS SDK와 DeleteContactList 함께 사용
- AWS SDK와 DeleteEmailIdentity 함께 사용
- AWS SDK와 DeleteEmailTemplate 함께 사용
- AWS SDK와 GetEmailIdentity 함께 사용
- AWS SDK와 ListContactLists 함께 사용
- AWS SDK와 ListContacts 함께 사용
- AWS SDK와 SendEmail 함께 사용

AWS SDK와 **CreateContact** 함께 사용

다음 코드 예시는 `CreateContact`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.

- 뉴스레터 시나리오

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Creates a contact and adds it to the specified contact list.
    /// </summary>
    /// <param name="emailAddress">The email address of the contact.</param>
    /// <param name="contactListName">The name of the contact list.</param>
    /// <returns>The response from the CreateContact operation.</returns>
    public async Task<bool> CreateContactAsync(string emailAddress, string
contactListName)
    {
        var request = new CreateContactRequest
        {
            EmailAddress = emailAddress,
            ContactListName = contactListName
        };

        try
        {
            var response = await _sesClient.CreateContactAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (AlreadyExistsException ex)
        {
```

```
            Console.WriteLine($"Contact with email address {emailAddress} already
exists in the contact list {contactListName}.");
            Console.WriteLine(ex.Message);
            return true;
        }
        catch (NotFoundException ex)
        {
            Console.WriteLine($"The contact list {contactListName} does not
exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while creating the contact:
{ex.Message}");
        }
        return false;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 CreateContact를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
      try {
        // Create a new contact with the provided email address in the
        CreateContactRequest contactRequest = CreateContactRequest.builder()
            .contactListName(CONTACT_LIST_NAME)
```

```
            .emailAddress(emailAddress)
            .build();

        sesClient.createContact(contactRequest);
        contacts.add(emailAddress);

        System.out.println("Contact created: " + emailAddress);

        // Send a welcome email to the new contact
        String welcomeHtml = Files.readString(Paths.get("resources/
coupon_newsletter/welcome.html"));
        String welcomeText = Files.readString(Paths.get("resources/
coupon_newsletter/welcome.txt"));

        SendEmailRequest welcomeEmailRequest = SendEmailRequest.builder()
            .fromEmailAddress(this.verifiedEmail)
            .destination(Destination.builder().toAddresses(emailAddress).build())
            .content(EmailContent.builder()
                .simple(
                    Message.builder()
                        .subject(Content.builder().data("Welcome to the Weekly
 Coupons Newsletter").build())
                        .body(Body.builder()
                            .text(Content.builder().data(welcomeText).build())
                            .html(Content.builder().data(welcomeHtml).build())
                            .build())
                        .build())
                .build())
            .build();
        SendEmailResponse welcomeEmailResponse =
 sesClient.sendEmail(welcomeEmailRequest);
        System.out.println("Welcome email sent: " +
 welcomeEmailResponse.messageId());
      } catch (AlreadyExistsException e) {
        // If the contact already exists, skip this step for that contact and
 proceed
        // with the next contact
        System.out.println("Contact already exists, skipping creation...");
      } catch (Exception e) {
        System.err.println("Error occurred while processing email address " +
 emailAddress + ": " + e.getMessage());
        throw e;
      }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateContact를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep
```

```
            try:
                # Create a new contact
                self.ses_client.create_contact(
                    ContactListName=CONTACT_LIST_NAME, EmailAddress=email
                )
                print(f"Contact with email '{email}' created successfully.")

                # Send the welcome email
                self.ses_client.send_email(
                    FromEmailAddress=self.verified_email,
                    Destination={"ToAddresses": [email]},
                    Content={
                        "Simple": {
                            "Subject": {
                                "Data": "Welcome to the Weekly Coupons
Newsletter"
                            },
                            "Body": {
                                "Text": {"Data": welcome_text},
                                "Html": {"Data": welcome_html},
                            },
                        }
                    },
                )
                print(f"Welcome email sent to '{email}'.")
                if self.sleep:
                    # 1 email per second in sandbox mode, remove in production.
                    sleep(1.1)
            except ClientError as e:
                # If the contact already exists, skip and proceed
                if e.response["Error"]["Code"] == "AlreadyExistsException":
                    print(f"Contact with email '{email}' already exists.
Skipping...")
                else:
                    raise e
```

- API 세부 정보는 AWS SDK for Python(Boto3) API 참조의 CreateContact를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn add_contact(client: &Client, list: &str, email: &str) -> Result<(),
 Error> {
    client
        .create_contact()
        .contact_list_name(list)
        .email_address(email)
        .send()
        .await?;

    println!("Created contact");

    Ok(())
}
```

- API 세부 정보는 AWS SDK for Rust API 참조의 CreateContact를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`CreateContactList`** 함께 사용

다음 코드 예시는 `CreateContactList`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.

- 뉴스레터 시나리오

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Creates a contact list with the specified name.
    /// </summary>
    /// <param name="contactListName">The name of the contact list.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> CreateContactListAsync(string contactListName)
    {
        var request = new CreateContactListRequest
        {
            ContactListName = contactListName
        };

        try
        {
            var response = await _sesClient.CreateContactListAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (AlreadyExistsException ex)
        {
            Console.WriteLine($"Contact list with name {contactListName} already
exists.");
            Console.WriteLine(ex.Message);
            return true;
        }
        catch (LimitExceededException ex)
        {
            Console.WriteLine("The limit for contact lists has been exceeded.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
```

```
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while creating the contact
list: {ex.Message}");
        }
        return false;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 CreateContactList를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    try {
      // 2. Create a contact list
      String contactListName = CONTACT_LIST_NAME;
      CreateContactListRequest createContactListRequest =
CreateContactListRequest.builder()
          .contactListName(contactListName)
          .build();
      sesClient.createContactList(createContactListRequest);
      System.out.println("Contact list created: " + contactListName);
    } catch (AlreadyExistsException e) {
      System.out.println("Contact list already exists, skipping creation: weekly-
coupons-newsletter");
    } catch (LimitExceededException e) {
      System.err.println("Limit for contact lists has been exceeded.");
      throw e;
    } catch (SesV2Exception e) {
      System.err.println("Error creating contact list: " + e.getMessage());
```

```
        throw e;
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateContactList를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep
```

```
        try:

 self.ses_client.create_contact_list(ContactListName=CONTACT_LIST_NAME)
            print(f"Contact list '{CONTACT_LIST_NAME}' created successfully.")
        except ClientError as e:
            # If the contact list already exists, skip and proceed
            if e.response["Error"]["Code"] == "AlreadyExistsException":
                print(f"Contact list '{CONTACT_LIST_NAME}' already exists.")
            else:
                raise e
```

- API 세부 정보는 AWS SDK for Python(Boto3) API 참조의 CreateContactList를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn make_list(client: &Client, contact_list: &str) -> Result<(), Error> {
    client
        .create_contact_list()
        .contact_list_name(contact_list)
        .send()
        .await?;

    println!("Created contact list.");

    Ok(())
}
```

- API 세부 정보는 AWS SDK for Rust API 참조의 CreateContactList를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`CreateEmailIdentity`** 함께 사용

다음 코드 예시는 `CreateEmailIdentity`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.

- 뉴스레터 시나리오

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Creates an email identity (email address or domain) and starts the
verification process.
    /// </summary>
    /// <param name="emailIdentity">The email address or domain to create and
verify.</param>
    /// <returns>The response from the CreateEmailIdentity operation.</returns>
    public async Task<CreateEmailIdentityResponse>
CreateEmailIdentityAsync(string emailIdentity)
    {
        var request = new CreateEmailIdentityRequest
        {
            EmailIdentity = emailIdentity
        };

        try
        {
            var response = await _sesClient.CreateEmailIdentityAsync(request);
            return response;
        }
```

```
        catch (AlreadyExistsException ex)
        {
            Console.WriteLine($"Email identity {emailIdentity} already exists.");
            Console.WriteLine(ex.Message);
            throw;
        }
        catch (ConcurrentModificationException ex)
        {
            Console.WriteLine($"The email identity {emailIdentity} is being
modified by another operation or thread.");
            Console.WriteLine(ex.Message);
            throw;
        }
        catch (LimitExceededException ex)
        {
            Console.WriteLine("The limit for email identities has been
exceeded.");
            Console.WriteLine(ex.Message);
            throw;
        }
        catch (NotFoundException ex)
        {
            Console.WriteLine($"The email identity {emailIdentity} does not
exist.");
            Console.WriteLine(ex.Message);
            throw;
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while creating the email
identity: {ex.Message}");
            throw;
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 CreateEmailIdentity를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   try {
     CreateEmailIdentityRequest createEmailIdentityRequest =
CreateEmailIdentityRequest.builder()
         .emailIdentity(verifiedEmail)
         .build();
     sesClient.createEmailIdentity(createEmailIdentityRequest);
     System.out.println("Email identity created: " + verifiedEmail);
   } catch (AlreadyExistsException e) {
     System.out.println("Email identity already exists, skipping creation: " +
verifiedEmail);
   } catch (NotFoundException e) {
     System.err.println("The provided email address is not verified: " +
verifiedEmail);
     throw e;
   } catch (LimitExceededException e) {
     System.err
         .println("You have reached the limit for email identities. Please
remove some identities and try again.");
     throw e;
   } catch (SesV2Exception e) {
     System.err.println("Error creating email identity: " + e.getMessage());
     throw e;
   }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateEmailIdentity를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep


        try:

 self.ses_client.create_email_identity(EmailIdentity=self.verified_email)
```

```
            print(f"Email identity '{self.verified_email}' created
successfully.")
        except ClientError as e:
            # If the email identity already exists, skip and proceed
            if e.response["Error"]["Code"] == "AlreadyExistsException":
                print(f"Email identity '{self.verified_email}' already exists.")
            else:
                raise e
```

- API 세부 정보는 AWS SDK for Python(Boto3) API 참조의 CreateEmailIdentity를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
        match self
            .client
            .create_email_identity()
            .email_identity(self.verified_email.clone())
            .send()
            .await
        {
            Ok(_) => writeln!(self.stdout, "Email identity created
successfully.")?,
            Err(e) => match e.into_service_error() {
                CreateEmailIdentityError::AlreadyExistsException(_) => {
                    writeln!(
                        self.stdout,
                        "Email identity already exists, skipping creation."
                    )?;
                }
                e => return Err(anyhow!("Error creating email identity: {}", e)),
            },
        }
```

- API 세부 정보는 AWS SDK for Rust API 참조의 CreateEmailIdentity를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`CreateEmailTemplate`** 함께 사용

다음 코드 예시는 `CreateEmailTemplate`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.

- 뉴스레터 시나리오

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Creates an email template with the specified content.
    /// </summary>
    /// <param name="templateName">The name of the email template.</param>
    /// <param name="subject">The subject of the email template.</param>
    /// <param name="htmlContent">The HTML content of the email template.</param>
    /// <param name="textContent">The text content of the email template.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> CreateEmailTemplateAsync(string templateName, string
subject, string htmlContent, string textContent)
    {
        var request = new CreateEmailTemplateRequest
        {
            TemplateName = templateName,
```

```
            TemplateContent = new EmailTemplateContent
            {
                Subject = subject,
                Html = htmlContent,
                Text = textContent
            }
        };

        try
        {
            var response = await _sesClient.CreateEmailTemplateAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (AlreadyExistsException ex)
        {
            Console.WriteLine($"Email template with name {templateName} already
exists.");
            Console.WriteLine(ex.Message);
        }
        catch (LimitExceededException ex)
        {
            Console.WriteLine("The limit for email templates has been
exceeded.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while creating the email
template: {ex.Message}");
        }

        return false;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 CreateEmailTemplate를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    try {
      // Create an email template named "weekly-coupons"
      String newsletterHtml = loadFile("resources/coupon_newsletter/coupon-
newsletter.html");
      String newsletterText = loadFile("resources/coupon_newsletter/coupon-
newsletter.txt");

      CreateEmailTemplateRequest templateRequest =
 CreateEmailTemplateRequest.builder()
          .templateName(TEMPLATE_NAME)
          .templateContent(EmailTemplateContent.builder()
              .subject("Weekly Coupons Newsletter")
              .html(newsletterHtml)
              .text(newsletterText)
              .build())
          .build();

      sesClient.createEmailTemplate(templateRequest);

      System.out.println("Email template created: " + TEMPLATE_NAME);
    } catch (AlreadyExistsException e) {
      // If the template already exists, skip this step and proceed with the next
      // operation
      System.out.println("Email template already exists, skipping creation...");
    } catch (LimitExceededException e) {
      // If the limit for email templates is exceeded, fail the workflow and
 inform
      // the user
      System.err.println("You have reached the limit for email templates. Please
 remove some templates and try again.");
      throw e;
    } catch (Exception e) {
```

```
        System.err.println("Error occurred while creating email template: " +
 e.getMessage());
        throw e;
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateEmailTemplate를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
```

```
        self.ses_client = ses_client
        self.sleep = sleep


        try:
            template_content = {
                "Subject": "Weekly Coupons Newsletter",
                "Html": load_file_content("coupon-newsletter.html"),
                "Text": load_file_content("coupon-newsletter.txt"),
            }
            self.ses_client.create_email_template(
                TemplateName=TEMPLATE_NAME, TemplateContent=template_content
            )
            print(f"Email template '{TEMPLATE_NAME}' created successfully.")
        except ClientError as e:
            # If the template already exists, skip and proceed
            if e.response["Error"]["Code"] == "AlreadyExistsException":
                print(f"Email template '{TEMPLATE_NAME}' already exists.")
            else:
                raise e
```

- API 세부 정보는 AWS SDK for Python(Boto3) API 참조의 CreateEmailTemplate를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
        let template_html =
            std::fs::read_to_string("../resources/newsletter/coupon-
newsletter.html")
                .unwrap_or_else(|_| "Missing coupon-
newsletter.html".to_string());
        let template_text =
```

```
            std::fs::read_to_string("../resources/newsletter/coupon-
newsletter.txt")
                .unwrap_or_else(|_| "Missing coupon-newsletter.txt".to_string());

        // Create the email template
        let template_content = EmailTemplateContent::builder()
            .subject("Weekly Coupons Newsletter")
            .html(template_html)
            .text(template_text)
            .build();

        match self
            .client
            .create_email_template()
            .template_name(TEMPLATE_NAME)
            .template_content(template_content)
            .send()
            .await
        {
            Ok(_) => writeln!(self.stdout, "Email template created
 successfully.")?,
            Err(e) => match e.into_service_error() {
                CreateEmailTemplateError::AlreadyExistsException(_) => {
                    writeln!(
                        self.stdout,
                        "Email template already exists, skipping creation."
                    )?;
                }
                e => return Err(anyhow!("Error creating email template: {}", e)),
            },
        }
```

- API 세부 정보는 AWS SDK for Rust API 참조의 CreateEmailTemplate를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **DeleteContactList** 함께 사용

다음 코드 예시는 DeleteContactList의 사용 방법을 보여 줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 뉴스레터 시나리오

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Deletes a contact list and all contacts within it.
    /// </summary>
    /// <param name="contactListName">The name of the contact list to delete.</
param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteContactListAsync(string contactListName)
    {
        var request = new DeleteContactListRequest
        {
            ContactListName = contactListName
        };

        try
        {
            var response = await _sesClient.DeleteContactListAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (ConcurrentModificationException ex)
        {
            Console.WriteLine($"The contact list {contactListName} is being
 modified by another operation or thread.");
            Console.WriteLine(ex.Message);
        }
        catch (NotFoundException ex)
        {
```

```
            Console.WriteLine($"The contact list {contactListName} does not
exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while deleting the contact
list: {ex.Message}");
        }

        return false;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 DeleteContactList를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    try {
      // Delete the contact list
      DeleteContactListRequest deleteContactListRequest =
DeleteContactListRequest.builder()
          .contactListName(CONTACT_LIST_NAME)
          .build();

      sesClient.deleteContactList(deleteContactListRequest);

      System.out.println("Contact list deleted: " + CONTACT_LIST_NAME);
```

```
    } catch (NotFoundException e) {
      // If the contact list does not exist, log the error and proceed
      System.out.println("Contact list not found. Skipping deletion...");
    } catch (Exception e) {
      System.err.println("Error occurred while deleting the contact list: " +
 e.getMessage());
      e.printStackTrace();
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 DeleteContactList를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
```

```
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep


        try:

self.ses_client.delete_contact_list(ContactListName=CONTACT_LIST_NAME)
            print(f"Contact list '{CONTACT_LIST_NAME}' deleted successfully.")
        except ClientError as e:
            # If the contact list doesn't exist, skip and proceed
            if e.response["Error"]["Code"] == "NotFoundException":
                print(f"Contact list '{CONTACT_LIST_NAME}' does not exist.")
            else:
                print(e)
```

- API 세부 정보는 AWS SDK for Python(Boto3) API 참조의 DeleteContactList를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
        match self
            .client
            .delete_contact_list()
            .contact_list_name(CONTACT_LIST_NAME)
            .send()
            .await
        {
            Ok(_) => writeln!(self.stdout, "Contact list deleted
successfully.")?,
            Err(e) => return Err(anyhow!("Error deleting contact list: {e}")),
```

```
        }
```

- API 세부 정보는 AWS SDK for Rust API 참조의 DeleteContactList를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`DeleteEmailIdentity`** 함께 사용

다음 코드 예시는 `DeleteEmailIdentity`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.

- 뉴스레터 시나리오

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Deletes an email identity (email address or domain).
    /// </summary>
    /// <param name="emailIdentity">The email address or domain to delete.</
param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteEmailIdentityAsync(string emailIdentity)
    {
        var request = new DeleteEmailIdentityRequest
        {
            EmailIdentity = emailIdentity
        };

        try
```

```
        {
            var response = await _sesClient.DeleteEmailIdentityAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (ConcurrentModificationException ex)
        {
            Console.WriteLine($"The email identity {emailIdentity} is being
modified by another operation or thread.");
            Console.WriteLine(ex.Message);
        }
        catch (NotFoundException ex)
        {
            Console.WriteLine($"The email identity {emailIdentity} does not
exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while deleting the email
identity: {ex.Message}");
        }

        return false;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 DeleteEmailIdentity를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
      try {
        // Delete the email identity
        DeleteEmailIdentityRequest deleteIdentityRequest =
 DeleteEmailIdentityRequest.builder()
            .emailIdentity(this.verifiedEmail)
            .build();

        sesClient.deleteEmailIdentity(deleteIdentityRequest);

        System.out.println("Email identity deleted: " + this.verifiedEmail);
      } catch (NotFoundException e) {
        // If the email identity does not exist, log the error and proceed
        System.out.println("Email identity not found. Skipping deletion...");
      } catch (Exception e) {
        System.err.println("Error occurred while deleting the email identity: " +
 e.getMessage());
        e.printStackTrace();
      }
    } else {
      System.out.println("Skipping email identity deletion.");
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 DeleteEmailIdentity를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
```

```
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep


            try:

 self.ses_client.delete_email_identity(EmailIdentity=self.verified_email)
                print(f"Email identity '{self.verified_email}' deleted
 successfully.")
            except ClientError as e:
                # If the email identity doesn't exist, skip and proceed
                if e.response["Error"]["Code"] == "NotFoundException":
                    print(f"Email identity '{self.verified_email}' does not
 exist.")
                else:
                    print(e)
```

- API 세부 정보는 AWS SDK for Python(Boto3) API 참조의 DeleteEmailIdentity를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
            match self
                .client
                .delete_email_identity()
                .email_identity(self.verified_email.clone())
                .send()
                .await
            {
                Ok(_) => writeln!(self.stdout, "Email identity deleted
successfully.")?,
                Err(e) => {
                    return Err(anyhow!("Error deleting email identity: {}", e));
                }
            }
```

- API 세부 정보는 AWS SDK for Rust API 참조의 DeleteEmailIdentity를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`DeleteEmailTemplate`** 함께 사용

다음 코드 예시는 `DeleteEmailTemplate`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.

- 뉴스레터 시나리오

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Deletes an email template.
    /// </summary>
    /// <param name="templateName">The name of the email template to delete.</
param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteEmailTemplateAsync(string templateName)
    {
        var request = new DeleteEmailTemplateRequest
        {
            TemplateName = templateName
        };

        try
        {
            var response = await _sesClient.DeleteEmailTemplateAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (NotFoundException ex)
        {
            Console.WriteLine($"The email template {templateName} does not
 exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
 later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
```

```
            Console.WriteLine($"An error occurred while deleting the email
template: {ex.Message}");
        }

        return false;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 DeleteEmailTemplate를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    try {
      // Delete the template
      DeleteEmailTemplateRequest deleteTemplateRequest =
DeleteEmailTemplateRequest.builder()
          .templateName(TEMPLATE_NAME)
          .build();

      sesClient.deleteEmailTemplate(deleteTemplateRequest);

      System.out.println("Email template deleted: " + TEMPLATE_NAME);
    } catch (NotFoundException e) {
      // If the email template does not exist, log the error and proceed
      System.out.println("Email template not found. Skipping deletion...");
    } catch (Exception e) {
      System.err.println("Error occurred while deleting the email template: " +
e.getMessage());
      e.printStackTrace();
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 DeleteEmailTemplate를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep


        try:
            self.ses_client.delete_email_template(TemplateName=TEMPLATE_NAME)
            print(f"Email template '{TEMPLATE_NAME}' deleted successfully.")
        except ClientError as e:
```

```
            # If the email template doesn't exist, skip and proceed
            if e.response["Error"]["Code"] == "NotFoundException":
                print(f"Email template '{TEMPLATE_NAME}' does not exist.")
            else:
                print(e)
```

- API 세부 정보는 AWS SDK for Python(Boto3) API 참조의 DeleteEmailTemplate를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
        match self
            .client
            .delete_email_template()
            .template_name(TEMPLATE_NAME)
            .send()
            .await
        {
            Ok(_) => writeln!(self.stdout, "Email template deleted
successfully.")?,
            Err(e) => {
                return Err(anyhow!("Error deleting email template: {e}"));
            }
        }
```

- API 세부 정보는 AWS SDK for Rust API 참조의 DeleteEmailTemplate를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **GetEmailIdentity** 함께 사용

다음 코드 예시는 GetEmailIdentity의 사용 방법을 보여 줍니다.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

이메일 주소가 확인되었는지 여부를 결정합니다.

```
async fn is_verified(client: &Client, email: &str) -> Result<(), Error> {
    let resp = client
        .get_email_identity()
        .email_identity(email)
        .send()
        .await?;

    if resp.verified_for_sending_status() {
        println!("The address is verified");
    } else {
        println!("The address is not verified");
    }

    Ok(())
}
```

- API 세부 정보는 AWS SDK for Rust API 참조의 GetEmailIdentity를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **ListContactLists** 함께 사용

다음 코드 예시는 ListContactLists의 사용 방법을 보여 줍니다.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn show_lists(client: &Client) -> Result<(), Error> {
    let resp = client.list_contact_lists().send().await?;

    println!("Contact lists:");

    for list in resp.contact_lists() {
        println!("  {}", list.contact_list_name().unwrap_or_default());
    }

    Ok(())
}
```

- API 세부 정보는 AWS SDK for Rust API 참조의 ListContactLists를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **`ListContacts`** 함께 사용

다음 코드 예시는 `ListContacts`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.

- 뉴스레터 시나리오

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Lists the contacts in the specified contact list.
    /// </summary>
    /// <param name="contactListName">The name of the contact list.</param>
    /// <returns>The list of contacts response from the ListContacts operation.</
returns>
    public async Task<List<Contact>> ListContactsAsync(string contactListName)
    {
        var request = new ListContactsRequest
        {
            ContactListName = contactListName
        };

        try
        {
            var response = await _sesClient.ListContactsAsync(request);
            return response.Contacts;
        }
        catch (NotFoundException ex)
        {
            Console.WriteLine($"The contact list {contactListName} does not
 exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
 later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
```

```
            Console.WriteLine($"An error occurred while listing the contacts:
{ex.Message}");
        }

        return new List<Contact>();
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 ListContacts를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
      ListContactsRequest contactListRequest = ListContactsRequest.builder()
          .contactListName(CONTACT_LIST_NAME)
          .build();

      List<String> contactEmails;
      try {
        ListContactsResponse contactListResponse =
sesClient.listContacts(contactListRequest);

        contactEmails = contactListResponse.contacts().stream()
            .map(Contact::emailAddress)
            .toList();
      } catch (Exception e) {
        // TODO: Remove when listContacts's GET body issue is resolved.
        contactEmails = this.contacts;
      }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 ListContacts를 참조하세요.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep


        try:
            contacts_response = self.ses_client.list_contacts(
                ContactListName=CONTACT_LIST_NAME
            )
```

```
        except ClientError as e:
            if e.response["Error"]["Code"] == "NotFoundException":
                print(f"Contact list '{CONTACT_LIST_NAME}' does not exist.")
                return
            else:
                raise e
```

- API 세부 정보는 AWS SDK for Python(Boto3) API 참조의 ListContacts를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn show_contacts(client: &Client, list: &str) -> Result<(), Error> {
    let resp = client
        .list_contacts()
        .contact_list_name(list)
        .send()
        .await?;

    println!("Contacts:");

    for contact in resp.contacts() {
        println!("  {}", contact.email_address().unwrap_or_default());
    }

    Ok(())
}
```

- API 세부 정보는 AWS SDK for Rust API 참조의 ListContacts를 참조하세요.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK와 **SendEmail** 함께 사용

다음 코드 예시는 SendEmail의 사용 방법을 보여 줍니다.

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Sends an email with the specified content and options.
    /// </summary>
    /// <param name="fromEmailAddress">The email address to send the email
from.</param>
    /// <param name="toEmailAddresses">The email addresses to send the email
to.</param>
    /// <param name="subject">The subject of the email.</param>
    /// <param name="htmlContent">The HTML content of the email.</param>
    /// <param name="textContent">The text content of the email.</param>
    /// <param name="templateName">The name of the email template to use
(optional).</param>
    /// <param name="templateData">The data to replace placeholders in the email
template (optional).</param>
    /// <param name="contactListName">The name of the contact list for
unsubscribe functionality (optional).</param>
    /// <returns>The MessageId response from the SendEmail operation.</returns>
    public async Task<string> SendEmailAsync(string fromEmailAddress,
List<string> toEmailAddresses, string? subject,
        string? htmlContent, string? textContent, string? templateName = null,
string? templateData = null, string? contactListName = null)
    {
        var request = new SendEmailRequest
        {
            FromEmailAddress = fromEmailAddress
        };
```

```
        if (toEmailAddresses.Any())
        {
            request.Destination = new Destination { ToAddresses =
toEmailAddresses };
        }

        if (!string.IsNullOrEmpty(templateName))
        {
            request.Content = new EmailContent()
            {
                Template = new Template
                {
                    TemplateName = templateName,
                    TemplateData = templateData
                }
            };
        }
        else
        {
            request.Content = new EmailContent
            {
                Simple = new Message
                {
                    Subject = new Content { Data = subject },
                    Body = new Body
                    {
                        Html = new Content { Data = htmlContent },
                        Text = new Content { Data = textContent }
                    }
                }
            };
        }

        if (!string.IsNullOrEmpty(contactListName))
        {
            request.ListManagementOptions = new ListManagementOptions
            {
                ContactListName = contactListName
            };
        }

        try
        {
```

```
            var response = await _sesClient.SendEmailAsync(request);
            return response.MessageId;
        }
        catch (AccountSuspendedException ex)
        {
            Console.WriteLine("The account's ability to send email has been
permanently restricted.");
            Console.WriteLine(ex.Message);
        }
        catch (MailFromDomainNotVerifiedException ex)
        {
            Console.WriteLine("The sending domain is not verified.");
            Console.WriteLine(ex.Message);
        }
        catch (MessageRejectedException ex)
        {
            Console.WriteLine("The message content is invalid.");
            Console.WriteLine(ex.Message);
        }
        catch (SendingPausedException ex)
        {
            Console.WriteLine("The account's ability to send email is currently
paused.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while sending the email:
{ex.Message}");
        }

        return string.Empty;
    }
```

- API 세부 정보는 AWS SDK for .NET API 참조의 SendEmail을 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

메시지를 전송합니다.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sesv2.model.Body;
import software.amazon.awssdk.services.sesv2.model.Content;
import software.amazon.awssdk.services.sesv2.model.Destination;
import software.amazon.awssdk.services.sesv2.model.EmailContent;
import software.amazon.awssdk.services.sesv2.model.Message;
import software.amazon.awssdk.services.sesv2.model.SendEmailRequest;
import software.amazon.awssdk.services.sesv2.model.SesV2Exception;
import software.amazon.awssdk.services.sesv2.SesV2Client;

/**
 * Before running this AWS SDK for Java (v2) example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class SendEmail {
    public static void main(String[] args) {
        final String usage = """

                                Usage:
                                    <sender> <recipient> <subject>\s

                                Where:
                                    sender - An email address that represents the
 sender.\s
```

```
                                recipient - An email address that represents the
 recipient.\s
                                subject - The subject line.\s
                            """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String sender = args[0];
        String recipient = args[1];
        String subject = args[2];

        Region region = Region.US_EAST_1;
        SesV2Client sesv2Client = SesV2Client.builder()
                .region(region)
                .build();

        // The HTML body of the email.
        String bodyHTML = "<html>" + "<head></head>" + "<body>" + "<h1>Hello!</
h1>"
                + "<p> See the list of customers.</p>" + "</body>" + "</html>";

        send(sesv2Client, sender, recipient, subject, bodyHTML);
    }

    public static void send(SesV2Client client,
                            String sender,
                            String recipient,
                            String subject,
                            String bodyHTML) {

        Destination destination = Destination.builder()
                .toAddresses(recipient)
                .build();

        Content content = Content.builder()
                .data(bodyHTML)
                .build();

        Content sub = Content.builder()
                .data(subject)
                .build();
```

```
        Body body = Body.builder()
                .html(content)
                .build();

        Message msg = Message.builder()
                .subject(sub)
                .body(body)
                .build();

        EmailContent emailContent = EmailContent.builder()
                .simple(msg)
                .build();

        SendEmailRequest emailRequest = SendEmailRequest.builder()
                .destination(destination)
                .content(emailContent)
                .fromEmailAddress(sender)
                .build();

        try {
            System.out.println("Attempting to send an email through Amazon SES "
                    + "using the AWS SDK for Java...");
            client.sendEmail(emailRequest);
            System.out.println("email was sent");

        } catch (SesV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

템플릿을 사용하여 메시지를 보냅니다.

```
      String coupons = Files.readString(Paths.get("resources/coupon_newsletter/
sample_coupons.json"));
      for (String emailAddress : contactEmails) {
        SendEmailRequest newsletterRequest = SendEmailRequest.builder()
            .destination(Destination.builder().toAddresses(emailAddress).build())
            .content(EmailContent.builder()
                .template(Template.builder()
```

```
                    .templateName(TEMPLATE_NAME)
                    .templateData(coupons)
                    .build())
                .build())
            .fromEmailAddress(this.verifiedEmail)
            .listManagementOptions(ListManagementOptions.builder()
                .contactListName(CONTACT_LIST_NAME)
                .build())
            .build();
        SendEmailResponse newsletterResponse =
 sesClient.sendEmail(newsletterRequest);
        System.out.println("Newsletter sent to " + emailAddress + ": " +
 newsletterResponse.messageId());
     }
```

헤더 정보가 포함된 메시지를 전송합니다.

```
public class SendwithHeader {

    public static void main(String[] args) {
        final String usage = """

            Usage:
                <sender> <recipient> <subject>\s

            Where:
                sender - An email address that represents the sender.\s
                recipient - An email address that represents the recipient.\s
                subject - The subject line.\s
            """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String sender = args[0];
        String recipient = args[1];
        String subject = args[2];
        Region region = Region.US_EAST_1;
        SesV2Client sesv2Client = SesV2Client.builder()
                .region(region)
```

```
                .build();

        String bodyHTML = """
                <html>
                    <head></head>
                    <body>
                        <h1>Hello!</h1>
                        <p>See the list of customers.</p>
                    </body>
                </html>
                """;

        sendWithHeader(sesv2Client, sender, recipient, subject, bodyHTML);
        sesv2Client.close();
    }

    /**
     * Sends an email using the AWS SES V2 client.
     *
     * @param sesv2Client the SES V2 client to use for sending the email
     * @param sender the email address of the sender
     * @param recipient the email address of the recipient
     * @param subject the subject of the email
     * @param bodyHTML the HTML content of the email body
     */
    public static void sendWithHeader(SesV2Client sesv2Client,
                                      String sender,
                                      String recipient,
                                      String subject,
                                      String bodyHTML) {
        EmailContent emailContent = EmailContent.builder()
                .simple(Message.builder()
                        .body(b -> b.html(c ->
 c.charset(UTF_8.name()).data(bodyHTML))
                                .text(c ->
 c.charset(UTF_8.name()).data(bodyHTML)))
                        .subject(c -> c.charset(UTF_8.name()).data(subject))
                        .headers(List.of(
                                MessageHeader.builder()
                                        .name("List-Unsubscribe")
                                        .value("<https://nutrition.co/?
address=x&topic=x>, <mailto:unsubscribe@nutrition.co?subject=TopicUnsubscribe>")
                                        .build(),
                                MessageHeader.builder()
```

```
                                        .name("List-Unsubscribe-Post")
                                        .value("List-Unsubscribe=One-Click")
                                        .build()))
                        .build())
                .build();

        SendEmailRequest request = SendEmailRequest.builder()
                .fromEmailAddress(sender)
                .destination(d -> d.toAddresses(recipient))
                .content(emailContent)
                .build();

        try {
            SendEmailResponse response = sesv2Client.sendEmail(request);
            System.out.println("Email sent! Message ID: " +
 response.messageId());
        } catch (SesV2Exception e) {
            System.err.println("Failed to send email: " +
 e.awsErrorDetails().errorMessage());
            throw new RuntimeException(e);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 SendEmail을 참조하십시오.

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

연락처 목록의 모든 구성원에게 메시지를 전송합니다.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
```

```
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep


                self.ses_client.send_email(
                    FromEmailAddress=self.verified_email,
                    Destination={"ToAddresses": [email]},
                    Content={
                        "Simple": {
                            "Subject": {
                                "Data": "Welcome to the Weekly Coupons
 Newsletter"
                            },
                            "Body": {
                                "Text": {"Data": welcome_text},
                                "Html": {"Data": welcome_html},
                            },
                        }
                    },
                )
                print(f"Welcome email sent to '{email}'.")
```

연락처 목록의 모든 구성원에게 템플릿을 사용하여 메시지를 전송합니다.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep


                self.ses_client.send_email(
                    FromEmailAddress=self.verified_email,
                    Destination={"ToAddresses": [email_address]},
                    Content={
                        "Template": {
                            "TemplateName": TEMPLATE_NAME,
                            "TemplateData": coupon_items,
                        }
                    },
                    ListManagementOptions={"ContactListName": CONTACT_LIST_NAME},
                )
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 SendEmail를 참조하세요.

Ruby

SDK for Ruby

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require 'aws-sdk-sesv2'
require_relative 'config' # Recipient and sender email addresses.

# Set up the SESv2 client.
client = Aws::SESV2::Client.new(region: AWS_REGION)

def send_email(client, sender_email, recipient_email)
  response = client.send_email(
    {
      from_email_address: sender_email,
      destination: {
        to_addresses: [recipient_email]
      },
      content: {
        simple: {
          subject: {
            data: 'Test email subject'
          },
          body: {
            text: {
              data: 'Test email body'
            }
          }
        }
      }
    }
  )
  puts "Email sent from #{SENDER_EMAIL} to #{RECIPIENT_EMAIL} with message ID:
 #{response.message_id}"
end
```

```
send_email(client, SENDER_EMAIL, RECIPIENT_EMAIL)
```

- API 세부 정보는 AWS SDK for Ruby API 참조의 SendEmail을 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

연락처 목록의 모든 구성원에게 메시지를 전송합니다.

```
async fn send_message(
    client: &Client,
    list: &str,
    from: &str,
    subject: &str,
    message: &str,
) -> Result<(), Error> {
    // Get list of email addresses from contact list.
    let resp = client
        .list_contacts()
        .contact_list_name(list)
        .send()
        .await?;

    let contacts = resp.contacts();

    let cs: Vec<String> = contacts
        .iter()
        .map(|i| i.email_address().unwrap_or_default().to_string())
        .collect();

    let mut dest: Destination = Destination::builder().build();
    dest.to_addresses = Some(cs);
    let subject_content = Content::builder()
```

```
        .data(subject)
        .charset("UTF-8")
        .build()
        .expect("building Content");
    let body_content = Content::builder()
        .data(message)
        .charset("UTF-8")
        .build()
        .expect("building Content");
    let body = Body::builder().text(body_content).build();

    let msg = Message::builder()
        .subject(subject_content)
        .body(body)
        .build();

    let email_content = EmailContent::builder().simple(msg).build();

    client
        .send_email()
        .from_email_address(from)
        .destination(dest)
        .content(email_content)
        .send()
        .await?;

    println!("Email sent to list");

    Ok(())
}
```

연락처 목록의 모든 구성원에게 템플릿을 사용하여 메시지를 전송합니다.

```
            let coupons = std::fs::read_to_string("../resources/newsletter/
sample_coupons.json")
                .unwrap_or_else(|_| r#"{"coupons":[]}"#.to_string());
            let email_content = EmailContent::builder()
                .template(
                    Template::builder()
                        .template_name(TEMPLATE_NAME)
                        .template_data(coupons)
                        .build(),
```

```
                )
                .build();

            match self
                .client
                .send_email()
                .from_email_address(self.verified_email.clone())

.destination(Destination::builder().to_addresses(email.clone()).build())
                .content(email_content)
                .list_management_options(
                    ListManagementOptions::builder()
                        .contact_list_name(CONTACT_LIST_NAME)
                        .build()?,
                )
                .send()
                .await
            {
                Ok(output) => {
                    if let Some(message_id) = output.message_id {
                        writeln!(
                            self.stdout,
                            "Newsletter sent to {} with message ID {}",
                            email, message_id
                        )?;
                    } else {
                        writeln!(self.stdout, "Newsletter sent to {}", email)?;
                    }
                }
                Err(e) => return Err(anyhow!("Error sending newsletter to {}:
{}", email, e)),
            }
```

- API 세부 정보는 AWS SDK for Rust API 참조의 SendEmail을 참조하십시오.

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDKs를 사용한 Amazon SES API v2 시나리오

다음 코드 예제에서는 Amazon SES API v2 AWS SDKs에서 일반적인 시나리오를 구현하는 방법을 보여줍니다. 이러한 시나리오에서는 Amazon SES API v2 내에서 또는 다른 AWS 서비스와 결합된 상태에서 여러 함수를 직접적으로 호출하여 특정 태스크를 수행하는 방법을 보여줍니다. 각 시나리오에는 전체 소스 코드에 대한 링크가 포함되어 있습니다. 여기에서 코드를 설정하고 실행하는 방법에 대한 지침을 찾을 수 있습니다.

시나리오는 컨텍스트에 맞는 서비스 작업을 이해하는 데 도움이 되도록 중급 수준의 경험을 대상으로 합니다.

예시

- AWS SDK를 사용한 전체 Amazon SES API v2 뉴스레터 시나리오

AWS SDK를 사용한 전체 Amazon SES API v2 뉴스레터 시나리오

다음 코드 예제에서는 Amazon SES API v2 뉴스레터 시나리오를 실행하는 방법을 보여줍니다.

.NET

SDK for .NET

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

시나리오를 실행합니다.

```
using System.Diagnostics;
using System.Text.RegularExpressions;
using Amazon.SimpleEmailV2;
using Amazon.SimpleEmailV2.Model;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using Microsoft.Extensions.Logging;
using Microsoft.Extensions.Logging.Console;
using Microsoft.Extensions.Logging.Debug;
```

```
namespace Sesv2Scenario;

public static class NewsletterWorkflow
{
    /*
      This scenario demonstrates how to use the Amazon Simple Email Service (SES)
 v2 to send a coupon newsletter to a list of subscribers.
      The scenario performs the following tasks:

      1. Prepare the application:
         - Create a verified email identity for sending and replying to emails.
         - Create a contact list to store the subscribers' email addresses.
         - Create an email template for the coupon newsletter.

      2. Gather subscriber email addresses:
         - Prompt the user for a base email address.
         - Create 3 variants of the email address using subaddress extensions
 (e.g., user+ses-weekly-newsletter-1@example.com).
         - Add each variant as a contact to the contact list.
         - Send a welcome email to each new contact.

      3. Send the coupon newsletter:
         - Retrieve the list of contacts from the contact list.
         - Send the coupon newsletter using the email template to each contact.

      4. Monitor and review:
         - Provide instructions for the user to review the sending activity and
 metrics in the AWS console.

      5. Clean up resources:
         - Delete the contact list (which also deletes all contacts within it).
         - Delete the email template.
         - Optionally delete the verified email identity.

    */

    public static SESv2Wrapper _sesv2Wrapper;
    public static string? _baseEmailAddress = null;
    public static string? _verifiedEmail = null;
    private static string _contactListName = "weekly-coupons-newsletter";
    private static string _templateName = "weekly-coupons";
    private static string _subject = "Weekly Coupons Newsletter";
    private static string _htmlContentFile = "coupon-newsletter.html";
    private static string _textContentFile = "coupon-newsletter.txt";
```

```
    private static string _htmlWelcomeFile = "welcome.html";
    private static string _textWelcomeFile = "welcome.txt";
    private static string _couponsDataFile = "sample_coupons.json";

    // Relative location of the resources folder.
    private static string _resourcesFilePathLocation = "../../../../resources/";

    public static async Task Main(string[] args)
    {
        // Set up dependency injection for the Amazon service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonSimpleEmailServiceV2>()
                    .AddTransient<SESv2Wrapper>()
            )
            .Build();

        ServicesSetup(host);

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Amazon SES v2 Coupon Newsletter
Scenario.");
            Console.WriteLine("This scenario demonstrates how to use the Amazon
Simple Email Service (SES) v2 " +
                              "\r\nto send a coupon newsletter to a list of
subscribers.");

            // Prepare the application.
            var emailIdentity = await PrepareApplication();

            // Gather subscriber email addresses.
            await GatherSubscriberEmailAddresses(emailIdentity);

            // Send the coupon newsletter.
            await SendCouponNewsletter(emailIdentity);
```

```
            // Monitor and review.
            MonitorAndReview(true);

            // Clean up resources.
            await Cleanup(emailIdentity, true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Amazon SES v2 Coupon Newsletter scenario is
complete.");
            Console.WriteLine(new string('-', 80));
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred: {ex.Message}");
        }
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _sesv2Wrapper = host.Services.GetRequiredService<SESv2Wrapper>();
    }

    /// <summary>
    /// Set up the resources for the scenario.
    /// </summary>
    /// <returns>The email address of the verified identity.</returns>
    public static async Task<string?> PrepareApplication()
    {
        var htmlContent = await File.ReadAllTextAsync(_resourcesFilePathLocation
+ _htmlContentFile);
        var textContent = await File.ReadAllTextAsync(_resourcesFilePathLocation
+ _textContentFile);

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("1. In this step, we will prepare the application:" +
                          "\r\n  - Create a verified email identity for sending
and replying to emails." +
```

```
                          "\r\n  - Create a contact list to store the
subscribers' email addresses." +
                          "\r\n  - Create an email template for the coupon
newsletter.\r\n");

       // Prompt the user for a verified email address.
       while (!IsEmail(_verifiedEmail))
       {
           Console.Write("Enter a verified email address or an email to verify:
");
           _verifiedEmail = Console.ReadLine();
       }

       try
       {
           // Create an email identity and start the verification process.
           await _sesv2Wrapper.CreateEmailIdentityAsync(_verifiedEmail);
           Console.WriteLine($"Identity {_verifiedEmail} created.");
       }
       catch (AlreadyExistsException)
       {
           Console.WriteLine($"Identity {_verifiedEmail} already exists.");
       }
       catch (Exception ex)
       {
           Console.WriteLine($"Error creating email identity: {ex.Message}");
       }

       // Create a contact list.
       try
       {
           await _sesv2Wrapper.CreateContactListAsync(_contactListName);
           Console.WriteLine($"Contact list {_contactListName} created.");
       }
       catch (AlreadyExistsException)
       {
           Console.WriteLine($"Contact list {_contactListName} already
exists.");
       }
       catch (Exception ex)
       {
           Console.WriteLine($"Error creating contact list: {ex.Message}");
       }
```

```
        // Create an email template.
        try
        {
            await _sesv2Wrapper.CreateEmailTemplateAsync(_templateName, _subject,
 htmlContent, textContent);
            Console.WriteLine($"Email template {_templateName} created.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine($"Email template {_templateName} already exists.");
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error creating email template: {ex.Message}");
        }

        return _verifiedEmail;
    }

    /// <summary>
    /// Generate subscriber addresses and send welcome emails.
    /// </summary>
    /// <param name="fromEmailAddress">The verified email address from
 PrepareApplication.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> GatherSubscriberEmailAddresses(string
 fromEmailAddress)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("2. In Step 2, we will gather subscriber email
 addresses:" +
                          "\r\n  - Prompt the user for a base email address." +
                          "\r\n  - Create 3 variants of the email address using
 subaddress extensions (e.g., user+ses-weekly-newsletter-1@example.com)." +
                          "\r\n  - Add each variant as a contact to the contact
 list." +
                          "\r\n  - Send a welcome email to each new contact.\r
\n");

        // Prompt the user for a base email address.
        while (!IsEmail(_baseEmailAddress))
        {
            Console.Write("Enter a base email address (e.g., user@example.com):
 ");
```

```
            _baseEmailAddress = Console.ReadLine();
        }

        // Create 3 variants of the email address using +ses-weekly-newsletter-1,
 +ses-weekly-newsletter-2, etc.
        var baseEmailAddressParts = _baseEmailAddress!.Split("@");
        for (int i = 1; i <= 3; i++)
        {
            string emailAddress = $"{baseEmailAddressParts[0]}+ses-weekly-
newsletter-{i}@{baseEmailAddressParts[1]}";

            try
            {
                // Create a contact with the email address in the contact list.
                await _sesv2Wrapper.CreateContactAsync(emailAddress,
 _contactListName);
                Console.WriteLine($"Contact {emailAddress} added to the
 {_contactListName} contact list.");
            }
            catch (AlreadyExistsException)
            {
                Console.WriteLine($"Contact {emailAddress} already exists in the
 {_contactListName} contact list.");
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Error creating contact {emailAddress}:
 {ex.Message}");
                return false;
            }

            // Send a welcome email to the new contact.
            try
            {
                string subject = "Welcome to the Weekly Coupons Newsletter";
                string htmlContent = await
 File.ReadAllTextAsync(_resourcesFilePathLocation + _htmlWelcomeFile);
                string textContent = await
 File.ReadAllTextAsync(_resourcesFilePathLocation + _textWelcomeFile);

                await _sesv2Wrapper.SendEmailAsync(fromEmailAddress, new
 List<string> { emailAddress }, subject, htmlContent, textContent);
                Console.WriteLine($"Welcome email sent to {emailAddress}.");
            }
```

```
            catch (Exception ex)
            {
                Console.WriteLine($"Error sending welcome email to
{emailAddress}: {ex.Message}");
                return false;
            }

            // Wait 2 seconds before sending the next email (if the account is in
the SES Sandbox).
            await Task.Delay(2000);
        }

        return true;
    }

    /// <summary>
    ///  Send the coupon newsletter to the subscribers in the contact list.
    /// </summary>
    /// <param name="fromEmailAddress">The verified email address from
PrepareApplication.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> SendCouponNewsletter(string fromEmailAddress)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("3. In this step, we will send the coupon newsletter:"
+
                          "\r\n  - Retrieve the list of contacts from the contact
list." +
                          "\r\n  - Send the coupon newsletter using the email
template to each contact.\r\n");


        // Retrieve the list of contacts from the contact list.
        var contacts = await _sesv2Wrapper.ListContactsAsync(_contactListName);
        if (!contacts.Any())
        {
            Console.WriteLine($"No contacts found in the {_contactListName}
contact list.");
            return false;
        }

        // Load the coupon data from the sample_coupons.json file.
        string couponsData = await
File.ReadAllTextAsync(_resourcesFilePathLocation + _couponsDataFile);
```

```
        // Send the coupon newsletter to each contact using the email template.
        try
        {
            foreach (var contact in contacts)
            {
                // To use the Contact List for list management, send to only one
address at a time.
                await _sesv2Wrapper.SendEmailAsync(fromEmailAddress,
                    new List<string> { contact.EmailAddress },
                    null, null, null, _templateName, couponsData,
_contactListName);
            }

            Console.WriteLine($"Coupon newsletter sent to contact list
{_contactListName}.");
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error sending coupon newsletter to contact list
{_contactListName}: {ex.Message}");
            return false;
        }

        return true;
    }

    /// <summary>
    /// Provide instructions for monitoring sending activity and metrics.
    /// </summary>
    /// <param name="interactive">True to run in interactive mode.</param>
    /// <returns>True if successful.</returns>
    public static bool MonitorAndReview(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("4. In step 4, we will monitor and review:" +
                          "\r\n  - Provide instructions for the user to review
the sending activity and metrics in the AWS console.\r\n");

        Console.WriteLine("Review your sending activity using the SES Homepage in
the AWS console.");
        Console.WriteLine("Press Enter to open the SES Homepage in your default
browser...");
        if (interactive)
```

```
        {
            Console.ReadLine();
            try
            {
                // Open the SES Homepage in the default browser.
                Process.Start(new ProcessStartInfo
                {
                    FileName = "https://console.aws.amazon.com/ses/home",
                    UseShellExecute = true
                });
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Error opening the SES Homepage:
 {ex.Message}");
                return false;
            }
        }

        Console.WriteLine("Review the sending activity and email metrics, then
 press Enter to continue...");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Clean up the resources used in the scenario.
    /// </summary>
    /// <param name="verifiedEmailAddress">The verified email address from
 PrepareApplication.</param>
    /// <param name="interactive">True if interactive.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Cleanup(string verifiedEmailAddress, bool
 interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("5. Finally, we clean up resources:" +
                          "\r\n  - Delete the contact list (which also deletes
 all contacts within it)." +
                          "\r\n  - Delete the email template." +
                          "\r\n  - Optionally delete the verified email identity.
\r\n");
```

```
        Console.WriteLine("Cleaning up resources...");

        // Delete the contact list (this also deletes all contacts in the list).
        try
        {
            await _sesv2Wrapper.DeleteContactListAsync(_contactListName);
            Console.WriteLine($"Contact list {_contactListName} deleted.");
        }
        catch (NotFoundException)
        {
            Console.WriteLine($"Contact list {_contactListName} not found.");
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error deleting contact list {_contactListName}:
{ex.Message}");
            return false;
        }

        // Delete the email template.
        try
        {
            await _sesv2Wrapper.DeleteEmailTemplateAsync(_templateName);
            Console.WriteLine($"Email template {_templateName} deleted.");
        }
        catch (NotFoundException)
        {
            Console.WriteLine($"Email template {_templateName} not found.");
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error deleting email template {_templateName}:
{ex.Message}");
            return false;
        }

        // Ask the user if they want to delete the email identity.
        var deleteIdentity = !interactive ||
            GetYesNoResponse(
                $"Do you want to delete the email identity
{verifiedEmailAddress}? (y/n) ");
        if (deleteIdentity)
        {
            try
```

```
            {
                await
 _sesv2Wrapper.DeleteEmailIdentityAsync(verifiedEmailAddress);
                Console.WriteLine($"Email identity {verifiedEmailAddress}
 deleted.");
            }
            catch (NotFoundException)
            {
                Console.WriteLine(
                    $"Email identity {verifiedEmailAddress} not found.");
            }
            catch (Exception ex)
            {
                Console.WriteLine(
                    $"Error deleting email identity {verifiedEmailAddress}:
 {ex.Message}");
                return false;
            }
        }
        else
        {
            Console.WriteLine(
                $"Skipping deletion of email identity {verifiedEmailAddress}.");
        }

        return true;
    }

    /// <summary>
    /// Helper method to get a yes or no response from the user.
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
    /// <returns>True if the user responds with a yes.</returns>
    private static bool GetYesNoResponse(string question)
    {
        Console.WriteLine(question);
        var ynResponse = Console.ReadLine();
        var response = ynResponse != null && ynResponse.Equals("y",
 StringComparison.InvariantCultureIgnoreCase);
        return response;
    }

    /// <summary>
```

```
    /// Simple check to verify a string is an email address.
    /// </summary>
    /// <param name="email">The string to verify.</param>
    /// <returns>True if a valid email.</returns>
    private static bool IsEmail(string? email)
    {
        if (string.IsNullOrEmpty(email))
            return false;
        return Regex.IsMatch(email, @"^[^@\s]+@[^@\s]+\.[^@\s]+$",
 RegexOptions.IgnoreCase);
    }
}
```

서비스 작업용 래퍼입니다.

```
using System.Net;
using Amazon.SimpleEmailV2;
using Amazon.SimpleEmailV2.Model;

namespace Sesv2Scenario;

/// <summary>
/// Wrapper class for Amazon Simple Email Service (SES) v2 operations.
/// </summary>
public class SESv2Wrapper
{

    private readonly IAmazonSimpleEmailServiceV2 _sesClient;

    /// <summary>
    /// Constructor for the SESv2Wrapper.
    /// </summary>
    /// <param name="sesClient">The injected SES v2 client.</param>
    public SESv2Wrapper(IAmazonSimpleEmailServiceV2 sesClient)
    {
        _sesClient = sesClient;
    }

    /// <summary>
    /// Creates a contact and adds it to the specified contact list.
    /// </summary>
    /// <param name="emailAddress">The email address of the contact.</param>
```

```
    /// <param name="contactListName">The name of the contact list.</param>
    /// <returns>The response from the CreateContact operation.</returns>
    public async Task<bool> CreateContactAsync(string emailAddress, string
contactListName)
    {
        var request = new CreateContactRequest
        {
            EmailAddress = emailAddress,
            ContactListName = contactListName
        };

        try
        {
            var response = await _sesClient.CreateContactAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (AlreadyExistsException ex)
        {
            Console.WriteLine($"Contact with email address {emailAddress} already
exists in the contact list {contactListName}.");
            Console.WriteLine(ex.Message);
            return true;
        }
        catch (NotFoundException ex)
        {
            Console.WriteLine($"The contact list {contactListName} does not
exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while creating the contact:
{ex.Message}");
        }
        return false;
    }

    /// <summary>
```

```
    /// Creates a contact list with the specified name.
    /// </summary>
    /// <param name="contactListName">The name of the contact list.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> CreateContactListAsync(string contactListName)
    {
        var request = new CreateContactListRequest
        {
            ContactListName = contactListName
        };

        try
        {
            var response = await _sesClient.CreateContactListAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (AlreadyExistsException ex)
        {
            Console.WriteLine($"Contact list with name {contactListName} already
exists.");
            Console.WriteLine(ex.Message);
            return true;
        }
        catch (LimitExceededException ex)
        {
            Console.WriteLine("The limit for contact lists has been exceeded.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while creating the contact
list: {ex.Message}");
        }
        return false;
    }

    /// <summary>
```

```
   /// Creates an email identity (email address or domain) and starts the
verification process.
   /// </summary>
   /// <param name="emailIdentity">The email address or domain to create and
verify.</param>
   /// <returns>The response from the CreateEmailIdentity operation.</returns>
   public async Task<CreateEmailIdentityResponse>
CreateEmailIdentityAsync(string emailIdentity)
   {
       var request = new CreateEmailIdentityRequest
       {
           EmailIdentity = emailIdentity
       };

       try
       {
           var response = await _sesClient.CreateEmailIdentityAsync(request);
           return response;
       }
       catch (AlreadyExistsException ex)
       {
           Console.WriteLine($"Email identity {emailIdentity} already exists.");
           Console.WriteLine(ex.Message);
           throw;
       }
       catch (ConcurrentModificationException ex)
       {
           Console.WriteLine($"The email identity {emailIdentity} is being
modified by another operation or thread.");
           Console.WriteLine(ex.Message);
           throw;
       }
       catch (LimitExceededException ex)
       {
           Console.WriteLine("The limit for email identities has been
exceeded.");
           Console.WriteLine(ex.Message);
           throw;
       }
       catch (NotFoundException ex)
       {
           Console.WriteLine($"The email identity {emailIdentity} does not
exist.");
           Console.WriteLine(ex.Message);
```

```
            throw;
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while creating the email
identity: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Creates an email template with the specified content.
    /// </summary>
    /// <param name="templateName">The name of the email template.</param>
    /// <param name="subject">The subject of the email template.</param>
    /// <param name="htmlContent">The HTML content of the email template.</param>
    /// <param name="textContent">The text content of the email template.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> CreateEmailTemplateAsync(string templateName, string
subject, string htmlContent, string textContent)
    {
        var request = new CreateEmailTemplateRequest
        {
            TemplateName = templateName,
            TemplateContent = new EmailTemplateContent
            {
                Subject = subject,
                Html = htmlContent,
                Text = textContent
            }
        };

        try
        {
            var response = await _sesClient.CreateEmailTemplateAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
```

```
        catch (AlreadyExistsException ex)
        {
            Console.WriteLine($"Email template with name {templateName} already
 exists.");
            Console.WriteLine(ex.Message);
        }
        catch (LimitExceededException ex)
        {
            Console.WriteLine("The limit for email templates has been
 exceeded.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
 later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while creating the email
 template: {ex.Message}");
        }

        return false;
    }

    /// <summary>
    /// Deletes a contact list and all contacts within it.
    /// </summary>
    /// <param name="contactListName">The name of the contact list to delete.</
param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteContactListAsync(string contactListName)
    {
        var request = new DeleteContactListRequest
        {
            ContactListName = contactListName
        };

        try
        {
            var response = await _sesClient.DeleteContactListAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
```

```
        }
        catch (ConcurrentModificationException ex)
        {
            Console.WriteLine($"The contact list {contactListName} is being
 modified by another operation or thread.");
            Console.WriteLine(ex.Message);
        }
        catch (NotFoundException ex)
        {
            Console.WriteLine($"The contact list {contactListName} does not
 exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
 later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while deleting the contact
 list: {ex.Message}");
        }

        return false;
    }

    /// <summary>
    /// Deletes an email identity (email address or domain).
    /// </summary>
    /// <param name="emailIdentity">The email address or domain to delete.</
param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteEmailIdentityAsync(string emailIdentity)
    {
        var request = new DeleteEmailIdentityRequest
        {
            EmailIdentity = emailIdentity
        };

        try
        {
            var response = await _sesClient.DeleteEmailIdentityAsync(request);
```

```
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (ConcurrentModificationException ex)
        {
            Console.WriteLine($"The email identity {emailIdentity} is being
 modified by another operation or thread.");
            Console.WriteLine(ex.Message);
        }
        catch (NotFoundException ex)
        {
            Console.WriteLine($"The email identity {emailIdentity} does not
 exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
 later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while deleting the email
 identity: {ex.Message}");
        }

        return false;
    }

    /// <summary>
    /// Deletes an email template.
    /// </summary>
    /// <param name="templateName">The name of the email template to delete.</
param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteEmailTemplateAsync(string templateName)
    {
        var request = new DeleteEmailTemplateRequest
        {
            TemplateName = templateName
        };

        try
        {
```

```
            var response = await _sesClient.DeleteEmailTemplateAsync(request);
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (NotFoundException ex)
        {
            Console.WriteLine($"The email template {templateName} does not
 exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
 later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while deleting the email
 template: {ex.Message}");
        }

        return false;
    }

    /// <summary>
    /// Lists the contacts in the specified contact list.
    /// </summary>
    /// <param name="contactListName">The name of the contact list.</param>
    /// <returns>The list of contacts response from the ListContacts operation.</
returns>
    public async Task<List<Contact>> ListContactsAsync(string contactListName)
    {
        var request = new ListContactsRequest
        {
            ContactListName = contactListName
        };

        try
        {
            var response = await _sesClient.ListContactsAsync(request);
            return response.Contacts;
        }
        catch (NotFoundException ex)
        {
```

```
            Console.WriteLine($"The contact list {contactListName} does not
exist.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while listing the contacts:
{ex.Message}");
        }

        return new List<Contact>();
    }

    /// <summary>
    /// Sends an email with the specified content and options.
    /// </summary>
    /// <param name="fromEmailAddress">The email address to send the email
from.</param>
    /// <param name="toEmailAddresses">The email addresses to send the email
to.</param>
    /// <param name="subject">The subject of the email.</param>
    /// <param name="htmlContent">The HTML content of the email.</param>
    /// <param name="textContent">The text content of the email.</param>
    /// <param name="templateName">The name of the email template to use
(optional).</param>
    /// <param name="templateData">The data to replace placeholders in the email
template (optional).</param>
    /// <param name="contactListName">The name of the contact list for
unsubscribe functionality (optional).</param>
    /// <returns>The MessageId response from the SendEmail operation.</returns>
    public async Task<string> SendEmailAsync(string fromEmailAddress,
List<string> toEmailAddresses, string? subject,
        string? htmlContent, string? textContent, string? templateName = null,
string? templateData = null, string? contactListName = null)
    {
        var request = new SendEmailRequest
        {
            FromEmailAddress = fromEmailAddress
```

```
        };

        if (toEmailAddresses.Any())
        {
            request.Destination = new Destination { ToAddresses =
toEmailAddresses };
        }

        if (!string.IsNullOrEmpty(templateName))
        {
            request.Content = new EmailContent()
            {
                Template = new Template
                {
                    TemplateName = templateName,
                    TemplateData = templateData
                }
            };
        }
        else
        {
            request.Content = new EmailContent
            {
                Simple = new Message
                {
                    Subject = new Content { Data = subject },
                    Body = new Body
                    {
                        Html = new Content { Data = htmlContent },
                        Text = new Content { Data = textContent }
                    }
                }
            };
        }

        if (!string.IsNullOrEmpty(contactListName))
        {
            request.ListManagementOptions = new ListManagementOptions
            {
                ContactListName = contactListName
            };
        }

        try
```

```
        {
            var response = await _sesClient.SendEmailAsync(request);
            return response.MessageId;
        }
        catch (AccountSuspendedException ex)
        {
            Console.WriteLine("The account's ability to send email has been
 permanently restricted.");
            Console.WriteLine(ex.Message);
        }
        catch (MailFromDomainNotVerifiedException ex)
        {
            Console.WriteLine("The sending domain is not verified.");
            Console.WriteLine(ex.Message);
        }
        catch (MessageRejectedException ex)
        {
            Console.WriteLine("The message content is invalid.");
            Console.WriteLine(ex.Message);
        }
        catch (SendingPausedException ex)
        {
            Console.WriteLine("The account's ability to send email is currently
 paused.");
            Console.WriteLine(ex.Message);
        }
        catch (TooManyRequestsException ex)
        {
            Console.WriteLine("Too many requests were made. Please try again
 later.");
            Console.WriteLine(ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred while sending the email:
 {ex.Message}");
        }

        return string.Empty;
    }
}
```

- API 세부 정보는 AWS SDK for .NET API 참조의 다음 주제를 참조하세요.

- CreateContact
- CreateContactList
- CreateEmailIdentity
- CreateEmailTemplate
- DeleteContactList
- DeleteEmailIdentity
- DeleteEmailTemplate
- ListContacts
- SendEmail.simple
- SendEmail.template

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    try {
      // 2. Create a contact list
      String contactListName = CONTACT_LIST_NAME;
      CreateContactListRequest createContactListRequest =
 CreateContactListRequest.builder()
          .contactListName(contactListName)
          .build();
      sesClient.createContactList(createContactListRequest);
      System.out.println("Contact list created: " + contactListName);
    } catch (AlreadyExistsException e) {
      System.out.println("Contact list already exists, skipping creation: weekly-
coupons-newsletter");
    } catch (LimitExceededException e) {
      System.err.println("Limit for contact lists has been exceeded.");
      throw e;
    } catch (SesV2Exception e) {
```

```
      System.err.println("Error creating contact list: " + e.getMessage());
      throw e;
    }

      try {
        // Create a new contact with the provided email address in the
        CreateContactRequest contactRequest = CreateContactRequest.builder()
            .contactListName(CONTACT_LIST_NAME)
            .emailAddress(emailAddress)
            .build();

        sesClient.createContact(contactRequest);
        contacts.add(emailAddress);

        System.out.println("Contact created: " + emailAddress);

        // Send a welcome email to the new contact
        String welcomeHtml = Files.readString(Paths.get("resources/
coupon_newsletter/welcome.html"));
        String welcomeText = Files.readString(Paths.get("resources/
coupon_newsletter/welcome.txt"));

        SendEmailRequest welcomeEmailRequest = SendEmailRequest.builder()
            .fromEmailAddress(this.verifiedEmail)
            .destination(Destination.builder().toAddresses(emailAddress).build())
            .content(EmailContent.builder()
                .simple(
                    Message.builder()
                        .subject(Content.builder().data("Welcome to the Weekly
 Coupons Newsletter").build())
                        .body(Body.builder()
                            .text(Content.builder().data(welcomeText).build())
                            .html(Content.builder().data(welcomeHtml).build())
                            .build())
                        .build())
                .build())
            .build();
        SendEmailResponse welcomeEmailResponse =
 sesClient.sendEmail(welcomeEmailRequest);
        System.out.println("Welcome email sent: " +
 welcomeEmailResponse.messageId());
      } catch (AlreadyExistsException e) {
        // If the contact already exists, skip this step for that contact and
 proceed
```

```
        // with the next contact
        System.out.println("Contact already exists, skipping creation...");
      } catch (Exception e) {
        System.err.println("Error occurred while processing email address " +
 emailAddress + ": " + e.getMessage());
        throw e;
      }
    }

      ListContactsRequest contactListRequest = ListContactsRequest.builder()
          .contactListName(CONTACT_LIST_NAME)
          .build();

      List<String> contactEmails;
      try {
        ListContactsResponse contactListResponse =
 sesClient.listContacts(contactListRequest);

        contactEmails = contactListResponse.contacts().stream()
            .map(Contact::emailAddress)
            .toList();
      } catch (Exception e) {
        // TODO: Remove when listContacts's GET body issue is resolved.
        contactEmails = this.contacts;
      }


      String coupons = Files.readString(Paths.get("resources/coupon_newsletter/
sample_coupons.json"));
      for (String emailAddress : contactEmails) {
        SendEmailRequest newsletterRequest = SendEmailRequest.builder()
            .destination(Destination.builder().toAddresses(emailAddress).build())
            .content(EmailContent.builder()
                .template(Template.builder()
                    .templateName(TEMPLATE_NAME)
                    .templateData(coupons)
                    .build())
                .build())
            .fromEmailAddress(this.verifiedEmail)
            .listManagementOptions(ListManagementOptions.builder()
                .contactListName(CONTACT_LIST_NAME)
                .build())
            .build();
```

```
        SendEmailResponse newsletterResponse =
 sesClient.sendEmail(newsletterRequest);
        System.out.println("Newsletter sent to " + emailAddress + ": " +
 newsletterResponse.messageId());
      }

    try {
      CreateEmailIdentityRequest createEmailIdentityRequest =
 CreateEmailIdentityRequest.builder()
          .emailIdentity(verifiedEmail)
          .build();
      sesClient.createEmailIdentity(createEmailIdentityRequest);
      System.out.println("Email identity created: " + verifiedEmail);
    } catch (AlreadyExistsException e) {
      System.out.println("Email identity already exists, skipping creation: " +
 verifiedEmail);
    } catch (NotFoundException e) {
      System.err.println("The provided email address is not verified: " +
 verifiedEmail);
      throw e;
    } catch (LimitExceededException e) {
      System.err
          .println("You have reached the limit for email identities. Please
 remove some identities and try again.");
      throw e;
    } catch (SesV2Exception e) {
      System.err.println("Error creating email identity: " + e.getMessage());
      throw e;
    }

    try {
      // Create an email template named "weekly-coupons"
      String newsletterHtml = loadFile("resources/coupon_newsletter/coupon-
newsletter.html");
      String newsletterText = loadFile("resources/coupon_newsletter/coupon-
newsletter.txt");

      CreateEmailTemplateRequest templateRequest =
 CreateEmailTemplateRequest.builder()
          .templateName(TEMPLATE_NAME)
          .templateContent(EmailTemplateContent.builder()
              .subject("Weekly Coupons Newsletter")
              .html(newsletterHtml)
              .text(newsletterText)
```

```
                .build())
            .build();

        sesClient.createEmailTemplate(templateRequest);

        System.out.println("Email template created: " + TEMPLATE_NAME);
      } catch (AlreadyExistsException e) {
        // If the template already exists, skip this step and proceed with the next
        // operation
        System.out.println("Email template already exists, skipping creation...");
      } catch (LimitExceededException e) {
        // If the limit for email templates is exceeded, fail the workflow and
inform
        // the user
        System.err.println("You have reached the limit for email templates. Please
remove some templates and try again.");
        throw e;
      } catch (Exception e) {
        System.err.println("Error occurred while creating email template: " +
e.getMessage());
        throw e;
      }

      try {
        // Delete the contact list
        DeleteContactListRequest deleteContactListRequest =
DeleteContactListRequest.builder()
            .contactListName(CONTACT_LIST_NAME)
            .build();

        sesClient.deleteContactList(deleteContactListRequest);

        System.out.println("Contact list deleted: " + CONTACT_LIST_NAME);
      } catch (NotFoundException e) {
        // If the contact list does not exist, log the error and proceed
        System.out.println("Contact list not found. Skipping deletion...");
      } catch (Exception e) {
        System.err.println("Error occurred while deleting the contact list: " +
e.getMessage());
        e.printStackTrace();
      }

        try {
          // Delete the email identity
```

```
        DeleteEmailIdentityRequest deleteIdentityRequest =
DeleteEmailIdentityRequest.builder()
            .emailIdentity(this.verifiedEmail)
            .build();

        sesClient.deleteEmailIdentity(deleteIdentityRequest);

        System.out.println("Email identity deleted: " + this.verifiedEmail);
      } catch (NotFoundException e) {
        // If the email identity does not exist, log the error and proceed
        System.out.println("Email identity not found. Skipping deletion...");
      } catch (Exception e) {
        System.err.println("Error occurred while deleting the email identity: " +
e.getMessage());
        e.printStackTrace();
      }
    } else {
      System.out.println("Skipping email identity deletion.");
    }

    try {
      // Delete the template
      DeleteEmailTemplateRequest deleteTemplateRequest =
DeleteEmailTemplateRequest.builder()
          .templateName(TEMPLATE_NAME)
          .build();

      sesClient.deleteEmailTemplate(deleteTemplateRequest);

      System.out.println("Email template deleted: " + TEMPLATE_NAME);
    } catch (NotFoundException e) {
      // If the email template does not exist, log the error and proceed
      System.out.println("Email template not found. Skipping deletion...");
    } catch (Exception e) {
      System.err.println("Error occurred while deleting the email template: " +
e.getMessage());
      e.printStackTrace();
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 다음 항목을 참조하세요.
  - CreateContact
  - CreateContactList

- CreateEmailIdentity
- CreateEmailTemplate
- DeleteContactList
- DeleteEmailIdentity
- DeleteEmailTemplate
- ListContacts
- SendEmail.simple
- SendEmail.template

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
def main():
    """
    The main function that orchestrates the execution of the workflow.
    """
    print(INTRO)
    ses_client = boto3.client("sesv2")
    workflow = SESv2Workflow(ses_client)
    try:
        workflow.prepare_application()
        workflow.gather_subscriber_email_addresses()
        workflow.send_coupon_newsletter()
        workflow.monitor_and_review()
    except ClientError as e:
        print_error(e)
    workflow.clean_up()



class SESv2Workflow:
```

```
    """
    A class to manage the SES v2 Coupon Newsletter Workflow.
    """

    def __init__(self, ses_client, sleep=True):
        self.ses_client = ses_client
        self.sleep = sleep


        try:

self.ses_client.create_contact_list(ContactListName=CONTACT_LIST_NAME)
            print(f"Contact list '{CONTACT_LIST_NAME}' created successfully.")
        except ClientError as e:
            # If the contact list already exists, skip and proceed
            if e.response["Error"]["Code"] == "AlreadyExistsException":
                print(f"Contact list '{CONTACT_LIST_NAME}' already exists.")
            else:
                raise e

            try:
                # Create a new contact
                self.ses_client.create_contact(
                    ContactListName=CONTACT_LIST_NAME, EmailAddress=email
                )
                print(f"Contact with email '{email}' created successfully.")

                # Send the welcome email
                self.ses_client.send_email(
                    FromEmailAddress=self.verified_email,
                    Destination={"ToAddresses": [email]},
                    Content={
                        "Simple": {
                            "Subject": {
                                "Data": "Welcome to the Weekly Coupons
Newsletter"
                            },
                            "Body": {
                                "Text": {"Data": welcome_text},
                                "Html": {"Data": welcome_html},
                            },
                        }
                    },
                )
```

```
                print(f"Welcome email sent to '{email}'.")
                if self.sleep:
                    # 1 email per second in sandbox mode, remove in production.
                    sleep(1.1)
            except ClientError as e:
                # If the contact already exists, skip and proceed
                if e.response["Error"]["Code"] == "AlreadyExistsException":
                    print(f"Contact with email '{email}' already exists.
Skipping...")
                else:
                    raise e

        try:
            contacts_response = self.ses_client.list_contacts(
                ContactListName=CONTACT_LIST_NAME
            )
        except ClientError as e:
            if e.response["Error"]["Code"] == "NotFoundException":
                print(f"Contact list '{CONTACT_LIST_NAME}' does not exist.")
                return
            else:
                raise e

                self.ses_client.send_email(
                    FromEmailAddress=self.verified_email,
                    Destination={"ToAddresses": [email]},
                    Content={
                        "Simple": {
                            "Subject": {
                                "Data": "Welcome to the Weekly Coupons
Newsletter"
                            },
                            "Body": {
                                "Text": {"Data": welcome_text},
                                "Html": {"Data": welcome_html},
                            },
                        }
                    },
                )
                print(f"Welcome email sent to '{email}'.")

                self.ses_client.send_email(
                    FromEmailAddress=self.verified_email,
                    Destination={"ToAddresses": [email_address]},
```

```
                    Content={
                        "Template": {
                            "TemplateName": TEMPLATE_NAME,
                            "TemplateData": coupon_items,
                        }
                    },
                    ListManagementOptions={"ContactListName": CONTACT_LIST_NAME},
                )

        try:

self.ses_client.create_email_identity(EmailIdentity=self.verified_email)
            print(f"Email identity '{self.verified_email}' created
successfully.")
        except ClientError as e:
            # If the email identity already exists, skip and proceed
            if e.response["Error"]["Code"] == "AlreadyExistsException":
                print(f"Email identity '{self.verified_email}' already exists.")
            else:
                raise e

        try:
            template_content = {
                "Subject": "Weekly Coupons Newsletter",
                "Html": load_file_content("coupon-newsletter.html"),
                "Text": load_file_content("coupon-newsletter.txt"),
            }
            self.ses_client.create_email_template(
                TemplateName=TEMPLATE_NAME, TemplateContent=template_content
            )
            print(f"Email template '{TEMPLATE_NAME}' created successfully.")
        except ClientError as e:
            # If the template already exists, skip and proceed
            if e.response["Error"]["Code"] == "AlreadyExistsException":
                print(f"Email template '{TEMPLATE_NAME}' already exists.")
            else:
                raise e

        try:

self.ses_client.delete_contact_list(ContactListName=CONTACT_LIST_NAME)
            print(f"Contact list '{CONTACT_LIST_NAME}' deleted successfully.")
        except ClientError as e:
            # If the contact list doesn't exist, skip and proceed
```

```
            if e.response["Error"]["Code"] == "NotFoundException":
                print(f"Contact list '{CONTACT_LIST_NAME}' does not exist.")
            else:
                print(e)

            try:

self.ses_client.delete_email_identity(EmailIdentity=self.verified_email)
                print(f"Email identity '{self.verified_email}' deleted
successfully.")
            except ClientError as e:
                # If the email identity doesn't exist, skip and proceed
                if e.response["Error"]["Code"] == "NotFoundException":
                    print(f"Email identity '{self.verified_email}' does not
exist.")
                else:
                    print(e)

        try:
            self.ses_client.delete_email_template(TemplateName=TEMPLATE_NAME)
            print(f"Email template '{TEMPLATE_NAME}' deleted successfully.")
        except ClientError as e:
            # If the email template doesn't exist, skip and proceed
            if e.response["Error"]["Code"] == "NotFoundException":
                print(f"Email template '{TEMPLATE_NAME}' does not exist.")
            else:
                print(e)
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 다음 주제를 참조하세요.
  - CreateContact
  - CreateContactList
  - CreateEmailIdentity
  - CreateEmailTemplate
  - DeleteContactList
  - DeleteEmailIdentity
  - DeleteEmailTemplate
  - ListContacts
  - SendEmail.simple

- SendEmail.template

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예 리포지토리에서 전체 예를 찾고 설정 및 실행하는 방법을 배워보세요.

```
        match self
            .client
            .create_contact_list()
            .contact_list_name(CONTACT_LIST_NAME)
            .send()
            .await
        {
            Ok(_) => writeln!(self.stdout, "Contact list created
successfully.")?,
            Err(e) => match e.into_service_error() {
                CreateContactListError::AlreadyExistsException(_) => {
                    writeln!(
                        self.stdout,
                        "Contact list already exists, skipping creation."
                    )?;
                }
                e => return Err(anyhow!("Error creating contact list: {}", e)),
            },
        }

            match self
                .client
                .create_contact()
                .contact_list_name(CONTACT_LIST_NAME)
                .email_address(email.clone())
                .send()
                .await
            {
                Ok(_) => writeln!(self.stdout, "Contact created for {}", email)?,
```

```
                Err(e) => match e.into_service_error() {
                    CreateContactError::AlreadyExistsException(_) => writeln!(
                        self.stdout,
                        "Contact already exists for {}, skipping creation.",
                        email
                    )?,
                    e => return Err(anyhow!("Error creating contact for {}: {}",
 email, e)),
                },
            }

        let contacts: Vec<Contact> = match self
            .client
            .list_contacts()
            .contact_list_name(CONTACT_LIST_NAME)
            .send()
            .await
        {
            Ok(list_contacts_output) => {
                list_contacts_output.contacts.unwrap().into_iter().collect()
            }
            Err(e) => {
                return Err(anyhow!(
                    "Error retrieving contact list {}: {}",
                    CONTACT_LIST_NAME,
                    e
                ))
            }
        };

            let coupons = std::fs::read_to_string("../resources/newsletter/
sample_coupons.json")
                .unwrap_or_else(|_| r#"{"coupons":[]}"#.to_string());
            let email_content = EmailContent::builder()
                .template(
                    Template::builder()
                        .template_name(TEMPLATE_NAME)
                        .template_data(coupons)
                        .build(),
                )
                .build();

            match self
                .client
```

```
                .send_email()
                .from_email_address(self.verified_email.clone())

.destination(Destination::builder().to_addresses(email.clone()).build())
                .content(email_content)
                .list_management_options(
                    ListManagementOptions::builder()
                        .contact_list_name(CONTACT_LIST_NAME)
                        .build()?,
                )
                .send()
                .await
            {
                Ok(output) => {
                    if let Some(message_id) = output.message_id {
                        writeln!(
                            self.stdout,
                            "Newsletter sent to {} with message ID {}",
                            email, message_id
                        )?;
                    } else {
                        writeln!(self.stdout, "Newsletter sent to {}", email)?;
                    }
                }
                Err(e) => return Err(anyhow!("Error sending newsletter to {}:
{}", email, e)),
            }

        match self
            .client
            .create_email_identity()
            .email_identity(self.verified_email.clone())
            .send()
            .await
        {
            Ok(_) => writeln!(self.stdout, "Email identity created
successfully.")?,
            Err(e) => match e.into_service_error() {
                CreateEmailIdentityError::AlreadyExistsException(_) => {
                    writeln!(
                        self.stdout,
                        "Email identity already exists, skipping creation."
                    )?;
                }
```

```
                e => return Err(anyhow!("Error creating email identity: {}", e)),
            },
        }

        let template_html =
            std::fs::read_to_string("../resources/newsletter/coupon-
newsletter.html")
                .unwrap_or_else(|_| "Missing coupon-
newsletter.html".to_string());
        let template_text =
            std::fs::read_to_string("../resources/newsletter/coupon-
newsletter.txt")
                .unwrap_or_else(|_| "Missing coupon-newsletter.txt".to_string());

        // Create the email template
        let template_content = EmailTemplateContent::builder()
            .subject("Weekly Coupons Newsletter")
            .html(template_html)
            .text(template_text)
            .build();

        match self
            .client
            .create_email_template()
            .template_name(TEMPLATE_NAME)
            .template_content(template_content)
            .send()
            .await
        {
            Ok(_) => writeln!(self.stdout, "Email template created
 successfully.")?,
            Err(e) => match e.into_service_error() {
                CreateEmailTemplateError::AlreadyExistsException(_) => {
                    writeln!(
                        self.stdout,
                        "Email template already exists, skipping creation."
                    )?;
                }
                e => return Err(anyhow!("Error creating email template: {}", e)),
            },
        }

        match self
            .client
```

```
            .delete_contact_list()
            .contact_list_name(CONTACT_LIST_NAME)
            .send()
            .await
        {
            Ok(_) => writeln!(self.stdout, "Contact list deleted
successfully.")?,
            Err(e) => return Err(anyhow!("Error deleting contact list: {e}")),
        }

            match self
                .client
                .delete_email_identity()
                .email_identity(self.verified_email.clone())
                .send()
                .await
            {
                Ok(_) => writeln!(self.stdout, "Email identity deleted
successfully.")?,
                Err(e) => {
                    return Err(anyhow!("Error deleting email identity: {}", e));
                }
            }

        match self
            .client
            .delete_email_template()
            .template_name(TEMPLATE_NAME)
            .send()
            .await
        {
            Ok(_) => writeln!(self.stdout, "Email template deleted
successfully.")?,
            Err(e) => {
                return Err(anyhow!("Error deleting email template: {e}"));
            }
        }
```

- API 세부 정보는 AWS SDK for Rust API 참조의 다음 주제를 참조하세요.
  - CreateContact
  - CreateContactList

- CreateEmailIdentity
- CreateEmailTemplate
- DeleteContactList
- DeleteEmailIdentity
- DeleteEmailTemplate
- ListContacts
- SendEmail.simple
- SendEmail.template

AWS SDK 개발자 안내서 및 코드 예제의 전체 목록은 섹션을 참조하세요AWS SDK에서 Amazon SES 사용. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

Amazon Simple Email Service의 보안

의 클라우드 보안 AWS 이 최우선 순위입니다. AWS 고객은 보안에 가장 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 이점을 누릴 수 있습니다.

보안은 AWS 와 사용자 간의 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 는 AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호할 책임이 있습니다. AWS 또한는 안전하게 사용할 수 있는 서비스를 제공합니다. 타사 감사자는 AWS 규정 준수 프로그램 일환으로 보안의 효과를 정기적으로 테스트하고 확인합니다. Amazon Simple Email Service에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 AWS 프로그램 제공 범위 내 서비스규정 준수 서비스를 참조하세요.
- 클라우드의 보안 - 사용자의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 여러분은 데이터의 민감도, 회사 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다

이 설명서는 Amazon Simple Email Service를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 보안 및 규정 준수 목표에 맞게 Amazon Simple Email Service를 구성하는 방법을 보여줍니다. 또한 Amazon Simple Email Service 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법을 알아봅니다.

> **Note**
>
> 이메일 스팸 및 맬웨어 배포를 포함한 AWS 리소스 침해를 보고해야 하는 경우 AWS Trust & Safety가 아닌 AWS 설명서 팀에서 양식을 수신하므로이 개발자 안내서의 페이지에 있는 피드백 링크를 사용하지 마세요. 대신 AWS 리소스 침해를 신고하려면 어떻게 해야 하나요? 페이지에서 지침에 따라 신뢰 및 안전 팀에 문의하여 AWS 모든 유형의 Amazon AWS 침해를 신고하세요.

내용

- Amazon Simple Email Service 데이터 보호
- Amazon SES의 Identity and Access Management
- Amazon SES의 로깅 및 모니터링
- Amazon Simple Email Service에 대한 규정 준수 확인
- Amazon Simple Email Service의 복원성

- Amazon Simple Email Service의 인프라 보안
- Amazon SES를 사용하여 VPC 엔드포인트 설정

Amazon Simple Email Service 데이터 보호

AWS 공동 책임 모델 Amazon Simple Email Service의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로 AWS 는 모든를 실행하는 글로벌 인프라를 보호할 책임이 있습니다 AWS 클라우드. 사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 사용하는 AWS 서비스 의 보안 구성과 관리 태스크에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 자격 증명을 보호하고 AWS 계정 AWS IAM Identity Center 또는 AWS Identity and Access Management (IAM)를 사용하여 개별 사용자를 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 사용하세요.
- SSL/TLS를 사용하여 AWS 리소스와 통신합니다. TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다 AWS CloudTrail. CloudTrail 추적을 사용하여 AWS 활동을 캡처하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail 추적 작업을 참조하세요.
- 내부의 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용합니다 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하세요.
- 명령줄 인터페이스 또는 API를 AWS 통해에 액세스할 때 FIPS 140-3 검증 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 Federal Information Processing Standard(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 Amazon Simple Email Service 또는 기타 AWS 서비스 에서 콘솔 AWS CLI, API 또는 AWS SDKs를 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.

내용

- Amazon SES에서 저장 데이터 암호화
- 전송 중 암호화
- Amazon SES에서 개인 데이터 삭제

Amazon SES에서 저장 데이터 암호화

기본적으로 Amazon SES는 모든 저장 데이터를 암호화합니다. 기본적으로 암호화를 사용하면 데이터를 보호하는 데 수반되는 운영 오버헤드와 복잡성을 줄이는 데 도움이 됩니다. 또한 암호화를 사용하면 엄격한 암호화 규정 준수 및 규제 요구 사항을 충족하는 Mail Manager 아카이브를 만들 수 있습니다.

SES는 다음의 암호화 옵션을 제공합니다.

- AWS 소유 키 - SES는 기본적으로 이를 사용합니다. AWS 소유 키를 보거나 관리하거나 사용하거나 사용을 감사할 수 없습니다. 하지만 데이터를 암호화하는 키를 보호하기 위해 어떤 작업을 수행하거나 어떤 프로그램을 변경할 필요가 없습니다. 자세한 내용은 AWS Key Management Service 개발자 안내서의 AWS 소유 키를 참조하세요.
- 고객 관리형 키 – SES는 사용자가 만들고 소유하고 관리하는 대칭형 고객 관리형 키를 사용하도록 지원합니다. 암호화를 완전히 제어할 수 있으므로 다음과 같은 태스크를 수행할 수 있습니다.
  - 키 정책 수립 및 유지
  - IAM 정책 및 권한 부여 수립 및 유지
  - 키 정책 활성화 및 비활성화
  - 키 암호화 자료 교체
  - 태그 추가
  - 키 별칭 만들기
  - 삭제를 위한 스케줄 키

  자체 키를 사용하려면 SES 리소스 생성 시 고객 관리형 키를 선택합니다.

  자세한 내용은 AWS Key Management Service 개발자 안내서의 고객 관리형 키를 참조하세요.

> **Note**
>
> SES는 AWS 소유 키를 사용하여 저장 시 암호화를 무료로 자동으로 활성화합니다.

그러나 고객 관리형 키 사용에는 AWS KMS 요금이 적용됩니다. 요금에 대한 자세한 내용은 AWS Key Management Service 요금을 참조하십시오.

고객 관리형 키 만들기

AWS Management Console또는 AWS KMS APIs.

대칭 고객 관리형 키를 만들려면

AWS Key Management Service 개발자 안내서의 대칭 암호화 KMS 키 만들기의 단계를 따르세요.

> **Note**
>
> 아카이빙하려면 키에서 다음 요구 사항을 충족해야 합니다.
>
> - 키는 대칭이어야 합니다.
> - 키 구성 요소 오리진은 `AWS_KMS`여야 합니다.
> - 키 사용은 `ENCRYPT_DECRYPT`여야 합니다.

키 정책

키 정책에서는 고객 관리형 키에 대한 액세스를 제어합니다. 모든 고객 관리형 키에는 키를 사용할 수 있는 사람과 키를 사용하는 방법을 결정하는 문장이 포함된 정확히 하나의 키 정책이 있어야 합니다. 고객 관리형 키를 만들 때 키 정책을 지정할 수 있습니다. 자세한 내용은 AWS Key Management Service 개발자 안내서의 고객 관리형 키에 대한 액세스 관리를 참조하세요.

Mail Manager 아카이빙에서 고객 관리형 키를 사용하려면 키 정책에서 다음 API 작업을 허용해야 합니다.

- kms:DescribeKey - SES가 키를 검증할 수 있도록 고객 관리형 키 세부 정보를 제공합니다.
- kms:GenerateDataKey – SES가 저장 데이터를 암호화하기 위한 데이터 키를 만들 수 있도록 허용합니다.
- kms:Decrypt – SES가 저장된 데이터를 API 클라이언트로 반환하기 전에 복호화할 수 있도록 허용합니다.

다음 예제는 일반적인 키 정책을 보여줍니다.

```
{
            "Sid": "Allow SES to encrypt/decrypt",
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
```

자세한 내용은 AWS Key Management Service 개발자 안내서의 정책에서의 권한 지정을 참조하세요.

문제 해결에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 키 액세스 문제 해결을 참조하세요.

Mail Manager 아카이빙을 위한 고객 관리형 키 지정

AWS 소유 키를 사용하는 대신 고객 관리형 키를 지정할 수 있습니다. 아카이브를 만들 때 KMS 키 ARN을 입력하여 데이터 키를 지정할 수 있습니다. 이는 Mail Manager 아카이빙에서 아카이브의 모든 고객 데이터를 암호화하는 데 사용됩니다.

- KMS 키 ARN - AWS KMS 고객 관리형 키의 키 식별자입니다. 키 ID, 키 ARN, 별칭 이름 또는 별칭 ARN을 입력합니다.

Amazon SES 암호화 컨텍스트

암호화 컨텍스트는 데이터에 대한 추가 컨텍스트 정보를 포함하는 선택적 키-값 페어 세트입니다.

AWS KMS 는 암호화 컨텍스트를 추가 인증 데이터로 사용하여 인증된 암호화를 지원합니다. 데이터 암호화 요청에 암호화 컨텍스트를 포함하면는 암호화 컨텍스트를 암호화된 데이터에 AWS KMS 바인딩합니다. 요청에 동일한 암호화 컨텍스트를 포함해야 이 데이터를 해독할 수 있습니다.

> **Note**
>
> Amazon SES는 아카이브를 만들기 위한 암호화 컨텍스트를 지원하지 않습니다. 대신 IAM 또는 KMS 정책을 사용할 수 있습니다. 예제 정책을 확인하려면 이 섹션 후반부의 아카이브 만들기 정책의 내용을 참조하세요.

Amazon SES 암호화 컨텍스트

SES는 모든 암호화 작업에서 동일한 암호화 컨텍스트를 사용합니다. 여기서 키는 AWS KMS `aws:ses:arn`이고 값은 리소스 Amazon 리소스 이름(ARN)입니다.

Example

```
"encryptionContext": {
    "aws:ses:arn": "arn:aws:ses:us-west-2:111122223333:ExampleResourceName/
ExampleResourceID"
}
```

모니터링을 위한 암호화 컨텍스트 사용

대칭형 고객 관리형 키를 사용하여 SES 리소스를 암호화하는 경우 감사 레코드 및 로그의 암호화 컨텍스트를 사용하여 고객 관리형 키가 사용되는 방식을 식별할 수도 있습니다. 암호화 컨텍스트는 AWS CloudTrail 또는 Amazon CloudWatch Logs에서 생성된 로그에도 나타납니다.

암호화 컨텍스트를 사용하여 고객 관리형 키에 대한 액세스 제어

그러나 암호화 컨텍스트를 사용하여 키 정책 및 IAM 정책에서 대칭 `conditions`에 대한 액세스를 제어할 수도 있습니다. 또한 권한 부여에서 암호화 컨텍스트 제약 조건을 사용할 수 있습니다.

SES는 권한 부여에서 암호화 컨텍스트 제약 조건을 사용하여 계정 및 리전에서 고객 관리형 키에 대한 액세스를 제어합니다. 권한 부여 제약 조건에 따라 권한 부여가 허용하는 작업은 지정된 암호화 컨텍스트를 사용해야 합니다.

Example

다음은 특정 암호화 컨텍스트에서 고객 관리형 키에 대한 액세스 권한을 부여하는 키 정책 설명의 예입니다. 이 정책 설명의 조건에 따라 권한 부여에는 암호화 컨텍스트를 지정하는 암호화 컨텍스트 제약 조건이 있어야 합니다.

```
{
    "Sid": "Enable DescribeKey",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ExampleReadOnlyRole"
     },
     "Action": "kms:DescribeKey",
     "Resource": "*"
},
{
     "Sid": "Enable CreateGrant",
     "Effect": "Allow",
     "Principal": {
         "AWS": "arn:aws:iam::111122223333:role/ExampleReadOnlyRole"
     },
     "Action": "kms:CreateGrant",
     "Resource": "*",
     "Condition": {
         "StringEquals": {
             "kms:EncryptionContext:aws:ses:arn": "arn:aws:ses:us-
west-2:111122223333:ExampleResourceName/ExampleResourceID"
          }
     }
}
```

아카이브 만들기 정책

다음 예제 정책은 아카이브 만들기를 사용 설정하는 방법을 보여줍니다. 정책은 모든 자산에서 작동합니다.

IAM 정책

```
{
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "ses:CreateArchive",
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "kms:DescribeKey",
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "ses.us-east-1.amazonaws.com",
                    "kms:CallerAccount": "012345678910"
                }
            }
        }
```

AWS KMS 정책

```
{
            "Sid": "Allow SES to encrypt/decrypt",
            "Effect": "Allow",
            "Principal": {
                "Service": "ses.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
```

Amazon SES에 대한 암호화 키 모니터링

Amazon SES 리소스와 함께 AWS KMS 고객 관리형 키를 사용하는 경우 AWS CloudTrail 또는 Amazon CloudWatch Logs를 사용하여 SES가 보내는 요청을 추적할 수 있습니다 AWS KMS.

다음 예제는 고객 관리형 키로 암호화된 데이터에 액세스`DescribeKey`하기 위해 SES에서 호출한 KMS 작업을 모니터링하기 위한 `GenerateDataKeyDecrypt`, 및 AWS CloudTrail 이벤트입니다.

GenerateDataKey

리소스에 대해 AWS KMS 고객 관리형 키를 활성화하면 SES가 고유한 테이블 키를 생성합니다. 리소스에 대한 AWS KMS고객 관리 AWS KMS 형 키를 지정하는 `GenerateDataKey` 요청을에 보냅니다.

Mail Manager 아카이브 리소스에 대해 AWS KMS 고객 관리형 키를 활성화하면 저장된 아카이브 데이터를 암호화할 `GenerateDataKey` 때이 키를 사용합니다.

다음 예제 이벤트는 `GenerateDataKey` 작업을 기록합니다.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ses.amazonaws.com"
    },
    "eventTime": "2021-04-22T17:07:02Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "encryptionContext": {
            "aws:ses:arn": "arn:aws:ses:us-west-2:111122223333:ExampleResourceName/
ExampleResourceID"
        },
        "keySpec": "AES_256",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333",
    "sharedEventID": "57f5dbee-16da-413e-979f-2c4c6663475e"
```

```
}
```

Decrypt

암호화된 리소스에 액세스하는 경우 SES는 저장된 암호화 데이터 키를 사용하여 암호화된 데이터에 액세스하는 `Decrypt` 작업을 직접적으로 호출합니다.

다음 예제 이벤트는 `Decrypt` 작업을 기록합니다.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ses.amazonaws.com"
    },
    "eventTime": "2021-04-22T17:10:51Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "encryptionContext": {
            "aws:ses:arn": "arn:aws:ses:us-west-2:111122223333:ExampleResourceName/
ExampleResourceID"
        },
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
```

```
    "eventCategory": "Management",
    "recipientAccountId": "111122223333",
    "sharedEventID": "dc129381-1d94-49bd-b522-f56a3482d088"
}
```

DescribeKey

SES는 `DescribeKey` 작업을 사용하여 리소스와 연결된 AWS KMS 고객 관리형 키가 계정 및 리전에 존재하는지 확인합니다.

다음 예제 이벤트는 `DescribeKey` 작업을 기록합니다.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-04-22T17:02:00Z"
            }
        },
        "invokedBy": "ses.amazonaws.com"
    },
    "eventTime": "2021-04-22T17:07:02Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "keyId": "00dd0db0-0000-0000-ac00-b0c000SAMPLE"
```

```
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333"
}
```

자세히 알아보기

다음 리소스에서 키에 대한 추가 정보를 확인할 수 있습니다.

- AWS Key Management Service 기본 개념에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서를 참조하세요.
- AWS Key Management Service의 보안 모범 사례 에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서를 참조하세요.

전송 중 암호화

기본적으로 Amazon SES는 opportunistic TLS를 사용합니다. 즉, Amazon SES가 수신 메일 서버에 대한 보안 연결을 항상 시도한다는 것입니다. 만약 보안 연결을 설정할 수 없는 경우 암호화하지 않고 해당 메시지를 전송합니다. Amazon SES가 보안 연결을 설정할 수 있다면 수신 이메일 서버에 해당 메시지를 전송하도록 이 작업을 변경할 수 있습니다. 자세한 내용은 Amazon SES 및 보안 프로토콜 단원을 참조하십시오.

Amazon SES에서 개인 데이터 삭제

사용 방식에 따라 Amazon SES는 개인 데이터로 간주되는 특정 데이터를 저장할 수 있습니다. 예를 들어 Amazon SES를 사용하여 이메일을 보내려면 최소 한 개 이상의 확인된 ID(이메일 주소 또는 도메

인)을 제공해야 합니다. Amazon SES 콘솔이나 Amazon SES API를 사용하여 이 개인 데이터를 영구적으로 삭제할 수 있습니다.

이 장에서는 개인 데이터로 간주될 수 있는 다양한 유형의 데이터를 삭제하는 절차를 설명합니다.

내용

- 계정 수준 금지 목록에서 이메일 주소 삭제
- Amazon SES를 사용하여 전송한 이메일에 대한 데이터 삭제
- ID에 대한 데이터 삭제
- 발신자 인증 데이터 삭제
- 수신 규칙과 관련된 데이터 삭제
- IP 주소 필터와 관련된 데이터 삭제
- 이메일 템플릿의 데이터 삭제
- 사용자 지정 확인 이메일 템플릿의 데이터 삭제
- AWS 계정을 닫아 모든 개인 데이터 삭제

계정 수준 금지 목록에서 이메일 주소 삭제

Amazon SES에는 계정 수준 금지 목록(선택 사항)이 포함되어 있습니다. 이 기능을 활성화하면 이메일 주소가 반송 메일 또는 수신 거부가 되면 금지 목록에 자동으로 추가됩니다. 이메일 주소는 삭제할 때까지 이 목록에 유지됩니다. 계정 수준 금지 목록에 대한 자세한 내용은 Amazon SES 계정 수준 금지 목록 사용 섹션을 참조하세요.

Amazon SES API v2에서 `DeleteSuppressedDestination` 작업을 사용하여 계정 수준 금지 목록에서 전자 메일 주소를 제거할 수 있습니다. 이 섹션에는 AWS CLI를 사용하여 이메일 주소를 삭제하는 절차가 나와 있습니다. AWS CLI설치 및 구성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요.

AWS CLI를 사용하여 계정 수준 금지 목록에서 주소를 제거하려면

- 명령줄에 다음 명령을 입력합니다.

  ```
  aws sesv2 delete-suppressed-destination --email-address recipient@example.com
  ```

  위의 명령에서 *recipient@example.com*을 계정 수준 금지 목록에서 제거할 이메일 주소로 바꿉니다.

Amazon SES를 사용하여 전송한 이메일에 대한 데이터 삭제

Amazon SES를 사용하여 이메일을 보낼 때 해당 이메일에 대한 정보를 다른 AWS 서비스로 보낼 수 있습니다. 예를 들어 이메일 이벤트(전송, 열기, 클릭 등)에 대한 정보를 Firehose로 보낼 수 있습니다. 이 이벤트 데이터에는 일반적으로 이메일을 보낸 이메일 주소와 IP 주소가 포함됩니다. 또한 이메일을 받는 모든 수신자의 이메일 주소가 포함됩니다.

Firehose를 사용하여 이메일 이벤트 데이터를 Amazon Simple Storage Service, Amazon OpenSearch Service, Amazon Redshift를 비롯한 여러 대상으로 스트리밍할 수 있습니다. 이 데이터를 제거하려면 먼저 Firehose로의 데이터 스트리밍을 중지한 후 이미 스트리밍된 데이터를 삭제해야 합니다. Amazon SES 이벤트 데이터를 Firehose로 스트리밍하는 것을 중지하려면 Firehose 이벤트 대상을 삭제해야 합니다.

Amazon SES 콘솔을 사용하여 Firehose 이벤트 대상을 제거하는 방법

1. https://console.aws.amazon.com/ses/에서 Amazon SES 콘솔을 엽니다.
2. Email Sending(이메일 보내기)에서 Configuration Sets(구성 세트)를 선택합니다.
3. 구성 세트 목록에서 Firehose 이벤트 대상을 포함하는 구성 세트를 선택합니다.
4. 삭제하려는 Firehose 이벤트 대상 옆에 있는 삭제( ) 버튼을 선택합니다.
5. 필요한 경우 Firehose가 다른 서비스에 쓴 데이터를 제거합니다. 자세한 내용은 the section called "저장된 이벤트 데이터 제거" 단원을 참조하십시오.

Amazon SES API를 사용하여 이벤트 대상을 삭제할 수도 있습니다. 다음 절차에서는 AWS Command Line Interface (AWS CLI)를 사용하여 Amazon SES API와 상호 작용합니다. AWS SDK를 사용하거나 HTTP 요청을 직접 수행하여 API와 상호 작용할 수도 있습니다.

를 사용하여 Firehose 이벤트 대상을 제거하려면 AWS CLI

1. 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws sesv2 delete-configuration-set-event-destination --configuration-set-
name configSet \
--event-destination-name eventDestination
```

   이 명령에서 *configSet*를 Firehose 이벤트 대상을 포함하는 구성 세트 이름으로 바꿉니다. *eventDestination*을 Firehose 이벤트 대상 이름으로 바꿉니다.

2. 필요한 경우 Firehose가 다른 서비스에 쓴 데이터를 제거합니다. 자세한 내용은 the section called "저장된 이벤트 데이터 제거" 단원을 참조하십시오.

저장된 이벤트 데이터 제거

다른 AWS 서비스에서 정보를 삭제하는 방법에 대한 자세한 내용은 다음 문서를 참조하세요.

- Amazon Simple Storage Service 사용자 안내서의 객체 및 버킷 삭제
- Amazon OpenSearch Service 개발자 가이드의 OpenSearch Service 도메인 삭제
- Amazon Redshift 클러스터 관리 가이드의 클러스터 삭제

Firehose를 사용하여에서 지원되지 AWS 않거나 관리되지 않는 타사 서비스인 Splunk로 이메일 데이터를 스트리밍할 수도 있습니다 AWS Management Console. Splunk에서 데이터를 제거하는 자세한 방법은 시스템 관리자에게 문의하거나, Splunk 웹 사이트의 설명서를 참조하세요.

ID에 대한 데이터 삭제

ID에는 Amazon SES를 사용하여 이메일을 보내는 데 사용하는 이메일 주소와 도메인이 포함되어 있습니다. 일부 관할권에서 이메일 주소나 도메인은 개인 식별 데이터로 간주될 수 있습니다.

Amazon SES 콘솔을 사용하여 ID를 삭제하려면

1. https://console.aws.amazon.com/ses/에서 Amazon SES 콘솔을 엽니다.
2. 자격 증명 관리에서 다음 중 하나를 수행합니다.
   - 도메인을 삭제하려면 도메인을 선택합니다.
   - 이메일 주소를 삭제하려면 이메일 주소를 선택합니다.
3. 삭제하려는 ID를 선택한 후 제거를 선택합니다.
4. 확인 대화 상자에서 Yes, Delete Identity(예, ID 삭제)를 선택합니다.

Amazon SES API를 사용하여 ID를 삭제할 수도 있습니다. 다음 절차는 AWS Command Line Interface (AWS CLI)을(를) 사용하여 Amazon SES API와 상호 작용합니다. AWS SDK를 사용하거나 HTTP 요청을 직접 수행하여 API와 상호 작용할 수도 있습니다.

를 사용하여 자격 증명을 삭제하려면 AWS CLI

- 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses delete-identity --identity sender@example.com
```

이 명령에서 *sender@example.com*을, 삭제하려는 ID로 바꿉니다.

발신자 인증 데이터 삭제

발신자 인증은 Amazon SES 구성 프로세스와 관련이 있기 때문에 다른 사용자가 나를 대신하여 이메일을 보낼 수 있습니다. 발신자 인증을 활성화하려면 Amazon SES에서 전송 권한 부여 사용에서 설명한 대로 정책을 만들어야 합니다. 이러한 정책에는 ID(사용자를 대신하여 이메일을 보내는 사람 또는 그룹과 연결됨) 외에도 자격 증명 AWS IDs(사용자 소유)이 포함됩니다. 발신자 인증 정책을 수정하거나 삭제하여 이 개인 데이터를 제거할 수 있습니다. 다음 절차는 이러한 정책을 삭제하는 방법을 보여줍니다.

Amazon SES 콘솔을 사용하여 발신자 인증 정책을 삭제하려면

1. https://console.aws.amazon.com/ses/에서 Amazon SES 콘솔을 엽니다.
2. 자격 증명 관리에서 다음 중 하나를 수행합니다.
   - 삭제하려는 발신자 인증 정책이 도메인과 연결되어 있으면 도메인을 선택합니다.
   - 삭제하려는 발신자 인증 정책이 이메일 주소와 연결되어 있으면 이메일 주소를 선택합니다.
3. Identity Policies(ID 정책)에서 삭제하려는 정책을 선택한 후 Remove Policy(정책 제거)를 선택합니다.

Amazon SES API를 사용하여 발신자 인증 정책을 삭제할 수도 있습니다. 다음 절차에서는 AWS Command Line Interface (AWS CLI)를 사용하여 Amazon SES API와 상호 작용합니다. AWS SDK를 사용하거나 HTTP 요청을 직접 수행하여 API와 상호 작용할 수도 있습니다.

를 사용하여 발신자 인증 정책을 삭제하려면 AWS CLI

- 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses delete-identity-policy --identity example.com --policy-name samplePolicy
```

이 명령에서 *example.com*을, 발신자 인증 정책을 포함하는 ID로 바꿉니다. *samplePolicy*를 발신자 인증 정책 이름으로 바꿉니다.

수신 규칙과 관련된 데이터 삭제

Amazon SES를 사용하여 이메일을 수신하는 경우 하나 이상의 ID(이메일 주소 또는 도메인)에 적용되는 수신 규칙을 만들 수 있습니다. 이러한 규칙은 지정한 ID로 전송된 수신 메일을 Amazon SES가 처리하는 방식을 결정합니다.

Amazon SES 콘솔을 사용하여 수신 규칙을 삭제하려면

1. https://console.aws.amazon.com/ses/에서 Amazon SES 콘솔을 엽니다.
2. 이메일 수신에서 Rule Sets(규칙 세트)를 선택합니다.
3. 수신 규칙이 활성 규칙 세트의 일부인 경우 View Active Rule Set(활성 규칙 세트 보기)를 선택합니다. 그렇지 않은 경우에는 삭제하려는 수신 규칙을 포함하는 규칙 세트를 선택합니다.
4. 수신 규칙 목록에서 삭제하려는 규칙을 선택합니다.
5. [Actions] 메뉴에서 [Delete]를 선택합니다.
6. 확인 대화 상자에서 삭제를 선택합니다.

Amazon SES API를 사용하여 수신 규칙을 삭제할 수도 있습니다. 다음 절차에서는 AWS Command Line Interface (AWS CLI)를 사용하여 Amazon SES API와 상호 작용합니다. AWS SDK를 사용하거나 HTTP 요청을 직접 수행하여 API와 상호 작용할 수도 있습니다.

를 사용하여 수신 규칙을 삭제하려면 AWS CLI

- 명령줄 프롬프트에 다음 명령을 입력합니다.

  ```
  aws ses delete-receipt-rule --rule-set myRuleSet --rule-name myReceiptRule
  ```

  이 명령에서 *myRuleSet*를, 수신 규칙을 포함하는 수신 규칙 세트 이름으로 바꿉니다. *myReceiptRule*을, 삭제하려는 수신 규칙 이름으로 바꿉니다.

IP 주소 필터와 관련된 데이터 삭제

Amazon SES를 사용하여 수신 이메일을 받는 경우 특정 IP 주소로부터 전송되는 메시지를 수락하거나 차단하는 필터를 만들 수 있습니다.

Amazon SES 콘솔을 사용하여 IP 주소 필터를 삭제하려면

1. https://console.aws.amazon.com/ses/에서 Amazon SES 콘솔을 엽니다.

2. 이메일 수신에서 IP Address Filters(IP 주소 필터)를 선택합니다.
3. IP 주소 필터 목록에서 제거하려는 필터를 선택한 후 삭제를 선택합니다.

Amazon SES API를 사용하여 IP 주소 필터를 삭제할 수도 있습니다. 다음 절차에서는 AWS Command Line Interface (AWS CLI)를 사용하여 Amazon SES API와 상호 작용합니다. AWS SDK를 사용하거나 HTTP 요청을 직접 수행하여 API와 상호 작용할 수도 있습니다.

를 사용하여 IP 주소 필터를 삭제하려면 AWS CLI

- 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses delete-receipt-filter --filter-name IPfilter
```

  이 명령에서 *IPfilter*를, 삭제하려는 IP 주소 필터의 이름으로 바꿉니다.

이메일 템플릿의 데이터 삭제

이메일을 보낼 때 이메일 템플릿을 사용하는 경우 그러한 템플릿을 어떻게 구성했는지에 따라 템플릿에 개인 데이터가 들어 있을 수 있습니다. 예를 들어 수신인이 자세한 정보를 문의할 수 있는 이메일 주소를 템플릿에 추가했을 수 있습니다.

이메일 템플릿은 Amazon SES API를 사용해야만 삭제할 수 있습니다.

를 사용하여 이메일 템플릿을 삭제하려면 AWS CLI

- 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses delete-template --template-name sampleTemplate
```

  이 명령에서 *sampleTemplate*을, 삭제하려는 이메일 템플릿의 이름으로 바꿉니다.

사용자 지정 확인 이메일 템플릿의 데이터 삭제

새로운 이메일 발신 주소를 확인하기 위해 사용자 지정 템플릿을 사용하는 경우 그러한 템플릿을 어떻게 구성했는지에 따라 템플릿에 개인 데이터가 들어 있을 수 있습니다. 예를 들어 수신인이 자세한 정보를 문의할 수 있는 이메일 주소를 확인 이메일 템플릿에 추가했을 수 있습니다.

사용자 지정 확인 이메일 템플릿은 Amazon SES API를 사용해야만 삭제할 수 있습니다.

를 사용하여 사용자 지정 확인 이메일 템플릿을 삭제하려면 AWS CLI

- 명령줄 프롬프트에 다음 명령을 입력합니다.

```
aws ses delete-custom-verification-email-template --template-
name verificationEmailTemplate
```

이 명령에서 *verificationEmailTemplate*을, 삭제하려는 사용자 지정 확인 이메일 템플릿의 이름으로 바꿉니다.

AWS 계정을 닫아 모든 개인 데이터 삭제

AWS 계정을 닫아서 Amazon SES에 저장된 모든 개인 데이터를 삭제할 수도 있습니다. 그러나이 작업은 다른 모든 AWS 서비스에 저장한 개인 또는 비개인 데이터도 모두 삭제합니다.

AWS 계정을 해지하면 AWS 계정의 데이터가 90일 동안 보존됩니다. 이 보관 기간이 끝나면 데이터가 영구적으로 삭제되며 되돌릴 수 없습니다.

AWS 계정을 해지하려면

AWS 계정을 해지하는 방법에 대한 전체 지침은 AWS 계정 해지에서 다룹니다.

Amazon SES의 Identity and Access Management

Amazon Simple Email Service(Amazon SES)와 함께 AWS Identity and Access Management (IAM)를 사용하여 사용자, 그룹 또는 역할이 수행할 수 있는 SES API 작업을 지정할 수 있습니다. (이 주제에서는 이러한 엔터티를 모두 사용자라고 칭합니다.) 또한 사용자가 이메일의 "From", 수신자 및 "Return-Path" 주소에 사용할 수 있는 이메일 주소를 제어할 수 있습니다.

예를 들어, 조직의 사용자가 이메일은 보낼 수 있지만 전송 통계 확인 등의 관리 작업은 수행하지 못하도록 하는 IAM 정책을 만들 수 있습니다. 또 다른 예를 들면, 사용자가 계정에서 SES를 통해 이메일을 보낼 수 있지만 특정 "From" 주소를 사용하는 경우에만 허용하는 정책을 작성할 수 있습니다.

IAM을 사용하려면 권한을 명시적으로 정의하는 문서인 IAM 정책을 정의하고 사용자에게 해당 정책을 연결합니다. IAM 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서를 참조하십시오. 정책에서 설정하는 제한 사항의 적용 외에는 사용자가 SES와 상호 작용하는 방법 또는 SES에서 요청을 수행하는 방법이 변경되지 않습니다.

> **Note**
>
> - 계정이 SES 샌드박스에 있는 경우 해당 제한으로 인해 일부 정책을 구현하지 못합니다. 프로덕션 액세스 권한 요청 단원을 참조하세요.
> - 또한 전송 권한 부여 정책을 사용하여 SES에 대한 액세스를 제어할 수 있습니다. IAM 정책은 개별 사용자가 수행할 수 있는 작업을 제한하지만 전송 권한 부여 정책은 확인된 개별 ID를 사용할 수 있는 방법을 제한합니다. 또한 전송 권한 부여 정책만 교차 계정 액세스 권한을 부여할 수 있습니다. 권한 부여 전송에 대한 자세한 내용은 Amazon SES에서 전송 권한 부여 사용 단원을 참조하세요.

기존 사용자의 SES SMTP 보안 인증 정보를 생성하는 방법에 대한 자세한 내용은 Amazon SES SMTP 자격 증명 획득 섹션을 참조하세요.

SES에 액세스하기 위한 IAM 정책 생성

이 단원에서는 특히 SES에서 IAM 정책을 사용하는 방법을 설명합니다. 일반적으로 IAM 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서를 참조하십시오.

다음 세 가지 이유로 SES에서 IAM을 사용할 수 있습니다.

- 이메일 전송 작업을 제한하기 위해
- 사용자가 전송하는 이메일의 "From", 수신자 및 "Return-Path" 주소를 제한하기 위해
- 일반적인 API 사용을 제어하기 위해(예: 사용자가 사용 권한이 있는 API를 호출할 수 있는 시간)

작업 제한

사용자가 수행할 수 있는 SES 작업을 제어하려면 IAM 정책의 `Action` 요소를 사용합니다. API 이름에 소문자 문자열 `ses:`를 접두사로 추가하여 `Action` 요소를 모든 SES API 작업으로 설정할 수 있습니다. 예를 들어, `Action`을 (모든 작업에 대해) `ses:SendEmail`, `ses:GetSendStatistics` 또는 `ses:*`로 설정할 수 있습니다.

그런 다음 `Action`에 따라 `Resource` 요소를 다음과 같이 지정합니다.

`Action` 요소가 이메일 전송 API(즉 **`ses:SendEmail`** 및/또는 **`ses:SendRawEmail`**)에 대한 액세스만 허용하는 경우:

- 사용자가의 자격 증명에서 전송하도록 허용하려면 `*Resource`로 AWS 계정설정합니다.

- 사용자가 전송할 수 있는 자격 증명을 제한하려면 `Resource`를 사용자가 사용할 수 있도록 허용한 자격 증명의 ARN으로 설정합니다.

`Action` 요소가 모든 API에 대한 액세스를 허용하는 경우:

- 사용자가 전송할 수 있는 자격 증명을 제한하지 않으려면 `Resource`를 *로 설정합니다.
- 사용자가 전송할 수 있는 자격 증명을 제한하려면 다음과 같은 두 가지 정책(또는 한 정책 안에 두 가지 문)을 만들어야 합니다.
  - 허용되는 비 이메일 전송 API의 명시적 목록으로 설정된 `Action`과 *로 설정된 `Resource`가 포함된 정책
  - 이메일 전송 API(`ses:SendEmail` 및/또는 `ses:SendRawEmail`) 중 하나로 설정된 `Action`과 사용자가 사용할 수 있도록 허용한 자격 증명의 ARN으로 설정된 `Resource`가 포함된 정책

사용 가능한 SES 작업 목록은 Amazon Simple Email Service API 참조를 참조하세요. 사용자가 SMTP 인터페이스를 사용할 경우 `ses:SendRawEmail`에 대한 액세스를 최소로 허용해야 합니다.

이메일 주소 제한

특정 이메일 주소로 사용자를 제한하려면 `Condition` 블록을 사용하면 됩니다. `Condition` 블록에서 IAM 사용 설명서의 설명과 같이 조건 키를 사용하여 조건을 지정합니다. 조건 키를 사용하여 다음과 같은 이메일 주소를 제어할 수 있습니다.

> **Note**
>
> 이러한 이메일 주소 조건 키는 아래 표에 나와 있는 API에만 적용됩니다.

조건 키	설명	API
`ses:Recipients`	To:, "CC" 및 "BCC" 주소가 포함된 수신자 주소를 제한합니다.	`SendEmail` , `SendRawEmail`
`ses:FromAddress`	"From" 주소를 제한합니다.	`SendEmail` , `SendRawEmail` , `SendBounce`

조건 키	설명	API
`ses:FromDisplayName`	표시 이름으로 사용된 "From" 주소를 제한합니다.	`SendEmail` , `SendRawEmail`
`ses:FeedbackAddress`	이메일 피드백 전달로 반송 메일과 불만 제기를 전송할 수 있는 주소인 "Return-Path" 주소를 제한합니다. 이메일 피드백 전달에 대한 자세한 내용은 이메일을 통해 Amazon SES 알림 수신 단원을 참조하세요.	`SendEmail` , `SendRawEmail`
`ses:MultiRegionEndpointId`	이메일을 보낼 때 사용되는 엔드포인트 ID를 제어할 수 있습니다.	`SendEmail` , `SendBulkEmail`

SES API 버전별 제한

`ses:ApiVersion` 조건 키를 사용하여 SES API 버전에 따라 SES에 대한 액세스를 제한할 수 있습니다.

> **Note**
>
> SES SMTP 인터페이스는 `ses:SendRawEmail`의 SES API 버전 2를 사용합니다.

일반 API 사용 제한

조건에서 AWS전체 키를 사용하면 사용자가 APIs. SES는 다음과 같은 AWS전체 정책 키만 구현합니다.

- `aws:CurrentTime`
- `aws:EpochTime`
- `aws:SecureTransport`
- `aws:SourceIp`
- `aws:SourceVpc`

- `aws:SourceVpce`
- `aws:UserAgent`
- `aws:VpcSourceIp`

이러한 키에 대한 자세한 내용은 IAM 사용 설명서를 참조하십시오.

SES에 대한 예제 IAM 정책

이 주제에서는 사용자의 SES에 대한 액세스를 특정 조건에서만 허용하는 정책 예를 다룹니다.

이 단원의 정책 예:

- 모든 SES 작업에 대한 모든 액세스 허용
- SES API 버전 2에 대한 액세스만 허용
- 이메일 전송 작업에 대한 액세스만 허용
- 전송 기간 제한
- 수신자 주소 제한
- "From" 주소 제한
- 이메일 발신자의 표시 이름 제한
- 반송 메일 및 수신 거부 피드백 대상 제한

모든 SES 작업에 대한 모든 액세스 허용

다음 정책은 사용자가 모든 SES 작업을 호출할 수 있도록 허용합니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "ses:*"
      ],
      "Resource":"*"
    }
  ]
```

```
}
```

SES API 버전 2에 대한 액세스만 허용

다음 정책은 사용자가 API 버전 2의 SES 작업만 호출할 수 있도록 허용합니다.

JSON

```
{
                    "Version":"2012-10-17",
                    "Statement":[
                        {
                            "Effect":"Allow",
                            "Action":[
                            "ses:*"
                            ],
                            "Resource":"*",
                            "Condition": {
                                "StringEquals" : {
                                "ses:ApiVersion" : "2"
                                }
                            }
                        }
                    ]
                }
```

이메일 전송 작업에 대한 액세스만 허용

다음 정책은 사용자가 SES를 사용하여 이메일을 전송하는 것을 허용하지만 SES 전송 통계에 액세스하는 것과 같은 관리 작업을 수행하는 것은 허용하지 않습니다.

JSON

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

```
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource":"*"
    }
  ]
}
```

전송 기간 제한

다음 정책은 사용자가 2018년 9월에만 SES 이메일 전송 API를 호출하도록 허용합니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource":"*",
      "Condition":{
        "DateGreaterThan":{
          "aws:CurrentTime":"2018-08-31T12:00Z"
        },
        "DateLessThan":{
          "aws:CurrentTime":"2018-10-01T12:00Z"
        }
      }
    }
  ]
}
```

수신자 주소 제한

다음 정책은 사용자가 SES 이메일 전송 API를 호출하도록 허용하지만 example.com 도메인 (`StringLike`는 대소문자 구분)의 수신자 주소에만 호출하도록 허용합니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource":"*",
      "Condition":{
        "ForAllValues:StringLike":{
          "ses:Recipients":[
            "*@example.com"
          ]
        }
      }
    }
  ]
}
```

"From" 주소 제한

다음 정책은 사용자가 SES 이메일 전송 API를 호출하도록 허용하지만 "From" 주소가 marketing@example.com인 경우에만 적용됩니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource":"*",
```

```
      "Condition":{
        "StringEquals":{
          "ses:FromAddress":"marketing@example.com"
        }
      }
    }
  ]
}
```

다음 정책은 사용자가 SendBounce API를 호출하도록 허용하지만 "From" 주소가 bounce@example.com인 경우에만 적용됩니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "ses:SendBounce"
      ],
      "Resource":"*",
      "Condition":{
        "StringEquals":{
          "ses:FromAddress":"bounce@example.com"
        }
      }
    }
  ]
}
```

이메일 발신자의 표시 이름 제한

다음 정책은 사용자가 SES 이메일 전송 API를 호출하도록 허용하지만 "From" 주소의 표시 이름에 Marketing(`StringLike`는 대소문자 구분)이 포함된 경우에만 적용됩니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource":"*",
      "Condition":{
        "StringLike":{
          "ses:FromDisplayName":"Marketing"
        }
      }
    }
  ]
}
```

반송 메일 및 수신 거부 피드백 대상 제한

다음 정책은 사용자가 SES 이메일 전송 API를 호출하도록 허용하지만 이메일의 "Return-Path"가 feedback@example.com으로 설정된 경우에만 적용됩니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource":"*",
      "Condition":{
        "StringEquals":{
```

```
            "ses:FeedbackAddress":"feedback@example.com"
          }
        }
      }
    ]
}
```

AWS Amazon Simple Email Service에 대한 관리형 정책

사용자, 그룹 및 역할에 권한을 추가하려면 직접 정책을 작성하는 것보다 AWS 관리형 정책을 사용하는 것이 더 쉽습니다. 팀에 필요한 권한만 제공하는 IAM 고객 관리형 정책을 생성하기 위해서는 시간과 전문 지식이 필요합니다. 빠르게 시작하려면 AWS 관리형 정책을 사용할 수 있습니다. 이러한 정책은 일반적인 사용 사례를 다루며 AWS 계정에서 사용할 수 있습니다. AWS 관리형 정책에 대한 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책을 참조하세요.

AWS 서비스는 AWS 관리형 정책을 유지 관리하고 업데이트합니다. AWS 관리형 정책에서는 권한을 변경할 수 없습니다. 서비스는 때때로 AWS 관리형 정책에 추가 권한을 추가하여 새 기능을 지원합니다. 이 유형의 업데이트는 정책이 연결된 모든 ID(사용자, 그룹 및 역할)에 적용됩니다. 서비스는 새 기능이 시작되거나 새 작업을 사용할 수 있게 될 때 AWS 관리형 정책을 업데이트할 가능성이 높습니다. 서비스는 AWS 관리형 정책에서 권한을 제거하지 않으므로 정책 업데이트로 인해 기존 권한이 손상되지 않습니다.

또한는 여러 서비스에 걸쳐 있는 직무에 대한 관리형 정책을 AWS 지원합니다. 예를 들어 ReadOnlyAccess AWS 관리형 정책은 모든 AWS 서비스 및 리소스에 대한 읽기 전용 액세스를 제공합니다. 서비스가 새 기능을 시작하면는 새 작업 및 리소스에 대한 읽기 전용 권한을 AWS 추가합니다. 직무 정책의 목록과 설명은 IAM 사용 설명서의 직무에 관한AWS 관리형 정책을 참조하세요.

AWS 관리형 정책: AmazonSESFullAccess

`AmazonSESFullAccess` 정책을 IAM 보안 인증에 연결할 수 있습니다. Amazon SES에 대한 전체 액세스 권한을 제공합니다.

이 정책의 권한을 보려면 AWS 관리형 정책 참조에서 AmazonSESFullAccess를 참조하세요.

AWS 관리형 정책: AmazonSESReadOnlyAccess

`AmazonSESReadOnlyAccess` 정책을 IAM 보안 인증에 연결할 수 있습니다. Amazon SES에 대한 읽기 전용 액세스 권한을 제공합니다.

이 정책의 권한을 보려면 AWS 관리형 정책 참조에서 AmazonSESReadOnlyAccess를 참조하세요.

AWS 관리형 정책: AmazonSESServiceRolePolicy

`AmazonSESServiceRolePolicy` 정책을 IAM 엔터티에 연결할 수 없습니다. 이 정책은 Amazon SES에 사용자를 대신하여 작업을 수행할 수 있도록 하는 서비스 연결 역할에 연결됩니다. 자세한 내용은 Amazon SES에 대한 서비스 연결 역할 권한 단원을 참조하십시오.

이 정책의 권한을 보려면 AWS 관리형 정책 참조에서 AmazonSESServiceRolePolicy를 참조하세요.

AWS 관리형 정책에 대한 Amazon Simple Email Service 업데이트

이 서비스가 이러한 변경 사항을 추적하기 시작한 이후부터 Amazon Simple Email Service의 AWS 관리형 정책 업데이트에 대한 세부 정보를 봅니다.

변경 사항	설명	날짜
Amazon Simple Email Service에서 새 관리형 정책 추가	Amazon Simple Email Service에서 SES가 자동으로 작업을 수행할 수 있도록 서비스 관련 역할 `AWSServiceRoleForAmazonSES` 에 `AmazonSESServiceRolePolicy` 추가	2024년 5월 13일
Amazon Simple Email Service에서 정책 정의 업데이트	Amazon Simple Email Service는 이 테이블(아래 행)의 이전 항목을 다음과 같이 명확히 정의: Amazon Simple Email Service는 AmazonSES ReadOnlyAccess 관리형 정책에 `ses:BatchGetMetricData` 를 추가했으며, 이를 통해 SES API BatchGetMetricData에 액세스할 수 있습니다.	2024년 4월 30일
Amazon Simple Email Service에서 정책 정의 업데이트	Amazon Simple Email Service가 AmazonSES ReadOnlyAccess 관리형 정책에 `ses:BatchGet*` 을 추가했으며, 이를 통해 SES API	2024년 2월 16일

변경 사항	설명	날짜
	BatchGetMetricData에 액세스할 수 있습니다.	
Amazon Simple Email Service에서 두 가지 정책 정의가 변경됨	Amazon Simple Email Service가 AmazonSESFullAccess 및 AmazonSESReadOnlyAccess AmazonSESReadOnlyAccess 정의 끝에서 " AWS 관리 콘솔을 통해" 제거됨	2023년 5월 3일
Amazon Simple Email Service에서 변경 사항 추적을 시작했습니다	Amazon Simple Email Service에서 AWS 관리형 정책에 대한 변경 사항 추적 시작	2023년 4월 5일

Amazon SES에 대해 서비스 연결 역할 사용

Amazon Simple Email Service(SES)는 AWS Identity and Access Management (IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 Amazon SES에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 SES에서 사전 정의하며 서비스가 사용자를 대신하여 다른 AWS 서비스를 호출하는 데 필요한 모든 권한을 포함합니다.

서비스 연결 역할을 사용하면 필요한 권한을 수동으로 추가할 필요가 없으므로 SES를 더 쉽게 설정할 수 있습니다. SES에서 서비스 연결 역할의 권한을 정의하므로 다르게 정의되지 않은 한, SES만 해당 역할을 수임할 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며 이 권한 정책은 다른 IAM 엔티티에 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 SES 리소스가 보호됩니다.

서비스 연결 역할을 지원하는 다른 서비스에 대한 자세한 내용은 AWS IAM으로 작업하는 서비스를 참조하고 서비스 연결 역할 열에서 예인 서비스를 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예 링크를 선택합니다.

Amazon SES에 대한 서비스 연결 역할 권한

SES는 AWSServiceRoleForAmazonSES라는 서비스 연결 역할을 사용합니다. SES는 SES 리소스를 대신하여 Amazon CloudWatch 기본 모니터링 지표를 게시할 수 있습니다.

AWSServiceRoleForAmazonSES 서비스 연결 역할은 다음 서비스를 신뢰하여 역할을 위임합니다.

- `ses.amazonaws.com`

AmazonSESServiceRolePolicy라는 역할 권한 정책은 SES가 지정된 리소스에서 다음 작업을 완료할 수 있도록 하는 AWS 관리형 정책입니다.

- 작업: `AWS/SES` CloudWatch 네임스페이스의 `cloudwatch:PutMetricData`. 이 작업은 SES가 지표 데이터를 CloudWatch `AWS/SES` 네임스페이스에 넣을 수 있는 권한을 부여합니다. CloudWatch에서 지원되는 SES 지표에 대한 자세한 내용은 Amazon SES의 로깅 및 모니터링 섹션을 참조하세요.
- 작업: `AWS/SES/MailManager` CloudWatch 네임스페이스의 `cloudwatch:PutMetricData`. 이 작업은 SES가 지표 데이터를 CloudWatch `AWS/SES/MailManager` 네임스페이스에 넣을 수 있는 권한을 부여합니다. CloudWatch에서 지원되는 SES 지표에 대한 자세한 내용은 Amazon SES의 로깅 및 모니터링 섹션을 참조하세요.
- 작업: `AWS/SES/Addons` CloudWatch 네임스페이스의 `cloudwatch:PutMetricData`. 이 작업은 SES가 지표 데이터를 CloudWatch `AWS/SES/Addons` 네임스페이스에 넣을 수 있는 권한을 부여합니다. CloudWatch에서 지원되는 SES 지표에 대한 자세한 내용은 Amazon SES의 로깅 및 모니터링 섹션을 참조하세요.

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.

Amazon SES에 대한 서비스 연결 역할 생성

서비스 링크 역할은 수동으로 생성할 필요가 없습니다. AWS Management Console AWS CLI, 또는 AWS API에서 SES 리소스를 생성하면 SES가 서비스 연결 역할을 생성합니다.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다. SES 리소스를 생성하면 SES에서 서비스 연결 역할을 다시 생성합니다.

Amazon SES에 대한 서비스 연결 역할 편집

SES는 AWSServiceRoleForAmazonSES 서비스 연결 역할을 편집하도록 허용하지 않습니다. 서비스 링크 역할을 생성한 후에는 다양한 개체가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다.

SES에 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 역할을 정리해야 수동으로 삭제할 수 있습니다.

서비스 연결 역할 정리

IAM을 사용하여 서비스 연결 역할을 삭제하려면 먼저 모든 SES 리소스를 삭제해야 합니다.

> **Note**
> 리소스를 삭제하려 할 때 SES 서비스가 역할을 사용 중이면 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

수동으로 서비스 연결 역할 삭제

IAM 콘솔 AWS CLI, 또는 AWS API를 사용하여 AWSServiceRoleForAmazonSES 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

Amazon SES 서비스 연결 역할이 지원되는 리전

SES에서는 서비스가 제공되는 모든 리전에서 서비스 연결 역할을 사용하도록 지원하지 않습니다. 다음 리전에서 AWSServiceRoleForAmazonSES 역할을 사용할 수 있습니다.

리전 이름	리전 자격 증명	SES 지원
미국 동부(버지니아 북부)	us-east-1	예
미국 동부(오하이오)	us-east-2	예
아시아 태평양(시드니)	ap-southeast-2	예
아시아 태평양(도쿄)	ap-northeast-1	예
유럽(프랑크푸르트)	eu-central-1	예
유럽(아일랜드)	eu-west-1	예

Amazon SES의 로깅 및 모니터링

모니터링은 Amazon SES 및 AWS 솔루션의 안정성, 가용성 및 성능을 유지하는 데 중요한 부분입니다.는 Amazon SES를 모니터링하고 잠재적 인시던트에 대응하는 데 도움이 되는 도구를 AWS 제공합니다.

- Amazon CloudWatch는 AWS 리소스와 AWS 실행 중인 애플리케이션을 실시간으로 모니터링합니다. 지표를 수집 및 추적하고, 사용자 지정 대시보드를 생성할 수 있으며, 지정된 지표가 지정한 임곗값에 도달하면 사용자에게 알리거나 조치를 취하도록 경보를 설정할 수 있습니다. 자세한 내용은 CloudWatch에서 Amazon SES 이벤트 데이터 가져오기 및 CloudWatch를 사용하여 평판 모니터링 경보 생성 섹션을 참조하세요.
- AWS CloudTrail는에 의해 또는를 대신하여 수행된 API 호출 및 관련 이벤트를 캡처 AWS 계정 하고 사용자가 지정한 Amazon S3 버킷에 로그 파일을 전송합니다. 호출한 사용자 및 계정 AWS, 호출이 수행된 소스 IP 주소, 호출이 발생한 시기를 식별할 수 있습니다. 자세한 내용은 를 사용하여 Amazon SES API 호출 로깅 AWS CloudTrail 단원을 참조하십시오.
- Amazon SES 이메일 전송 이벤트를 사용하면 이메일 전송 전략을 세밀하게 조정할 수 있습니다. Amazon SES는 다수의 전송, 배달, 확인, 클릭, 반송 메일, 불만 제기 및 거부를 포함하는 세부 정보를 캡처합니다. 자세한 내용은 전송 활동 모니터링 단원을 참조하십시오.
- Amazon SES 평판 지표는 계정에 대한 반송 메일 및 수신 거부율을 추적합니다. 자세한 내용은 발신자 평판 모니터링 단원을 참조하십시오.

를 사용하여 Amazon SES API 호출 로깅 AWS CloudTrail

Amazon SES는 SES에서 사용자 AWS CloudTrail, 역할 또는 서비스가 수행한 작업에 대한 레코드를 제공하는 AWS 서비스와 통합됩니다. CloudTrail은 SES에 대한 API 호출을 이벤트로 캡처합니다. 캡처되는 호출에는 SES 콘솔의 호출과 SES API 작업에 대한 코드 호출이 포함됩니다. 추적을 생성하면 SES 이벤트를 포함한 CloudTrail 이벤트를 지속적으로 Amazon S3 버킷에 배포할 수 있습니다. 트레일을 구성하지 않은 경우에도 CloudTrail 콘솔의 이벤트 기록에서 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 SES에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

구성 및 사용 방법을 포함하여 CloudTrail에 대한 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하세요.

CloudTrail의 SES 정보

CloudTrail은 계정을 생성할 AWS 계정 때에서 활성화됩니다. 지원되는 이벤트 활동이 SES에서 발생하면 해당 활동은 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. 에서 최근 이벤트를 보고 검색하고 다운로드할 수 있습니다 AWS 계정. 자세한 설명은 CloudTrail 이벤트 기록으로 이벤트 보기를 참조하세요.

SES 이벤트를 AWS 계정포함하여에 이벤트를 지속적으로 기록하려면 추적을 생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 콘솔에서 추적을 생성하면 기본적으로 모든 AWS 리전에 추적이 적용됩니다. 추적은 AWS 파티션의 모든 리전에서 이벤트를 로깅하고 지정한 Amazon S3 버킷으로 로그 파일을 전송합니다. 추가적으로, CloudTrail 로그에서 수집된 이벤트 데이터를 추가 분석 및 처리하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음 자료를 참조하세요.

- 트레일 생성 개요
- CloudTrail 지원 서비스 및 통합
- CloudTrail에서 Amazon SNS 알림 구성
- 여러 리전으로부터 CloudTrail 로그 파일 받기 및 여러 계정으로부터 CloudTrail 로그 파일 받기

모든 이벤트 또는 로그 항목에는 요청을 생성했던 사용자에 대한 정보가 포함됩니다. 자격 증명을 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청이 루트 또는 AWS Identity and Access Management (IAM) 사용자 자격 증명으로 이루어졌는지 여부입니다.
- 역할 또는 페더레이션 사용자에 대한 임시 보안 인증을 사용하여 요청이 생성되었는지 여부.
- 요청이 다른 AWS 서비스에 의해 이루어졌는지 여부입니다.

자세한 설명은 CloudTrail userIdentity 요소를 참조하세요.

CloudTrail의 SES 데이터 이벤트

데이터 이벤트는 리소스 상에서, 또는 리소스 내에서 수행되는 리소스 작업에 대한 정보를 제공합니다. 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.

> **Note**
>
> SES SMTP 인터페이스를 통한 이메일 전송 활동은 CloudTrail 이벤트에 로깅되지 않습니다. 포괄적인 활동 로깅을 위해 SES APIs 참조 및 SES API v2 참조의 최신 SES API를 사용합니다.

다음 표에는 데이터 이벤트를 로깅할 수 있는 SES 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 APIs를 표시하는 열을 사용하여 고급 이벤트 선택기를 구성할 때 지정할 `resources.type` 값이 표시됩니다. CloudTrail에 로깅되는 데이터 API 열에는 리소스 유형에 대해 CloudTrail에 로깅된 API 호출이 표시됩니다.

데이터 이벤트에 대한 SES 리소스 유형

데이터 이벤트 유형(콘솔)	resources.type 값	CloudTrail에 로깅되는 데이터 API
SES 자격 증명	AWS::SES::EmailIdentity	SES: SendEmail SendRawEmail SendTemplatedEmail SendBulkTemplatedEmail SES v2: SendEmail SendBulkEmail
SES 구성 세트	AWS::SES::ConfigurationSet	
SES 템플릿	AWS::SES::템플릿	SES: SendTemplatedEmail SendBulkTemplatedEmail

데이터 이벤트 유형(콘솔)	resources.type 값	CloudTrail에 로깅되는 데이터 API
		SES v2: SendEmail SendBulkEmail

다음 예제에서는 `--advanced-event-selectors` 파라미터를 사용하여 모든 SES 이메일 ID에 대한 모든 데이터 이벤트를 로깅하는 방법을 보여줍니다.

```
aws cloudtrail put-event-selectors \
--region Region \
--trail-name TrailName \
--advanced-event-selectors
'[
    {
        "Name": "Log SES data plane actions for all email identities",
        "FieldSelectors": [
        { "Field": "eventCategory", "Equals": ["Data"] },
        { "Field": "resources.type", "Equals": ["AWS::SES::EmailIdentity"] }
       ]
    }
]'
```

고급 이벤트 선택기를 추가로 세분화하여 `eventName`, `readOnly`및 `resources.ARN` 필드를 필터링하여 중요한 이벤트만 로깅할 수 있습니다. 이러한 필드에 대한 자세한 내용은 API 참조의 AdvancedFieldSelector를 참조하세요. AWS CloudTrail 데이터 이벤트를 로깅하는 방법에 대한 자세한 예는 추적에 대한 데이터 이벤트 로깅을 참조하세요.

SES 로깅을 위한 CloudTrail 로그 전송 시나리오

CloudTrail은 계정 및 리소스 소유권, 자격 증명 유형, 리전과 같은 요소를 기반으로 로그를 제공합니다. 다음 매트릭스는 이러한 요소의 특정 조합에 따라 로그가 전달되는 대상과 위치를 설명합니다.

시나리오 유형	계정 역할	리소스	요청 흐름	로그 전달
단일 교차 계정	계정 A: 리소스 소유자	이메일 자격 증명	B → A의 이메일 자격 증명	A와 B 모두에 전달된 로그

시나리오 유형	계정 역할	리소스	요청 흐름	로그 전달
	계정 B: 요청자	피드백 전달 이메일	B → A의 피드백 이메일	A와 B 모두에 전달된 로그
다중 교차 계정	계정 A:피드백 이메일 소유자 계정 B:이메일 자격 증명 소유자 계정 C:requester	피드백 이메일(A) 이메일 자격 증명(B)	C → A의 피드백 이메일 + B의 이메일 자격 증명	A, B, C로 전송된 로그
글로벌 엔드포인트(단일 계정)	계정 A: 소유자 및 요청자	글로벌 엔드포인트(기본:eu-west-1 및 보조:us-west-2)	A → 글로벌 엔드포인트	요청을 처리한 리전의 A에 전달된 로그(eu-west-1 또는 us-west-2)
글로벌 엔드포인트(교차 계정)	계정 A:이메일 자격 증명 소유자 계정 B:요청자	이메일 자격 증명(A) 글로벌 엔드포인트(B)(eu-west-1 및 us-west-2)	B → 글로벌 엔드포인트를 통한 A의 이메일 자격 증명	요청을 처리한 리전의 A와 B 모두에 전달된 로그(eu-west-1 또는 us-west-2)

Note

- CloudTrail은 항상 로그를 요청자 계정으로 전송합니다.
- 리소스 소유자는 작업을 수행하지 않은 경우에도 로그를 수신합니다.
- 글로벌 엔드포인트의 경우 두 계정 모두 구성된 모든 리전에서 CloudTrail 구독이 필요합니다.
- 리전 장애 중에는 모든 로그가 정상 리전에 표시됩니다.

CloudTrail의 SES 관리 이벤트

SES는 관리 이벤트를 CloudTrail에 전달합니다. 관리 이벤트에는 내에서 리소스를 생성하고 관리하는 것과 관련된 작업이 포함됩니다 AWS 계정. Amazon SES에서 관리 이벤트에는 자격 증명 또는 수신 규칙을 생성 및 삭제하는 등의 작업이 포함됩니다. SES API 작업에 대한 자세한 내용은 SES API 참조 및 SES API v2 참조를 참조하세요.

SES에 대한 CloudTrail 로그 파일 항목

추적이란 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 입력할 수 있게 하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 이벤트는 모든 소스로부터의 단일 요청을 나타내며 요청 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 들어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 직접 호출의 주문 스택 트레이스가 아니므로 특정 순서로 표시되지 않습니다.

다음 예에서는 이러한 이벤트 유형의 CloudTrail 로그를 보여줍니다.

이벤트 유형

- DeleteIdentity
- VerifyEmailIdentity
- 간단한 콘텐츠가 포함된 SendEmail
- 템플릿 콘텐츠가 포함된 SendEmail

DeleteIdentity

```
{
  "Records":[
    {
        "eventVersion": "1.11",
        "userIdentity": {
            "type": "AssumedRole",
            "principalId": "AROA4DO2KAWIPZEXAMPLE:myUserName",
            "arn": "arn:aws:sts::111122223333:assumed-role/users/myUserName",
            "accountId": "111122223333",
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "sessionContext": {
                "sessionIssuer": {
                    "type": "Role",
                    "principalId": "AROA4DO2KAWIPZEXAMPLE",
```

```
                    "arn": "arn:aws:iam::111122223333:role/admin-role",
                    "accountId": "111122223333",
                    "userName": "myUserName"
                },
                "attributes": {
                    "creationDate": "2025-02-27T09:53:35Z",
                    "mfaAuthenticated": "false"
                }
            }
        },
        "eventTime": "2025-02-27T09:54:31Z",
        "eventSource": "ses.amazonaws.com",
        "eventName": "DeleteIdentity",
        "awsRegion": "us-east-1",
        "sourceIPAddress": "192.0.2.0",
        "userAgent": "aws-cli/2.23.4",
        "requestParameters": {
            "identity": "sender@example.com"
        },
        "responseElements": null,
        "requestID": "50b87bfe-ab23-11e4-9106-5b36376f9d12",
        "eventID": "0ffa308d-1467-4259-8be3-c749753be325",
        "readOnly": false,
        "eventType": "AwsApiCall",
        "managementEvent": true,
        "recipientAccountId": "111122223333",
        "eventCategory": "Management",
        "tlsDetails": {
            "tlsVersion": "TLSv1.3",
            "cipherSuite": "TLS_AES_128_GCM_SHA256",
            "clientProvidedHostHeader": "email.us-east-1.amazonaws.com"
        }
    }
  ]
}
```


VerifyEmailIdentity


```
{
  "Records":[
    {
        "eventVersion": "1.11",
        "userIdentity": {
```

```
            "type": "AssumedRole",
            "principalId": "AROA4DO2KAWIPZEXAMPLE:myUserName",
            "arn": "arn:aws:sts::111122223333:assumed-role/users/myUserName",
            "accountId": "111122223333",
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "sessionContext": {
                "sessionIssuer": {
                    "type": "Role",
                    "principalId": "AROA4DO2KAWIPZEXAMPLE",
                    "arn": "arn:aws:iam::111122223333:role/admin-role",
                    "accountId": "111122223333",
                    "userName": "myUserName"
                },
                "attributes": {
                    "creationDate": "2025-02-27T09:53:35Z",
                    "mfaAuthenticated": "false"
                }
            }
        },
        "eventTime": "2025-02-27T09:56:20Z",
        "eventSource": "ses.amazonaws.com",
        "eventName": "VerifyEmailIdentity",
        "awsRegion": "us-east-1",
        "sourceIPAddress": "192.0.2.0",
        "userAgent": "aws-cli/2.23.4",
        "requestParameters": {
            "emailAddress": "sender@example.com"
        },
        "responseElements": null,
        "requestID": "eb2ff803-ac09-11e4-8ff5-a56a3119e253",
        "eventID": "5613b0ff-d6c6-4526-9b53-a603a9231725",
        "readOnly": false,
        "eventType": "AwsApiCall",
        "managementEvent": true,
        "recipientAccountId": "111122223333",
        "eventCategory": "Management",
        "tlsDetails": {
            "tlsVersion": "TLSv1.3",
            "cipherSuite": "TLS_AES_128_GCM_SHA256",
            "clientProvidedHostHeader": "email.us-east-1.amazonaws.com"
        }
    }
]
```

```
}
```

간단한 콘텐츠가 포함된 SendEmail

```
{
  "Records":[{
       "eventTime": "2025-01-24T11:43:00Z",
        "eventSource": "ses.amazonaws.com",
        "eventName": "SendEmail",
        "awsRegion": "us-east-1",
        "sourceIPAddress": "192.0.2.0",
        "userAgent": "aws-cli/2.23.4 md/awscrt#0.23.4",
        "requestParameters": {
            "destination": {
                "bccAddresses": ["HIDDEN_DUE_TO_SECURITY_REASONS"],
                "toAddresses": ["HIDDEN_DUE_TO_SECURITY_REASONS"],
                "ccAddresses": ["HIDDEN_DUE_TO_SECURITY_REASONS"]
            },
            "message": {
                "subject": {
                    "charset": "UTF-8",
                    "data": "HIDDEN_DUE_TO_SECURITY_REASONS"
                },
                "body": {
                    "html": {
                        "charset": "UTF-8",
                        "data": "HIDDEN_DUE_TO_SECURITY_REASONS"
                    },
                    "text": {
                        "charset": "UTF-8",
                        "data": "HIDDEN_DUE_TO_SECURITY_REASONS"
                    }
                }
            },
            "source": "sender@example.com"
        },
        "responseElements": null,
        "additionalEventData": {
            "sesMessageId": "01000100a11a11aa-00aa0a00-00a0-48a8-aaa7-
a174a83b456a-000000"
        },
        "requestID": "ab2cc803-ac09-11d7-8bb8-a56a3119e476",
        "eventID": "eb834e01-f168-435f-92c0-c36278378b6e",
```

```
        "readOnly": true,
        "resources": [{
            "accountId": "111122223333",
            "type": "AWS::SES::EmailIdentity",
            "ARN": "arn:aws:ses:us-east-1:111122223333:identity/sender@example.com"
        }],
        "eventType": "AwsApiCall",
        "managementEvent": false,
        "recipientAccountId": "111122223333",
        "eventCategory": "Data",
        "tlsDetails": {
            "tlsVersion": "TLSv1.3",
            "cipherSuite": "TLS_AES_128_GCM_SHA256",
            "clientProvidedHostHeader": "email.us-east-1.amazonaws.com"
        }
    }
  ]
}
```

템플릿 콘텐츠가 포함된 SendEmail

```
{
        "eventVersion": "1.11",
        "userIdentity": {
            "type": "AssumedRole",
            "principalId": "AROA4DO2KAWIPZEXAMPLE:myUserName",
            "arn": "arn:aws:sts::111122223333:assumed-role/users/myUserName",
            "accountId": "111122223333",
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "sessionContext": {
                "sessionIssuer": {
                    "type": "Role",
                    "principalId": "AROA4DO2KAWIPZEXAMPLE",
                    "arn": "arn:aws:iam::111122223333:role/admin-role",
                    "accountId": "111122223333",
                    "userName": "admin-role"
                },
                "attributes": {
                    "creationDate": "2025-03-05T18:51:06Z",
                    "mfaAuthenticated": "false"
                }
            }
        },
```

```
        "eventTime": "2025-03-05T19:16:29Z",
        "eventSource": "ses.amazonaws.com",
        "eventName": "SendEmail",
        "awsRegion": "us-east-1",
        "sourceIPAddress": "192.0.2.0",
        "userAgent": "aws-cli/2.23.4",
        "requestParameters": {
            "fromEmailAddress": "sender@example.com",
            "destination": {
                "toAddresses": ["HIDDEN_DUE_TO_SECURITY_REASONS"],
                "bccAddresses": ["HIDDEN_DUE_TO_SECURITY_REASONS"],
                "ccAddresses": ["HIDDEN_DUE_TO_SECURITY_REASONS"]
            },
            "emailTags": [{
                "value": "test",
                "name": "campaign"
            }, {
                "value": "cli-test",
                "name": "sender"
            }],
            "replyToAddresses": ["HIDDEN_DUE_TO_SECURITY_REASONS"],
            "content": {
                "template": {
                    "templateData": "HIDDEN_DUE_TO_SECURITY_REASONS",
                    "templateName": "TestTemplate"
                }
            }
        },
        "responseElements": null,
        "additionalEventData": {
            "sesMessageId": "01000100a11a11aa-00aa0a00-00a0-48a8-aaa7-
a174a83b456a-000000"
        },
        "requestID": "50b87bfe-ab23-11e4-9106-5b36376f9d12",
        "eventID": "0ffa308d-1467-4259-8be3-c749753be325",
        "readOnly": true,
        "resources": [{
            "accountId": "111122223333",
            "type": "AWS::SES::EmailIdentity",
            "ARN": "arn:aws:ses:us-east-1:111122223333:identity/sender@example.com"
        }, {
            "accountId": "111122223333",
            "type": "AWS::SES::Template",
            "ARN": "arn:aws:ses:us-east-1:111122223333:template/TestTemplate"
```

```
        }],
        "eventType": "AwsApiCall",
        "managementEvent": false,
        "recipientAccountId": "111122223333",
        "eventCategory": "Data",
        "tlsDetails": {
            "tlsVersion": "TLSv1.3",
            "cipherSuite": "TLS_AES_128_GCM_SHA256",
            "clientProvidedHostHeader": "email.us-east-1.amazonaws.com"
        }
    }
```

Amazon Simple Email Service에 대한 규정 준수 확인

타사 감사자는 여러 규정 준수 프로그램의 일환으로 Amazon Simple Email Service의 보안 및 AWS 규정 준수를 평가합니다. 여기에는 SOC, PCI, FedRAMP, HIPAA 등이 포함됩니다.

특정 규정 준수 프로그램 범위의 AWS 서비스 목록은 규정 준수 프로그램 AWS 제공 범위 내 서비스규정 준수 프로그램 . 일반 정보는 AWS 규정 준수 프로그램.

를 사용하여 타사 감사 보고서를 다운로드할 수 있습니다 AWS Artifact. 자세한 내용은 Downloading Reports inDownloading AWS Artifact을 참조하세요.

Amazon Simple Email Service 사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률과 규정에 따라 결정됩니다. AWS 은(는) 규정 준수를 지원할 다음과 같은 리소스를 제공합니다.

- 보안 및 규정 준수 퀵 스타트 가이드 -이 배포 가이드에서는 아키텍처 고려 사항에 대해 설명하고 보안 및 규정 준수 중심 기준 환경을 배포하기 위한 단계를 제공합니다 AWS.
- AWS 규정 준수 리소스 -이 워크북 및 가이드 모음은 산업 및 위치에 적용될 수 있습니다.
- AWS Config 개발자 안내서의 규칙을 사용하여 리소스 평가 - 리소스 구성이 내부 관행, 업계 지침 및 규정을 얼마나 잘 준수하는지 AWS Config평가합니다.
- AWS Security Hub -이 AWS 서비스는 보안 업계 표준 및 모범 사례 준수 여부를 확인하는 데 도움이 AWS 되는 내 보안 상태에 대한 포괄적인 보기를 제공합니다.

Amazon Simple Email Service의 복원성

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다. 리전은 물리적으로 분리되고 격리된 다수의 가용 영역을 제공하며, 이러한 영역은 짧은 지연 시간, 높은 처리량 및 높은 중복성을 갖

춘 네트워크를 통해 연결되어 있습니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 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 참조하세요.

Amazon Simple Email Service의 인프라 보안

관리형 서비스인 Amazon Simple Email Service는 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스 및가 인프라를 AWS 보호하는 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하세요. 인프라 보안 모범 사례를 사용하여 AWS 환경을 설계하려면 보안 원칙 AWS Well-Architected Framework의 인프라 보호를 참조하세요.

AWS 에서 게시한 API 호출을 사용하여 네트워크를 통해 Amazon Simple Email Service에 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.

또한 요청은 액세스 키 ID 및 IAM 위탁자와 관련된 보안 암호 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 자격 증명을 생성하여 요청에 서명할 수 있습니다.

Amazon SES를 사용하여 VPC 엔드포인트 설정

많은 Amazon SES 고객이 내부 시스템에서 퍼블릭 인터넷에 연결하는 기능을 제한하는 기업 정책을 마련하고 있습니다. 이러한 정책은 이러한 고객이 퍼블릭 Amazon SES 엔드포인트를 사용하지 못하도록 합니다.

유사한 정책이 있는 경우 Amazon Virtual Private Cloud를 사용해 이러한 제한 내에서 작업할 수 있습니다. Amazon VPC를 사용하면의 격리된 영역에 있는 가상 네트워크에 AWS 리소스를 배포할 수 있습니다 AWS 클라우드. Amazon VPC에 대한 자세한 내용은 Amazon VPC 사용 설명서를 참조하세요.

안전하고 확장 가능한 방식으로 VPC 엔드포인트를 통해 Amazon VPC에서 SES로 직접 연결할 수 있습니다. 인터페이스 VPC 엔드포인트를 사용하면 아웃바운드 트래픽 방화벽을 열 필요가 없기 때문에

더 나은 보안을 확보할 수 있을 뿐만 아니라 Amazon VPC 엔드포인트 사용의 다른 이점도 얻을 수 있습니다.

VPC 엔드포인트를 사용하면 SES로 향하는 트래픽이 인터넷을 통해 전송되지 않으며 Amazon 네트워크를 벗어나지 않으므로 가용성 위험이나 네트워크 트래픽의 대역폭 제약 없이 VPC에서 SES에 안전하게 연결할 수 있습니다. 다중 계정 인프라 전반에서 SES를 중앙 집중화하고 인터넷 게이트웨이를 사용할 필요 없이 계정에 서비스로 제공할 수 있습니다.

> **제한 사항**
>
> - Amazon SES는 다음 `use1-az2`, `use1-az3`, `use1-az5`, `usw1-az2`, `usw2-az4`, `apne2-az4`, `cac1-az3` 및 `cac1-az4` 가용 영역에서 VPC 엔드포인트를 지원하지 않습니다.
> - VPC 내에서 사용되는 SMTP 엔드포인트는 현재 계정에 사용 중인 AWS 리전 으로 제한됩니다.

Amazon VPC에서 SES를 설정하는 연습 예제

사전 조건

이 섹션의 절차를 완료하려면 먼저 다음 단계를 수행해야 합니다.

- 기존 Virtual Private Cloud(VPC) 보유 또는 새 VPC 생성 절차는 Amazon VPC로 시작하기를 참조하세요.
- VPC에서 이후 단계에서 생성한 VPC 엔드포인트와의 연결을 테스트할 Amazon EC2 인스턴스를 시작합니다. 자세한 내용은 기본 VPC를 참조하세요.

> **Note**
>
> SES용 VPC 엔드포인트는 모든 리소스와 함께 사용할 수 있지만 테스트 방법을 쉽게 하기 위해 이 예제에서는 EC2 인스턴스를 리소스로 사용합니다. Amazon EC2는 기본적으로 포트 25를 통한 이메일 트래픽을 제한하므로 TCP 465, 587, 2465 또는 2587과 같은 TCP 25 이외의 다른 포트를 사용해야 합니다. 자세한 내용은 포트 25를 사용하여 전송된 이메일에 대한 제한을 참조하세요.

Amazon VPC에서 SES 설정

SES와 사용할 VPC 엔드포인트를 설정하는 프로세스는 몇 가지 별도의 단계로 구성됩니다. 먼저 인스턴스가 SMTP 포트와 통신할 수 있도록 하는 보안 그룹을 만든 다음, Amazon SES용 VPC 엔드포인트를 생성하고, 마지막으로 VPC 엔드포인트에 대한 연결을 테스트하여 제대로 구성되었는지 확인해야 합니다.

1단계: 보안 그룹 생성

이 단계에서는 Amazon EC2 인스턴스가 생성할 VPC 인터페이스 엔드포인트와 통신할 수 있도록 하는 보안 그룹을 생성합니다.

보안 그룹을 생성하려면

1. Amazon EC2 콘솔의 탐색 창의 네트워크 및 보안에서 Security Groups(보안 그룹)을 선택합니다.
2. 보안 그룹 생성을 선택합니다.
3. 기본 세부 정보에서 다음을 수행합니다.
   - 보안 그룹 이름에 보안 그룹을 식별하는 고유한 이름을 입력합니다.
   - 설명에 보안 그룹의 목적을 설명하는 텍스트를 입력합니다.
   - VPC에 대해 Amazon SES를 사용할 VPC를 선택합니다.
4. 인바운드 규칙에서 규칙 추가를 선택합니다.
5. 새 인바운드 규칙에 대해 다음을 수행합니다.
   - 유형에 대해 사용자 지정 TCP를 선택합니다.
   - 포트 범위에 이메일을 전송하는 데 사용할 포트 번호를 입력합니다. 다음 **465**, **587**, **2465**, 또는 **2587** 포트 번호를 사용할 수 있습니다.
   - 소스 유형에 대해 사용자 지정을 선택합니다.
   - 소스에서 VPC 엔드포인트를 사용하여 SES 서비스와 통신할 리소스가 포함된 기타 보안 그룹 ID 또는 프라이빗 IP CIDR 범위를 입력합니다.
   - (액세스를 허용하려는 각 CIDR 범위 또는 보안 그룹에 대해 4~5단계를 반복합니다.)
6. 완료되면 보안 그룹 생성을 선택합니다.

2단계: VPC 엔드포인트 생성

Amazon VPC에서 VPC 엔드포인트를 사용하면 VPC를 지원되는 AWS 서비스에 연결할 수 있습니다. 이 예에서 Amazon EC2 보안 그룹이 Amazon SES에 연결할 수 있도록 Amazon VPC를 구성합니다.

VPC 엔드포인트를 생성하려면

1. https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 엽니다.
2. PrivateLink 및 Lattice에서 엔드포인트를 선택합니다.
3. Create Endpoint(엔드포인트 생성)를 선택하여 Create Endpoint(엔드포인트 생성) 페이지를 엽니다.
4. (선택 사항) Endpoint settings(엔드포인트 설정) 패널에서 Name tag(이름 태그) 필드에 태그를 생성합니다.
5. Service category(서비스 범주)에서 AWS services를 선택합니다.
6. Services(서비스) 패널의 검색 표시줄에서 smtp로 필터링한 다음 해당 라디오 버튼을 선택합니다.
7. VPC 패널에서 검색 창 내부를 클릭하고 목록 상자에서 VPC를 선택합니다(the section called “사전 조건” 참조).
8. Subnets(서브넷) 패널에서 Availability Zones(가용 영역) 및 Subnet IDs(서브넷 ID)를 선택합니다.

> **Note**
>
> Amazon SES는 `use1-az2`, `use1-az3`, `use1-az5`, `usw1-az2`, `usw2-az4`, `apne2-az4`, `cac1-az3` 및 `cac1-az4` 가용 영역에서 VPC 엔드포인트를 지원하지 않습니다.

9. Security groups(보안 그룹) 패널에서 이전에 생성한 보안 그룹을 선택합니다.
10. (선택 사항) 태그 패널에서 하나 이상의 태그를 생성할 수 있습니다.
11. Create endpoint(엔드포인트 생성)을 선택합니다. Amazon VPC에서 엔드포인트를 생성하는 동안 약 5분 정도 기다립니다. 엔드포인트를 사용할 준비가 되면 Status(상태) 열의 값이 Available(사용 가능)로 변경됩니다.

(선택 사항) 3단계: VPC 엔드포인트 연결 테스트

VPC 엔드포인트 구성 프로세스를 완료하면 연결을 테스트하여 VPC 엔드포인트가 올바르게 구성되었는지 확인할 수 있습니다. 대부분의 운영 체제에 포함된 명령줄 도구를 사용하여 연결을 테스트할 수 있습니다.

VPC 엔드포인트에 대한 연결을 테스트하려면

1. 방금 생성한 이메일-SMTP VPC 엔드포인트와 동일한 VPC에서 Amazon EC2 인스턴스를 시작합니다.

Linux 인스턴스 연결 방법에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Linux 인스턴스에 연결을 참조하세요.

Windows 인스턴스 연결에 대한 자세한 내용은 Amazon EC2 사용 설명서에서 시작하기 튜토리얼을 참조하세요.

2. 예를 들어 SES SMTP 인터페이스를 사용하여 테스트 이메일을 보낼 수 있습니다.

> **Note**
>
> Amazon SES를 통해 이메일을 보내기 전에 이메일 주소 또는 도메인을 확인해야 합니다. 자격 증명 확인에 대한 자세한 내용은 Amazon SES에서 자격 증명 생성 및 확인 단원을 참조하세요.

Amazon SES 문제 해결

이 섹션에는 문제가 발생했을 때 도움이 될 수 있는 다음의 주제가 포함되어 있습니다.

- 발생할 수 있는 도메인 확인 문제에 대한 자세한 내용은 도메인 및 이메일 주소 확인 문제 단원을 참조하세요.
- DKIM 관련 문제에 대한 해결 방법은 Amazon SES의 DKIM 문제 해결 단원을 참조하세요.
- 이메일을 전송할 때 발생할 수 있는 일반적인 전송 문제 및 각각 취할 수 있는 교정 작업의 목록은 Amazon SES 전송 문제 단원을 참조하세요.
- 수신자가 Amazon SES를 통해 전송된 이메일을 수신할 때 발생할 수 있는 문제에 대한 설명은 Amazon SES로부터 수신하는 이메일에서 발생하는 문제 단원을 참조하십시오.
- 반송 메일, 불만 제기 및 전송 알림 관련 문제에 대한 해결 방법은 Amazon SES 알림 문제 단원을 참조하세요.
- Amazon SES를 통해 이메일을 전송할 때 발생할 수 있는 오류의 목록은 Amazon SES 이메일 전송 오류 단원을 참조하십시오.
- API 또는 SMTP 인터페이스를 사용하여 Amazon SES로 다중 호출을 수행할 때 이메일 전송 속도를 높이는 방법에 대한 팁은 Amazon SES 처리량 증가 단원을 참조하십시오.
- Amazon SES에서 반환하는 SMTP 응답 코드 목록 및 Simple Mail Transfer Protocol(SMTP) 인터페이스를 통해 Amazon SES를 사용할 때 발생할 수 있는 일반적인 문제에 대한 해결 방법은 Amazon SES SMTP 문제 단원을 참조하십시오.
- Amazon SES API v2가 반환하는 공통 오류 코드의 목록은 공통 오류 단원을 참조하십시오.
- 전송 검토 프로세스와 관련된 일반적인 문제 및 이를 처리하는 방법에 대한 설명은 Amazon SES 전송 검토 프로세스 FAQ 단원을 참조하세요.
- DNS 기반 블랙홀 목록(DNSBL)이 Amazon SES를 통한 전송에 미치는 영향에 대한 논의는 DNS 블랙홀 목록(DNSBL) FAQ 단원을 참조하십시오.

Amazon SES API를 직접 호출하는 경우 수신할 수 있는 HTTP 오류는 Amazon Simple Email Service API 참조를 참조하십시오.

> **Note**
>
> 기술 지원을 요청해야 하는 경우이 개발자 안내서의 페이지에 있는 피드백 링크를 사용하지 마세요. 양식은 AWS Support가 아닌 AWS 설명서 팀에서 받기 때문입니다. 대신 문의처 페이지에서 사용 가능한 다양한 지원 옵션을 살펴보세요.

내용

- Amazon SES 일반 문제
- 도메인 및 이메일 주소 확인 문제
- Amazon SES의 DKIM 문제 해결
- Amazon SES 전송 문제
- Amazon SES로부터 수신하는 이메일에서 발생하는 문제
- Amazon SES 알림 문제
- Amazon SES 이메일 전송 오류
- Amazon SES 처리량 증가
- Amazon SES SMTP 문제

Amazon SES 일반 문제

이 페이지의 정보는 Amazon SES를 사용할 때 발생할 수 있는 문제를 설명하며 이를 진단하는 데 도움이 됩니다.

변경 사항이 즉시 표시되는 것은 아닙니다.

사용자들이 전세계 데이터 센터의 컴퓨터들을 통해 액세스하는 서비스인 Amazon SES는 최종 일관성이라고 하는 분산 컴퓨팅 모델을 사용합니다. Amazon SES(또는 기타 AWS 서비스)에서 변경한 내용은 가능한 모든 엔드포인트에서 표시되는 데 시간이 걸립니다. 일부 지연은 서버에서 서버로, 전 세계의 리전에서 리전으로 데이터를 보내는 데 걸리는 시간으로 인해 발생합니다. 대부분의 경우, 이 지연에는 몇 분이 채 소요되지 않습니다.

지연이 발생할 수 있는 영역에는 다음이 포함됩니다.

- 구성 세트 생성 및 수정 – 구성 세트를 생성하거나 수정하는 경우(예: 전용 IP 풀을 기존 구성 세트와 연결)하는 경우) 생성 또는 수정한 시간부터 변경 사항이 활성화될 때까지 잠시 지연될 수 있습니다.
- 이벤트 대상 생성 및 수정 - 이벤트 대상을 생성하거나 수정할 때(예: Amazon SES에 이메일 전송 데이터를 다른 AWS 서비스로 전송하도록 지시하기 위해) 이벤트 대상을 생성하거나 수정한 시간과 이메일 전송 이벤트가 실제로 지정된 대상에 도착하는 시간 사이에 지연이 발생할 수 있습니다.

도메인 및 이메일 주소 확인 문제

Amazon SES에서 도메인 또는 이메일 주소를 확인하려면 Amazon SES 콘솔 또는 Amazon SES API를 사용하여 해당 절차를 시작합니다. 이 단원에는 확인 절차 관련 문제를 해결하는 데 도움이 될 수 있는 정보가 포함되어 있습니다.

> **Note**
>
> 다음 절차에서 DNS 레코드에 대한 참조는 사용한 DKIM 형식에 따라 CNAME 또는 TXT 레코드를 참조할 수 있습니다. Easy DKIM은 CNAME 레코드를 사용하고 BYODKIM은 TXT 레코드를 사용합니다. 각 Easy DKIM 또는BYODKIM에 대한 자세한 확인 절차가 제공됩니다.

일반적인 도메인 확인 문제

the section called “도메인 자격 증명 확인” 내 절차를 사용하여 도메인을 확인하다가 문제가 발생할 경우 아래와 같은 예상 원인 및 해결 방법을 검토합니다.

- 소유하지 않은 도메인을 확인하려고 함 – 소유하지 않은 도메인은 확인할 수 없습니다. 예를 들어 gmail.com 도메인의 주소에서 Amazon SES를 통해 이메일을 보내려면 해당 이메일 주소를 구체적으로 확인해야 합니다. 전체 gmail.com 도메인을 확인할 수 없습니다.
- 프라이빗 도메인을 확인하려고 합니다(You're attempting to verify a private domain) - DNS 레코드를 퍼블릭 DNS를 통해 확인할 수 없는 경우 도메인을 확인할 수 없습니다.
- DNS 공급자가 DNS 레코드 이름에 밑줄을 허용하지 않음 – 소수의 DNS 공급자는 도메인의 레코드 이름에 밑줄(_)을 포함하도록 허용하지 않습니다. 그러나 DKIM 레코드 이름에서 밑줄은 필수입니다. DNS 공급자가 레코드 이름에 밑줄을 입력하는 것을 허용하지 않는 경우 해당 공급자의 고객 지원 팀에 문의하세요.
- DNS 공급자가 DNS 레코드의 끝에 도메인 이름을 추가함 – 일부 DNS 공급자는 DNS 레코드의 속성 이름에 도메인 이름을 자동으로 추가합니다. 예를 들어 속성 이름이 _domainkey.example.com인 레코드를 생성하는 경우 공급자는 도메인 이름을 추가하여 _domainkey.example.com.example.com을 생성할 수 있습니다. 도메인 이름 중복을 방지하려면 DNS 레코드를 입력할 때 도메인 이름 끝에 마침표를 추가하세요. 이 단계에서는 레코드에 도메인 이름을 추가할 필요가 없음을 DNS 공급자에게 알려줍니다.
- DNS 공급자가 DNS 레코드 값 수정 – 일부 공급자는 소문자만 사용하도록 DNS 레코드 값을 자동으로 수정합니다. Amazon SES는 도메인 확인 절차를 시작할 때 Amazon SES가 제공한 값과 속성 값이 정확히 일치하는 확인 레코드를 감지한 경우에만 도메인을 확인합니다. 도메인에 대한 DNS 공급

자가 소문자만 사용하도록 DNS 레코드 값을 변경하는 경우 DNS 공급자에게 연락하여 추가 지원을 요청하세요.

- 동일한 도메인을 여러 번 확인하려는 경우 - 서로 다른 리전에서 전송하거나 동일한 도메인을 사용하여 여러 AWS 계정에서 전송하기 때문에 도메인을 두 번 이상 확인해야 할 수 있습니다. DNS 공급자가 속성 이름이 동일한 DNS 레코드를 두 개 이상 허용하지 않는 경우 두 개의 도메인을 확인할 수 있습니다. DNS 공급자가 이를 허용하는 경우 동일한 DNS 레코드에 여러 속성 값을 할당할 수 있습니다. 예를 들어 Amazon Route 53에서 사용자의 DNS를 관리하는 경우 사용자는 다음 단계를 완료하여 동일한 CNAME 레코드에 여러 값을 설정할 수 있습니다.

  1. Route 53 콘솔에서 첫 번째 리전에서 도메인을 확인할 때 생성한 CNAME 레코드를 선택합니다.
  2. 값 상자에서 기존 속성 값의 끝으로 이동한 다음 Enter를 누릅니다.
  3. 추가 리전에 대한 속성 값을 추가한 다음 레코드 세트를 저장합니다.

  DNS 공급자가 동일한 DNS 레코드에 여러 값을 할당하도록 허용하지 않는 경우 DNS 레코드의 속성 이름에서 도메인을 _domainkey로 한 번 확인한 후 다음 번 확인할 때는 속성 이름에서 _domainkey를 제거합니다. 이 솔루션의 단점은 동일한 도메인을 두 번만 확인할 수 있다는 것입니다.

도메인 확인 설정 확인

Amazon SES 도메인 확인 DNS 레코드가 올바로 DNS 서버에 게시되었는지 다음 절차를 사용하여 확인할 수 있습니다. 이 절차는 Windows 및 Linux에서 사용 가능한 nslookup 도구를 사용합니다. Linux의 경우, dig도 사용할 수 있습니다.

이 지침에서 명령은 Windows 7에서 실행되었으며 사용된 예제 도메인은 CNAME 레코드를 사용하는 Easy DKIM으로 구성된 ses-example.com입니다.

이 절차에서, 먼저 사용자의 도메인에 서비스하는 CNAME 서버를 찾은 후 해당 서버에 TXT 레코드를 보기 위한 쿼리를 전송합니다. 사용자의 도메인에 서비스하는 DNS 서버로 쿼리하는 이유는 이들 서버가 도메인에 대한 가장 최신 정보를 포함하고 있으며, 이 정보가 다른 DNS 서버로 전파되려면 시간이 걸릴 수 있기 때문입니다.

도메인 확인 CNAME 레코드가 올바로 DNS 서버에 게시되었는지 확인하려면

1. 다음 단계에 따라 도메인의 이름 서버를 찾습니다.

   a. 명령줄로 이동합니다. Windows 7에서 명령줄로 이동하려면 [Start]를 선택하고 cmd를 입력하십시오. Linux 기반 운영 체제에서, 터미널 창을 엽니다.

b. 명령 프롬프트에서 다음을 입력합니다. 여기서 <domain>은 사용자의 도메인입니다. 그러면 사용자의 도메인에 서비스하는 모든 이름 서버가 나열됩니다.

```
nslookup -type=NS <domain>
```

도메인이 ses-example.com이라면 이 명령이 다음과 같습니다.

```
nslookup -type=NS ses-example.com
```

이 명령의 출력은 사용자의 도메인에 서비스하는 모든 이름 서버를 나열합니다. 다음 단계에서 이들 서버 중 하나에 쿼리합니다.

2. 다음 단계에 따라 CNAME 레코드가 올바로 게시되었는지 확인합니다. Amazon SES가 Easy DKIM 인증을 위해 세 개의 CNAME 레코드를 생성하므로 세 가지 각각에 대해 다음 절차를 반복하세요.

a. 명령 프롬프트에서 다음을 입력합니다. 여기서 <임의의 문자열>은 SES에서 생성된 CNAME이고, <도메인>은 사용자의 도메인이며, <이름 서버>는 1단계에서 검색한 이름 서버 중 하나입니다.

```
nslookup -type=CNAME <random string>_domainkey.<domain> <name server>
```

ses-example.com 예에서, 1단계에서 검색된 이름 서버가 ns1.name-server.net이고, SES에서 생성된 <임의의 문자열>이 `4hzwn5lmznmmjyl2pqf2agr3uzzzzxyz`라면 다음을 입력합니다.

```
nslookup -type=CNAME 4hzwn5lmznmmjyl2pqf2agr3uzzzzxyz_domainkey.ses-example.com
 ns1.name-server.net
```

b. 명령의 출력에서 `canonical name =` 이하의 문자열이 Amazon SES 콘솔의 자격 증명 목록에서 도메인을 선택하면 표시되는 CNAME 값과 일치하는지 확인합니다.

예에서는 `4hzwn5lmznmmjyl2pqf2agr3uzzzzxyz_domainkey.ses-example.com` 값이 `4hzwn5lmznmmjyl2pqf2agr3uzzzzxyz.dkim.amazonses.com`인 CNAME 레코드를 찾습니다. 이 레코드가 올바로 게시되었다면 명령이 다음을 출력할 것으로 기대할 수 있습니다.

```
4hzwn5lmznmmjyl2pqf2agr3uzzzzxyz_domainkey.ses-example.com canonical name =
 "4hzwn5lmznmmjyl2pqf2agr3uzzzzxyz.dkim.amazonses.com"
```

일반적인 이메일 확인 문제

- 확인 이메일이 도착하지 않음 – 이메일 주소 자격 증명 확인의 절차를 완료했지만 몇 분 내에 확인 이메일을 받지 못하는 경우 다음 단계를 완료합니다.
  - 확인하려는 이메일 주소의 스팸 또는 정크 메일 폴더를 선택합니다.
  - 확인하려는 주소가 이메일을 수신할 수 있는지 확인합니다. 별도의 이메일 주소(개인 이메일 주소 등)를 사용하여 확인하려는 주소로 테스트 이메일을 보냅니다.
  - Amazon SES 콘솔에서 확인된 주소 목록을 확인합니다. 확인하려는 이메일 주소에 오류가 없는지 확인합니다.

Amazon SES의 DKIM 문제 해결

이 단원에서는 Amazon SES에서 DKIM 인증을 구성할 때 발생할 수 있는 몇 가지 문제에 대해 설명합니다. DKIM을 설정하려고 할 때 문제가 발생하면 아래의 가능한 원인과 해결 방법을 검토하세요.

DKIM을 성공적으로 설정했지만 메시지에 DKIM 서명이 없습니다.

Easy DKIM 또는 BYODKIM을 사용하여 도메인에 대한 DKIM을 구성했지만 보내는 메시지에 DKIM 서명이 없는 경우 다음을 수행하세요.

- 해당하는 자격 증명에 대해 DKIM이 활성화되어 있는지 확인합니다. Amazon SES 콘솔에서 자격 증명에 대해 DKIM을 활성화하려면 자격 증명 목록에서 이메일 도메인을 선택합니다. 도메인의 세부 정보 페이지에서 DKIM을 확장하고 활성화를 선택하여 DKIM을 활성화합니다.
- 동일한 도메인의 확인된 이메일 주소에서 보내고 있는지 확인합니다. 도메인에 대해 DKIM을 설정하면 개별적으로 확인한 이메일 주소를 제외하고 해당 도메인에서 보내는 모든 메시지가 DKIM으로 서명됩니다. 개별 확인된 이메일 주소는 별도의 설정을 사용합니다. 예를 들어 example.com 도메인에 대해 DKIM을 구성하고 이메일 주소 mary@example.com을 개별적으로 확인한 경우(단, 주소에 대해 DKIM을 구성하지는 않은 경우) mary@example.com에서 보낸 이메일은 DKIM 인증 없이 전송됩니다. 계정의 자격 증명 목록에서 이메일 주소 자격 증명을 삭제하여 이 문제를 해결할 수 있습니다.
- 둘 이상의 AWS 리전에서 동일한 자격 증명을 사용하는 경우 각 리전에 대해 DKIM을 별도로 구성해야 합니다. 마찬가지로 두 개 이상의 AWS 계정에서 동일한 도메인을 사용하는 경우 각 계정

에 대해 DKIM을 구성해야 합니다. 특정 리전 또는 계정에서 필요한 DNS 레코드를 제거할 경우 Amazon SES가 DKIM 서명을 해당 리전 또는 계정에 대해서만 비활성화합니다. DKIM 서명이 비활성화되면 Amazon SES에서 이메일로 알림을 보냅니다.

Amazon SES 콘솔에서 도메인의 DKIM 세부 정보에 DKIM: 발신자 확인을 기다리는 중... DKIM 확인 상태: 확인 대기 중이 표시됩니다.

Easy DKIM 또는 BYODKIM - 자체 DKIM 사용의 절차를 완료하여 도메인에 대해 DKIM을 구성했지만 Amazon SES 콘솔에 여전히 DKIM 확인 대기 중으로 표시되는 경우 다음을 수행하십시오.

- 최대 72시간이 걸릴 수 있습니다. 드문 경우이지만 DNS 레코드가 Amazon SES에 표시되기까지 시간이 걸릴 수 있습니다.
- CNAME 레코드(Easy DKIM의 경우) 또는 TXT 레코드(BYODKIM의 경우)가 올바른 이름을 사용하는지 확인합니다. 일부 DNS 공급자는 생성한 레코드에 도메인 이름을 자동으로 추가합니다. 예를 들어, 이름이 `example._domainkey.example.com`인 레코드를 생성하는 경우 DNS 공급자가 이 문자열의 끝에 도메인 이름을 추가하여 `example._domainkey.example.com.example.com` 결과를 생성할 수 있습니다. 자세한 내용은 DNS 공급자의 설명서를 참조하세요.

Amazon SES에서 DKIM 설정이 취소되었다(또는 취소될 예정)는 이메일이 전송됩니다.

이는 Amazon SES가 더 이상 DNS 서버에서 필요한 CNAME 레코드(Easy DKIM을 사용한 경우) 또는 필요한 TXT 레코드(BYODKIM을 사용한 경우)를 찾을 수 없음을 의미합니다. 이 알림 이메일은 DKIM 설정이 취소 상태로 되고 DKIM 서명이 비활성화되기 전에 DNS 레코드를 다시 게시해야 하는 시간을 안내합니다. DKIM 설정이 취소되면 DKIM 설정 절차를 처음부터 다시 시작해야 합니다.

BYODKIM을 설정하려고 하면 DKIM 확인 프로세스가 실패합니다.

프라이빗 키가 올바른 형식을 사용하는지 확인합니다. 프라이빗 키는 PKCS #1 또는 PKCS #8 형식이어야 하며 1024비트 또는 2048비트 RSA 암호화를 사용해야 합니다. 또한 프라이빗 키는 base64로 인코딩되어야 합니다.

BYODKIM을 설정하는 동안 도메인에 대한 퍼블릭 키를 지정하려고 하면 **`BadRequestException`** 오류가 발생합니다.

`BadRequestException` 오류가 발생하면 다음을 수행하세요.

- 퍼블릭 키에 대해 지정하는 선택기에 1 ~ 63자의 영숫자 문자가 포함되어 있는지 확인합니다. 선택기에는 마침표나 다른 기호 또는 구두점을 포함할 수 없습니다.
- 퍼블릭 키에서 머리글, 바닥글 및 모든 줄 바꿈을 제거했는지 확인합니다.

Easy DKIM을 사용하는 경우 DNS 서버는 Amazon SES DKIM CNAME 레코드를 성공적으로 반환하지만 도메인 확인 TXT 레코드에 대해 **SERVFAIL**을(를) 반환합니다.

> DNS 공급자가 CNAME 레코드를 리디렉션하지 못할 수도 있습니다. AmazonSES 및 ISP 쿼리는 TXT 레코드를 쿼리합니다. DKIM 사양을 준수하기 위해 DNS 서버는 TXT 레코드 쿼리 및 CNAME 레코드 쿼리 모두에 응답해야 합니다. DNS 공급자가 TXT 레코드 쿼리에 응답할 수 없는 경우 대안은 DNS 호스팅 공급자로 Route 53을 사용하는 것입니다.

이메일에 두 개의 DKIM 서명이 포함되어 있습니다.

> `d=amazonses.com`을 포함하는 추가 DKIM 서명은 Amazon SES에서 자동으로 추가됩니다. 이 서명은 무시할 수 있습니다.

Amazon SES 전송 문제

Amazon SES로 요청이 성공하면 메시지는 대부분 즉시 전송됩니다. 일부의 경우에는 약간의 지연이 있을 수 있습니다. 어떤 경우에도 이메일이 전송된다는 확신을 할 수 있습니다.

하지만 Amazon SES가 메시지를 전송할 때 여러 요인에 의해 전송이 실패할 수 있으며, 일부 경우에는 전송한 메시지가 도달하지 않아야만 전송 실패를 인지할 수 있습니다. 다음 프로세스를 사용하여 이러한 상황을 해결합니다.

이메일이 도달하지 않을 경우 다음을 시도합니다.

- 문제의 이메일에 대해 `SendEmail` 또는 `SendRawEmail` 요청을 전송했는지, 올바른 응답을 수신했는지 확인합니다. 프로그래밍 방식으로 이러한 요청을 생성하는 경우 소프트웨어 로그에서 프로그램이 요청을 전송하고 올바른 응답을 수신했는지 확인합니다.
- 문제가 실제로 미전송이 아니라 지연일 수 있으므로 블로그 기사 Three places where your email could get delayed when sending through SES를 참조하세요.
- 발신자의 이메일 주소("From" 주소)가 유효한 주소인지 확인합니다. 또한 반송 메일이 전송되는 "Return-Path" 주소도 확인합니다. 이메일이 반송된 경우 오류 메시지가 설명을 제공합니다.
- AWS Service Health Dashboard를 확인하여 Amazon SES에 알려진 문제가 없는지 확인하십시오.
- 이메일 수신자 또는 수신자의 ISP에게 문의합니다. 수신자가 올바른 이메일 주소를 사용하는지 확인하고, 수신자의 ISP에서 전송 문제가 발생한 적이 있는지 문의합니다. 또한 이메일이 도달했지만 스팸으로 필터링되었는지도 확인합니다.
- 유료 AWS Support 플랜에 가입한 경우 새 기술 지원 케이스를 열 수 있습니다. AWS와 대화를 나눌 때 관련 수신자 주소와 함께 `SendEmail` 또는 `SendRawEmail` 응답에서 반환된 요청 ID나 메시지 ID를 알려주시기 바랍니다.

- 문제가 영구적 전송 실패가 아니라 실제로는 지연인지 시간을 두고 확인합니다. 스패머에 대항하기 위해 일부 ISP는 알 수 없는 발송 메일 서버로부터의 수신 메시지를 일시적으로 거부합니다. 그레이리스팅이라고 하는 이 프로세스 때문에 전송 지연이 발생할 수 있습니다. Amazon SES가 이러한 메시지를 다시 시도합니다. 그레이리스팅이 문제인 경우 ISP가 이러한 재시도 중 한 번에서 이메일을 수락할 수 있습니다.
- 고객의 이익을 가장 먼저 생각한다고 해도 메시지 발송률에 영향을 미치는 상황은 언제든지 발생할 수 있습니다. the section called “긍정적인 발신자 평판 유지”에서 이메일 커뮤니케이션이 의도한 대상에게 확실히 도달할 수 있도록 하는 방법을 참조하세요.

Amazon SES로부터 수신하는 이메일에서 발생하는 문제

이 섹션에서는 Amazon SES에서 보낸 이메일을 받을 때 발생할 수 있는 몇 가지 일반적인 문제에 대해 설명합니다.

이메일 클라이언트는 이메일 소스로 “amazonses.com을 경유하여 전송”을 표시합니다.

일부 이메일 클라이언트는 발신자의 도메인이 이메일을 보낸 도메인(이 경우 amazonses.com)과 일치하지 않으면 “경유” 도메인을 표시합니다. 자세한 내용은 Gmail 지원 웹사이트에서 발신자 이름 옆의 추가 정보를 참조하세요. 또는 DomainKeys Identified Mail(DKIM)을 설정할 수 있습니다. DKIM을 사용하여 이메일을 인증할 경우, 일반적으로 이메일 클라이언트는 "경유" 도메인을 표시하지 않습니다. DKIM 서명이 이메일이 해당 도메인으로부터 전송되었음을 증명하기 때문입니다. DKIM 설정에 대한 자세한 내용은 Amazon SES에서 DKIM을 사용하여 이메일 인증 단원을 참조하세요.

> **Note**
>
> SES 사용자로부터 스팸 또는 기타 원치 않는 이메일 메시지를 받은 경우 이메일 클라이언트의 스팸 신고 도구를 사용하고 문의하기 아래에 나온 SES 이메일 침해 신고 단계를 따릅니다.

메시지에 깨진 문자 또는 의미 없는 문자가 포함되어 있습니다.

메시지에 ASCII 문자 집합에 없는 문자(예: 억양 표시가 되어 있는 라틴 문자, 중국어 문자 또는 아랍어 문자)가 포함되어 있는 경우 HTML 문자 인코딩을 사용하여 해당 문자를 인코딩해야 합니다. Email On Acid 웹사이트의 HTML 문자 변환기와 같은 웹 기반 도구를 사용하여 이메일의 문자를 인코딩할 수 있습니다.

또는 MIME 메시지를 직접 수집할 수 있습니다. MIME 메시지에서 메시지가 UTF-8 인코딩을 사용하도록 지정할 수 있습니다. UTF-8 인코딩을 사용하는 경우 메시지에서 ASCII가 아닌 문자를 직접 사용할 수 있습니다. MIME 메시지 생성을 마쳤으면 SendRawEmail API 또는 SendMail API v2를 사용하여 메시지를 보낼 수 있습니다.

이 문제의 일반적인 원인 중 하나는 Microsoft Word의 스마트 따옴표 기능입니다. Word에서 콘텐츠를 복사하여 이메일에 붙여 넣는 경우가 많으면 이 문제가 발생할 수 있습니다. 스마트 따옴표 기능은 곧은 따옴표 문자("...")를 둥근 따옴표 문자(“...”)로 바꿉니다. 둥근 따옴표 문자는 표준 ASCII 문자가 아닙니다. 결과적으로 일부 이메일 클라이언트에서 "??"로 렌더링될 수 있습니다. 또는 “â€œ”와 같은 문자 그룹으로 렌더링될 수 있습니다. 이 문제를 해결하려면 Word에서 스마트 따옴표 기능을 사용하지 않도록 설정하면 됩니다. 또는 이전 단락의 SendRaweMail 솔루션을 사용할 수 있습니다. 이 기능을 사용하지 않도록 설정하는 방법에 대한 자세한 내용은 Microsoft Office 지원 웹사이트에서 Word의 스마트 따옴표를 참조하세요.

Amazon SES 알림 문제

반송 메일, 수신 거부 또는 전송 알림에서 문제가 발생할 경우 아래의 예상 원인 및 해결 방법을 검토합니다.

- Amazon SNS를 통해 반송 메일 알림을 수신하지만 알림이 어떤 수신자에게 해당하는지는 알 수 없음—향후에는 반송 메일 알림을 특정 수신자와 연결하기 위해 다음 옵션을 사용할 수 있습니다.
  - Amazon SES는 사용자가 추가한 사용자 지정 메시지 ID를 유지하지 않으므로 Amazon SES가 이메일을 수락할 때 사용자에게 반환하는 Amazon SES 메시지 ID와 식별자 사이의 매핑을 저장합니다.
  - 각 Amazon SES 호출에서, 여러 수신자에게 단일 메시지를 전송하는 대신 단일 수신자에게 전송합니다.
  - 이메일을 통한 피드백 전달을 활성화할 수 있습니다. 그러면 전체 반송 메일 메시지가 사용자에게 전달됩니다.
- Amazon SNS 또는 이메일 피드백 전달을 통해 수신 거부 또는 전송 알림을 수신하지만 알림이 어떤 수신자에게 해당하는지는 알 수 없음 - 일부 ISP는 Amazon SES로 수신 거부 알림을 전달하기 전에 수신 거부한 수신자의 이메일 주소를 수정합니다. 수신자의 이메일 주소를 찾기 위한 최선의 옵션은 Amazon SES가 이메일을 수락할 때 사용자에게 반환하는 Amazon SES 메시지 ID와 식별자 사이의 자체 매핑을 저장하는 것입니다. Amazon SES는 사용자가 추가하는 사용자 지정 메시지 ID를 보유하지 않습니다.
- 소유하지 않은 Amazon SNS 주제로 이동하는 알림을 설정하려는 경우—주제의 소유자가 사용자 계정이 해당 주제에 대한 `SNS:Publish` 작업을 호출하도록 허용하는 Amazon SNS 액세스 정책을 구

성해야 합니다. IAM 정책 사용을 통해 Amazon SNS 주제에 대한 액세스를 제어하는 방법에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 가이드의 Amazon SNS 주제에 대한 액세스 관리를 참조하십시오.

Amazon SES 이메일 전송 오류

이 주제는 Amazon SES를 통해 이메일을 전송할 때 발생할 수 있는 이메일 전송 고유 오류에 대해 알아봅니다. Amazon SES를 통해 이메일을 전송하려 할 때 Amazon SES 호출이 실패할 경우, Amazon SES가 애플리케이션으로 오류 메시지를 반환하고 이메일을 전송하지 않습니다. 이 오류 메시지를 확인하는 방법은 Amazon SES를 호출하는 방법에 따라 달라집니다.

- Amazon SES API를 직접 호출하는 경우 쿼리 작업이 오류를 반환합니다. 오류는 `MessageRejected`이거나 Amazon Simple Email Service API 참조의 일반 오류 주제에 지정된 오류 중 하나일 수 있습니다.
- 예외를 지원하는 프로그래밍 언어를 사용하는 AWS SDK를 사용하여 Amazon SES를 호출하면 Amazon SES에서 예외가 발생할 수 있습니다. 예외의 유형은 SDK 및 오류에 따라 달라집니다. 예를 들어, 예외는 Amazon SES`MessageRejectedException`(실제 이름은 SDK에 따라 다를 수 있음) 또는 일반 AWS 예외일 수 있습니다. 예외의 유형과 상관없이 예외의 오류 유형 및 오류 메시지를 통해 더 많은 정보를 확인할 수 있습니다.
- SMTP 인터페이스를 통해 Amazon SES를 호출하는 경우 애플리케이션에 따라 오류를 경험하는 방식이 달라집니다. 일부 애플리케이션은 구체적 오류 메시지를 표시할 수 있지만 다른 애플리케이션은 그렇지 않을 수 있습니다. Amazon SES에서 반환하는 SMTP 응답 코드 목록은 Amazon SES에서 반환하는 SMTP 응답 코드 단원을 참조하십시오.

> **Note**
>
> 이메일을 전송하기 위한 Amazon SES 호출이 실패할 경우 해당 이메일에 대해서는 비용이 부과되지 않습니다.

다음은 이메일을 전송할 때 Amazon SES가 오류를 반환할 수 있는 Amazon SES 고유 문제의 유형입니다. 이러한 오류는 Amazon Simple Email Service API 참조의 일반 AWS 오류 주제에 지정된 `MalformedQueryString` 것과 같은 일반 오류에 추가됩니다.

- 이메일 주소가 확인되지 않음. 다음 자격 증명이 region 리전에서 identity1, identity2, identity3을 확인하는 데 실패했습니다. - Amazon SES에서 확인되지 않은 이메일 주소 또는 도메인에서 이메일을

보내려고 합니다. 이 오류는 "From", "Source", "Sender" 또는 "Return-Path" 주소에 해당할 수 있습니다. 계정이 여전히 Amazon SES 샌드박스에 있을 경우 Amazon SES 메일박스 시뮬레이터에서 제공하는 수신자를 제외한 모든 수신자 이메일 주소를 확인해야 합니다. Amazon SES에서 실패한 자격 증명을 전부 표시할 수 없는 경우 오류 메시지가 말줄임표로 끝납니다.

> **Note**
>
> Amazon SES에는 엔드포인트가 여러 AWS 리전 개 있으며 이메일 주소 확인 상태는 각각 다릅니다 AWS 리전. 사용 AWS 리전 하려는의 각 발신자에 대한 확인 프로세스를 완료해야 합니다.

- 계정이 일시 중지됨—사용자 계정의 이메일 전송 기능이 일시 중지됩니다. Amazon SES 콘솔에 계속 액세스하여 대부분의 작업을 수행할 수는 있습니다. 그러나 이메일을 전송할 경우 이 메시지를 받습니다.

  사용자 계정의 이메일 전송 기능이 일시 중지되면 AWS 계정과 연결된 이메일 주소로 알림이 자동 전송됩니다. 자세한 내용은 the section called “전송 검토 프로세스 FAQ” 단원을 참조하십시오.

- 스로틀링 - 애플리케이션이 초당 너무 많은 메시지를 전송하려고 시도하고 있거나 최근 24시간 동안 너무 많은 이메일을 전송했을 수 있습니다. 이러한 경우 오류 메시지는 다음 예와 유사할 수 있습니다.
  - 일일 메시지 할당량 초과—24시간 기준으로 허용된 최대 메시지 수를 전송했습니다. 일일 할당량을 초과한 경우 다음 24시간 기간까지 기다렸다가 추가로 이메일을 전송할 수 있습니다.
  - 최대 송신률 초과 - 최대 송신률이 허용하는 초당 이메일보다 많은 이메일을 전송하려 했습니다. 전송 속도를 초과한 경우 계속 이메일을 전송할 수는 있지만 전송 속도를 낮춰야 합니다. 자세한 내용은 AWS 메시징 및 타겟팅 블로그의 "제한 - 최대 전송 속도 초과" 오류를 처리하는 방법을 참조하세요.
  - 최대 SigV2 SMTP 전송 속도 초과 - 2019년 1월 10일 이전에 생성된 SMTP 자격 증명을 사용하여 메시지를 보내려고 합니다. SMTP 자격 증명은 이전 버전의 AWS 서명을 사용하여 생성되었습니다. 보안을 위해 이 날짜 이전에 만든 자격 증명을 삭제하고 새로운 자격 증명으로 교체하세요. IAM 콘솔을 사용하여 이전 자격 증명을 삭제할 수 있습니다. 자격 증명 생성에 대한 자세한 내용은 the section called “SMTP 자격 증명 획득” 단원을 참조하십시오.

  정기적으로 전송 활동을 모니터링하여 얼마나 발신 할당량에 근접해 있는지 확인해야 합니다. 자세한 내용은 Amazon SES 발신 할당량 모니터링 단원을 참조하세요. 발신 할당량에 대한 일반적인 정보는 Amazon SES 발신 한도 관리 단원을 참조하세요. 발신 할당량을 높이는 방법에 대한 자세한 내용은 Amazon SES 발신 할당량 높이기 단원을 참조하세요.

> **Important**
>
> 스로틀링 오류를 설명하는 오류 텍스트가 일일 할당량 또는 최대 전송 속도 초과와 관련되지 않은 경우 시스템 전체 문제로 인해 전송 용량이 감소한 것일 수 있습니다. 서비스 상태에 대한 자세한 내용은 AWS Service Health Dashboard를 참조하십시오.

- 지정된 수신자가 없음—수신자를 입력하지 않았습니다.
- 이메일 주소에 비 ASCII 문자가 포함되어 있음 - 이메일 주소 문자열은 7비트 ASCII여야 합니다. 주소의 도메인 부분에 유니코드 문자가 포함된 이메일 주소와 메시지를 주고받으려면 퓨니코드를 사용하여 도메인을 인코딩해야 합니다. 이메일 주소의 로컬 부분(@ 기호 앞부분)은 물론 "대화명"에도 퓨니코드를 사용할 수 없습니다. "대화명"에 유니코드 문자를 사용하려면 Amazon SES API v2를 사용하여 원시 이메일 보내기에서 설명한 것과 같이 MIME 인코딩된 단어 구문을 사용하여 "대화명"을 인코딩해야 합니다. 퓨니코드에 대한 자세한 내용은 RFC 3492를 참조하세요.
- Mail FROM 도메인이 확인되지 않음 - Amazon SES가 지정된 MAIL FROM 도메인을 사용하는 데 필요한 MX 레코드를 읽지 못했습니다. 사용자 지정 MAIL FROM 도메인 설정에 대한 자세한 내용은 사용자 지정 MAIL FROM 도메인 사용 단원을 참조하세요.
- 구성 세트가 존재하지 않음 - 사용자가 지정한 구성 세트가 존재하지 않습니다. 구성 세트는 이메일 전송 이벤트를 게시하는 데 사용하는 선택적 파라미터입니다. 자세한 내용은 Amazon SES 이벤트 게시를 사용하여 이메일 전송 모니터링 단원을 참조하세요.

Amazon SES 처리량 증가

이메일을 전송할 때 최대 전송 속도가 허용하는 한 얼마든지 Amazon SES를 호출할 수 있습니다. (최대 전송 속도에 대한 자세한 내용은 Amazon SES 발신 한도 관리를 참조하세요.) 하지만 각 Amazon SES 호출은 완료하려면 시간이 걸립니다.

Amazon SES API 또는 SMTP 인터페이스를 사용하여 Amazon SES를 여러 번 호출할 경우 다음 팁이 처리량을 개선하는 데 도움이 될 수 있습니다.

- 현재 성능을 측정하여 병목을 식별—한 가지 가능한 성능 테스트는 여러 개의 테스트 이메일을 애플리케이션의 코드 루프에서 최대한 빠르게 전송하는 것입니다. 각 `SendEmail` 요청의 왕복 지연 시간을 측정합니다. 그런 다음 동일 시스템에서 애플리케이션의 인스턴스를 점진적으로 추가하며 실행하고 네트워크 지연 시간에 대한 영향을 관찰합니다. 시스템 리소스 병목 또는 네트워크 병목이 존재하는지 정확히 식별하기 위해 이 테스트를 여러 시스템 또는 여러 네트워크에서 실행할 수도 있습니다.

- (API만 해당) 영구 HTTP 연결 사용을 고려—각 API 요청에 대해 별도의 새 HTTP 연결을 설정하는 오버헤드를 감수하는 대신 영구 HTTP 연결을 사용합니다. 즉, 여러 API 요청에서 동일한 HTTP 연결을 재사용합니다.
- 다중 스레드 사용을 고려—애플리케이션이 단일 스레드를 사용할 때 애플리케이션 코드가 Amazon SES API를 호출한 후 API 응답을 동기 방식으로 대기합니다. 이메일 전송은 일반적으로 I/O 바운드 작업이며 여러 스레드에서 작업을 수행하면 처량이 개선됩니다. 원하는 만큼 여러 개의 스레드를 사용하여 동시에 발송할 수 있습니다.
- 다중 프로세스 사용을 고려—다중 프로세스를 사용하면 동시에 활성화되는 Amazon SES 연결이 늘어나므로 처리량이 증가할 수 있습니다. 예를 들어 원하는 이메일을 여러 버킷으로 분할한 후 여러 인스턴스의 이메일 전송 스크립트를 동시에 실행합니다.
- 로컬 메일 릴레이 사용을 고려—애플리케이션은 메시지를 신속하게 로컬 메일 서버로 전송할 수 있습니다. 그러면 메시지를 버퍼링하고 비동기 방식으로 Amazon SES로 전송하는 데 도움이 될 수 있습니다. 일부 메일 서버는 전송 동시성을 지원합니다. 즉 애플리케이션이 단일 스레드 방식으로 이메일을 메일 서버로 전송하더라도 메일 서버는 Amazon SES로 전송할 때 다중 스레드를 사용합니다. 자세한 내용은 기존 이메일 서버와 Amazon SES 통합 단원을 참조하십시오.
- 애플리케이션을 Amazon SES API 엔드포인트에 더 가까이 호스팅하는 것을 고려— 애플리케이션을 Amazon SES API 엔드포인트와 인접한 데이터 센터 또는 Amazon SES API 엔드포인트와 동일한 AWS 리전의 Amazon EC2 인스턴스에서 호스팅하는 것을 고려할 수 있습니다. 그러면 애플리케이션과 Amazon SES 사이의 네트워크 지연 시간이 감소하여 처리량 증가에 도움이 될 수 있습니다. Amazon SES를 사용할 수 있는 리전 목록은 AWS 일반 참조의 Amazon Simple Email Service(Amazon SES)를 참조하세요.
- 다중 시스템 사용을 고려—호스트 시스템의 시스템 구성에 따라 단일 IP 주소와의 동시 HTTP 연결 수에 제한이 있을 수 있습니다. 이로 인해 단일 시스템에서 허용되는 동시 연결 수를 초과하면 병렬 연결의 이점이 제한될 수 있습니다. 이것이 병목이라면 여러 시스템을 사용한 동시 Amazon SES 요청을 고려할 수 있습니다.
- SMTP 엔드포인트 대신 Amazon SES 쿼리 API 사용을 고려 - Amazon SES 쿼리 API를 사용하면 단일 네트워크 호출을 통해 이메일 전송 요청을 제출할 수 있지만 SMTP 엔드포인트와의 인터페이스는 여러 네트워크 요청(예: EHLO, MAIL FROM, RCPT TO, DATA, QUIT)으로 구성된 SMTP 변환을 포함합니다. Amazon SES 쿼리 API에 대한 자세한 내용은 Amazon SES API를 사용하여 이메일 보내기 단원을 참조하십시오.
- Amazon SES 메일박스 시뮬레이터를 사용하여 최대 처리량을 테스트 - 구현하려는 변경 사항을 테스트하기 위해 메일박스 시뮬레이터를 사용할 수 있습니다. 메일박스 시뮬레이터를 사용하면 일일 발신 할당량을 소진하지 않고 시스템의 최대 처리량을 측정할 수 있습니다. 사서함 시뮬레이터에 대한 자세한 내용은 수동으로 메일박스 시뮬레이터 사용 단원을 참조하세요.

SMTP 인터페이스를 통해 Amazon SES에 액세스하는 경우, 처리량에 영향을 미칠 수 있는 특정 SMTP 관련 문제는 Amazon SES SMTP 문제 단원을 참조하십시오.

Amazon SES SMTP 문제

이 단원에는 Amazon SES Simple Mail Transfer Protocol(SMTP) 인터페이스를 통한 이메일 전송과 관련된 몇 가지 일반적인 문제에 대한 해결 방법이 나와 있습니다. 또한 Amazon SES에서 반환하는 SMTP 응답 코드 목록도 포함되어 있습니다.

Amazon SES SMTP 인터페이스를 통한 이메일 전송에 대한 자세한 내용은 Amazon SES SMTP 인터페이스를 사용하여 이메일 보내기 단원을 참조하십시오.

- Amazon SES SMTP 엔드포인트에 연결할 수 없습니다.

  Amazon SES SMTP 엔드포인트 연결 문제는 대부분 다음 문제와 관련이 있습니다.

  - 잘못된 자격 증명 - SMTP 엔드포인트에 연결하는 데 사용하는 자격 증명은 자격 AWS 증명과 다릅니다. SMTP 자격 증명을 얻으려면 Amazon SES SMTP 자격 증명 획득 단원을 참조하세요. 자격 증명에 대한 자세한 내용은 Amazon SES 자격 증명 유형을 참조하세요.
  - 네트워크 또는 방화벽 문제 – 네트워크가 이메일을 전송하려는 포트를 통한 아웃바운드 연결을 차단한 것일 수 있습니다. 로컬 네트워크의 문제로 인해 연결 문제가 발생하는지 확인하려면 *port*를 사용하려는 포트(일반적으로 465, 587, 2465 또는 2587)로 바꿔 명령줄에 `telnet email-smtp.us-west-2.amazonaws.com` *port* 명령을 입력합니다.

    이 명령을 사용하여 SMTP 서버에 연결할 수 있으며 TLS 래퍼 또는 STARTTLS를 사용하여 Amazon SES에 연결하려는 경우 명령줄을 사용하여 Amazon SES SMTP 인터페이스에 대한 연결 테스트에 나온 절차를 완료합니다.

    `telnet` 또는 `openssl`을(를) 사용하여 Amazon SES SMTP 엔드포인트에 연결할 수 없는 경우 네트워크 안의 무언가(예: 방화벽)가 사용하려는 포트를 통한 아웃바운드 연결을 차단하는 것입니다. 네트워크 관리자와 함께 문제를 진단 및 해결하세요.

- Amazon EC2 인스턴스에서 포트 25를 사용하여 Amazon SES로 보내려는데 시간 초과 오류를 수신합니다.

  Amazon EC2는 기본적으로 포트 25를 제한합니다. 이러한 제한을 제거하려면 이메일 전송 제한 사항 제거를 위한 Amazon EC2 요청을 제출하세요. 포트 465 또는 포트 587을 사용하여 Amazon SES에 연결할 수도 있으며 이들 포트에서는 연결이 제한되지 않습니다.

- 네트워크 오류 때문에 이메일이 삭제됨

애플리케이션이 Amazon SES SMTP 엔드포인트에 연결할 때 재시도 로직을 사용하는지, 애플리케이션이 네트워크 오류 시 메시지 전송을 감지 및 재시도할 수 있는지 확인합니다. SMTP는 버보스 프로토콜이며 이 프로토콜을 사용하여 이메일을 보내려면 네트워크를 여러 번 왕복해야 합니다. SMTP의 특성상 네트워크 오류의 가능성이 증가합니다.

- SMTP 엔드포인트와 연결이 끊김.

  연결 끊김은 주로 다음 문제로 인해 발생합니다.

  - MTU 크기 – 시간 초과 오류 메시지를 수신하는 경우 Amazon SES SMTP 인터페이스와 연결하기 위해 사용 중인 컴퓨터의 네트워크 인터페이스의 최대 전송 단위(MTU)가 너무 큰 것일 수 있습니다. 이 문제를 해결하려면 해당 컴퓨터의 MTU 크기를 1,500바이트로 설정합니다.

    Windows, Linux 및 macOS 운영 체제에서 MTU 크기를 설정하는 방법에 대한 자세한 내용은 Amazon Redshift 관리 가이드의 쿼리는 클라이언트에서 중단되고 클러스터에 도달하지 않는 것으로 나타남을 참조하세요.

    Amazon EC2 인스턴스의 MTU 크기 설정에 대한 자세한 내용은 Amazon EC2 사용 설명서의 EC2 인스턴스에 대한 네트워크 최대 전송 단위(MTU)를 참조하세요.

  - 장기간 연결 – Amazon SES SMTP 엔드포인트는 Elastic Load Balancer(ELB) 뒤의 Amazon EC2 인스턴스 플릿에서 실행됩니다. 시스템이 최신 상태 및 내결함성을 유지하기 위해 활성 Amazon EC2 인스턴스가 주기적으로 종료되고 새 인스턴스로 교체됩니다. 애플리케이션이 ELB를 통해 Amazon EC2 인스턴스와 연결되므로 Amazon EC2 인스턴스가 종료되면 연결이 무효화됩니다. 단일 SMTP 연결을 통해 고정된 수의 메시지를 전송한 후, 또는 SMTP 연결이 일정 시간 활성 상태를 유지한 후 새 SMTP 연결을 설정해야 합니다. 애플리케이션이 호스팅되는 위치와 Amazon SES로 이메일을 제출하는 방식에 따라 적절한 임계값을 찾는 실험이 필요합니다.

- 네트워크에서 IP 주소 허용 목록을 작성할 수 있도록 Amazon SES SMTP 메일 서버의 IP 주소를 알고 싶은 경우.

  Amazon SES SMTP 엔드포인트의 IP 주소는 로드 밸런서 뒤에 있습니다. 따라서 이 IP 주소는 자주 바뀝니다. Amazon SES 엔드포인트의 모든 IP 주소에 대한 명확한 목록을 제공하는 것은 불가능합니다. 개별 IP 주소 허용 목록을 작성하는 대신 `amazonses.com` 도메인 허용 목록을 작성하는 것이 좋습니다.

Amazon SES에서 반환하는 SMTP 응답 코드

이 단원에는 Amazon SES SMTP 인터페이스에서 반환하는 응답 코드 목록이 나와 있습니다.

400 오류가 수신되는 SMTP 요청을 재시도해야 합니다. 갈수록 더 오래 대기했다가 요청을 재시도하도록 시스템을 구현하는 것이 좋습니다. 예를 들어, 재시도하기 전에는 5초를 대기하고, 그 다음 재시도 시에는 10초를, 그 다음에는 30초를 대기하는 식입니다. 세 번째 재시도에 실패하면 20분을 기다린 다음 이러한 프로세스를 반복합니다. 지수 재시도 정책을 사용하는 구현의 예를 보려면 AWS 메시징 및 타겟팅 블로그에서 '제한 - 최대 전송 속도 초과' 오류를 처리 하는 방법을 참조하십시오.

> **Note**
>
> AWS SDKs 재시도 로직을 자동으로 구현하지만 SMTP 대신 HTTPS 인터페이스를 사용합니다.

500 오류가 발생하면 요청을 다시 제출하기 전에 요청을 수정해 문제를 해결해야 합니다. 예를 들어 AWS 인증 자격 증명이 유효하지 않은 경우 요청을 다시 제출하기 전에 올바른 자격 증명을 사용하도록 애플리케이션을 업데이트해야 합니다.

설명	응답 코드	추가 정보
인증 성공	`235 Authentication successful`	SMTP 클라이언트가 성공적으로 연결되어 SMTP 서버에 로그인했습니다.
전송 성공	`250 Ok` *`MessageID`*	*`MessageID`* 는 Amazon SES가 메시지를 식별하기 위해 사용하는 고유한 문자열입니다.
서비스 사용 불가	`421 Too many concurrent SMTP connections`	현재, SMTP 서버에 대한 연결이 너무 많아 Amazon SES가 요청을 처리할 수 없습니다.
로컬 처리 오류	`451 Temporary service failure`	Amazon SES에서 요청을 처리할 수 없습니다. 요청이 처리되지 못하도록 막는 요청 관련 문제가 있을 수 있습니다.
제한 시간	`451 Timeout waiting for data from client`	요청 간에 너무 많은 시간이 경과하여 SMTP 서버가 연결을 차단했습니다.

설명	응답 코드	추가 정보
일일 전송 할당량을 초과함	`454 Throttling failure: Daily message quota exceeded`	Amazon SES가 24시간 기준으로 전송을 허용한 최대 이메일 수를 초과했습니다. 자세한 내용은 Amazon SES 발신 한도 관리 단원을 참조하십시오.
최대 전송 속도를 초과함	`454 Throttling failure: Maximum sending rate exceeded`	Amazon SES가 초당 전송할 수 있도록 허용한 최대 이메일 수를 초과했습니다. 자세한 내용은 Amazon SES 발신 한도 관리 단원을 참조하십시오.
SMTP 자격 증명 검증 시 Amazon SES 문제	`454 Temporary authentication failure`	이 문제의 원인일 수 있는 문제에는 다음이 포함되지만 이에 국한되지는 않습니다. • 이메일 발송 애플리케이션과 Amazon SES 간 암호화에 문제가 있습니다. Amazon SES에 연결할 때는 암호화된 연결을 사용해야 합니다. 자세한 내용은 Amazon SES SMTP 엔드포인트에 연결 단원을 참조하십시오. • Amazon SES에 문제가 있는 것일 수 있습니다. AWS Service Health Dashboard에서 업데이트를 받으십시오.
요청 수신 중 오류 발생	`454 Temporary service failure`	Amazon SES에서 요청을 수신하지 못했습니다. 따라서 메시지가 전송되지 않았습니다.

설명	응답 코드	추가 정보
잘못된 자격 증명	`530 Authentication required`	이메일을 보내는 데 사용하는 애플리케이션이 Amazon SES SMTP 인터페이스에 연결할 때 인증을 시도하지 않았습니다.
인증 자격 증명이 유효하지 않음	`535 Authentication Credentials Invalid`	이메일을 전송하는 데 사용하는 애플리케이션이 Amazon SES에 대해 올바른 SMTP 자격 증명을 제공하지 않았습니다. SMTP 자격 증명은 AWS 자격 증명과 동일하지 않습니다. 자세한 내용은 Amazon SES SMTP 자격 증명 획득 단원을 참조하십시오.
Amazon SES에 가입되지 않은 계정	`535 Account not subscribed to SES`	SMTP 자격 증명을 소유 AWS 계정 한가 Amazon SES에 가입되지 않았습니다.
메시지가 너무 김	`552 Message is too long.`	전송하려는 메시지가 최대 메시지 크기보다 큽니다.
Amazon SES에 가입되지 않은 계정	`535 Account not subscribed to SES`	SMTP 자격 증명을 소유 AWS 계정 한가 Amazon SES에 가입되지 않았습니다.
MAIL FROM 구문 오류	`553 <email-address> Invalid email address`	SMTP 메시지의 MAIL FROM 부분에 구문 오류가 있습니다. 올바른 형식을 따르고 있는지 확인하고 이메일 주소를 '<>'로 묶는 것을 잊지 마세요.
RCPT TO 구문 오류	`553 <email-address> address unknown`	SMTP 메시지의 RCPT TO 부분에 구문 오류가 있습니다. 올바른 형식을 따르고 있는지 확인하고 이메일 주소를 '<>'로 묶는 것을 잊지 마세요.

설명	응답 코드	추가 정보
사용자가 Amazon SES SMTP 엔드포인트를 호출할 권한이 없음	`554 Access denied: User` *`UserARN`* `is not authorized to perform ses:SendRawEmail on resource` *`IdentityARN`*	SMTP 자격 증명을 소유한 사용자의 AWS Identity and Access Management (IAM) 정책 또는 Amazon SES 전송 권한 부여 정책은 Amazon SES SMTP 엔드포인트를 호출할 수 없습니다.
확인되지 않은 이메일 주소	`554 Message rejected: Email address is not verified. The following identities failed the check in region` *`region`*`:` *`identity0`* `,` *`identity1`* `,` *`identity2`*	Amazon SES 계정에서 이메일을 전송하도록 확인되지 않은 이메일 주소 또는 도메인에서 이메일을 전송하려고 합니다. 이 오류는 "From", "Source", "Sender" 또는 "Return-Path" 주소에 해당할 수 있습니다. 계정이 여전히 샌드박스에 있을 경우 Amazon SES 메일박스 시뮬레이터에서 제공하는 수신자를 제외한 모든 수신자 이메일 주소를 확인해야 합니다. Amazon SES에서 확인 점검에 실패한 ID를 모두 볼 수 없는 경우 오류 메시지가 마침표 세 개(...)로 끝납니다. **Note** Amazon SES에는 여러 AWS 리전에 엔드포인트가 있으며 이메일 주소 확인 상태는 각각 다릅니다 AWS 리전. 사용 AWS 리전 하려는의 각 발신자에 대한 확인 프로세스를 완료해야 합니다.

> **Note**
>
> 이 페이지의 문제 해결 단계를 따라 해결되지 않는 SMTP 문제의 경우 문의하기 아래에 나온 SES 지원 옵션을 사용해 봅니다.

Amazon SES FAQ

이 단원에는 Amazon SES 사용과 관련하여 몇 가지 자주 묻는 질문에 대한 답변이 포함되어 있습니다.

이 단원에는 다음 주제에 대한 FAQ가 포함되어 있습니다.

- 전용 IP 주소(관리형) FAQs
- Amazon SES 전송 검토 프로세스 FAQ
- DNS 블랙홀 목록(DNSBL) FAQ
- Amazon SES 이메일 전송 지표 FAQ

전용 IP 주소(관리형) FAQs

전용 IP 주소(관리형)는 전용 IP 관리, 규모 조정 및 워밍업을 위한 다양한 자동 기능을 제공하지만이 자동화의 범위와 SES의 책임에 대해 일부 오해가 있었습니다. "관리형"은 SES가 IP 평판 및 목록 문제의 모든 측면을 완전히 처리한다는 것을 의미합니다. 이러한 오해를 명확히 하려면 서비스가 규모 조정 및 워밍업과 같은 기술적 측면을 자동화하지만 전송 평판을 유지하고 평판 블록 목록(RBL)에 등록하는 것과 같은 평판 관련 문제를 관리할 책임은 사용자에게 있다는 점을 강조해야 합니다.

이러한 FAQs 기능 범위에 대한 일반적인 오해를 해결하고 사용자와 SES 간의 공동 책임 모델을 명확히 합니다. 이 FAQ에서는 "관리형" 측면은 기술 인프라 관리를 의미하지만 전송 평판을 적극적으로 모니터링 및 유지하고 반송률을 낮게 유지하며 대부분의 RBL 목록 제거 요청을 직접 처리해야 합니다.

Q1. 전용 IP 주소(관리형)가 RBL에 나열되지 않도록 SES에 요청할 수 있습니까?

전용 IP 주소(관리형)가 수신자 메일 공급자의 RBL에 나열된 경우 SES는 책임을 지지 않으며 RBL 관리자에게 직접 제거를 요청해야 합니다. 반송 메일 알림과 SMTP 응답 메시지를 모두 추적하여 블록을 식별하여 전용 IPs(관리형)를 모니터링하는 것이 중요합니다. 이 모니터링은 이메일 전송 평판을 보호하는 데 도움이 되며 발생할 수 있는 모든 RBL 인시던트를 신속하게 해결하여 일관된 이메일 전송 가능성을 보장합니다.

Q2. Spamhaus에서 제공하지 않는 RBL에 나열된 현재 IP 주소를 대체하기 위해 새 전용 IP 주소(관리형)를 할당하도록 SES에 요청할 수 있습니까?

아니요. SES는 전용 IP 주소를 교체하지 않습니다. 전용 IP 주소, 관리형 또는 표준을 담당하므로 RBL에 등록되는 이유를 파악하고 직접 등록을 취소해야 합니다.

Q3. SES는 내 계정에 할당된 전용 IP 주소(관리형)의 반송률을 모니터링하고 반송률이 높아지면 주소를 교체할 수 있나요?

아니요. SES는 계정의 반송률이 높을 때 전용 IP 주소를 교체하지 않습니다. 전용 IP 주소(관리형)를 임대하면 계정만 해당 전용 IP 주소를 통해 이메일을 보낼 수 있는 독점적 권한을 가지므로 SES는 계정을 모니터링할 수 없습니다. 수신 거부 관리 및 반송률 2% 미만 유지를 포함하여 발신자 평판 및 이메일 구성 요소를 관리하는 것은 사용자의 책임입니다.

Q4. 방금 전용 IP 주소(관리형)를 임대했는데 주소를 통해 전송된 이메일이 RBL에 있기 때문에 바운스되고 있습니다. SES는 계정에 임대하기 전에 전용 IP 주소(관리형)의 평판을 확인하나요?

예. SES는 SES 계정에 임대하기 전에 전용 IP 주소(관리형)(30일)를 재설정합니다. 평판은 일반적으로 대부분의 주요 공급자로 재설정됩니다. SES는 주소가 Spamhaus와 같은 RBL에 있지 않은지 확인합니다. 그러나 SES는 더 작은 리전 RBLs과 같이 사용 가능한 모든 RBLs 모니터링하지 않습니다. B2B 포커스 또는 해당 도메인을 사용하는 리전 공급자 포커스로 인해 RBL에 대한 우려 사항이 있는 경우 전용 IP 주소(관리형)의 평판 상태를 직접 검토해야 합니다.

Q5. SES가 전용 IP 주소(관리형)가 Spamhus 이외의 RBL에 나열될 때 조치를 취하지 않는 경우, 이를 사용해야 하는 이유는 무엇입니까?

전용 IP 주소(관리형)는 트래픽을 기반으로 IP 주소를 추가하고 제거하여 오토 스케일링 측면에서 관리됩니다. 또한 ISP별 워밍업 관리 측면에서 시간을 절약하므로 관리형 풀은 과거 전송 패턴을 기반으로 모든 IP 주소를 통과할 수 있는 이메일 수를 추적합니다. 더 많은 이점은에서 확인할 수 있습니다the section called “이점 및 특징”.

Q6. 계정에 임대된 전용 IP 주소(관리형)를 추적하려면 어떻게 해야 하나요?

Amazon Data Firehose 또는 Amazon SNS 주제에 대해 정의된 이벤트 게시 대상과 함께 SES 구성 세트를 사용할 수 있습니다. SES 전송 이벤트에는 태그가 포함됩니다`ses:outgoing-ip`. 따라서 전용 IP 주소(관리형)의 평판으로 인해 이메일이 반송된 경우 반송 이벤트의 `ses:outgoing-ip` 태그에서 문제가 되는 전용 IP 주소(관리형)를 찾을 수 있습니다.

Amazon SES 전송 검토 프로세스 FAQ

Amazon SES를 통해 전송된 이메일을 모니터링하여 서비스가 악의적이거나 원치 않거나 품질이 낮은 이메일을 전달하는 데 사용되지 않도록 합니다. 사용자가 이러한 범주 중 하나에 속하는 콘텐츠를 전송하고 있다고 판단되면 해당 계정에 대해 조치를 취합니다. 이를 전송 검토 프로세스라고 합니다.

대부분의 경우 계정에서 문제가 감지되면 해당 계정을 검토 중 상태로 전환합니다. 다른 경우에는 계정의 이메일 전송 기능을 일시 중지합니다. 이러한 조치를 취하여 각 계정의 발신자 평판을 보호하고 다른 SES 사용자가 서비스 중단 및 전송 문제를 겪지 않도록 합니다.

내용

- 검토 중인 계정 FAQ
- 전송 일시 중지 FAQ
- 반송 메일 FAQ
- 수신 거부 FAQ
- 스팸 트랩 FAQ
- 수동 조사 FAQ

검토 중인 계정 FAQ

Q1. 내 계정이 검토 중이라는 메시지를 수신했습니다. 테스트 기간 알림이란 무엇인가요?

계정에서 전송된 이메일과 관련된 문제가 발견되어 문제를 해결할 시간을 주는 것입니다. 평소와 같이 이메일을 계속 전송할 수 있지만 계정 검토의 원인이 된 문제를 해결해야 합니다. 검토 기간이 끝나기 전에 문제를 해결하지 않으면 추가 이메일을 보내는 기능이 일시 중지될 수 있습니다.

Q2. 내 계정이 검토 중인 경우 항상 알림을 받나요?

예. AWS 계정과 연결된 이메일 주소로 알림을 받게 됩니다.

Q3. 내 계정이 검토 중이라는 알림을 받지 못한 이유는 무엇인가요?

계정이 검토되면 AWS 계정과 연결된 이메일 주소로 알림이 자동으로 전송됩니다. 이 이메일 주소는 AWS 계정을 생성할 때 지정한 주소입니다. 경우에 따라 이 이메일 주소는 SES를 사용하여 이메일을 전송하는 데 사용하는 주소와 다를 수 있습니다.

평판 지표를 정기적으로 참조하여 발신자 평판을 모니터링하는 것이 좋습니다. 또한 Amazon CloudWatch에서 자동화 경보를 설정할 수도 있습니다. 이 경보는 평판 측정치가 특정 임계값을 초과할 때 알림을 보낼 수 있습니다. 또한 휴대폰으로 텍스트 메시지를 전송하는 등의 다른 방법으로 연락하도록 Amazon CloudWatch를 구성할 수 있습니다.

Q4. SES 계정이 검토 중이라는 사실이 다른 AWS 서비스 사용에 영향을 미치나요?

SES 계정을 검토하는 동안에는 다른 AWS 서비스를 계속 사용할 수 있습니다. 하지만 아웃바운드 통신(예: Amazon SNS)을 전송하는 다른 AWS 서비스에 대한 서비스 할당량 증가를 요청하면 SES 계정의 검토 기간이 해제될 때까지 해당 요청이 거부될 수 있습니다.

Q5. 내 계정이 검토 중인 경우 어떻게 해야 하나요?

다음과 같이 해야 합니다.

- 가능하면 문제를 해결할 때까지 메일 전송을 중지하세요. 계정이 검토 중인 동안에도 이메일을 계속 전송할 수 있습니다. 그러나 변경하지 않고 메일을 계속 전송하면 문제가 의도치 않게 악화될 수 있습니다.
- Amazon SES에서 보낸 이메일에서 문제에 대한 요약을 살펴보세요.
- 전송에 대해 조사하여 전송의 어떤 측면이 문제를 유발했는지 알아보세요.
- 문제를 해결할 것으로 생각되는 변경 사항을 적용한 후 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다.
- 반드시 Amazon SES에서 요구하는 모든 정보를 제공하세요. 요구하는 정보는 사례를 평가하기 위해 필요합니다.

Q6. 검토는 어떻게 요청하면 되나요?

계정 검토에 대한 결정을 검토해 달라고 요청할 수 있습니다. 검토를 요청하려면 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다.

요청 시 다음 정보를 제공합니다.

- 계정 검토의 원인이 된 이벤트의 근본 원인에 관한 정보.
- 문제 해결을 위해 변경한 사항의 목록. 나중에 이행할 계획인 단계가 아닌 이미 이행한 단계만 포함합니다.

- 이러한 변경을 통해 향후 같은 문제가 다시 발생하지 않도록 하는 방법에 대한 정보.

계정을 검토 중으로 전환하게 된 이벤트의 특성에 따라 추가 정보가 필요할 수 있습니다. 요청에 포함해야 하는 정보의 목록은 사용자가 겪은 문제와 관련된 FAQ 주제를 참조하세요.

Q7. 검토 요청이 수락되지 않은 경우에는 어떻게 하나요?

검토 요청을 수락하지 않은 이유에 대한 정보와 함께 사용자의 요청에 응답해드립니다. 경우에 따라 문제를 해결했으며 변경을 통해 향후 문제가 다시 발생하지 않음을 증명할 수 있는 경우 다른 요청을 제출할 수 있습니다.

Q8. 문제를 진단하는 데 도움을 받을 수 있나요?

Amazon SES에서는 일반적으로 문제에 대한 종합적인 개요만을 제공합니다(예: 반송 메일과 관련하여 문제가 있습니다). 근본 원인은 직접 조사하셔야 합니다.

Q9. 내 계정이 더 이상 검토 상태가 아닌지 어떻게 알 수 있나요?

평판 지표에는 사용자 계정의 현재 상태에 대한 정보가 포함되어 있습니다. 자세한 내용은 평판 지표를 사용하여 반송 메일 및 수신 거부률 추적 단원을 참조하십시오.

Q10. 문제가 발생할 때마다 내 계정이 검토 상태가 되나요?

아니요. 경우에 따라 먼저 사용자 계정을 검토하지 않고 계정의 이메일 전송 기능을 일시 중지할 수 있습니다. 예:

- 문제가 매우 심각한 경우.
- 과거에 같은 문제로 인해 사용자 계정을 여러 번 검토한 경우. 따라서 계정 검토의 원인이 된 특정 사건이 아니라 근본적인 문제를 해결하는 것이 중요합니다. 예를 들어 특정 캠페인으로 인해 사용자 계정을 검토한 경우 해당 캠페인을 중지하기보다는 근본적인 조치를 취해야 합니다. 문제가 있는 캠페인의 속성이 무엇인지 확인하고 추후 캠페인에 같은 문제가 발생하지 않도록 준비된 프로세스를 갖추어야 합니다.

이러한 경우 계정의 이메일 전송 기능을 일시 중지하면 알림이 자동으로 전송됩니다.

Q11. 검토 기간이 만료되기 직전에 문제를 해결한 경우에는 어떻게 하나요?

에 로그인 AWS Management Console 하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 사례에 대한 회신에서 문제가 해결되었음을 알려주세요.

Q12. AWS 담당자 또는 Premium Support로부터 도움을 받을 수 있나요?

이미 AWS 계정 담당자와 협력하고 있는 경우 계정이 검토 중일 때 해당 담당자에게 자동으로 연락합니다. 계정 담당자는 문제를 더 잘 이해하는 데 도움이 되는 추가 정보를 제공할 수 있습니다. Premium Support를 사용하는 경우 해당 팀에 문의하여 추가 지원을 받아야 합니다.

전송 일시 중지 FAQ

Q1. 내 계정의 이메일 전송 기능이 일시 중지되었다는 메시지를 수신했습니다. 테스트 기간 알림이란 무엇인가요?

전송한 이메일에 있는 심각한 문제 때문에 사용자 계정의 이메일 전송 기능을 일시 중지했습니다. 대부분의 경우 다음과 같은 이유 중 하나로 인해 계정을 일시 중지합니다.

- 이전에 사용자 계정을 검토했습니다. 계정 검토의 원인이 된 문제가 검토 기간 종료 전에 해결되지 않았으므로 사용자 계정의 이메일 전송 기능을 일시 중지했습니다.
- 같은 문제로 인해 사용자 계정을 여러 번 검토했습니다.
- 사용자 계정이 AWS 서비스 약관을 위반한 이메일을 전송했습니다. 이러한 위반이 심각한 경우 먼저 사용자 계정을 검토하지 않고 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

Q2. 내 계정의 이메일 전송 기능이 일시 중지된 경우 항상 알림을 받나요?

예. AWS 계정과 연결된 이메일 주소로 알림을 받게 됩니다.

Q3. 내 계정의 이메일 전송 기능이 일시 중지되었습니다. 그런데도 알림을 받지 못한 이유는 무엇인가요?

계정의 이메일 전송 기능이 일시 중지되면 해당 계정과 연결된 이메일 주소로 알림이 자동 전송됩니다.

> **Note**
>
> AWS 계정을 생성할 때 이메일 주소를 제공해야 합니다. 이 주소는 언제든지 변경할 수 있습니다. AWS 계정과 연결된 주소 변경에 대한 자세한 내용은 AWS 결제 및 비용 관리 사용 설명서의 AWS 계정 관리를 참조하세요.

Amazon CloudWatch를 사용하여 반송 메일이나 수신 거부율이 너무 높을 때 알리는 경보를 만들 수 있습니다. 경보를 만들어 두면 사용자의 계정에서 이메일을 보내지 못하게 일시 중지 조치를 유발하는 요

인에 대해 조기 경고를 받을 수 있어 편리합니다. 그러나 일시 중지로 인해 이메일을 받지 못하게 만드는 요인은 반송 메일과 수신 거부뿐만이 아닙니다. CloudWatch의 경보 만들기에 대한 자세한 내용은 CloudWatch를 사용하여 평판 모니터링 경보 생성 섹션을 참조하세요.

계정 대시보드를 사용하여 계정의 현재 상태를 확인할 수도 있습니다. 예를 들어 사용자 계정의 이메일 전송 기능이 현재 일시 중지되어 있으면 계정 대시보드의 계정 상태(Account status) 섹션에 일시 중지됨(Paused) 상태가 표시됩니다. 계정에서 정상적으로 이메일을 보낼 수 있으면 정상(Healthy) 상태가 표시됩니다.

마지막으로 https://phd.aws.amazon.com/ AWS Health Dashboard (PHD)를 확인하여 계정의 이메일 전송 기능이 현재 일시 중지되었는지 확인할 수 있습니다. 계정의 이메일 전송 기능을 일시 중지하면 PHD의 이벤트 로그(Event log) 섹션에 SES 전송이 일시 중지됨(SES sending paused) 이벤트를 자동으로 추가합니다. SES sending paused(SES 전송이 일시 중지됨) 이벤트는 계정의 이메일 전송 기능이 현재 일시 중지되었는지 여부에 상관없이 항상 Closed(종결) 상태입니다. 이벤트 로그에는 전송 일시 중지 이벤트가 발생할 때 AWS 계정과 연결된 이메일 주소로 전송한 이메일의 사본도 포함됩니다.

CloudWatch를 사용하여 Personal Health Dashboard에 새 이벤트가 나타날 때 알려 주도록 경보를 만들 수 있습니다. 자세한 내용은 AWS Health 사용 설명서의 CloudWatch Events에서 AWS Health 이벤트 모니터링을 참조하세요.

Q4. 내 계정의 이메일 전송 기능이 일시 중지되었습니다. 이것이 다른 AWS 서비스 사용 능력에 영향을 미칩니까?

계정의 이메일 전송 기능이 일시 중지된 동안에도 다른 AWS 서비스를 계속 사용할 수 있습니다. 하지만 아웃바운드 통신(예: Amazon SNS)을 전송하는 다른 AWS 서비스에 대한 서비스 할당량 증가를 요청하면 사용자 계정의 이메일 전송 기능이 복원될 때까지 해당 요청이 거부될 수 있습니다.

Q5. 내 계정의 이메일 전송 기능이 일시 중지된 경우 어떻게 해야 하나요?

다음과 같이 해야 합니다.

- Amazon SES에서 보낸 이메일에서 문제에 대한 요약을 살펴보세요.
- 전송에 대해 조사하여 전송의 어떤 측면이 문제를 유발했는지 알아보세요.
- 문제를 해결할 것으로 생각되는 변경 사항을 적용한 후 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다.
- 반드시 Amazon SES에서 요구하는 모든 정보를 제공하세요. 요구하는 정보는 사례를 평가하기 위해 필요합니다.

Q6. 검토란 무엇인가요?

계정 검토에 대한 결정을 검토해 줄 것을 요청할 수 있습니다. 검토 요청에 대한 자세한 내용은 다음 질문을 참조하세요.

Q7. 검토는 어떻게 요청하면 되나요?

검토를 요청하려면 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다.

요청 시 다음 정보를 제공합니다.

- 문제의 원인에 대한 정보.
- 문제 해결을 위해 변경한 사항의 목록. 나중에 이행할 계획인 단계가 아닌 이미 이행한 단계만 포함합니다.
- 이러한 변경을 통해 향후 같은 문제가 다시 발생하지 않도록 하는 방법에 대한 정보.

계정의 이메일 전송 기능을 일시 중지하게 만든 이벤트의 특성에 따라 추가 정보가 필요할 수 있습니다. 요청에 포함해야 하는 정보의 목록은 사용자가 겪은 문제와 관련된 FAQ 주제를 참조하세요.

Q8. 요청이 수락되지 않은 경우에는 어떻게 하나요?

검토 요청을 수락하지 않은 이유에 대한 정보와 함께 사용자의 요청에 응답해드립니다. 경우에 따라 문제를 해결했으며 변경을 통해 향후 문제가 다시 발생하지 않음을 증명할 수 있는 경우 다른 요청을 제출할 수 있습니다.

Q9. 문제를 진단하는 데 도움을 받을 수 있나요?

Amazon SES에서는 일반적으로 문제에 대한 종합적인 개요만을 제공합니다(예: 반송 메일과 관련하여 문제가 있습니다). 문제를 해결하는 것은 사용자의 책임입니다.

Q10. 내 계정의 이메일 전송 기능이 복원되었는지 어떻게 알 수 있나요?

평판 지표에는 사용자 계정의 현재 상태에 대한 정보가 포함되어 있습니다. 자세한 내용은 평판 지표를 사용하여 반송 메일 및 수신 거부률 추적 단원을 참조하십시오.

Q11. AWS 담당자 또는 Premium Support로부터 도움을 받을 수 있나요?

이미 AWS 계정 담당자와 협력하고 있는 경우 계정의 이메일 전송 기능을 일시 중지하면 해당 담당자에게 자동으로 연락합니다. 계정 담당자는 문제를 더 잘 이해하는 데 도움이 되는 추가 정보를 제공할 수 있습니다. Premium Support를 사용하는 경우 해당 팀에 문의하여 추가 지원을 받아야 합니다.

반송 메일 FAQ

Q1. 반송 메일에 신경을 쓰는 이유는 무엇인가요?

반송 메일 발생률은 이메일 공급자 및 스팸 방지 기관 등에서 질이 나쁜 이메일을 전송하는 발신자를 감지하는 데 사용됩니다. 반송 메일 비율이 높으면 이메일이 받은 편지함이 아닌 스팸 폴더로 전송될 수 있습니다.

Q2. 내 계정이 검토 중이거나 내 계정의 반송 메일 발생률로 인해 내 전송 기능이 일시 중지되었다는 알림을 수신한 경우 어떻게 해야 하나요?

문제의 원인을 확인한 다음 해결합니다. 문제를 해결할 것으로 생각되는 변경 사항을 적용한 후 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다. 또한 다음 정보를 포함합니다.

- 반송 메일을 추적하는 데 사용하는 방법
- 새 수신자의 이메일 주소로 전송하기 전에 주소가 유효한지 확인하는 방법 예: Q11. 반송 메일을 최소화하려면 어떻게 해야 하나요?에서 어떤 권장 사항을 따르고 있나요?

Q3. 반송 메일 발생률에는 어떤 유형의 반송 메일이 포함되나요?

반송 메일 발생률에는 확인되지 않은 도메인으로 보내는 하드 바운스 메일만 포함됩니다. 하드 바운스는 "이메일 주소가 존재하지 않음"과 같은 영구적 전송 실패입니다. "메일박스 가득 참"과 같은 일시적 및 간헐적 실패 또는 차단된 IP 주소로 인한 반송 메일은 반송 메일 발생률에 반영되지 않습니다.

Q4. 내 계정의 검토 원인이 되거나 내 전송 기능의 일시 중지 원인이 될 수 있는 반송 메일 발생률을 공개하나요?

최상의 결과를 얻으려면 반송 메일 발생률을 2% 미만으로 유지해야 합니다. 반송 메일 발생률이 이것보다 높으면 이메일 배달에 영향을 미칠 수 있습니다.

반송 메일 발생률이 5% 이상이면 사용자 계정을 검토합니다. 반송 메일 발생률이 10% 이상이면 높은 반송 메일 발생률의 원인이 된 문제를 해결할 때까지 사용자 계정의 추가 이메일 전송 기능을 일시 중지할 수 있습니다.

Q5. 반송 메일 발생률은 얼마 동안 계산되나요?

고정된 기간을 기준으로 반송 메일 발생률을 계산하지는 않습니다. 발신자마다 전송 속도가 다르기 때문입니다. 그 대신 대표 볼륨을 살펴봅니다. 대표 볼륨은 일반적인 전송 실행을 나타내는 이메일의 양입니다. 대량 발신자와 소량 발신자 모두에게 공정을 기하기 위해 대표 볼륨은 사용자마다 다르게 적용되며 사용자의 전송 패턴 변화에 따라 바뀝니다.

Q6. SES 콘솔 또는 GetSendStatistics API의 정보를 사용하여 제 바운스 발생률을 계산할 수 있나요?

아니요. 반송 메일 발생률은 대표 볼륨을 사용하여 계산됩니다(Q5. 반송 메일 발생률은 얼마 동안 계산되나요? 참조). 바운스 발생률은 전송 속도에 따라 SES 콘솔 또는 `GetSendStatistics`에서 가져올 수 있는 것보다 시간상 더 앞서 나갈 수 있습니다. 또한 확인되지 않은 도메인으로 보낸 이메일만 반송 메일 발생률을 계산할 때 고려됩니다. 그러나 그러한 방법을 사용하여 반송 메일 발생률을 정기적으로 모니터링하면 사용자 계정을 검토하거나 사용자 계정의 이메일 전송 기능을 일시 중지하게 만드는 수준에 도달하기 전에 문제를 발견할 수 있으므로 좋은 지표가 됩니다.

Q7. 반송된 이메일 주소를 어떻게 알 수 있나요?

SES에서 보내는 바운스 알림을 검토합니다. 이메일을 통해 Amazon SES 알림 수신에서 설명한 것처럼 SES에서 알림을 전달하는 이메일 주소는 원본 메시지를 보낸 방식에 따라 다릅니다. Amazon SES에 대한 이벤트 알림 설정에서 설명한 것처럼 Amazon Simple Notification Service(Amazon SNS)를 통해서 반송 메일 알림을 설정할 수도 있습니다. 추가적으로 조사하지 않고 목록에서 반송된 주소를 제거하는 작업만으로는 근본적인 문제를 해결할 수 없습니다. 반송 메일을 줄이기 위해 할 수 있는 일에 대한 자세한 내용은 Q11. 반송 메일을 최소화하려면 어떻게 해야 하나요?를 참조하세요.

Q8. 반송 메일을 모니터링하지 않은 경우 반송된 주소 목록을 받을 수 있나요?

아니요. 반송된 주소의 전체 목록을 제공할 수 없습니다. 사용자는 자신의 계정에 대한 반송 메일을 모니터링하고 조치를 취할 책임이 있습니다.

Q9. 반송 메일을 어떻게 처리해야 하나요?

메일 그룹에서 반송된 주소를 제거하고 해당 주소에 대한 메일 전송을 즉시 중지해야 합니다. 소량을 전송할 경우 이메일을 통해 반송 메일을 모니터링하고 메일 그룹에서 반송된 주소를 수동으로 제거하는 작업만으로도 충분할 수 있습니다. 볼륨이 큰 경우에는 반송 메일을 받는 메일박스를 프로그래밍 방식으로 처리하거나 Amazon SNS를 통해 반송 메일 알림을 설정하여 이 프로세스에 대한 자동화를 설정할 수 있습니다. 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하십시오.

Q10. 발신 할당량에 도달해서 내 이메일이 반송될 수 있나요?

아니요. 반송 메일은 발신 할당량과 관계가 없습니다. 발신 할당량을 초과할 경우 이메일 전송 시 SES API 또는 SMTP 인터페이스에서 오류가 발생합니다.

Q11. 반송 메일을 최소화하려면 어떻게 해야 하나요?

먼저 반송 메일에 대해 잘 파악하고 있어야 합니다(Q7. 반송된 이메일 주소를 어떻게 알 수 있나요? 참조). 그런 다음 아래 지침을 따릅니다.

- 이메일 주소를 구입하거나 빌리거나 공유하지 마세요. 이메일 수신을 명시적으로 요청한 수신자에게만 이메일을 보내세요.
- 목록에서 반송된 이메일 주소를 제거하세요.
- 웹 양식에서 사용자에게 이메일 주소를 두 번 입력하도록 요청하고 두 개의 주소가 일치하는지 확인한 후에만 양식을 제출할 수 있도록 하세요.
- 이중 옵트인을 사용하여 신규 사용자를 등록하세요. 다시 말해, 신규 사용자가 등록하면 메일을 추가로 받기 전에 클릭해야 하는 확인 이메일을 보내세요. 이렇게 하면 누군가가 타인을 등록하는 것뿐만 아니라 실수로 등록하는 것을 방지할 수 있습니다.
- 최근에 메일을 보내지 않은 주소로 보내야 하는 경우(따라서 주소가 유효한지 아직 확실히 알지 못할 경우), 전체 전송의 소량만 보내도록 하세요. 자세한 내용은 Amazon SES 블로그 게시물 Never send to old addresses, but what if you have to?를 참조하세요.
- 가상 주소 사용을 조장하도록 등록을 구성하지 않았는지 확인하세요. 예를 들어 수신자가 주소를 확인할 때까지 부가 가치나 혜택을 제공하지 마세요.
- "친구에게 이메일 보내기" 기능이 있는 경우, CAPTCHA 또는 이와 유사한 메커니즘을 사용하여 이 기능이 자동으로 사용되지 않도록 하세요. 또한 사용자가 임의의 콘텐츠를 삽입하도록 허용하지 마세요.
- 시스템 알림에 SES를 사용하는 경우, 메일을 수신할 수 있는 실제 주소로 알림을 보내는지 확인하세요. 또한 필요하지 않은 알림은 끄도록 하세요.
- 새로운 시스템을 테스트하는 경우 이메일을 수신할 수 있는 실제 주소로 보내거나 SES 메일박스 시뮬레이터를 사용하세요. 자세한 내용은 수동으로 메일박스 시뮬레이터 사용 단원을 참조하십시오.

수신 거부 FAQ

Q1. 수신 거부란 무엇인가요?

수신 거부는 수신자가 이메일 수신을 원치 않는다고 보고하는 경우에 발생합니다. 수신자가 이메일 클라이언트에서 '스팸 보고' 버튼을 클릭했거나, 이메일 공급자에게 수신 거부를 했거나, SES로 직접 또는 다른 방법을 통해 알렸을 수 있습니다. 이 주제에서는 수신 거부에 대한 일반 정보를 제공합니다. 알림에 수신 거부의 소스에 대한 구체적 정보가 포함되어 있을 경우 다음 관련 주제도 읽어 보세요.

- 피드백 루프를 통한 SES 수신 거부 FAQ
- 수신자로부터 직접 수신한 SES 수신 거부 FAQ
- 이메일 공급자를 통한 SES 수신 거부 FAQ

Q2. 수신 거부에 신경을 쓰는 이유는 무엇인가요?

수신 거부 발생률은 이메일 공급자 및 스팸 방지 기관 등에서 발신자가 이메일 수신을 등록하지 않은 수신자에게 이메일을 전송하거나 발신자가 수신자가 등록한 유형과 다른 콘텐츠를 전송하고 있다는 지표로 흔히 사용됩니다.

Q3. 내 계정이 검토 중이거나 수신 거부 문제로 인해 내 전송 기능이 일시 중지되었다는 알림을 수신한 경우 어떻게 해야 하나요?

목록 취득 프로세스와 이메일의 콘텐츠를 검토하여 수신자가 수신하는 이메일을 환영하지 않은 이유를 파악하세요. 문제의 원인을 확인한 다음 해결합니다. 문제를 해결할 것으로 생각되는 변경 사항을 적용한 후 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다.

Q4. 수신 거부를 최소화하려면 어떻게 해야 하나요?

먼저 SES가 사용자에 알릴 수 있는, 즉 SES가 피드백 루프(피드백 루프를 통한 SES 수신 거부 FAQ 섹션 참조)를 통해 수신하는 수신 거부를 모니터링해야 합니다. 그런 다음 아래 지침을 따릅니다.

- 이메일 주소를 구입하거나 빌리거나 공유하지 마세요. 메일을 명시적으로 요청한 주소만 사용하세요.
- 이중 옵트인을 사용하여 신규 사용자를 등록하세요. 다시 말해, 사용자가 등록하면 메일을 추가로 받기 전에 클릭해야 하는 확인 이메일을 보내세요. 이렇게 하면 누군가가 타인을 등록하는 것뿐만 아니라 실수로 등록하는 것을 방지할 수 있습니다.

- 전송하는 메일에 대한 반응을 모니터링하고 메시지를 열거나 클릭하지 않은 수신자에게는 전송을 중지하세요.
- 신규 사용자가 등록하면 이들이 수신할 이메일의 유형에 대해 명확히 알려주고 사용자가 등록한 메일 유형만 전송하도록 하세요. 예를 들어, 사용자가 뉴스 업데이트에 등록한 경우에는 광고를 보내지 마세요.
- 메일의 서식을 잘 지정하고 전문적인 인상을 주도록 하세요.
- 수신자가 메일의 발신자를 혼동하지 않도록 명확하게 표시하세요.
- 사용자가 쉽고 확실하게 메일 구독을 취소할 수 있도록 하세요.

피드백 루프를 통한 SES 수신 거부 FAQ

이 주제에서는 SES가 피드백 루프를 통해 이메일 공급자로부터 받는 수신 거부에 대한 정보를 제공합니다. 모든 유형의 수신 거부에 적용되는 일반 정보는 수신 거부 FAQ 단원을 참조하세요.

Q1. 이 유형의 수신 거부는 어떻게 보고됩니까?

대부분의 이메일 클라이언트 프로그램은 "스팸으로 표시" 등으로 표시된 버튼을 제공합니다. 이 버튼을 누르면 메시지가 스팸 폴더로 이동한 후 이메일 공급자로 전달됩니다. 또한, 대부분의 이메일 공급자는 침해 주소(예: abuse@example.com)를 유지합니다. 이 주소를 통해 사용자는 원치 않는 이메일을 전달하고 공급자에게 이러한 이메일을 방지하는 작업을 할 것을 요청할 수 있습니다. SES가 이메일 공급자와 피드백 루프(FBL)를 설정한 경우 이메일 공급자가 수신 거부를 SES로 보냅니다.

> **Note**
>
> SES는 메시지를 보낼 때 Feedback-ID 헤더를 자동으로 설정하여 메일박스 공급자가 수신 거부 및 스팸 비율과 같은 전송 통계를 집계하고 사용할 수 있도록 합니다. SES가 제공하는 Feedback-ID 헤더 값은 다음과 같이 구성됩니다.
>
> - `FeedBackId:((SESInternalID):(AmazonSES))`, 여기서 각 항목은 다음과 같습니다.
> - SESInternalID는 SES가 수신 거부 정보를 수집하는 데 사용하는 식별자입니다.
> - AmazonSES는 SES를 전송 플랫폼으로 식별하는 정적 태그입니다.
>
> 필요에 따라 SES가 제공하는 표준 Feedback-ID 헤더 값 외에도 `ses:feedback-id-a` 및 `ses:feedback-id-b` 메시지 태그를 사용하여 자체 맞춤형 피드백 ID(최대 2개)를 지정할 수 있습니다. the section called “이메일 캠페인에 대한 세분화된 피드백” 섹션을 참조하세요.

Q2. 이러한 수신 거부는 SES 콘솔에 표시되고 GetSendStatistics API가 반환하는 수신 거부 발생률 통계에 포함되나요?

예. 하지만 수신 거부 발생률 통계에는 SES로 피드백을 제공하지 않는 이메일 공급자로부터의 수신 거부는 포함하지 않습니다. 피드백을 제공하는 도메인의 수신 거부 발생률이 나머지 전송 또한 대표할 가능성이 높습니다.

Q3. 이러한 수신 거부를 어떻게 통보받을 수 있습니까?

이메일 또는 Amazon SNS 알림을 통해 통보받을 수 있습니다. Amazon SES에 대한 이벤트 알림 설정 섹션의 설정 지침을 참조하세요.

Q4. 이메일 또는 Amazon SNS를 통해 수신 거부 알림을 수신하면 어떻게 해야 합니까?

먼저 메일 그룹에서 수신 거부를 한 주소를 제거하고 해당 주소에 대한 메일 전송을 즉시 중지해야 합니다. 구독 취소 요청에 대한 이메일에도 회신하지 마세요. 수신 거부를 받는 메일박스를 프로그래밍 방식으로 처리하거나 Amazon SNS를 통해 수신 거부 알림을 설정하여 이 프로세스에 대한 자동화를 설정할 수 있습니다. 자세한 내용은 Amazon SES에 대한 이벤트 알림 설정 단원을 참조하십시오.

그런 다음 전송을 주의 깊게 살펴서 수신자가 메일을 거부한 이유를 알아보고 근본적인 문제를 해결하세요. 수신 거부를 하는 사람에 비해 수신 거부를 하지는 않았지만(또는 못했지만) 메일을 환영하지 않는 사람이 훨씬 더 많을 수 있습니다. 따라서 수신 거부를 하는 수신자를 제거하기만 하는 것은 근본적인 문제를 해결하는 것이 아닙니다.

Q5. 내 계정의 검토 원인이 되거나 내 계정의 이메일 전송 기능이 일시 중지되는 원인이 될 수 있는 SES 수신 거부 발생률을 공개하나요?

최상의 결과를 얻으려면 수신 거부 발생률을 0.1% 미만으로 유지해야 합니다. 수신 거부 발생률이 이것보다 높으면 이메일 배달에 영향을 미칠 수 있습니다.

수신 거부 발생률이 0.1% 이상이면 사용자 계정을 검토합니다. 수신 거부 발생률이 0.5% 이상이면 높은 수신 거부 발생률의 원인이 된 문제를 해결할 때까지 사용자 계정의 추가 이메일 전송 기능을 일시 중지할 수 있습니다.

Q6. 얼마 동안 수신 거부 발생률이 계산되나요?

고정된 기간을 기준으로 수신 거부 발생률을 계산하지는 않습니다. 발신자마다 전송 속도가 다르기 때문입니다. 그 대신 대표 볼륨을 살펴봅니다. 대표 볼륨은 일반적인 전송 실행을 나타내는 이메일의 양입니다. 대량 발신자와 소량 발신자 모두에게 공정을 기하기 위해 대표 볼륨은 사용자마다 다르게 적용되며 사용자의 전송 패턴 변화에 따라 바뀝니다. 또한 수신 거부 발생률은 모든 이메일을 기준으로 계

산되지 않습니다. 그 대신 SES로 수신 거부 피드백을 보내는 도메인으로 전송된 메일에 대한 수신 거부 백분율로 계산됩니다.

Q7. SES 콘솔 또는 GetSendStatistics API의 지표를 사용하여 자체 수신 거부 발생률을 계산할 수 있나요?

아니요. 여기에는 다음과 같은 두 가지 주요 이유가 있습니다.

- 수신 거부 발생률은 대표 볼륨을 사용하여 계산됩니다(Q6. 얼마 동안 수신 거부 발생률이 계산되나요? 참조). 수신 거부 발생률은 전송 속도에 따라 SES 콘솔 또는 `GetSendStatistics` API에서 가져올 수 있는 것보다 시간상 더 앞서 나갈 수 있습니다. 이런 이유로, 이러한 방법을 정기적으로 사용하여 사용자 계정에 대한 수신 거부 발생률을 모니터링하는 것이 좋습니다. 이러한 방법으로 수신 거부 발생률을 모니터링하면 이메일 전송에 영향을 줄 수 있는 수준에 도달하기 전에 문제를 파악하는 데 필요한 정보를 얻을 수 있습니다.
- 수신 거부 발생률 계산 시 모든 이메일을 포함시키지는 않습니다. 수신 거부 발생률은 SES로 수신 거부 피드백을 보내는 도메인으로 전송된 메일에 대한 수신 거부 백분율로 계산됩니다.

Q8. 수신 거부를 한 이메일 주소를 어떻게 알 수 있나요?

SES가 이메일 또는 Amazon SNS를 통해 전송하는 수신 거부 알림을 검사합니다(Amazon SES에 대한 이벤트 알림 설정 섹션 참조). 하지만 이메일 공급자마다 제공하는 정보량이 다르며, 일부 공급자는 수신자의 이메일 주소를 수정한 후에 SES로 수신 거부 알림을 전송하기도 합니다. 이후에 수신자의 이메일 주소를 찾기 위한 최선의 옵션은 SES가 이메일을 수락할 때 사용자에게 반환하는 SES 메시지 ID와 식별자 사이의 자체 매핑을 저장하는 것입니다. SES는 사용자가 추가하는 사용자 지정 메시지 ID를 보유하지 않습니다.

Q9. 수신 거부를 모니터링하지 않은 경우 수신 거부된 주소 목록을 받을 수 있나요?

포괄적인 목록을 제공해 드리지는 않습니다. 그러나 이메일이나 Amazon SNS를 통해 추후 수신 거부를 모니터링할 수 있습니다.

Q10. 샘플 이메일을 받아볼 수 있습니까?

샘플 이메일은 요청하시더라도 보내드릴 수 없습니다만, 이 정보는 수신 거부 알림에서 확인할 수 있습니다. 자세한 내용은 Q8. 수신 거부를 한 이메일 주소를 어떻게 알 수 있나요? 단원을 참조하십시오.

수신자로부터 직접 수신한 SES 수신 거부 FAQ

이 주제에서는 SES가 수신자로부터 직접 수신하는 수신 거부에 대한 정보를 제공합니다. 모든 유형의 수신 거부에 적용되는 일반 정보는 수신 거부 FAQ 단원을 참조하세요.

Q1. 이 유형의 수신 거부는 어떻게 보고됩니까?

여러 수신자가 이메일 또는 여타 수단을 통해 SES로 직접 귀하의 메일에 대해 문의했습니다.

Q2. 이러한 수신 거부는 SES 콘솔에 표시되고 GetSendStatistics API가 반환하는 수신 거부 발생률 통계에 포함되나요?

아니요. SES 콘솔 또는 `GetSendStatistics` API를 통해 검색하는 수신 거부 발생률 통계에는 SES가 피드백 루프를 통해 받는 수신 거부만 포함됩니다. 이러한 유형의 수신 거부에 대한 자세한 내용은 피드백 루프를 통한 SES 수신 거부 FAQ 섹션을 참조하세요.

Q3. 이메일 피드백 알림 또는 Amazon SNS를 통한 수신 거부에 대한 알림을 받지 못한 이유는 무엇입니까?

이메일 피드백 전달 및 Amazon SNS 알림은 SES가 피드백 루프를 통해 받는 수신 거부만 포함합니다. 수신자가 SES에 직접 제기하는 수신 거부에 대해서는 알림을 받지 않습니다.

Q4. 수신 거부를 한 이메일 주소를 어떻게 알 수 있나요?

수신 거부한 수신자의 신원을 보호하기 위해 이메일에 대해 수신 거부한 이메일 주소는 제공되지 않습니다.

목록에서 개별 수신자를 제거하는 데 집중하기 보다는 수신을 거부하게 만든 문제를 확인하는 것이 좋습니다. 먼저 고객 유치 과정을 검토하고 이메일 수신을 명시적으로 요청하지 않은 고객을 목록에서 제거하는 것이 좋습니다. 또한 이메일의 콘텐츠를 분석하여 수신자가 수신을 거부하는 이유를 이해해야 합니다.

Q5. 샘플 이메일을 받아볼 수 있습니까?

수신 거부한 수신자의 신원을 보호하기 위해 수신자가 수신 거부한 이메일의 사본은 제공되지 않습니다.

Q6. 내 계정이 검토 중이거나 직접 수신 거부로 인해 내 전송 기능이 일시 중지되었다는 알림을 수신한 경우 어떻게 해야 하나요?

이메일 수신을 특정하게 등록한 수신자에게만 메시지를 전송하도록 발송 과정을 즉시 변경하세요. 또한 수신자가 수신을 등록한 유형의 콘텐츠를 전송하고 있는지 확인하세요. 문제를 해결할 것으로 생각되는 변경 사항을 적용한 후 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다.

3주 이내에 검토를 요청하지 않고 수신자 직접 수신 거부를 계속 받으면 계정의 이메일 전송 기능이 일시 중지될 수 있습니다.

이메일 공급자를 통한 SES 수신 거부 FAQ

이 주제에서는 SES가 이메일 공급자(메일박스 공급자라고도 함)를 통해 받는 수신 거부에 대한 정보를 제공합니다. 모든 유형의 수신 거부에 적용되는 일반 정보는 수신 거부 FAQ 단원을 참조하세요.

Q1. 이 유형의 수신 거부는 어떻게 보고됩니까?

한 이메일 공급자가 고객 중 상당 수가 귀사의 이메일을 스팸으로 표시했다고 SES로 보고했습니다. 이 보고는 피드백 루프를 통한 SES 수신 거부 FAQ에서 설명된 피드백 루프가 아닌 다른 수단을 통해 SES로 제공되었습니다.

Q2. 이러한 수신 거부는 SES 콘솔에 표시되고 GetSendStatistics API가 반환하는 수신 거부 발생률 통계에 포함되나요?

아니요. SES 콘솔 또는 `GetSendStatistics` API를 통해 검색하는 수신 거부 발생률 통계에는 SES가 피드백 루프를 통해 받는 수신 거부만 포함됩니다.

Q3. 이메일 피드백 알림 또는 Amazon SNS를 통한 수신 거부에 대한 알림을 받지 못한 이유는 무엇입니까?

이메일 피드백 전달 및 Amazon SNS 알림은 SES가 피드백 루프를 통해 받는 수신 거부만 포함합니다.

Q4. 수신 거부를 한 이메일 주소를 어떻게 알 수 있나요?

이메일 공급자는 일반적으로 이 정보를 공개하지 않습니다. 하지만 목록에서 개별 수신자를 제거하는 데 집중하기 보다는 근본적인 문제를 찾고 해결하는 데 집중해야 합니다. 목록 취득 프로세스와 이메일의 콘텐츠를 검토하여 수신자가 이메일을 환영하지 않은 이유를 파악하는 것부터 시작하세요.

Q5. 샘플 이메일을 받아볼 수 있습니까?

아니요. 이메일 공급자는 일반적으로 샘플 이메일을 제공하지 않습니다.

Q6. 내 계정이 검토 중이거나 이메일 공급자 수신 거부로 인해 내 전송 기능이 일시 중지되었다는 알림을 수신한 경우 어떻게 해야 하나요?

문제의 원인을 확인한 다음 해결합니다. 문제를 해결할 것으로 생각되는 변경 사항을 적용한 후 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생

하지 않도록 하는 방법을 설명합니다. 3주 이내에 검토를 요청하지 않고 공급자로부터 수신 거부를 계속 받으면 계정의 추가 이메일 전송 기능이 일시 중지될 수 있습니다.

스팸 트랩 FAQ

Q1. 스팸 트랩이란 무엇인가요?

스팸 트랩이란 인터넷 서비스 제공업체(ISP), 이메일 공급자 또는 스팸 방지 기관에서 관리하는 특수한 이메일 주소입니다. 합법적인 방식으로는 이러한 주소로부터 이메일을 수신하도록 등록할 수 없기 때문에 스팸 트랩을 관리하는 기관에서는 누군가가 이러한 주소로 메일을 보낼 경우 의심스러운 방식으로 이메일을 사용할 가능성이 있다고 파악합니다.

Q2. 스팸 트랩은 어떻게 설정하나요?

스팸 트랩 주소는 다양한 방식으로 설정할 수 있습니다. 이전에 유효했으나 장기간 사용하지 않는 (따라서 반송되는) 주소에서 변환할 수 있습니다. 스팸 트랩 목적으로 설정된 주소인 경우도 있습니다. 이러한 주소는 추측하기 어려운 특이한 주소일 수 있으며, 실제 주소와 유사한 주소인 경우(예: 일반적인 도메인 이름에 오타 삽입)도 있습니다. 스팸 트랩은 다양한 방식으로 인터넷에 배포하여 "퍼뜨려지는" 경우도 흔히 있습니다.

Q3. 스팸 트랩으로 전송하는지 여부를 SES에서 어떻게 아나요?

스팸 트랩을 운영하는 일부 기관은 SES 발신자가 스팸 트랩으로 전송했을 시 SES 알림을 보냅니다.

Q4. SES는 스팸 트랩 보고서를 어떻게 사용하나요?

보고서를 검토합니다. 계정이 스팸 트랩에 이메일을 보내고 있다고 판단되면 계정을 검토하고 기본 문제를 해결하도록 요청합니다. 검토 기간이 끝나기 전에 문제를 해결하지 않으면 계정의 추가 이메일 전송 기능이 일시 중지될 수 있습니다. 스팸 트랩 문제가 매우 심각한 경우 먼저 사용자 계정을 검토하지 않고 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

Q5. 내 계정이 검토 중이거나 스팸 트랩 문제로 인해 내 전송 기능이 일시 중지되었다는 알림을 수신한 경우 어떻게 해야 하나요?

먼저 계정 검토 또는 이메일 전송 기능 일시 중지의 원인이 된 문제를 해결해야 합니다. 다음으로 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다. 이러한 조치가 문제를 적절하게 해결할 것으로 생각되면 검토 기간을 취소하거나 사용자 계정에서 전송 일시 중지를 제거할 것입니다.

스팸 트랩 히트의 보고 방식으로 인해 이러한 조치가 문제를 해결했는지 확인할 수 있으려면 3주 이상 걸릴 수 있습니다.

Q6. 스팸 트랩으로 몇 회 전송할 경우 사용자 계정을 검토하거나 계정의 이메일 전송 기능을 일시 중지하나요?

계정에 조치가 취해질 수 있는 일정한 스팸 트랩 횟수는 공개되지 않습니다. 그러나 스팸 트랩 횟수가 적으면 발신자로서의 평판에 매우 부정적인 영향을 줄 수 있으므로 스팸 트랩 보고서를 심각하게 받아들여야 합니다.

Q7. 스팸 트랩 주소를 공개하나요?

아니요. 스팸 트랩이 효과를 나타내려면 스팸 트랩을 비밀로 유지해야 합니다. 스팸 트랩 기관에서는 스팸 트랩 전송이 발생했다는 사실만 공개합니다. 실제 스팸 트랩 주소는 공개하지 않습니다.

Q8. 스팸 트랩으로 전송하는 일을 방지하려면 어떻게 해야 하나요?

스팸 트랩으로 전송하는 위험을 줄이려면 다음 지침을 따르세요.

- 이메일 주소를 구입하거나 빌리거나 공유하지 마세요. 메일을 명시적으로 요청한 주소만 사용하세요.
- 웹 양식에서 사용자에게 이메일 주소를 두 번 입력하도록 요청하고 두 개의 주소가 일치하는지 확인한 후에만 양식을 제출할 수 있도록 하세요.
- 이중 옵트인을 사용하여 신규 사용자를 등록하세요. 다시 말해, 사용자가 등록하면 메일을 추가로 받기 전에 클릭해야 하는 확인 이메일을 보내세요.
- 하드 반송 메일 주소가 스팸 트랩으로 변환되기 전에 미리 목록에서 제거하세요.
- 수신자의 반응을 모니터링하고 최근에 귀하의 이메일 또는 웹 사이트와 교류하지 않은 수신자에게 전송하는 것을 중지하세요. "활성 사용자"로 간주하기 위해 인정되는 기간은 사용 사례에 따라 다르지만, 일반적으로 사용자가 수개월 동안 귀하의 이메일을 열거나 클릭하지 않은 경우 귀하의 메일 수신을 원한다는 증거가 없는 한 그러한 사용자를 메일 목록에서 제거하는 것을 고려해야 합니다.
- 최근에 귀하와 상호 작용하지 않은 사람과 의도적으로 접촉하기 위한 일환으로 재참여 캠페인을 시행할 경우 상당한 주의를 기울이세요. 이러한 시도는 상당히 위험할 수 있으며 스팸 트랩 전송뿐만 아니라 반송 메일 및 수신 거부 등의 문제를 야기할 가능성이 높습니다.
- 메일 발송 목록의 전체 수신자에게 옵트인 메시지를 발송하고 확인 링크를 클릭한 수신자만 유지합니다. 이 절차는 목록에서 비활성 수신자를 제거할 뿐만 아니라 스팸 트랩 주소를 제거하는 데도 도움이 됩니다. 하지만 메일 그룹에 다수의 불량 주소가 포함되어 있다고 판단하거나 사용자 계정에 이미 반송 메일 문제가 있는 경우에는 이 기법을 사용하지 않는 것이 좋습니다. 왜냐하면 사용자 계정의 반송 메일 발생률이 더욱 늘어날 수 있기 때문입니다.

수동 조사 FAQ

Q1. 내 계정이 검토 중이거나 수동 조사로 인해 내 전송 기능이 일시 중지되었다는 알림을 수신한 경우 어떻게 해야 하나요?

SES 조사자가 전송과 관련된 중요한 문제를 발견했습니다. 일반적인 문제는 다음과 같습니다.

- 전송이 AWS 이용 정책(AUP)을 위반했습니다.
- 수신자가 원치 않는 이메일로 보입니다.
- 콘텐츠가 피싱과 관련되어 있습니다(시뮬레이션된 피싱 포함).
- 그렇지 않으면 콘텐츠가 SES에서 지원하지 않는 사용 사례와 관련이 있습니다.

문제를 해결할 수 있다고 판단되는 경우 일정 시간 동안 사용자 계정을 검토합니다. 사용자 계정을 검토하는 동안 문제 해결을 위해 이메일 전송 실행을 변경해야 합니다.

문제를 해결할 수 없다고 판단되거나 문제가 매우 심각한 경우 먼저 사용자 계정을 검토하지 않고 계정의 이메일 전송 기능을 일시 중지할 수 있습니다.

Q2. 이메일 전송의 수동 검토 원인이 될 수 있는 문제는 무엇인가요?

사용자 계정의 수동 검토 원인이 될 수 있는 문제는 몇 가지 있습니다. 그 이유는 다음과 같으나 이에 국한되지는 않습니다.

- 수신자가 사용자 계정에서 전송된 이메일에 대해 수신 거부하기 위해 SES에 문의했습니다.
- 이메일 전송 패턴에서 비정상적인 변화를 감지했습니다.
- 스팸 필터를 통해 원치 않거나 품질이 낮은 콘텐츠의 전형적인 이메일 특성을 찾았습니다.

사용자 계정을 검토하거나 계정의 이메일 전송 기능을 일시 중지하면 알림이 전송됩니다. 대부분의 경우 이 알림에는 문제에 대한 정보가 포함되어 있으며 취할 수 있는 다음 조치에 대한 정보가 나와 있습니다.

Q3. "원치 않는" 이메일이란 무엇인가요?

원치 않는 이메일이란 수신자가 명시적으로 수신을 요청하지 않은 이메일입니다. 수신자가 특정 유형의 메일(예: 알림)에 등록했지만 다른 유형의 메일(예: 광고)이 전송된 경우가 이에 포함됩니다.

사용자 계정을 검토하거나 계정의 이메일 전송 기능을 일시 중지하면 알림이 전송됩니다. 원치 않는 이메일 문제로 인해 이러한 작업 중 하나를 수행하고 있다는 알림이 수신되면 AWS 콘솔에 로그인하고

지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 포함하는 정보는 다음과 같습니다.

- 전송하는 모든 메시지가 수신자 측에서 특정하게 요청한 것이며 AWS 이용 정책을 준수하고 있습니까?
- 귀하 또는 귀하의 웹 사이트와 상호 작용하는 고객에게 이메일을 요청하는 것 이외의 방식으로 이메일 주소를 수집하였습니까? 메일 그룹을 입수한 방식을 설명해야 합니다.
- 구독 및 구독 취소 프로세스는 어떻게 작동합니까? 옵트인 및 옵트아웃 링크를 포함시켜야 합니다.

Q4. 내 계정이 검토 중이거나 수동 검토로 인해 내 전송 기능이 일시 중지되었다는 알림을 수신한 경우 어떻게 해야 하나요?

문제의 원인을 확인한 다음 해결합니다. 문제를 해결할 것으로 생각되는 변경 사항을 적용한 후 AWS 콘솔에 로그인하고 지원 센터로 이동합니다. 사용자를 대신하여 개설된 사례에 회신합니다. 메시지에 문제 해결을 위해 취한 조치에 관한 세부 정보를 제공하고 이러한 조치를 통해 향후 문제가 다시 발생하지 않도록 하는 방법을 설명합니다. 이러한 조치가 문제를 적절하게 해결할 것으로 생각되면 사용자 계정에 대한 검토 기간을 취소할 것입니다.

Q5. "수정할 수 있다"고 보는 문제 유형은 어떤 것인가요?

일반적으로 과거의 전송 사례가 우수했으며 전송을 대부분 계속하면서도 문제가 되는 전송을 중단하기 위해 취할 수 있는 조치가 있는 경우 상황을 수정할 수 있다고 봅니다. 예를 들어, 세 가지 유형의 이메일을 보내는 데 그 중 한 가지 유형만 문제가 되는 경우 문제가 되는 해당 전송만 중지하고 나머지 전송은 계속 진행할 수 있습니다.

Q6. 문제의 원인을 찾을 수 없는 경우에는 어떻게 하나요?

AWS 콘솔에 로그인하고 지원 센터로 이동할 수 있습니다. 사용자를 대신하여 개설된 사례에 회신하고 문제를 일으킨 메일의 샘플을 요청합니다.

DNS 블랙홀 목록(DNSBL) FAQ

도메인 이름 시스템 기반 블랙홀 목록(DNSBL)(실시간 블랙홀 목록(RBL), 거부 목록, 블록리스트 또는 차단 목록으로 불리기도 함)은 원치 않는 이메일을 보내는 것으로 의심되는 IP 주소를 이메일 공급자에게 알리기 위한 것입니다.

DNSBL마다 이메일 전달률에 미치는 영향이 다릅니다. 이 주제에서는 DNSBL이 Amazon SES를 사용하여 보내는 이메일의 전달에 미치는 영향과 DNSBL에서 Amazon SES IP 주소를 제거하기 위한 정책을 설명합니다.

> **Note**
>
> 이 주제는 이메일 공급자가 수신 메시지를 차단하는 데 사용하는 DNSBL에 대한 내용입니다. 이전에 반송 메일을 생성한 이메일 주소의 수신자에게 보낸 이메일이 Amazon SES에서 차단되는 방법에 대한 자세한 내용은 Amazon SES 전역 금지 목록 섹션을 참조하세요.

Q1. DNSBL은 이메일 전송에 어떤 영향을 주나요?

DNSBL마다 메시지의 성공적인 전송에 미치는 영향이 다릅니다. Gmail, Hotmail, AOL, Yahoo를 비롯한 주요 이메일 공급자들은 Spamhaus에서 제공하는 것과 같이 매우 소수의 DNSBL을 인식하는 것으로 보입니다. 당사의 경험에 의하면 일부 메일 시스템은 특정 DNSBL를 강조하지만 다른 DNSBL는 대부분 영향이 적습니다.

마지막으로 자체적인 내부 거부 목록이 있는 이메일 공급자가 많습니다. 이메일 공급자들은 이러한 목록을 매우 면밀히 감시하고 대중과 공유하는 일이 거의 없습니다. 이러한 목록 중 하나에 IP 주소가 등록되어 있으면 해당 공급자를 사용하는 수신자에 대한 사용자의 이메일 전송 기능이 심각한 영향을 받을 수 있습니다.

Q2. IP 주소는 어떻게 DNSBL로 등록되나요?

IP 주소가 DNSBL에 등록되는 경로에는 여러 가지가 있습니다. IP 주소는 스팸 트랩에 이메일을 보내면 DNSBL에 추가될 수 있습니다. 스팸 트랩은 실제 사용자에게 속하지 않은 이메일 주소입니다. 스팸 트랩은 스팸을 수집하고 스패머를 식별하기 위한 목적으로만 존재합니다. 일부 DNSBL는 개인 사용자가 IP 주소를 제출하는 것도 허용합니다. 사용자가 전체 IP 주소 범위를 제출하는 것을 허용하는 DNSBL도 더러 있습니다. 다른 DNSBL은 이메일 관리자의 기여를 통해 유지되며 관리자가 자체 시스템을 침해하고 있다고 생각하는 IP 주소를 포함할 수 있습니다.

Q3. Amazon SES는 DNSBL에 IP 주소가 등록되는 것을 어떻게 예방하고 있나요?

당사 시스템은 침해 징후를 검사합니다. IP 주소가 DNSBL에 추가될 수 있는 전송 패턴이나 기타 특성을 감지하면 발신자에게 알림을 보냅니다. 상황이 심각하거나 알림 전송 후에도 발신자가 문제를 해결

하지 않으면 문제를 해결할 때까지 발신자의 이메일 전송 기능을 일시 중지하게 됩니다. 이러한 방식으로 전송 정책을 적용하면 IP 주소가 DNSBL에 등록될 가능성이 줄어듭니다.

Q4. Amazon SES가 DNSBL에서 IP 주소를 제거할 수 있습니까?

SES 공유 IPs 경우 전체 Amazon SES 서비스에서의 전송에 영향을 미치거나 Gmail, Yahoo, AOL, Hotmail과 같은 주요 이메일 공급자를 사용하는 수신자에게 이메일을 보내는 기능에 영향을 미칠 수 있는 DNSBLs을 적극적으로 모니터링합니다. Spamhaus가 제공하는 DNSBL이 이 카테고리에 속합니다. 이러한 기준 중 하나를 충족하는 목록에 IP 주소 중 하나가 표시되면 해당 주소가 DNSBL에서 최대한 빨리 제거되도록 즉각적인 조치를 취합니다.

당사는 전체 Amazon SES 서비스를 통한 전송 작업에 영향을 미칠 가능성이 낮거나 주요 이메일 공급자에게 전송 시 미칠 영향이 크지 않은 DNSBL은 모니터링하지 않습니다. SORBS 및 UCEPROTECT가 제공하는 DNSBL이 이 카테고리에 속합니다. 이러한 목록을 운영하는 공급자의 특정 등록 및 등록 취소 관행 때문에 IP 주소를 이러한 목록에서 제거할 수 없습니다.

수신자 메일 공급자의 RBL에 나열된 전용 IP 주소(관리형 또는 표준)의 경우 IPs 목록을 삭제하는 것은 SES의 책임이 아니며 RBL 관리자에게 직접 제거를 요청해야 합니다.

Q5. 이메일 공급자가 발신 IP 주소가 Spamhaus가 아닌 다른 DNSBL에 등록되어 있다는 이유로 제 이메일 주소를 거부합니다. 어떻게 해야 합니까?

먼저 메시지가 DNSBL 때문에 차단된 것이 맞는지 확인하세요. 발신 IP 주소가 DNSBL에 추가되어 이메일이 거부된 것이라면 다음 예제처럼 이름을 기준으로 DNSBL 공급자가 언급된 반송 메일 알림을 받게 됩니다.

```
554 5.7.1 Service unavailable; Client host [192.0.2.0] blocked using DNSBLName;
 See: http://www.example.com/query/ip/192.0.2.0
```

반송 메일 알림을 받았지만 앞서 나온 예제에 표시된 메시지와 비슷한 정보가 포함되어 있지 않다면 이메일 공급자가 DNSBL에 추가되는 것과 관련 없는 이유로 메시지를 거부했을 가능성이 큽니다.

발신 IP 주소가 DNSBL에 등록되어 있다는 이유로 이메일 공급자가 이메일을 차단하고 있다는 사실을 확인할 수 있다면 몇 가지 취할 수 있는 조치가 있습니다.

- 메시지를 거부한 도메인의 포스트마스터에게 문의하여 스팸 필터링 정책의 예외를 요청하세요. 일부 포스트마스터는 지원 프로세스가 있으며 이 프로세스를 설명하는 포스트마스터 페이지를 게시할 수 있습니다. 문의하려는 도메인이 포스트마스터 지원 정책을 게시하지 않으면 postmaster@*example.com*으로 이메일을 보내 포스트마스터에게 문의할 수 있습니다. 여기서

`example.com`은 해당 도메인입니다. 포스트마스터 메일박스를 사용하려면 RFC 5321에 따라 도메인이 필요합니다.

포스트마스터에게 문의할 경우에는 수신한 반송 메일 코드, 보내려는 이메일의 헤더, 이메일 전송 시 DNSBL이 미치는 영향, 이메일이 부적절하게 차단되고 있다고 생각하는 이유에 대한 정보를 제공하세요. 적법한 이메일을 보내고 있음을 증명하기 위해 포스트마스터에게 제공할 수 있는 정보가 많을수록 포스트마스터가 예외를 적용할 가능성이 높습니다.

- 이메일 공급자가 응답하지 않거나 정책을 변경할 의지가 없다면 전용 IP 주소 사용을 고려해 보세요. 전용 IP 주소는 사용자 본인만 사용할 수 있는 주소입니다. 올바른 전송 실행을 구현함으로써 참여율은 높게 유지하고 반송 메일, 수신 거부, 스팸 트랩 횟수를 낮게 유지할 수 있습니다. 올바른 전송 실행은 사용자 주소가 DNSBL에 등록되지 않도록 하는 데 도움이 됩니다.

Q6. Gmail, Hotmail 또는 다른 주요 공급자에게 보내는 메일이 스팸 폴더로 전송됩니다. 제 발신 IP 주소가 DNSBL에 등록되었기 때문인가요?

아닐 것입니다. IP 주소가 Spamhaus의 DNSBL 중 하나와 같이 중대한 영향을 미치는 DNSBL에 등록된 경우, 주요 이메일 공급자는 이메일을 스팸 폴더로 보내는 것이 아니라 해당 IP 주소의 이메일을 완전히 거부합니다.

주요 이메일 공급자가 이메일을 거부하지 않고 수락하면 대개 받은 편지함 또는 스팸 폴더에 메시지를 넣을지 여부를 결정할 때 사용자 참여를 고려합니다. 사용자 참여란 이전에 사용자에게 보낸 메시지와 사용자가 상호 작용하는 방식을 의미합니다.

메시지가 고객의 받은 편지함에 도착할 확률을 높이려면 다음과 같은 모범 사례를 모두 실행해야 합니다.

- 이메일 주소를 임차하거나 구매하지 마세요 . 목록 임차 또는 구매는 AWS 이용 정책(AUP)을 위반하는 행위이며 어떠한 경우에도 Amazon SES에서 허용되지 않습니다.
- 이메일 수신을 명시적으로 요청한 고객에게만 이메일을 보내세요. 이메일 수신을 명시적으로 동의하지 않은 수신자에게 이메일을 보내는 것은 전 세계의 여러 국가 및 관할 구역에서 불법입니다.
- 지난 30~90일 동안 보낸 메시지를 열어 보지 않았거나 메시지의 링크를 클릭하지 않은 고객에게는 이메일 발신을 중지하세요. 이 단계는 참여율을 높게 유지하는 데 도움이 될 수 있으므로 추후에 보내는 메시지가 수신자의 받은 편지함에 도착할 가능성이 높아집니다.
- 보내는 각 메시지에서 일관된 디자인 요소와 작문 스타일을 사용해 고객이 메시지 발신자를 쉽게 식별할 수 있게 하세요.
- SPF 및 DKIM 등의 이메일 인증 메커니즘을 사용합니다.

- 고객이 콘텐츠 구독을 위해 웹 양식을 사용할 경우, 이메일을 보내 고객이 이메일 수신을 원한다는 사실을 확인받으세요. 고객이 이메일 수신을 원한다고 확인할 때까지는 추가로 이메일을 보내지 마세요. 이 프로세스를 확인된 옵트인 또는 이중 옵트인이라고 합니다.
- 고객이 쉽게 구독을 취소할 수 있게 하고, 구독 해제 요청은 즉각 처리하세요.
- 링크가 포함된 이메일을 보낼 경우 해당 링크를 Spamhaus DBL(Domain Block List)과 비교해 점검하세요. 링크를 테스트하려면 Spamhaus 웹 사이트에서 Domain Lookup Tool을 사용하세요.

이러한 실행을 구현하면 발신자 평판을 개선하여 전송한 이메일이 수신자의 받은 편지함에 도달할 가능성을 높일 수 있습니다. 또한 이러한 실행을 구현하면 사용자 계정에 대한 반송 메일과 수신 거부 발생률을 낮게 유지하는 데 도움이 되고 스팸 트랩에 이메일을 보낼 위험을 낮출 수 있습니다.

Amazon SES 이메일 전송 지표 FAQ

Amazon SES는 전송하는 이메일에 대한 몇 가지 지표를 수집합니다. 이러한 지표는 이메일 프로그램의 효율성을 분석하거나, 반송 또는 불만 제기 비율 같이 중요한 통계를 모니터링하는 데 사용됩니다.

이번 단원에는 이메일 전송 지표와 관련하여 다음 지표에 대한 FAQ가 포함되어 있습니다.

- 일반 질문
- 열기 추적
- 클릭 추적

> **Note**
>
> 이벤트 추적은 수신자의 이메일 서비스 공급자(ESP) 와 Amazon SES의 제어 범위를 벗어난 개인 정보 보호 설정을 구성한 방법에 따라 달라집니다. 다음과 같은 조건에서 추적 이벤트 수를 왜곡할 수 있습니다(부정확한 수 반환).
>
> - 이메일 수신자가 개인 정보를 보호하는 이메일 서비스 공급자(ESP)를 사용하고 있습니다.
> - 이메일 수신자가 ESP에 데이터를 공유할 수 있는 권한을 명시적으로 부여하지 않습니다.
> - 이메일 수신자의 ESP가 이미지 또는 링크를 캐시하며, SES가 초기 열림 수만 계산하고 후속 열림 수는 계산할 수 없습니다.

일반 질문

Q1. 이메일 전송 후 Amazon SES는 얼마나 오랫동안 이메일 확인 및 클릭 지표를 수집합니까?

Amazon SES는 각 이메일 전송 후 60일 동안 열기 및 클릭 지표를 수집합니다.

Q2. 사용자가 이메일 1개를 여러 차례 확인하거나, 혹은 이메일 1개의 링크를 여러 차례 클릭하는 경우 이러한 이벤트가 각각 따로 추적됩니까?

수신자가 이메일을 여러 번 열면 Amazon SES는 각 열기를 고유한 열기 이벤트로 계산합니다. 마찬가지로 수신자가 동일한 링크를 여러 번 클릭하면 Amazon SES는 각 클릭을 고유한 클릭 이벤트로 계산합니다. 그러나 이러한 수는 위의 메모 상자에 설명된 시나리오에 따라 왜곡될 수 있습니다.

Q3. 이메일 확인 및 클릭 지표가 집계됩니까? 혹은 수신자 수준으로 측정 가능합니까?

이메일 확인 및 클릭 수는 수신자 수준으로 추적됩니다. 따라서 이메일 확인 및 클릭 추적을 통해 어떤 수신자가 이메일을 확인했는지, 혹은 이메일의 링크를 클릭했는지 알 수 있습니다.

Q4. Amazon SES API를 사용하여 열기 및 클릭 지표를 검색할 수 있습니까?

Amazon SES API는 열기 및 클릭 지표를 검색하는 방법을 제공하지 않습니다. 하지만 CloudWatch API를 사용하여 Amazon SES에 대한 열기 및 클릭 지표를 검색할 수 있습니다. 예를 들어 AWS CLI 를 사용하여 다음 명령을 실행하여 CloudWatch API를 사용하여 클릭 지표를 검색할 수 있습니다.

```
aws cloudwatch get-metric-statistics --namespace AWS/SES --metric-name Click \
  --statistics Sum --period 86400 --start-time 2017-01-01T00:00:00Z \
  --end-time 2017-12-31T23:59:59Z
```

위에 표시된 명령은 2017년의 각 일자에 대한 총 클릭 이벤트 수를 검색합니다. 확인 지표를 검색하려면 `metric-name` 파라미터의 값을 `Open`으로 변경합니다. 또한 `start-time` 및 `end-time` 파라미터를 수정하여 분석 기간을 변경하거나 `period` 파라미터를 변경하여 더 세분화된 분석을 수행할 수 있습니다.

열기 추적

Q1. 열기 추적은 어떻게 작동합니까?

Amazon SES를 통해 전송되는 이메일마다 1x1픽셀의 투명 GIF 이미지가 삽입되며, 이러한 이미지 파일에 대한 고유 참조가 포함되어 있어서 이 이미지를 다운로드하면 SES에서 어떤 메시지를 누가 열었는지 정확히 알 수 있습니다.

기본적으로 이 픽셀은 이메일 하단에 삽입됩니다. 그러나 일부 이메일 공급자의 애플리케이션에서는 특정 크기를 초과하면 이메일의 미리 보기가 잘리고 메시지의 나머지 부분을 볼 수 있는 링크가 제공될 수 있습니다. 이 시나리오에서는 SES 픽셀 추적 이미지가 로드되지 않으며 추적하려는 오픈율에 집계되지 않습니다. 이 문제를 해결하려면 필요에 따라 이메일의 시작 부분에 픽셀을 배치하거나 이메일 본문에 `{{ses:openTracker}}` 자리표시자를 삽입하여 다른 위치에 픽셀을 배치하면 됩니다. SES가 자리표시자가 있는 메시지를 받으면 열기 추적 픽셀 이미지로 대체됩니다.

> **Important**
>
> `{{ses:openTracker}}` 자리 표시자를 두 개 이상 추가하면 `400 BadRequestException` 오류 코드가 반환되므로 자리 표시자를 하나만 추가하도록 합니다.

이러한 추적 픽셀이 추가되더라도 이메일의 모습은 바뀌지 않습니다.

Q2. 이메일 확인 추적 기능은 기본적으로 활성화됩니까?

열기 추적은 기본적으로 모든 Amazon SES 사용자에게 활성화됩니다. 이메일 추적 기능을 사용하려면 다음 방법을 따라야 합니다.

1. 구성 세트를 생성합니다.
2. 구성 세트에서 이벤트 대상을 생성합니다.
3. 이벤트 대상을 구성하여 이메일 확인 이벤트 알림을 대상에 게시합니다.
4. 이메일 확인을 추적할 모든 이메일에 1단계에서 생성한 구성 세트를 지정합니다.

구성 세트의 이벤트 대상을 통해 열기 추적을 사용 설정하는 방법에 대한 자세한 내용은 the section called “이벤트 대상 만들기” 섹션을 참조하세요. SMTP 이메일에서 서식 지정, 원시, 템플릿 이메일 방식으로 픽셀 자리표시자를 사용할 수 있습니다.

이벤트 게시를 사용하여 이메일 전송 모니터링 방법에 대해 자세히 알아보세요.

Q3. 특정 이메일에서 이메일 확인 추적 픽셀을 생략할 수 있습니까?

이메일에서 이메일 확인 추적 픽셀을 생략할 수 있는 방법은 두 가지가 있습니다. 첫 번째는 구성 세트를 지정하지 않고 이메일을 전송하는 방법입니다. 두 번째는 이메일 확인 이벤트에 대한 데이터를 게시하지 않도록 구성하여 구성 세트를 지정하는 방법입니다.

Q4. 일반 텍스트 이메일에 대해서도 이메일 확인을 추적합니까?

열기 추적은 HTML 이메일에서만 사용할 수 있습니다. 이메일 확인 추적 기능은 이미지를 추가해야 하기 때문에 텍스트 전용(비-HTML) 이메일 클라이언트를 사용하여 이메일을 확인하는 사용자는 이메일 확인 지표를 수집할 수 없습니다.

클릭 추적

Q1. 클릭 추적은 어떻게 작동합니까?

Amazon SES는 클릭을 추적할 목적으로 이메일 본문에 포함된 각 링크를 수정합니다. 수신자가 링크를 열면 Amazon SES 서버로 먼저 전송되고 나서 바로 대상 주소로 보내집니다. 이메일 확인 추적 기능과 마찬가지로 각 리디렉션 링크는 고유성을 갖습니다. 그렇기 때문에 어떤 수신자가 링크를 클릭했는지, 언제 클릭했는지, 그리고 어떤 이메일에서 링크를 클릭했는지 Amazon SES에서 알 수 있습니다.

> **Important**
>
> 하나의 메시지를 여러 수신자에게 보낸 경우 각 수신자는 동일한 클릭 추적 링크를 저장하게 됩니다. 개별 수신자의 클릭 동작을 추적하려면 한 번에 한 수신자에게 이메일을 보내십시오.

Q2. 클릭 추적을 비활성화할 수 있습니까?

`ses:no-track` 속성을 이메일의 HTML 본문에 있는 앵커 태그에 추가하면 개별 링크에 대한 클릭 추적을 비활성화할 수 있습니다. 예를 들어 AWS 홈 페이지에 연결하면 일반 앵커 링크는 다음과 유사합니다.

```
<a href="https://aws.amazon.com">Amazon Web Services</a>
```

해당 링크에 대한 클릭 추적을 비활성화하려면 다음과 유사하게 수정합니다.

```
<a ses:no-track href="aws.amazon.com">Amazon Web Services</a>
```

`ses:no-track`은(는) 표준 HTML 속성이 아니므로 Amazon SES는 수신자의 수신함에 도착하는 이메일 버전에서 이를 자동으로 제거합니다.

특정 구성 세트를 사용하여 보내는 모든 메시지에 대해 클릭 추적을 비활성화할 수도 있습니다. 클릭 추적을 비활성화하려면 클릭 이벤트가 캡처되지 않도록 구성 세트 이벤트 대상을 수정합니다.

구성 세트의 이벤트 대상을 통해 클릭 추적을 사용 중지하는 방법에 대한 자세한 내용은 the section called "이벤트 대상 만들기" 섹션을 참조하세요.

이벤트 게시를 사용하여 이메일 전송 모니터링 방법에 대해 자세히 알아보세요.

Q3. 각 이메일에서 몇 개의 링크를 추적할 수 있습니까?

클릭 추적 시스템은 최대 250개의 링크를 추적할 수 있습니다.

Q4. 일반 텍스트 이메일의 링크에서도 클릭 지표가 수집됩니까?

HTML 이메일에서 클릭 수를 추적하는 것만 가능합니다.

Q5. 링크에 고유 식별자로 태그를 지정할 수 있습니까?

`ses:tags` 속성을 사용하면 이메일의 링크에 제한 없이 태그를 키-값 페어로 추가할 수 있습니다. 이 속성을 사용할 때는 인라인 CSS 속성을 전달하는 데 사용한 것과 동일한 형식으로 키와 값을 지정합니다. 즉, 키를 입력하고 콜론(:)을 입력한 후 값을 입력합니다. 키-값 페어를 여러 개 전달해야 할 경우 각 페어를 세미콜론(;)으로 구분하십시오.

예를 들어 `product:book, genre:fiction, subgenre:scifi, type:newrelease` 태그를 링크에 추가한다고 가정하겠습니다. 그러면 태그가 추가된 링크는 다음과 같은 모습이 됩니다.

```
<a ses:tags="product:book;genre:fiction;subgenre:scifi;type:newrelease;"
    href="http://www.amazon.com/…/">New Releases in Science Fiction</a>
```

이 태그들은 사용자가 클릭한 특정 링크에 대한 추가 분석이 가능하도록 이벤트 게시 대상으로 전달됩니다.

> **Note**
>
> 링크 태그에는 숫자 0~9, 문자 A~Z(대문자 및 소문자 모두 가능), 하이픈(-), 밑줄(_)이 포함될 수 있습니다.

Q6. 추적되는 링크는 HTTP 또는 HTTPS 프로토콜을 사용합니까?

추적 링크는 이메일의 원래 링크와 동일한 프로토콜을 사용합니다.

예를 들어 이메일에 `https://www.amazon.com`에 대한 링크가 포함되어 있는 경우 해당 링크는 HTTPS 프로토콜을 사용하는 추적 링크로 바뀝니다. 이메일에 `http://www.example.com`에 대한 링크가 포함되어 있는 경우 해당 링크는 HTTP를 사용하는 추적 링크로 바뀝니다. 위에서 언급한 두 링크가 이메일에 모두 포함되어 있는 경우 HTTPS 링크는 HTTPS 프로토콜을 사용하는 추적 링크로 바뀌고, HTTP 링크는 HTTP 프로토콜을 사용하는 추적 링크로 바뀝니다.

Q7. 제 이메일의 링크가 추적되지 않습니다. 이유가 무엇입니까?

Amazon SES는 이메일의 링크에 올바르게 인코딩된 URL이 포함되었다고 가정합니다. 특히 링크의 URL은 RFC 3986을 준수해야 합니다. 이메일의 링크가 올바르게 인코딩되지 않은 경우에도 수신자는 이메일에서 링크를 볼 수 있으나 Amazon SES는 해당 링크의 클릭 이벤트를 추적하지 않습니다.

잘못된 인코딩과 관련된 문제는 일반적으로 쿼리 문자열이 포함된 URL에서 발생합니다. 예를 들어 이메일의 링크 URL의 쿼리 문자열에 인코딩되지 않은 공백 문자가 포함된 경우 이를테면 다음 예제에서처럼 ‘John‘과 ‘Doe‘ 사이에 공백이 있는 경우(예: http://www.example.com/path/to/page?name=John Doe) Amazon SES는 해당 링크를 추적하지 않습니다. 그러나 URL이 인코딩된 공백 문자를 사용하는 경우(예: http://www.example.com/path/to/page?name=John%20Doe) Amazon SES는 이 링크를 평소대로 추적합니다.

빠른 찾기 인덱스

다음 색인은 사용 방법 또는 개념의 두 가지 검색 방법을 제공하여 Amazon SES에서 항목을 빠르게 찾을 수 있도록 생성되었습니다. 방법은 작업을 수행하는 "방법"을 설명하는 반면 개념은 더 큰 그림을 설명합니다.

> 여러분의 의견을 알려주세요.
>
> 오른쪽 위의 피드백 버튼을 사용하여 알려주세요...
>
> - 이 인덱스가 도움이 되었나요?
> - 이 인덱스에 추가하려는 방법 또는 개념이 있나요?
> - 다르게 분류해야 한다고 생각했던 것이 있었나요?

SES 사용 방법 및 개념 링크

How-tos

SES 방법 링크는 알파벳순으로 나열되며 선택하신 작업을 수행하는 “방법”을 보여주는 해당 섹션으로 연결됩니다.

- 방법 알아보기...
  - 사용자 지정 MAIL FROM 도메인 설정의 일부로 SPF 레코드 추가
  - IP 풀 할당
  - 스팸 이메일 수신 차단
  - 사용자 지정 열기/클릭 도메인 구성
  - SNS 알림 구성
  - SMTP 엔드포인트에 연결
  - 구성 세트 생성
  - 도메인 자격 증명 생성
  - 이메일 주소 자격 증명 생성
  - 이벤트 대상 생성
  - IP 주소 필터 생성
  - 전용 IP(관리형) 사용을 위한 관리형 IP 풀 생성

- 수신 규칙 생성
- CloudWatch를 사용하여 평판 경보 생성
- 사용자 지정 정책을 사용하여 전송 권한 부여 정책 생성
- 정책 생성기를 사용하여 전송 권한 부여 정책 생성
- 전용 IP 주소(표준)용 표준 전용 IP 풀 생성
- 자격 증명 삭제
- 개인 데이터 삭제
- 자격 증명 편집
- 이메일 피드백 전달 활성화
- 평판 지표 내보내기
- 샌드박스에서 나가기
- SES 시작하기
- 가상 배달 가능성 관리자 시작하기
- 이메일 수신을 위한 권한 부여
- 처리량 높이기
- 전송 할당량 높이기
- 기존 이메일 서버와 통합
- API 호출 로깅
- 구성 세트 관리
- Easy DKIM 및 BYODKIM 관리
- 전송 및 평판 지표 모니터링
- 전송 통계 모니터링
- 사용량 통계 모니터링
- 전송 할당량 모니터링
- 자격 증명의 DKIM 레코드 얻기
- SMTP 보안 인증 정보 획득
- 구성 세트 수준 금지로 계정 수준 금지 재정의
- 이메일 주소 자격 증명의 상속된 DKIM 서명 재정의
- 이메일 전송 일시 중지

- MX 레코드 게시

- AWS 리소스 남용 보고
- 전용 IP 주소 요청
- 기술 지원 요청
- 가상 배달 가능성 관리자 어드바이저를 사용하여 전달 및 평판 문제 해결
- CloudWatch에서 이벤트 데이터 검색
- Kinesis Data Firehose에서 이벤트 데이터 검색
- SNS에서 이벤트 데이터 검색
- AWS SDK를 사용하여 이메일 전송
- 프로그래밍 방식으로 이메일 전송
- SES API를 사용하여 이메일 전송
- SMTP를 사용하여 이메일 전송
- CLI 또는 SES API를 사용하여 첨부 파일이 포함된 원시 이메일 전송
- 메일박스 시뮬레이터를 사용하여 테스트 이메일 전송
- BYODKIM(자체 DKIM 사용) 설정
- DMARC 정책 설정
- Easy DKIM 설정
- 이메일 수신 설정
- 이벤트 게시 설정
- MAIL FROM 도메인 설정
- 전송 권한 부여 설정(자격 증명 소유자 작업)
- 전송 권한 부여 설정(위임 발신자 작업)
- 이메일 전송 시 구성 세트 지정
- SMTP 인터페이스와 연결 테스트
- 반송 및 불만 제기 비율 추적
- 상속된 DKIM 서명 속성 이해
- 평판 지표 사용
- 소프트웨어 패키지를 사용하여 이메일 보내기
- 구독 관리 사용
- 템플릿을 사용하여 이메일 보내기

- 계정 수준 금지 목록 사용

- 도메인 자격 증명 확인
- 이메일 주소 자격 증명 확인
- 자격 증명 보기
- 가상 배달 가능성 관리자 대시보드를 사용하여 계정의 전달 가능성 지표를 개괄적 및 세부적 수준에서 확인
- 전용 IP에 대한 SNDS 지표 보기
- 전용 IP 주소 워밍업

Concepts

SES 개념 링크는 알파벳순으로 나열되며 선택하신 개념을 설명하는 해당 챕터와 섹션으로 연결됩니다.

- 다음에 대한 정보 찾기...
  - AWS 리소스 남용, 보고서
  - 계정 대시보드
  - 계정 수준 금지 목록
  - 이메일 수신 작업 옵션
  - 헤더 추가 작업
  - 연결 형식, 지원되지 않음
  - 반송 메일 응답 작업, 반환
  - BYODKIM(자체 DKIM 사용)
  - BYOIP(자체 IP 사용)
  - 코드 예제
  - 규정 준수 검증
  - 구성 세트 수준 금지
  - 구성 세트
  - 콘텐츠 인코딩
  - 계정 간 알림 레거시 지원
  - 사용자 지정 MAIL FROM 도메인
  - 데이터 보호

  - 전용 IP 주소

- 전용 IP 주소(관리형)
- 전용 IP 주소(표준)
- DKIM, 이메일 인증
- DMARC(Domain-based Message Authentication, Reporting and Conformance)
- DKIM을 통해 DMARC 준수
- SPF를 통해 DMARC 준수
- Easy DKIM
- 이메일 피드백 전달 대상
- 이메일 수신 인증
- 이메일 수신 개념
- 이메일 수신 콘솔 연습
- 이메일 수신 맬웨어 검사
- 이메일 수신 권한
- 이메일 수신 사용 사례
- 이메일 수신 제한 사항
- 이메일 전송 인증 방법
- 엔드포인트
- 이벤트 알림
- 이메일을 통한 이벤트 알림
- SNS를 통한 이벤트 알림
- 이벤트 게시
- FAQ(Frequently Asked Questions)
- 전역 금지 목록
- 지원되는 헤더 필드
- 자격 증명 관리
- ID 및 액세스 관리
- 인프라 보안
- Amazon WorkMail과 통합 작업
- IP 주소 필터를 사용한 IP 기반 제어

- Lambda 함수 작업, 호출

- 목록 관리
- 목록 및 구독
- 로깅 및 모니터링
- 맬웨어 탐지
- 수동 DKIM 서명
- 이벤트 게시를 사용하여 이메일 전송 모니터링
- 발신자 평판 모니터링
- 전송 활동 모니터링
- 할당량
- 수신 규칙
- 수신 규칙을 사용한 수신자 기반 제어
- 리전
- 평판 지표
- 평판 지표 메시지
- 복원성
- S3 버킷 작업, 전달
- 샌드박스 - 나가기
- 보안
- 보안 프로토콜, 지원됨
- 전송 권한 부여
- 전송 권한 부여 정책 구조
- 전송 권한 부여 정책 예제
- 전송 권한 부여 프로세스
- 전용 IP에 대한 SNDS 지표
- SNS 알림 내용
- SNS 알림 예제
- SNS 주제 작업, 게시
- SPF(발신자 정책 프레임워크)
- 규칙 세트 중지 작업

- 구독 관리

- 지원, 기술 요청
- 사용자 지정 이메일 확인 템플릿
- 문제 해결
- 확인된 자격 증명
- 가상 배달 가능성 관리자
- VPC 엔드포인트

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