사용 설명서

Outposts에서의 Amazon S3

API 버전 2006-03-01

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

Outposts에서의 Amazon S3: 사용 설명서

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

Amazon의 상표 및 브랜드 디자인은 Amazon 외 제품 또는 서비스와 함께, Amazon 브랜드 이미지를 떨어뜨리거나 고객에게 혼동을 일으킬 수 있는 방식으로 사용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon과 제휴 관계이거나 관련이 있거나 후원 관계와 관계없이 해당 소유자의 자산입니다.

Table of Contents

S3 on Outposts란 무엇인가요? 1
- S3 on Outposts 작동 방식 1
  - 리전 2
  - 버킷 2
  - Objects 3
  - 키 3
  - S3 버전 관리 3
  - 버전 ID 4
  - 스토리지 클래스 및 암호화 4
  - 버킷 정책 4
  - S3 on Outposts 액세스 포인트 5
- S3 on Outposts 기능 5
  - 액세스 관리 5
  - 스토리지 로깅 및 모니터링 6
  - 강력한 일관성 6
- 관련 서비스 6
- S3 on Outposts 액세스 7
  - AWS Management Console 7
  - AWS Command Line Interface 7
  - AWS SDK 7
- S3 on Outposts 요금 지불 8
- 다음 단계 8

Outpost 설정 9
- 새 Outpost 주문 9

S3 on Outposts의 차이점 10
- 사양 10
- 지원되는 API 작업 11
- S3 on Outposts에서 지원하는 Amazon S3 AWS CLI 명령 11
- 지원되지 않는 Amazon S3 기능 11
- 네트워크 제한 12

S3 on Outposts 시작하기 14
- S3 콘솔 사용 14
  - 버킷, 액세스 포인트 및 엔드포인트 생성 15
  - 다음 단계 17

AWS CLI 및 Java용 SDK 사용 ... 17
1단계: 버킷 만들기 ... 18
2단계: 액세스 포인트 생성 ... 19
3단계: 엔드포인트 생성 ... 20
4단계: S3 on Outposts 버킷에 객체 업로드 ... 22
S3 on Outposts에 대한 네트워킹 ... 23
네트워킹 액세스 유형 선택 ... 23
S3 on Outposts 버킷과 객체에 액세스 ... 23
교차 계정 탄력적 네트워크 인터페이스를 사용하여 연결 관리 ... 24
S3 on Outposts 버킷 작업 ... 25
버킷 ... 25
액세스 포인트 ... 25
엔드포인트 ... 26
S3 on Outposts에 대한 API 작업 ... 26
S3 on Outposts 버킷의 생성 및 관리 ... 28
버킷 생성 ... 28
태그 추가 ... 32
버킷 정책 사용 ... 33
버킷 정책 추가 ... 33
버킷 정책 보기 ... 36
버킷 정책 삭제 ... 37
버킷 정책 예제 ... 38
버킷 나열 ... 41
버킷 가져오기 ... 43
버킷 삭제 ... 44
액세스 포인트 작업 ... 45
액세스 포인트 생성 ... 46
액세스 포인트에 버킷 스타일 별칭 사용 ... 48
액세스 포인트 구성 보기 ... 52
액세스 포인트 나열 ... 53
액세스 포인트 삭제 ... 54
액세스 포인트 정책 추가 ... 55
액세스 포인트 정책 보기 ... 57
엔드포인트 작업 ... 58
엔드포인트 생성 ... 60
엔드포인트 나열 ... 62

엔드포인트 삭제 ... 63
S3 on Outposts 객체 작업 ... 65
객체 업로드 ... 66
객체 복사 ... 68
Java용 AWS SDK 사용 ... 69
객체 가져오기 ... 70
객체 나열 ... 73
객체 삭제 ... 76
HeadBucket 사용 ... 80
멀티파트 업로드 수행 ... 82
S3 on Outposts 버킷에 있는 객체의 멀티파트 업로드 수행 ... 82
멀티파트 업로드를 사용하여 S3 on Outposts 버킷에 있는 대형 객체 복사 ... 85
S3 on Outposts 버킷에 있는 객체의 부분 나열 ... 87
S3 on Outposts 버킷에서 진행 중인 멀티파트 업로드 목록 검색 ... 88
미리 서명된 URL 사용 ... 89
미리 서명된 URL 기능 제한 ... 90
미리 서명된 URL을 생성할 수 있는 사용자 ... 92
S3 on Outposts는 미리 서명된 URL의 만료 날짜 및 시간을 언제 확인하나요? ... 92
객체 공유 ... 93
객체 업로드 ... 98
로컬 Amazon EMR을 사용하는 Amazon S3 on Outposts ... 102
Amazon S3 on Outposts 버킷 생성 ... 103
Amazon S3 on Outposts를 사용하여 Amazon EMR 사용 시작 ... 104
권한 부여 및 인증 캐시 ... 109
권한 부여 및 인증 캐시 구성 ... 110
SigV4A 서명 검증 ... 110
보안 ... 111
IAM 설정 ... 111
S3 on Outposts 정책의 보안 주체 ... 113
Outposts 기반 S3의 ARN ... 114
S3 on Outposts에 대한 정책 예제 ... 116
엔드포인트에 대한 권한 ... 117
S3 on Outposts의 서비스 연결 역할 ... 119
데이터 암호화 ... 119
S3 on Outposts에 대한 AWS PrivateLink ... 120
규제 및 제한 ... 121

S3 on Outposts 인터페이스 엔드포인트 액세스 122
온프레미스 DNS 구성 업데이트 123
VPC 엔드포인트 생성 124
VPC 엔드포인트 정책 및 버킷 정책 생성 124
Signature Version 4(SigV4) 정책 키 126
서명 버전 4 관련 조건 키를 사용하는 버킷 정책 예제 127
AWS 관리형 정책 129
AWSS3OnOutpostsServiceRolePolicy 130
정책 업데이트 130
서비스 링크 역할 사용 131
S3 on Outposts에 대한 서비스 연결 역할 권한 131
S3 on Outposts에 대한 서비스 연결 역할 생성 134
S3 on Outposts에 대한 서비스 연결 역할 편집 134
S3 on Outposts에 대한 서비스 연결 역할 삭제 135
S3 on Outposts 서비스 연결 역할에 대해 지원되는 리전 135
S3 on Outposts 스토리지 관리 136
S3 버전 관리에 대한 관리 136
수명 주기 구성 생성 및 관리 138
콘솔 사용 139
AWS CLI 및 Java용 SDK 사용 143
S3 on Outposts에 대한 객체 복제 146
복제 구성 147
Outposts에서 S3 복제 기능의 요구 사항 148
복제 가능한 객체 149
복제 불가능한 객체 149
Outposts에서 S3 복제 기능이 지원하지 않는 것은 무엇인가요? 150
복제 설정 150
복제 관리 168
S3 on Outposts 공유 175
사전 조건 175
절차 176
사용 예제: 177
기타 서비스 179
S3 on Outposts 모니터링 181
CloudWatch 지표 181
CloudWatch 지표 181

Amazon CloudWatch Events 183
CloudTrail 로그 184
S3 on Outposts 객체에 대해 CloudTrail 로깅 활성화 185
Amazon S3 on Outposts AWS CloudTrail 로그 파일 항목 187
S3 on Outposts로 개발하기 190
지원되는 리전 190
S3 on Outposts API 191
객체 관리를 위한 Amazon S3 API 작업 191
버킷 관리를 위한 Amazon S3 Control API 작업 192
Outposts 관리를 위한 S3 on Outposts API 작업 193
S3 제어 클라이언트 구성 194
IPv6을 통해 요청 194
IPv6 시작하기 195
듀얼 스택 엔드포인트를 사용하여 요청 195
IAM 정책에서 IPv6 주소 사용 196
IP 주소 호환성 테스트 197
AWS PrivateLink로 IPv6 사용 198
듀얼 스택 엔드포인트 사용 201

Amazon S3 on Outposts란 무엇인가요?

AWS Outposts는 진정으로 일관된 하이브리드 환경을 위해 거의 모든 데이터 센터, 콜로케이션 공간 또는 온프레미스 시설에 동일한 AWS 인프라, AWS 서비스, API 및 도구를 제공하는 완전관리형 서비스입니다. AWS Outposts는 온프레미스 시스템, 로컬 데이터 처리, 데이터 상주 및 로컬 시스템과 상호 의존하는 애플리케이션의 마이그레이션에 대해 짧은 지연 시간으로 액세스해야 하는 워크로드에 적합합니다. 자세한 내용은 AWS Outposts 사용 설명서의 AWS Outposts이란 무엇입니까? 섹션을 참조하십시오.

Amazon S3 on Outposts를 사용하면 Outposts에 S3 버킷을 생성하고 온프레미스에서 객체를 손쉽게 저장 및 검색할 수 있습니다. S3 on Outposts는 OUTPOSTS라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outposts 버킷과 통신합니다.

액세스 정책, 암호화, 태그 지정을 포함하여 Amazon S3에서와 같이 Outposts 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다.

- S3 on Outposts 작동 방식
- S3 on Outposts 기능
- 관련 서비스
- S3 on Outposts 액세스
- S3 on Outposts 요금 지불
- 다음 단계

S3 on Outposts 작동 방식

S3 on Outposts는 데이터를 Outpost의 버킷 내 객체로 저장하는 객체 스토리지 서비스입니다. 객체는 데이터 파일 및 해당 파일을 설명하는 모든 메타데이터입니다. 버킷은 객체에 대한 컨테이너입니다.

S3 on Outposts에 데이터를 저장하려면 먼저 버킷을 생성합니다. 버킷을 생성할 때 버킷 이름과 버킷을 저장할 Outpost를 지정합니다. S3 on Outposts 버킷에 액세스하여 객체 작업을 수행하려면 다음으로 액세스 포인트를 생성하고 구성합니다. 액세스 포인트로 요청을 라우팅하려면 엔드포인트도 생성해야 합니다.

액세스 포인트는 모든 AWS 서비스 또는 S3에 데이터를 저장하는 고객 애플리케이션에 대한 데이터 액세스를 간소화합니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 객체 작업을 수행하는 데 사용할 수 있는 버킷에 연결된 네트워크 엔드포인트입니다. 각 액세스 포인트에는 고유한 권한 및 네트워크 제어가 있습니다.

AWS Management Console, AWS CLI, AWS SDK 또는 REST API를 사용하여 S3 on Outposts 버킷, 액세스 포인트 및 엔드포인트를 생성하고 관리할 수 있습니다. S3 on Outposts 버킷의 객체를 업로드하고 관리하려면 AWS CLI, AWS SDK 또는 REST API를 사용하면 됩니다.

리전

AWS Outposts 프로비저닝 중 사용자 또는 AWS가 버킷 작업 및 원격 측정을 위해 선택한 AWS 리전 또는 Outposts 홈 리전으로 Outpost를 다시 연결하는 서비스 링크 연결을 생성합니다. Outpost는 상위 AWS 리전에 대한 연결에 의존합니다. Outposts 랙은 연결이 끊긴 작업 또는 연결 없음으로 제한된 환경을 위해 설계되지 않았습니다. 자세한 내용은 AWS Outposts 사용 설명서의 AWS 리전으로의 Outpost 연결을 참조하세요.

버킷

버킷은 S3 on Outposts에 저장된 객체에 대한 컨테이너입니다. 버킷에 저장할 수 있는 객체 수에는 제한이 없습니다. 또한 Outpost마다 계정당 버킷을 최대 100개까지 포함할 수 있습니다.

버킷을 생성할 때 버킷 이름을 입력하고 버킷이 속할 Outpost를 선택합니다. 버킷을 생성한 후에는 버킷 이름을 변경하거나 버킷을 다른 Outpost로 이전할 수 없습니다. 버킷 이름은 Amazon S3 버킷 이름 지정 규칙을 따라야 합니다. S3 on Outposts에서 버킷 이름은 Outpost 및 AWS 계정마다 고유합니다. S3 on Outposts 버킷을 식별하려면 `outpost-id`, `account-id` 및 버킷 이름이 필요합니다.

다음의 예시는 S3 on Outposts 버킷에 대한 Amazon 리소스 이름(ARN) 형식을 보여줍니다. ARN은 Outpost의 홈 리전과 Outpost 계정, Outpost ID 및 버킷 이름으로 구성됩니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/bucket/bucket-name
```

모든 객체는 어떤 버킷에 포함됩니다. Outposts 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 객체 작업에 버킷을 지정할 때 액세스 포인트 이름이 포함된 액세스 포인트 ARN 또는 액세스 포인트 별칭을 사용합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

다음 예제는 `outpost-id`, `account-id` 및 액세스 포인트 이름을 포함하는 Outposts의 S3에 대한 액세스 포인트 ARN 형식을 보여줍니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

버킷에 대한 자세한 내용은 S3 on Outposts 버킷 작업을 참조하십시오.

Objects

객체는 S3 on Outposts에 저장되는 기본 엔터티입니다. 객체는 객체 데이터와 메타데이터로 구성됩니다. 메타데이터는 객체를 설명하는 이름-값 페어의 집합입니다. 여기에는 마지막으로 수정한 날짜와 같은 몇 가지 기본 메타데이터 및 `Content-Type` 같은 표준 HTTP 메타데이터가 포함됩니다. 객체를 저장할 때 사용자 지정 메타데이터를 지정할 수도 있습니다. 객체는 키 (또는 이름)을 통해 버킷 내에서 고유하게 식별됩니다.

Amazon S3 on Outposts를 사용할 경우 객체 데이터는 항상 Outpost에 저장됩니다. AWS가 Outpost 랙을 설치하는 경우 데이터 상주 요구 사항을 충족하기 위해 데이터가 Outpost에 로컬로 유지됩니다. 객체는 Outpost에서 벗어나지 않으며 AWS 리전에 있지 않습니다. AWS Management Console이 리전 내에 호스팅되므로 콘솔을 사용하여 Outpost의 객체를 업로드하거나 관리할 수 없습니다. 그러나 REST API, AWS Command Line Interface(AWS CLI), AWS SDK를 사용하여 액세스 포인트를 통해 객체를 업로드하고 관리할 수 있습니다.

키

객체 키(또는 키 이름)는 버킷 내 객체에 대한 고유한 식별자입니다. 버킷 내 모든 객체는 정확히 하나의 키를 갖습니다. 버킷과 객체 키의 조합은 각 객체를 고유하게 식별합니다.

다음 예제는 S3 on Outposts 객체에 대한 ARN 형식을 보여줍니다. 여기에는 Outpost의 홈 리전에 대한 AWS 리전 코드, AWS 계정 ID, Outpost ID, 버킷 이름 및 객체 키가 포함됩니다.

```
arn:aws:s3-outposts:us-west-2:123456789012:outpost/ op-01ac5d28a6a232904/bucket/amzn-s3-demo-bucket1/object/myobject
```

객체 키에 대한 자세한 내용은 S3 on Outposts 객체 작업 섹션을 참조하십시오.

S3 버전 관리

Outposts 버킷에 S3 버전 관리를 사용하면 동일 버킷 내에 여러 개의 객체 변형을 보유할 수 있습니다. S3 버전 관리를 사용하여 버킷에 저장된 모든 버전의 모든 객체를 보존, 검색 및 복원할 수 있습니다. S3 버전 관리는 의도치 않은 사용자 작업 및 애플리케이션 장애로부터 복구하는 데 도움이 됩니다.

자세한 내용은 S3 on Outposts 버킷의 S3 버전 관리에 대한 관리 단원을 참조하십시오.

버전 ID

버킷에 S3 버전 관리를 활성화하면 S3 on Outposts에서 버킷에 추가되는 각 객체에 고유한 버전 ID를 생성합니다. 버전 관리를 사용 설정할 때 버킷에 이미 존재하는 객체에는 `null`의 버전 ID가 있습니다. 이러한(또는 다른) 객체를 PutObject와 같은 기타 작업으로 수정하는 경우 새 객체가 고유한 버전 ID를 가집니다.

자세한 내용은 S3 on Outposts 버킷의 S3 버전 관리에 대한 관리 단원을 참조하십시오.

스토리지 클래스 및 암호화

S3 on Outposts는 새로운 스토리지 클래스인 S3 Outposts(`OUTPOSTS`)를 제공합니다. S3 Outposts 스토리지 클래스는 AWS Outposts의 버킷에 저장된 객체에만 사용할 수 있습니다. 다른 S3 스토리지 클래스를 S3 on Outposts에 사용하려고 하면 S3 on Outposts에서 `InvalidStorageClass` 오류를 반환합니다.

기본적으로 S3 Outposts(`OUTPOSTS`) 스토리지 클래스에 저장된 객체는 Amazon S3 관리형 암호화 키를 통한 서버 측 암호화(SSE-S3)를 사용하여 암호화됩니다. 자세한 내용은 S3 on Outposts의 데이터 암호화 단원을 참조하십시오.

버킷 정책

버킷 정책은 버킷과 해당 버킷의 객체에 액세스 권한을 부여할 수 있는 리소스 기반 AWS Identity and Access Management(IAM) 정책입니다. 버킷 소유자만 정책을 버킷에 연결할 수 있습니다. 버킷에 연결된 권한은 버킷 소유자가 소유한 모든 버킷의 객체에 적용됩니다. 버킷 정책은 크기가 20KB로 제한됩니다.

버킷 정책에는 AWS에서 표준인 JSON 기반 IAM 정책 언어가 사용됩니다. 버킷 정책을 사용하여 버킷의 객체에 대한 권한을 추가하거나 거부할 수 있습니다. 버킷 정책은 정책의 요소를 기반으로 요청을 허용 또는 거부합니다. 이러한 요소에는 요청자, S3 on Outposts 작업, 리소스 및 요청의 측면 또는 조건(예: 요청을 수행하는 데 사용된 IP 주소)이 포함될 수 있습니다. 예를 들어 버킷 소유자가 업로드된 객체를 완전히 제어할 수 있도록 하면서 S3 on Outposts 버킷에 객체를 업로드할 수 있는 교차 계정 권한을 부여하는 버킷 정책을 생성할 수 있습니다.

버킷 정책에서는 ARN 및 기타 값에 와일드카드 문자(`*`)를 사용하여 객체의 하위 집합에 권한을 부여할 수 있습니다. 예를 들어, 공통 접두사로 시작하거나 `.html`과 같은 지정된 확장자로 끝나는 객체 그룹에 대한 액세스를 제어할 수 있습니다.

S3 on Outposts 액세스 포인트

S3 on Outposts 액세스 포인트는 해당 엔드포인트를 사용하여 데이터에 액세스하는 방법을 설명하는 전용 액세스 정책이 포함된 명명된 네트워크 엔드포인트입니다. 액세스 포인트는 S3 on Outposts의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 S3 객체 작업을 수행하는 데 사용할 수 있는 버킷에 연결됩니다.

객체 작업에 버킷을 지정할 때 액세스 포인트 이름이 포함된 액세스 포인트 ARN 또는 액세스 포인트 별칭을 사용합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

액세스 포인트에는 해당 액세스 포인트를 통해 이루어진 모든 요청에 대해 S3 on Outposts가 적용하는 고유한 권한 및 네트워크 제어가 있습니다. 각 액세스 포인트는 기본 버킷에 연결된 버킷 정책과 함께 작동하는 사용자 지정 액세스 포인트 정책을 적용합니다.

자세한 내용은 S3 on Outposts 버킷과 객체에 액세스 단원을 참조하십시오.

S3 on Outposts 기능

액세스 관리

S3 on Outposts는 버킷 및 객체에 대한 액세스 감사 및 관리 기능을 제공합니다. 기본적으로 S3 on Outposts 버킷 및 객체는 프라이빗입니다. 생성한 S3 on Outposts 리소스에만 액세스할 수 있습니다.

특정 사용 사례를 지원하는 세분화된 리소스 권한을 부여하거나 S3 on Outposts 리소스의 권한을 감사하기 위해 다음 기능을 사용할 수 있습니다.

- S3 퍼블릭 액세스 차단 - 버킷과 객체에 대한 퍼블릭 액세스를 차단합니다. Outposts에 있는 버킷의 경우 퍼블릭 액세스 차단 기능은 기본적으로 항상 사용으로 설정되어 있습니다.
- AWS Identity and Access Management(IAM) – IAM은 S3 on Outposts resources 리소스를 포함하여 AWS 리소스에 대한 액세스를 안전하게 제어하는 데 도움이 되는 웹 서비스입니다. IAM을 사용하면 사용자가 액세스할 수 있는 AWS 리소스를 제어하는 권한을 중앙에서 관리할 수 있습니다. IAM을 사용하여 리소스를 사용하도록 인증(로그인) 및 권한 부여(권한 있음)된 대상을 제어합니다.
- S3 on Outposts 액세스 포인트 - S3 on Outposts의 공유 데이터 세트에 대한 데이터 액세스를 관리합니다. 액세스 포인트는 전용 액세스 정책이 포함된 네트워크 엔드포인트입니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 객체 작업을 수행하는 데 사용할 수 있는 버킷에 연결됩니다.
- 버킷 정책 - IAM 기반 정책 언어를 사용하여 S3 버킷과 그 안에 있는 객체에 대한 리소스 기반 권한을 구성합니다.

- AWS Resource Access Manager(AWS RAM) - AWS 계정, 조직 내 또는 AWS Organizations의 조직 단위(OU) 간에 S3 on Outposts 용량을 안전하게 공유합니다.

스토리지 로깅 및 모니터링

S3 on Outposts는 S3 on Outposts 리소스가 사용되는 방식을 모니터링하고 제어하는 데 사용할 수 있는 로깅 및 모니터링 도구를 제공합니다. 자세한 내용은 모니터링 도구를 참조하세요.

- S3 on Outposts에 대한 Amazon CloudWatch 지표 – 리소스의 운영 상태를 추적하고 용량 가용성을 파악합니다.
- S3 on Outposts에 대한 Amazon CloudWatch Events 이벤트 – Amazon Simple Queue Service(Amazon SQS), Amazon Simple Notification Service(Amazon SNS) 및 AWS Lambda를 비롯하여 지원되는 모든 CloudWatch Events 대상을 통해 알림을 수신하도록 하는 S3 on Outposts API 이벤트에 대한 규칙을 생성합니다.
- S3 on Outposts에 대한 AWS CloudTrail 로그 – S3 on Outposts의 사용자, 역할 또는 AWS 서비스가 수행한 조치를 기록합니다. CloudTrail 로그는 S3 버킷 수준 및 객체 수준 작업에 대한 자세한 API 추적을 제공합니다.

강력한 일관성

S3 on Outposts는 모든 AWS 리전의 S3 on Outposts 버킷에 있는 객체의 PUT 및 DELETE 요청에 대해 강력한 쓰기 후 읽기(read-after-write) 일관성을 제공합니다. 이는 새 객체에 대한 쓰기, 기존 객체를 덮어쓰는 PUT 요청 및 DELETE 요청 모두에 적용됩니다. 또한 S3 on Outposts 객체 태그 및 객체 메타데이터(예: HEAD 객체)는 강력한 일관성을 갖습니다. 자세한 내용은 Amazon S3 사용 설명서의 Amazon S3 데이터 일관성 모델을 참조하세요.

관련 서비스

데이터를 S3 on Outposts로 로드한 후에는 해당 데이터를 다른 AWS 서비스에 사용할 수 있습니다. 가장 자주 사용하게 될 서비스는 다음과 같습니다.

- Amazon Elastic Compute Cloud(Amazon EC2) – AWS 클라우드에서 안전하고 확장 가능한 컴퓨팅 용량을 제공합니다. Amazon EC2를 사용하면 하드웨어에 선투자할 필요성이 감소되어 더 빠르게 애플리케이션을 개발하고 배포할 수 있습니다. Amazon EC2를 사용하여 원하는 수의 가상 서버를 구축하고 보안 및 네트워킹을 구성하며 스토리지를 관리할 수 있습니다.

- Amazon Elastic Block Store(Amazon EBS) on Outposts – Amazon EBS local snapshots on Outposts를 사용하여 Outpost의 볼륨 스냅샷을 S3 on Outposts에 로컬로 저장합니다.
- Amazon Relational Database Service(Amazon RDS) on Outposts – Amazon RDS 로컬 백업을 사용하여 Amazon RDS 백업을 Outpost에 로컬로 저장합니다.
- AWS DataSync – Outposts와 AWS 리전 간 데이터 전송을 자동화하고, 전송할 대상, 전송 시기 및 사용할 네트워크 대역폭의 양을 선택할 수 있습니다. S3 on Outposts는 와(과) 통합됩니다AWS DataSync 대량의 로컬 처리가 필요한 온프레미스 애플리케이션의 경우, S3 on Outposts는 온프레미스 객체 스토리지를 제공하여 네트워크 변화로 인한 데이터 전송 및 버퍼를 최소화하고 Outposts와 AWS 리전 간에 데이터를 쉽게 전송할 수 있는 기능을 제공합니다.

S3 on Outposts 액세스

다음 방법 중 하나를 사용하여 S3 on Outposts에서 작업할 수 있습니다.

AWS Management Console

이 콘솔은 S3 on Outposts 및 AWS 리소스를 관리하기 위한 웹 기반 사용자 인터페이스입니다. AWS 계정에 가입한 사용자는 AWS Management Console에 로그인한 후 AWS Management Console 홈 페이지에서 S3를 선택하여 S3 on Outposts에 액세스할 수 있습니다. 그런 다음 왼쪽 탐색 창에서 Outposts 버킷(Outposts buckets)을 선택합니다.

AWS Command Line Interface

AWS 명령줄 도구를 통해 시스템 명령줄에서 명령을 실행하거나 스크립트를 구축하여 AWS(S3 등) 작업을 수행할 수 있습니다.

이AWS Command Line Interface(AWS CLI)는 광범위한 AWS 서비스의 명령을 제공합니다. AWS CLI는 Windows, macOS, Linux에서 지원됩니다. 시작하려면 AWS Command Line Interface 사용 설명서를 참조하세요. S3 on Outposts에 사용할 수 있는 명령에 대한 자세한 내용은 AWS CLI 명령 참조의 s3api, s3control 및 s3outposts를 참조하세요.

AWS SDK

AWS에서는 다양한 프로그래밍 언어 및 플랫폼(Java, Python, Ruby, .NET, iOS, Android 등)을 위한 라이브러리와 샘플 코드로 구성된 소프트웨어 개발 키트(SDK)를 제공합니다. AWS SDK를 사용하면 편리하게 S3 on Outposts 및 AWS에 프로그래밍 방식으로 액세스할 수 있습니다. S3 on Outposts가 Amazon S3와 동일한 SDK를 사용하므로 S3 on Outposts는 동일한 S3 API, 자동화 및 도구를 사용하는 일관된 환경을 제공합니다.

S3 on Outposts는 REST 서비스입니다. 기본 REST API를 래핑하고 프로그래밍 태스크를 간소화하는 AWS SDK 라이브러리를 사용하여 S3 on Outposts에 요청을 전송할 수 있습니다. 예를 들어 SDK는 서명 계산, 암호화 방식으로 요청 서명, 오류 관리 및 자동으로 요청 재시도와 같은 작업을 처리합니다. 다운로드 및 설치 방법을 비롯하여 AWS SDK에 대한 자세한 내용은 AWS에서의 구축을 위한 도구를 참조하세요.

S3 on Outposts 요금 지불

Amazon EC2 인스턴스 유형, Amazon EBS 범용 SSD(Solid State Drive) 볼륨(gp2) 및 S3 on Outposts의 조합을 제공하는 다양한 AWS Outposts 랙 구성을 구매할 수 있습니다. 요금에는 제공, 설치, 인프라 서비스 유지 보수, 소프트웨어 매치 및 업그레이드가 포함됩니다.

자세한 내용은 AWS Outposts 랙 요금을 참조하세요.

다음 단계

S3 on Outposts 작업에 대한 자세한 내용은 다음 주제를 참조하세요.

- Outpost 설정
- Amazon S3 on Outposts와 Amazon S3의 차이점
- Amazon S3 on Outposts 시작하기
- S3 on Outposts에 대한 네트워킹
- S3 on Outposts 버킷 작업
- S3 on Outposts 객체 작업
- S3 on Outposts의 보안
- S3 on Outposts 스토리지 관리
- Amazon S3 on Outposts로 개발

Outpost 설정

Amazon S3 on Outposts를 시작하려면 Outpost with Amazon S3 용량을 시설에 배포해야 합니다. Outpost 및 S3 용량 주문 옵션에 대한 자세한 내용은 AWS Outposts 섹션을 참조하세요. Outposts에 S3 용량이 있는지 확인하려면 ListOutpostsWithS3 API 호출을 사용하면 됩니다. 사양 및 S3 on Outposts와 Amazon S3의 차이점은 Amazon S3 on Outposts와 Amazon S3의 차이점 섹션을 참조하세요.

자세한 내용은 다음 항목을 참조하십시오.

주제

- 새 Outpost 주문

새 Outpost 주문

새 Outpost with S3 용량을 주문해야 하는 경우 AWS Outposts 랙 요금에서 Amazon Elastic Compute Cloud (Amazon EC2), Amazon Elastic Block Store (Amazon EBS), Amazon S3의 용량 옵션을 확인합니다.

원하는 구성을 선택한 후 AWS Outposts 사용 설명서의 Outpost 생성 및 Outpost 용량 주문에 나오는 단계를 따릅니다.

Amazon S3 on Outposts와 Amazon S3의 차이점

Amazon S3 on Outposts는 온프레미스 AWS Outposts 환경에 객체 스토리지를 제공합니다. S3 on Outposts를 사용하면 데이터를 온프레미스 애플리케이션에 가깝게 유지하여 로컬 처리, 데이터 상주 및 까다로운 성능 요구 사항을 충족할 수 있습니다. Amazon S3 API와 기능을 사용하므로 S3 on Outposts를 통해 손쉽게 Outposts에서 데이터를 저장, 보호, 태그 지정, 보고하고, 액세스를 제어하며, 일관된 하이브리드 환경을 위해 온프레미스 시설로 AWS 인프라를 확장할 수 있습니다.

S3 on Outposts가 어떻게 고유한지에 대해 자세히 알아보려면 다음 주제를 참조하세요.

주제

- S3 on Outposts 사양
- S3 on Outposts에서 지원하는 API 작업
- S3 on Outposts에서 지원하는 Amazon S3 AWS CLI 명령
- Amazon S3 기능은 S3 on Outposts에서 지원되지 않습니다.
- S3 on Outposts 네트워크 요구 사항

S3 on Outposts 사양

- 최대 Outposts 버킷 크기는 50TB입니다.
- AWS 계정당 최대 Outposts 버킷 수는 100개입니다.
- Outposts 버킷은 액세스 포인트 및 엔드포인트를 통해서만 액세스할 수 있습니다.
- Outposts 버킷당 최대 액세스 포인트의 수는 10개입니다.
- 액세스 포인트 정책은 크기가 20KB로 제한됩니다.
- Outpost 소유자는 AWS Resource Access Manager를 사용하여 조직 내 AWS Organizations 액세스를 관리할 수 있습니다. Outpost에 액세스해야 하는 모든 계정은 AWS Organizations의 소유자 계정과 동일한 조직 내에 있어야 합니다.
- S3 on Outposts 버킷 소유자 계정은 항상 버킷에 있는 모든 객체의 소유자입니다.
- S3 on Outposts 버킷 소유자 계정만 버킷에 대한 작업을 수행할 수 있습니다.
- 객체 크기 제한은 Amazon S3과 일치합니다.
- S3 on Outposts에 저장된 모든 객체는 OUTPOSTS 스토리지 클래스에 저장됩니다.

- 기본적으로 OUTPOSTS 스토리지 클래스에 저장된 모든 객체는 Amazon S3 관리형 암호화 키(SSE-S3)와 함께 서버 측 암호화를 사용하여 저장됩니다. 또한 고객 제공 암호화 키(SSE-C)와 함께 서버 측 암호화를 사용하여 객체를 저장하도록 명시적으로 선택할 수도 있습니다.
- Outpost에 객체를 저장할 공간이 충분하지 않으면 API는 ICE(Insufficient Capacity Exception)를 반환합니다.

S3 on Outposts에서 지원하는 API 작업

S3 on Outposts에서 지원하는 API 작업 목록은 S3 on Outposts API 작업 단원을 참조하세요.

S3 on Outposts에서 지원하는 Amazon S3 AWS CLI 명령

다음의 Amazon S3 AWS CLI 명령은 현재 Amazon S3 on Outposts에서 지원되지 않습니다. 자세한 내용은 AWS CLI 명령 참조의 Available Commands를 참조하세요.

- 동일한 Outposts 버킷 내 `cp`, `mv` 및 `sync` 또는 로컬 환경과 Outposts 버킷 사이.
- `ls`
- `presign`
- `rm`

Amazon S3 기능은 S3 on Outposts에서 지원되지 않습니다.

다음의 Amazon S3 기능은 현재 Amazon S3 on Outposts에서 지원되지 않습니다. 해당 기능을 사용하려는 모든 시도가 거부됩니다.

- 조건부 요청
- ACL(액세스 제어 목록)
- CORS(Cross-Origin Resource Sharing)
- S3 배치 작업
- S3 인벤토리 보고서
- 기본 버킷 암호화 변경
- 퍼블릭 버킷
- 멀티 팩터 인증(MFA) 삭제

- (객체 삭제 및 불완전한 멀티파트 업로드 중단과 구분) S3 수명 주기 전환
- S3 객체 잠금 법적 보존
- 객체 잠금 보존
- AWS Key Management Service(AWS KMS) 키(SSE-KMS)를 사용한 서버 측 암호화
- S3 Replication Time Control(S3 RTC)
- Amazon CloudWatch 요청 지표
- 지표 구성
- Transfer Acceleration
- S3 이벤트 알림
- 요청자 지불 버킷
- S3 Select
- AWS Lambda 이벤트
- 서버 액세스 로깅
- HTTP POST 요청
- SOAP
- 웹 사이트 액세스

S3 on Outposts 네트워크 요구 사항

- 요청을 S3 on Outposts 액세스 포인트로 라우팅하려면 S3 on Outposts 엔드포인트를 생성하고 구성해야 합니다. S3 on Outposts에 대한 엔드포인트에는 다음과 같은 제한이 적용됩니다.
  - Outpost에 있는 각 Virtual Private Cloud(VPC)에는 연결된 엔드포인트가 하나씩 있을 수 있으며, Outpost당 엔드포인트를 최대 100개 보유할 수 있습니다.
  - 여러 액세스 포인트를 동일한 엔드포인트에 매핑할 수 있습니다.
  - 엔드포인트는 다음 CIDR 범위의 하위 공간에 있는 CIDR 블록이 있는 VPC에만 추가할 수 있습니다.
    - 10.0.0.0/8
    - 172.16.0.0/12
    - 192.168.0.0/16
- Outpost에 대한 엔드포인트는 겹치지 않는 CIDR 블록이 있는 VPC에서만 생성할 수 있습니다.
- 엔드포인트는 Outposts 서브넷 내에서만 생성할 수 있습니다.

- 엔드포인트를 만드는 데 사용하는 서브넷에는 사용할 수 있는 S3 on Outposts용 IP 주소 4개가 포함되어야 합니다.
- 고객 소유 IP 주소 풀(CoIP 풀)을 지정하는 경우 사용할 수 있는 S3 on Outposts용 IP 주소 4개가 포함되어야 합니다.
- VPC별로 Outposts당 하나의 엔드포인트만 생성할 수 있습니다.

Amazon S3 on Outposts 시작하기

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다.

Amazon S3 on Outposts를 사용하면 Amazon S3에서와 같이 AWS Outposts에서 Amazon S3 API 및 기능(예: 객체 스토리지, 액세스 정책, 암호화, 태그 지정 등)을 사용할 수 있습니다. S3 on Outposts에 대한 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

주제

- AWS Management Console를 사용하여 시작하기
- AWS CLI 및 Java용 SDK를 사용하여 시작하기

AWS Management Console를 사용하여 시작하기

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

콘솔을 사용하여 S3 on Outposts를 시작하려면 다음 주제를 참조하세요. AWS CLI 또는 AWS SDK for Java를 사용하여 시작하려면 AWS CLI 및 Java용 SDK를 사용하여 시작하기 섹션을 참조하세요.

주제

- 버킷, 액세스 포인트 및 엔드포인트 생성
- 다음 단계

버킷, 액세스 포인트 및 엔드포인트 생성

다음 절차에서는 S3 on Outposts에서 첫 번째 버킷을 생성하는 방법을 보여줍니다. 콘솔을 사용하여 버킷을 생성할 때 버킷에 즉시 객체를 저장할 수 있도록 버킷과 연결된 액세스 포인트 및 엔드포인트도 생성합니다.

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. [Outposts 버킷 생성(Create Outposts bucket)]을 선택합니다.
4. 버킷 이름(Bucket name)에 버킷의 도메인 이름 시스템(DNS)을 준수하는 이름을 입력합니다.

   버킷 이름은 다음과 같아야 합니다.

   - AWS 계정, Outpost 및 Outpost의 홈 AWS 리전 내에서 고유해야 합니다.
   - 3-63자여야 합니다.
   - 대문자가 없어야 합니다.
   - 소문자 또는 숫자로 시작해야 합니다.

   버킷을 생성한 후에는 해당 이름을 변경할 수 없습니다. 버킷 이름 지정에 대한 자세한 내용은 Amazon S3 사용 설명서의 범용 버킷 이름 지정 규칙을 참조하세요.

   > **Important**
   > 버킷 이름에 계정 번호와 같은 중요한 정보를 포함하지 마세요. 버킷 이름은 버킷의 객체를 가리키는 URL에 표시됩니다.

5. Outpost에서 버킷이 상주할 Outpost를 선택합니다.
6. Bucket Versioning(버킷 버전 관리)에서 Outposts 버킷의 S3에 대한 S3 버전 관리 상태를 다음 옵션 중 하나로 설정합니다.

   - Disable(비활성화)(기본값) - 버킷이 버전 관리되지 않은 상태로 유지됩니다.
   - Enable(활성화) - 버킷 내 객체에 S3 버전 관리를 활성화합니다. 버킷에 추가된 모든 객체는 고유한 버전 ID 을 받습니다.

S3 버전 관리에 대한 자세한 내용은 S3 on Outposts 버킷의 S3 버전 관리에 대한 관리 섹션을 참조하십시오.

7. (선택 사항) Outposts 버킷과 연결할 선택적 태그를 추가합니다. 태그를 사용하여 개별 프로젝트나 프로젝트 그룹에 대해 기준을 추적하거나 비용 할당 태그를 사용하여 버킷에 레이블을 지정할 수 있습니다.

   기본적으로 Outposts 버킷에 저장된 모든 객체는 Amazon S3 관리형 암호화 키(SSE-S3)와 함께 서버 측 암호화를 사용하여 저장됩니다. 또한 고객 제공 암호화 키(SSE-C)와 함께 서버 측 암호화를 사용하여 객체를 저장하도록 명시적으로 선택할 수도 있습니다. 암호화 유형을 변경하려면 REST API, AWS Command Line Interface(AWS CLI) 또는 AWS SDK를 사용해야 합니다.

8. [Outposts 액세스 포인트 설정(Outposts access point settings)] 단원에서 액세스 포인트 이름을 입력합니다.

   S3 액세스 포인트는 S3의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 S3 객체 작업을 수행하는 데 사용할 수 있는 Outposts 버킷에 연결된 네트워크 엔드포인트입니다. 자세한 내용은 액세스 포인트 단원을 참조하십시오.

   액세스 포인트 이름은 이 리전 및 Outpost의 계정 내에서 고유해야 하며 액세스 포인트 규제 및 제한을 준수해야 합니다.

9. 이 Amazon S3 on Outposts 액세스 포인트에 대한 VPC를 선택합니다.

   VPC가 없으면 VPC 생성(Create VPC)을 선택합니다. 자세한 내용은 Amazon S3 사용 설명서의 가상 프라이빗 클라우드(VPC)로 제한된 액세스 포인트 만들기를 참조하세요.

   Virtual Private Cloud(VPC)를 사용하면 사용자가 정의한 가상 네트워크로 AWS 리소스를 시작할 수 있습니다. 이 가상 네트워크는 AWS의 확장 가능한 인프라를 사용한다는 이점과 함께 고객의 자체 데이터 센터에서 운영하는 기존 네트워크와 매우 유사합니다.

10. (기존 VPC에 대한 선택 사항) 엔드포인트에 대해 엔드포인트 서브넷(Endpoint subnet)을 선택합니다.

    서브넷은 VPC의 IP 주소 범위입니다. 원하는 서브넷이 없으면 서브넷 생성(Create subnet)을 선택합니다. 자세한 내용은 S3 on Outposts에 대한 네트워킹 단원을 참조하십시오.

11. (기존 VPC에 대한 선택 사항) 엔드포인트에 대해 엔드포인트 보안 그룹(Endpoint security group)을 선택합니다.

    보안 그룹은 인바운드 및 아웃바운드 트래픽을 제어하는 가상 방화벽 역할을 합니다.

12. (기존 VPC에 대한 선택 사항) 다음과 같은 엔드포인트 액세스 유형(Endpoint access type)을 선택합니다.
    - 프라이빗(Private) – VPC와 함께 사용할 경우 선택합니다.
    - 고객 소유 IP(Customer owned IP) – 온프레미스 네트워크 내에서 고객 소유 IP 주소 풀(CoIP)과 함께 사용할 경우 선택합니다.
13. (선택 사항) Outpost 액세스 포인트 정책(Outpost access point policy)을 지정합니다. 콘솔에는 정책에서 사용할 수 있는 액세스 포인트의 Amazon 리소스 이름(ARN)이 자동으로 표시됩니다.
14. [Outposts 버킷 생성(Create Outposts bucket)]을 선택합니다.

> **Note**
>
> Outpost 엔드포인트가 생성되고 버킷이 사용할 준비가 되려면 최대 5분 정도 걸릴 수 있습니다. 추가 버킷 설정을 구성하려면 세부 정보 보기(View details)를 선택합니다.

다음 단계

Amazon S3 on Outposts를 사용할 경우 객체 데이터는 항상 Outpost에 저장됩니다. AWS가 Outpost 랙을 설치하는 경우 데이터 상주 요구 사항을 충족하기 위해 데이터가 Outpost에 로컬로 유지됩니다. 객체는 Outpost에서 벗어나지 않으며 AWS 리전에 있지 않습니다. AWS Management Console이 리전 내에 호스팅되므로 콘솔을 사용하여 Outpost의 객체를 업로드하거나 관리할 수 없습니다. 그러나 REST API, AWS Command Line Interface(AWS CLI), AWS SDK를 사용하여 액세스 포인트를 통해 객체를 업로드하고 관리할 수 있습니다.

S3 on Outposts 버킷, 액세스 포인트 및 엔드포인트를 생성한 후에는 AWS CLI 또는 Java용 SDK를 사용하여 버킷에 객체를 업로드할 수 있습니다. 자세한 내용은 S3 on Outposts 버킷에 객체 업로드 단원을 참조하십시오.

AWS CLI 및 Java용 SDK를 사용하여 시작하기

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태

깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

S3 on Outposts를 시작하려면 버킷, 액세스 포인트 및 엔드포인트를 생성해야 합니다. 그러면 객체를 버킷에 업로드할 수 있습니다. 다음 예제에서는 AWS CLI 및 Java용 SDK를 사용하여 S3 on Outposts를 시작하는 방법을 보여줍니다. 콘솔을 사용하여 시작하려면 AWS Management Console를 사용하여 시작하기 섹션을 참조하세요.

주제

- 1단계: 버킷 만들기
- 2단계: 액세스 포인트 생성
- 3단계: 엔드포인트 생성
- 4단계: S3 on Outposts 버킷에 객체 업로드

1단계: 버킷 만들기

다음 AWS CLI 및 Java용 SDK 예제에서는 S3 on Outposts 버킷 생성 방법을 보여줍니다.

AWS CLI

Example

다음 예제에서는 AWS CLI를 사용하여 S3 on Outposts 버킷(`s3-outposts:CreateBucket`)을 생성합니다. 이 명령을 실행하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
aws s3control create-bucket --bucket example-outposts-bucket --outpost-
id op-01ac5d28a6a232904
```

SDK for Java

Example

다음 예제에서는 SDK for Java를 사용하여 S3 on Outposts 버킷(`s3-outposts:CreateBucket`)을 생성합니다.

```
import com.amazonaws.services.s3control.model.*;
```

```
public String createBucket(String bucketName) {

    CreateBucketRequest reqCreateBucket = new CreateBucketRequest()
            .withBucket(bucketName)
            .withOutpostId(OutpostId)
            .withCreateBucketConfiguration(new CreateBucketConfiguration());

    CreateBucketResult respCreateBucket =
 s3ControlClient.createBucket(reqCreateBucket);
    System.out.printf("CreateBucket Response: %s%n", respCreateBucket.toString());

    return respCreateBucket.getBucketArn();

}
```

2단계: 액세스 포인트 생성

Amazon S3 on Outposts 버킷에 액세스하려면 액세스 포인트를 생성하고 구성해야 합니다. 다음 예제는 AWS CLI 및 Java용 SDK를 사용하여 액세스 포인트를 생성하는 방법을 보여줍니다.

액세스 포인트는 Amazon S3의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 Amazon S3 객체 작업을 수행하는 데 사용할 수 있는 버킷에 연결된 네트워크 엔드포인트입니다. S3 on Outposts를 통해 Outposts 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 액세스 포인트는 가상 호스트 스타일의 주소 지정만 지원합니다.

AWS CLI

Example

다음 AWS CLI 예제에서는 Outposts 버킷에 대한 액세스 포인트를 생성합니다. 이 명령을 실행하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
aws s3control create-access-point --account-id 123456789012
 --name example-outposts-access-point --bucket "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket" --vpc-configuration VpcId=example-vpc-12345
```

SDK for Java

Example

다음 SDK for Java 예제에서는 Outposts 버킷에 대한 액세스 포인트를 생성합니다. 이 예제를 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.services.s3control.model.*;

public String createAccessPoint(String bucketArn, String accessPointName) {

    CreateAccessPointRequest reqCreateAP = new CreateAccessPointRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withName(accessPointName)
            .withVpcConfiguration(new VpcConfiguration().withVpcId("vpc-12345"));

    CreateAccessPointResult respCreateAP =
 s3ControlClient.createAccessPoint(reqCreateAP);
    System.out.printf("CreateAccessPoint Response: %s%n", respCreateAP.toString());

    return respCreateAP.getAccessPointArn();

}
```

3단계: 엔드포인트 생성

요청을 Amazon S3 on Outposts 액세스 포인트로 라우팅하려면 S3 on Outposts 엔드포인트를 생성하고 구성해야 합니다. 엔드포인트를 생성하려면 Outposts 홈 리전에 대한 서비스 링크와의 활성 연결이 필요합니다. Outpost에 있는 각 Virtual Private Cloud(VPC)에는 연결된 엔드포인트가 하나씩 있을 수 있습니다. 엔드포인트 할당량에 대한 자세한 내용은 S3 on Outposts 네트워크 요구 사항 단원을 참조하세요. Outposts 버킷에 액세스하고 객체 작업을 수행하려면 엔드포인트를 생성해야 합니다. 자세한 내용은 엔드포인트 단원을 참조하십시오.

다음 예제는 AWS CLI 및 Java용 SDK를 사용하여 앤드포인트를 생성하는 방법을 보여줍니다. 엔드포인트 생성 및 관리에 필요한 권한에 대한 자세한 내용은 S3 on Outposts 엔드포인트에 대한 권한 섹션을 참조하세요.

AWS CLI

Example

다음 AWS CLI 예제에서는 VPC 리소스 액세스 유형을 사용하여 Outposts의 엔드포인트를 생성합니다. VPC는 서브넷에서 파생되었습니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1
```

다음 AWS CLI 예제에서는 고객 소유 IP 주소 풀(CoIP 풀) 액세스 유형을 사용하여 Outpost에 대한 엔드포인트를 생성합니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1 --access-type CustomerOwnedIp --
customer-owned-ipv4-pool ipv4pool-coip-12345678901234567
```

SDK for Java

Example

다음 SDK for Java 예제에서는 Outpost에 대한 엔드포인트를 생성합니다. 이 예제를 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.CreateEndpointRequest;
import com.amazonaws.services.s3outposts.model.CreateEndpointResult;

public void createEndpoint() {
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    CreateEndpointRequest createEndpointRequest = new CreateEndpointRequest()
                .withOutpostId("op-0d79779cef3c30a40")
                .withSubnetId("subnet-8c7a57c5")
                .withSecurityGroupId("sg-ab19e0d1")
                .withAccessType("CustomerOwnedIp")
                .withCustomerOwnedIpv4Pool("ipv4pool-coip-12345678901234567");
```

```
    // Use .withAccessType and .withCustomerOwnedIpv4Pool only when the access type
 is
    // customer-owned IP address pool (CoIP pool)
    CreateEndpointResult createEndpointResult =
 s3OutpostsClient.createEndpoint(createEndpointRequest);
    System.out.println("Endpoint is created and its ARN is " +
 createEndpointResult.getEndpointArn());
}
```

4단계: S3 on Outposts 버킷에 객체 업로드

객체를 업로드하려면 S3 on Outposts 버킷에 객체 업로드 단원을 참조하세요.

S3 on Outposts에 대한 네트워킹

Amazon S3 on Outposts를 사용하여 로컬 데이터 액세스, 데이터 처리, 데이터 레지던시를 필요로 하는 애플리케이션을 위해 온프레미스에서 객체를 저장하고 검색할 수 있습니다. 이 단원에서는 S3 on Outposts에 액세스하기 위한 네트워킹 요구 사항에 대해 설명합니다.

주제

- 네트워킹 액세스 유형 선택
- S3 on Outposts 버킷과 객체에 액세스
- 교차 계정 탄력적 네트워크 인터페이스

네트워킹 액세스 유형 선택

VPC 내부 또는 온프레미스 네트워크에서 S3 on Outposts에 액세스할 수 있습니다. 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 이 연결을 사용하면 AWS 네트워크 내에서 VPC와 S3 on Outposts 버킷 간의 트래픽이 유지됩니다. 엔드포인트를 생성할 때 엔드포인트 액세스 유형을 `Private`(VPC 라우팅의 경우) 또는 `CustomerOwnedIp`(고객 소유 IP 주소 풀[CoIP 풀]의 경우) 중에서 지정해야 합니다.

- `Private`(VPC 라우팅의 경우) – 액세스 유형을 지정하지 않으면 S3 on Outposts는 기본값으로 `Private`을 사용합니다. `Private` 액세스 유형을 사용할 경우 VPC의 인스턴스는 Outpost의 리소스와 통신하는 데 퍼블릭 IP 주소를 필요로 하지 않습니다. VPC 내에서 S3 on Outposts 작업을 수행할 수 있습니다. 직접 VPC 라우팅을 통해 온프레미스 네트워크에서 이 유형의 엔드포인트로 액세스할 수 있습니다. 자세한 내용은 AWS Outposts 사용 설명서의 로컬 게이트웨이 라우팅 테이블을 참조하세요.
- `CustomerOwnedIp`(CoIP 풀의 경우) – 기본값으로 `Private` 액세스 유형이 아닌 `CustomerOwnedIp`를 선택할 경우 IP 주소 범위를 지정해야 합니다. 이 액세스 유형을 사용하여 온프레미스 네트워크에서 및 VPC 내에서 S3 on Outposts 작업을 수행할 수 있습니다. VPC 내에서 S3 on Outposts에 액세스할 때 트래픽이 로컬 게이트웨이의 대역폭으로 제한됩니다.

S3 on Outposts 버킷과 객체에 액세스

S3 on Outposts 버킷과 객체에 액세스하려면 다음 항목이 있어야 합니다.

- VPC에 대한 액세스 포인트

- 동일 VPC에 대한 엔드포인트
- Outpost와 AWS 리전 사이의 활성 연결. Outposts를 리전에 연결하는 방법에 대한 자세한 내용은 AWS Outpost 사용 설명서의 AWS 리전에 Outpost 연결을 참조하세요.

S3 on Outposts의 버킷과 객체에 액세스하는 방법에 대한 자세한 내용은 S3 on Outposts 버킷 작업 및 S3 on Outposts 객체 작업 단원을 참조하세요.

교차 계정 탄력적 네트워크 인터페이스

S3 on Outposts 엔드포인트는 Amazon 리소스 이름(ARN)을 가진 명명된 리소스입니다. 이러한 엔드포인트가 생성되면 AWS Outposts는 다수의 교차 계정 탄력적 네트워크 인터페이스를 설정합니다. S3 on Outposts 교차 계정 탄력적 네트워크 인터페이스는 다른 네트워크 인터페이스와 비슷하지만, S3 on Outposts가 교차 계정 탄력적 네트워크 인터페이스를 Amazon EC2 인스턴스에 연결한다는 한 가지 예외가 있습니다.

S3 on Outposts 도메인 이름 시스템(DNS)은 교차 계정 탄력적 네트워크 인터페이스를 통해 요청을 로드 밸런싱합니다. S3 on Outposts는 Amazon EC2 콘솔의 네트워크 인터페이스 창에서 볼 수 있는 AWS 계정에서 교차 계정 탄력적 네트워크 인터페이스를 생성합니다.

CoIP 풀 액세스 유형을 사용하는 엔드포인트의 경우, S3 on Outposts는 구성된 CoIP 풀의 교차 계정 탄력적 네트워크 인터페이스를 사용하여 IP 주소를 할당하고 연결합니다.

S3 on Outposts 버킷 작업

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 단원을 참조하세요.

Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. S3 on Outposts 버킷 및 객체에 액세스하려면 VPC에 대한 액세스 포인트와 엔드포인트가 있어야 합니다. 자세한 내용은 S3 on Outposts에 대한 네트워킹 단원을 참조하십시오.

버킷

S3 on Outposts에서 버킷 이름은 Outpost마다 고유하며, Outpost 홈 리전에 대한 AWS 리전 코드, AWS 계정 ID, Outpost ID 및 이들을 식별하기 위한 버킷 이름이 필요합니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/bucket/bucket-name
```

자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하십시오.

액세스 포인트

Amazon S3 on Outposts는 Outposts 버킷에 액세스할 수 있는 유일한 수단으로 Virtual Private Cloud(VPC) 전용 액세스 포인트를 지원합니다.

액세스 포인트는 Amazon S3의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 Amazon S3 객체 작업을 수행하는 데 사용할 수 있는 버킷에 연결된 네트워크 엔드포인트입니다. S3 on Outposts를 통해 Outposts 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 액세스 포인트는 가상 호스트 스타일의 주소 지정만 지원합니다.

다음 예제는 S3 on Outposts 액세스 포인트에 사용할 ARN 형식을 보여줍니다. 액세스 포인트 ARN에는 Outpost 홈 리전에 대한 AWS 리전 코드, AWS 계정 ID, Outpost ID 및 액세스 포인트 이름이 포함되어 있습니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

엔드포인트

요청을 S3 on Outposts 액세스 포인트로 라우팅하려면 S3 on Outposts 엔드포인트를 생성하고 구성해야 합니다. S3 on Outposts 엔드포인트를 사용하면 VPC를 Outposts 버킷에 프라이빗으로 연결할 수 있습니다. S3 on Outposts 엔드포인트는 S3 on Outposts 버킷에 대한 진입점의 가상 통합 자원 식별자(URI)입니다. 수평으로 확장된 고가용성 중복 VPC 구성 요소입니다.

Outpost에 있는 각 Virtual Private Cloud(VPC)에는 연결된 엔드포인트가 하나씩 있을 수 있으며, Outpost당 엔드포인트를 최대 100개 보유할 수 있습니다. Outpost 버킷에 액세스하고 객체 작업을 수행하려면 이러한 엔드포인트를 생성해야 합니다. 이런 엔드포인트를 생성할 때 S3와 S3 on Outposts에서 동일한 작업이 작동하여 API 모델 및 동작을 동일하게 유지할 수 있습니다.

S3 on Outposts에 대한 API 작업

Outpost 버킷 API 작업을 관리하기 위해 S3 on Outposts는 Amazon S3 엔드포인트와 다른 별도의 엔드포인트를 호스팅합니다. 이 엔드포인트는 `s3-outposts.region.amazonaws.com`입니다.

Amazon S3 API 작업을 사용하려면 버킷 및 객체에 서명할 때 올바른 ARN 형식을 사용해야 합니다. 요청이 Amazon S3(`s3-control.region.amazonaws.com`)에 대한 것인지 S3 on Outposts(`s3-outposts.region.amazonaws.com`)에 대한 것인지를 Amazon S3가 결정할 수 있도록 API 작업에 ARN을 전달해야 합니다. ARN 형식에 따라, S3가 요청에 적절하게 서명하고 요청을 라우팅할 수 있습니다.

요청이 Amazon S3 제어 영역으로 전송될 때마다 SDK는 ARN에서 구성 요소를 추출하고 ARN에서 추출한 *`outpost-id`* 값을 가진 추가 헤더 `x-amz-outpost-id`를 포함합니다. ARN의 서비스 이름은 S3 on Outposts 엔드포인트로 라우팅되기 전에 요청에 서명하는 데 사용됩니다. 이 동작은 `s3control` 클라이언트에 의해 처리된 전체 API 작업에 적용됩니다.

다음 표에는 Amazon S3 on Outposts에 대한 확장 API 작업과 Amazon S3에 대한 변경 사항이 나와 있습니다.

API	S3 on Outposts 파라미터 값
`CreateBucket`	버킷 이름을 ARN, Outposts ID로
`ListRegionalBuckets`	Outpost ID

API	S3 on Outposts 파라미터 값
DeleteBucket	버킷 이름을 ARN으로
DeleteBucketLifecycleConfiguration	버킷 이름을 ARN으로
GetBucketLifecycleConfiguration	버킷 이름을 ARN으로
PutBucketLifecycleConfiguration	버킷 이름을 ARN으로
GetBucketPolicy	버킷 이름을 ARN으로
PutBucketPolicy	버킷 이름을 ARN으로
DeleteBucketPolicy	버킷 이름을 ARN으로
GetBucketTagging	버킷 이름을 ARN으로
PutBucketTagging	버킷 이름을 ARN으로
DeleteBucketTagging	버킷 이름을 ARN으로
CreateAccessPoint	액세스 포인트 이름을 ARN으로
DeleteAccessPoint	액세스 포인트 이름을 ARN으로
GetAccessPoint	액세스 포인트 이름을 ARN으로
GetAccessPoint	액세스 포인트 이름을 ARN으로
ListAccessPoints	액세스 포인트 이름을 ARN으로
PutAccessPointPolicy	액세스 포인트 이름을 ARN으로
GetAccessPointPolicy	액세스 포인트 이름을 ARN으로
DeleteAccessPointPolicy	액세스 포인트 이름을 ARN으로

S3 on Outposts 버킷의 생성 및 관리

S3 on Outposts 버킷의 생성과 관리에 대한 자세한 내용은 다음 주제를 참조하세요.

S3 on Outposts 버킷 생성

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

> **Note**
>
> 버킷은 버킷을 생성하는 AWS 계정이 소유하며 이 계정은 버킷에 작업을 커밋할 수 있는 유일한 계정입니다. 버킷에는 Outpost, 태그, 기본 암호화, 액세스 포인트 설정 등의 구성 속성이 있습니다. 액세스 포인트 설정에는 버킷의 객체에 액세스하기 위한 Virtual Private Cloud(VPC) 및 액세스 포인트 정책 그리고 기타 메타데이터가 포함됩니다. 자세한 내용은 S3 on Outposts 사양 단원을 참조하십시오.
>
> Virtual Private Cloud(VPC)의 인터페이스 VPC 엔드포인트를 통해 버킷 및 엔드포인트 관리 액세스를 제공하기 위해 AWS PrivateLink를 사용하는 버킷을 생성하려면, S3 on Outposts에 대한 AWS PrivateLink를 참조하세요.

다음 예제에서는 AWS Management Console, AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 버킷을 생성하는 방법을 보여줍니다.

S3 콘솔 사용

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.

3. [Outposts 버킷 생성(Create Outposts bucket)]을 선택합니다.
4. 버킷 이름(Bucket name)에 버킷의 도메인 이름 시스템(DNS)을 준수하는 이름을 입력합니다.

   버킷 이름은 다음과 같아야 합니다.

   - AWS 계정, Outpost 및 Outpost의 홈 AWS 리전 내에서 고유해야 합니다.
   - 3-63자여야 합니다.
   - 대문자가 없어야 합니다.
   - 소문자 또는 숫자로 시작해야 합니다.

   버킷을 생성한 후에는 해당 이름을 변경할 수 없습니다. 버킷 이름 지정에 대한 자세한 내용은 Amazon S3 사용 설명서의 범용 버킷 이름 지정 규칙을 참조하세요.

   > **Important**
   > 버킷 이름에 계정 번호와 같은 중요한 정보를 포함하지 마세요. 버킷 이름은 버킷의 객체를 가리키는 URL에 표시됩니다.

5. Outpost에서 버킷이 상주할 Outpost를 선택합니다.
6. Bucket Versioning(버킷 버전 관리)에서 Outposts 버킷의 S3에 대한 S3 버전 관리 상태를 다음 옵션 중 하나로 설정합니다.

   - Disable(비활성화)(기본값) - 버킷이 버전 관리되지 않은 상태로 유지됩니다.
   - Enable(활성화) - 버킷 내 객체에 S3 버전 관리를 활성화합니다. 버킷에 추가된 모든 객체는 고유한 버전 ID 을 받습니다.

   S3 버전 관리에 대한 자세한 내용은 S3 on Outposts 버킷의 S3 버전 관리에 대한 관리 섹션을 참조하십시오.

7. (선택 사항) Outposts 버킷과 연결할 선택적 태그를 추가합니다. 태그를 사용하여 개별 프로젝트나 프로젝트 그룹에 대해 기준을 추적하거나 비용 할당 태그를 사용하여 버킷에 레이블을 지정할 수 있습니다.

   기본적으로 Outposts 버킷에 저장된 모든 객체는 Amazon S3 관리형 암호화 키(SSE-S3)와 함께 서버 측 암호화를 사용하여 저장됩니다. 또한 고객 제공 암호화 키(SSE-C)와 함께 서버 측 암호화를 사용하여 객체를 저장하도록 명시적으로 선택할 수도 있습니다. 암호화 유형을 변경하려면 REST API, AWS Command Line Interface(AWS CLI) 또는 AWS SDK를 사용해야 합니다.

8. [Outposts 액세스 포인트 설정(Outposts access point settings)] 단원에서 액세스 포인트 이름을 입력합니다.

   S3 액세스 포인트는 S3의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 S3 객체 작업을 수행하는 데 사용할 수 있는 Outposts 버킷에 연결된 네트워크 엔드포인트입니다. 자세한 내용은 액세스 포인트 단원을 참조하십시오.

   액세스 포인트 이름은 이 리전 및 Outpost의 계정 내에서 고유해야 하며 액세스 포인트 규제 및 제한을 준수해야 합니다.

9. 이 Amazon S3 on Outposts 액세스 포인트에 대한 VPC를 선택합니다.

   VPC가 없으면 VPC 생성(Create VPC)을 선택합니다. 자세한 내용은 Amazon S3 사용 설명서의 가상 프라이빗 클라우드(VPC)로 제한된 액세스 포인트 만들기를 참조하세요.

   Virtual Private Cloud(VPC)를 사용하면 사용자가 정의한 가상 네트워크로 AWS 리소스를 시작할 수 있습니다. 이 가상 네트워크는 AWS의 확장 가능한 인프라를 사용한다는 이점과 함께 고객의 자체 데이터 센터에서 운영하는 기존 네트워크와 매우 유사합니다.

10. (기존 VPC에 대한 선택 사항) 엔드포인트에 대해 엔드포인트 서브넷(Endpoint subnet)을 선택합니다.

    서브넷은 VPC의 IP 주소 범위입니다. 원하는 서브넷이 없으면 서브넷 생성(Create subnet)을 선택합니다. 자세한 내용은 S3 on Outposts에 대한 네트워킹 단원을 참조하십시오.

11. (기존 VPC에 대한 선택 사항) 엔드포인트에 대해 엔드포인트 보안 그룹(Endpoint security group)을 선택합니다.

    보안 그룹은 인바운드 및 아웃바운드 트래픽을 제어하는 가상 방화벽 역할을 합니다.

12. (기존 VPC에 대한 선택 사항) 다음과 같은 엔드포인트 액세스 유형(Endpoint access type)을 선택합니다.

    - 프라이빗(Private) – VPC와 함께 사용할 경우 선택합니다.
    - 고객 소유 IP(Customer owned IP) – 온프레미스 네트워크 내에서 고객 소유 IP 주소 풀(CoIP)과 함께 사용할 경우 선택합니다.

13. (선택 사항) Outpost 액세스 포인트 정책(Outpost access point policy)을 지정합니다. 콘솔에는 정책에서 사용할 수 있는 액세스 포인트의 Amazon 리소스 이름(ARN)이 자동으로 표시됩니다.

14. [Outposts 버킷 생성(Create Outposts bucket)]을 선택합니다.

> **Note**
>
> Outpost 엔드포인트가 생성되고 버킷이 사용할 준비가 되려면 최대 5분 정도 걸릴 수 있습니다. 추가 버킷 설정을 구성하려면 세부 정보 보기(View details)를 선택합니다.

AWS CLI 사용

Example

다음 예제에서는 AWS CLI를 사용하여 S3 on Outposts 버킷(`s3-outposts:CreateBucket`)을 생성합니다. 이 명령을 실행하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
aws s3control create-bucket --bucket example-outposts-bucket --outpost-
id op-01ac5d28a6a232904
```

Java용 AWS SDK 사용

Example

다음 예제에서는 SDK for Java를 사용하여 S3 on Outposts 버킷(`s3-outposts:CreateBucket`)을 생성합니다.

```
import com.amazonaws.services.s3control.model.*;

public String createBucket(String bucketName) {

    CreateBucketRequest reqCreateBucket = new CreateBucketRequest()
            .withBucket(bucketName)
            .withOutpostId(OutpostId)
            .withCreateBucketConfiguration(new CreateBucketConfiguration());

    CreateBucketResult respCreateBucket =
 s3ControlClient.createBucket(reqCreateBucket);
    System.out.printf("CreateBucket Response: %s%n", respCreateBucket.toString());

    return respCreateBucket.getBucketArn();

}
```

S3 on Outposts 버킷에 대한 태그 추가

Amazon S3 on Outposts 버킷에 대한 태그를 추가하여 스토리지 비용을 추적하거나 개별 프로젝트 또는 프로젝트 그룹에 대한 기타 기준을 추적할 수 있습니다.

> **Note**
> 버킷은 버킷을 생성하는 AWS 계정이 소유하며 이 계정은 태그를 변경할 수 있는 유일한 계정입니다.

S3 콘솔 사용

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 편집할 태그가 있는 Outposts 버킷을 선택합니다.
4. 속성(Properties) 탭을 선택합니다.
5. 태그에서 편집을 선택합니다.
6. 새 태그 추가(Add new tag)를 선택하고 키(Key) 및 값(Value)(선택 사항)을 입력합니다.

   개별 프로젝트 또는 프로젝트 그룹에 대한 기타 기준을 추적하려면 Outposts 버킷에 연결할 태그를 추가합니다.
7. Save changes(변경 사항 저장)를 선택합니다.

AWS CLI 사용

다음 AWS CLI 예제에서는 현재 폴더에서 태그(*tagging.json*)를 지정하는 JSON 문서를 사용하여 S3 on Outposts 버킷에 태그 지정 구성을 적용합니다. 이 예제를 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체합니다.

```
aws s3control put-bucket-tagging --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --tagging file://tagging.json

tagging.json
```

```
{
   "TagSet": [
     {
       "Key": "organization",
       "Value": "marketing"
     }
   ]
}
```

다음 AWS CLI 예제에서는 명령줄에서 직접 S3 on Outposts 버킷에 태깅 구성을 적용합니다.

```
aws s3control put-bucket-tagging --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --tagging 'TagSet=[{Key=organization,Value=marketing}]'
```

이 명령에 대한 자세한 내용은 AWS CLI 참조의 put-bucket-tagging을 참조하세요.

버킷 정책을 사용하여 Amazon S3 on Outposts 버킷에 대한 액세스 관리

버킷 정책은 버킷과 해당 버킷의 객체에 액세스 권한을 부여할 수 있는 리소스 기반 AWS Identity and Access Management(IAM) 정책입니다. 버킷 소유자만 정책을 버킷에 연결할 수 있습니다. 버킷에 연결된 권한은 버킷 소유자가 소유한 모든 버킷의 객체에 적용됩니다. 버킷 정책은 크기가 20KB로 제한됩니다. 자세한 내용은 버킷 정책 단원을 참조하십시오.

버킷 정책을 업데이트하여 Amazon S3 on Outposts 버킷에 대한 액세스를 관리할 수 있습니다. 자세한 내용은 다음 항목을 참조하십시오.

주제

- Amazon S3 on Outposts 버킷의 버킷 정책 추가 또는 편집
- Amazon S3 on Outposts 버킷의 버킷 정책 보기
- Amazon S3 on Outposts 버킷의 버킷 정책 삭제
- 버킷 정책 예제

Amazon S3 on Outposts 버킷의 버킷 정책 추가 또는 편집

버킷 정책은 버킷과 해당 버킷의 객체에 액세스 권한을 부여할 수 있는 리소스 기반 AWS Identity and Access Management(IAM) 정책입니다. 버킷 소유자만 정책을 버킷에 연결할 수 있습니다. 버킷에 연

결된 권한은 버킷 소유자가 소유한 모든 버킷의 객체에 적용됩니다. 버킷 정책은 크기가 20KB로 제한됩니다. 자세한 내용은 버킷 정책 단원을 참조하십시오.

다음 주제에서는 AWS Management Console, AWS Command Line Interface(AWS CLI) 또는 AWS SDK for Java를 사용하여 Amazon S3 on Outposts 버킷 정책을 업데이트하는 방법을 보여줍니다.

S3 콘솔 사용

버킷 정책 생성 또는 편집

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 편집할 버킷 정책이 있는 Outposts 버킷을 선택합니다.
4. 권한 탭을 선택합니다.
5. 새 정책을 생성하거나 정책을 편집하려면 Outposts 버킷 정책(Outposts bucket policy) 섹션에서 편집(Edit)을 선택합니다.

   이제 S3 on Outposts 버킷 정책을 추가하거나 편집할 수 있습니다. 자세한 내용은 S3 on Outposts로 IAM 설정 단원을 참조하십시오.

AWS CLI 사용

다음 AWS CLI 예제에서는 Outposts 버킷에 정책을 적용합니다.

1. 다음 버킷 정책을 JSON 파일로 저장합니다. 이 예시에서 파일의 이름은 `policy1.json`으로 지정됩니다. *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
{
   "Version":"2012-10-17",
   "Id":"testBucketPolicy",
   "Statement":[
      {
         "Sid":"st1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
```

```
        "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket"
      }
   ]
}
```

2. `put-bucket-policy` CLI 명령의 일부로 JSON 파일을 제출합니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3control put-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --policy file://policy1.json
```

Java용 AWS SDK 사용

다음 SDK for Java 예제에서는 Outposts 버킷에 정책을 적용합니다.

```
import com.amazonaws.services.s3control.model.*;

public void putBucketPolicy(String bucketArn) {

    String policy = "{\"Version\":\"2012-10-17\",\"Id\":\"testBucketPolicy\",
\"Statement\":[{\"Sid\":\"st1\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"" +
 AccountId+ "\"},\"Action\":\"s3-outposts:*\",\"Resource\":\"" + bucketArn + "\"}]}";

    PutBucketPolicyRequest reqPutBucketPolicy = new PutBucketPolicyRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withPolicy(policy);

    PutBucketPolicyResult respPutBucketPolicy =
 s3ControlClient.putBucketPolicy(reqPutBucketPolicy);
    System.out.printf("PutBucketPolicy Response: %s%n",
 respPutBucketPolicy.toString());

}
```

Amazon S3 on Outposts 버킷의 버킷 정책 보기

버킷 정책은 버킷과 해당 버킷의 객체에 액세스 권한을 부여할 수 있는 리소스 기반 AWS Identity and Access Management(IAM) 정책입니다. 버킷 소유자만 정책을 버킷에 연결할 수 있습니다. 버킷에 연결된 권한은 버킷 소유자가 소유한 모든 버킷의 객체에 적용됩니다. 버킷 정책은 크기가 20KB로 제한됩니다. 자세한 내용은 버킷 정책 단원을 참조하십시오.

다음 주제에서는 AWS Management Console, AWS Command Line Interface(AWS CLI) 또는 AWS SDK for Java를 사용하여 Amazon S3 on Outposts 버킷 정책을 확인하는 방법을 보여줍니다.

S3 콘솔 사용

버킷 정책 생성 또는 편집

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 편집할 권한이 있는 Outposts 버킷을 선택합니다.
4. [Permissions] 탭을 선택합니다.
5. Outposts 버킷 정책(Outposts bucket policy) 섹션에서 기존 버킷 정책을 검토할 수 있습니다. 자세한 내용은 S3 on Outposts로 IAM 설정 단원을 참조하십시오.

AWS CLI 사용

다음 AWS CLI 예제에서는 Outposts 버킷에 대한 정책을 가져옵니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3control get-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket
```

Java용 AWS SDK 사용

다음 SDK for Java 예제에서는 Outposts 버킷에 대한 정책을 가져옵니다.

```
import com.amazonaws.services.s3control.model.*;

public void getBucketPolicy(String bucketArn) {
```

```
    GetBucketPolicyRequest reqGetBucketPolicy = new GetBucketPolicyRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    GetBucketPolicyResult respGetBucketPolicy =
 s3ControlClient.getBucketPolicy(reqGetBucketPolicy);
    System.out.printf("GetBucketPolicy Response: %s%n",
 respGetBucketPolicy.toString());

}
```

Amazon S3 on Outposts 버킷의 버킷 정책 삭제

버킷 정책은 버킷과 해당 버킷의 객체에 액세스 권한을 부여할 수 있는 리소스 기반 AWS Identity and Access Management(IAM) 정책입니다. 버킷 소유자만 정책을 버킷에 연결할 수 있습니다. 버킷에 연결된 권한은 버킷 소유자가 소유한 모든 버킷의 객체에 적용됩니다. 버킷 정책은 크기가 20KB로 제한됩니다. 자세한 내용은 버킷 정책 단원을 참조하십시오.

다음 주제에서는 AWS Management Console 또는 AWS Command Line Interface(AWS CLI)를 사용하여 Amazon S3 on Outposts 버킷 정책을 확인하는 방법을 보여줍니다.

S3 콘솔 사용

버킷 정책 삭제

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 편집할 권한이 있는 Outposts 버킷을 선택합니다.
4. [Permissions] 탭을 선택합니다.
5. Outposts 버킷 정책 단원에서 삭제(Delete)를 선택합니다.
6. 삭제를 확인합니다.

AWS CLI 사용

다음 예제에서는 AWS CLI를 사용하여 S3 on Outposts 버킷(`s3-outposts:DeleteBucket`)의 버킷 정책을 삭제합니다. 이 명령을 실행하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
aws s3control delete-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket
```

버킷 정책 예제

S3 on Outposts 버킷 정책을 사용하면 적절한 권한을 가진 사용자만 객체에 액세스할 수 있도록 하여 S3 on Outposts 버킷의 객체 액세스를 보호할 수 있습니다. 인증받은 사용자라도 적절한 권한이 없다면 S3 on Outposts 리소스에 액세스하지 못하게 할 수도 있습니다.

이 섹션에서는 S3 on Outposts 버킷 정책의 일반적인 사용 사례에 대한 예제를 제시합니다. 이러한 정책을 테스트하려면 *user input placeholders*를 (귀하의 버킷 이름 등) 자체 정보로 대체합니다.

객체 집합으로의 권한을 부여 또는 거부하려면 Amazon 리소스 이름(ARN) 및 기타 값에 와일드카드 문자(*)를 사용하면 됩니다. 예를 들어, 공통 접두사로 시작하거나 `.html`과 같은 지정된 확장자로 끝나는 객체 그룹에 대한 액세스를 제어할 수 있습니다.

AWS Identity and Access Management(IAM) 정책 언어에 대한 자세한 내용은 S3 on Outposts로 IAM 설정를 참조하십시오.

> **Note**
>
> Amazon S3 콘솔을 사용하여 s3outposts 권한을 테스트할 경우 콘솔이 요구하는 추가 권한, 즉 `s3outposts:createendpoint` 및 `s3outposts:listendpoints` 등을 부여해야 합니다.

버킷 정책 생성을 위한 추가 리소스

- S3 on Outposts 버킷 정책을 만들 때 사용할 수 있는 IAM 정책 작업, 리소스 및 조건 키 목록은 Actions, resources, and condition keys for Amazon S3 on Outposts를 참조하세요.
- S3 on Outposts 정책 생성에 대한 지침은 Amazon S3 on Outposts 버킷의 버킷 정책 추가 또는 편집 섹션을 참조하세요.

주제

- 특정 IP 주소를 기반으로 Amazon S3 on Outposts 버킷에 대한 액세스 관리

특정 IP 주소를 기반으로 Amazon S3 on Outposts 버킷에 대한 액세스 관리

버킷 정책은 버킷과 해당 버킷의 객체에 액세스 권한을 부여할 수 있는 리소스 기반 AWS Identity and Access Management(IAM) 정책입니다. 버킷 소유자만 정책을 버킷에 연결할 수 있습니다. 버킷에 연결된 권한은 버킷 소유자가 소유한 모든 버킷의 객체에 적용됩니다. 버킷 정책은 크기가 20KB로 제한됩니다. 자세한 내용은 버킷 정책 섹션을 참조하세요.

액세스를 특정 IP 주소로 제한

다음 예제에서는 지정된 IP 주소 범위에서 요청된 것이 아닌 한, 어떤 사용자도 지정된 버킷 내의 객체에 S3 on Outposts 작업을 수행하지 못하도록 거부합니다.

> **Note**
>
> 특정 IP 주소에 대한 액세스를 제한할 때는 S3 on Outposts 버킷에 액세스할 수 있는 VPC 엔드포인트, VPC 소스 IP 주소 또는 외부 IP 주소도 지정해야 합니다. 그렇지 않으면 적절한 권한이 이미 갖춰지지 않은 상태에서 모든 사용자가 S3 on Outposts 버킷의 객체에 대해 s3outposts 작업을 수행하는 것을 정책에서 거부하는 경우 버킷에 대한 액세스 권한을 상실하게 될 수 있습니다.

이 정책의 `Condition` 문은 *`192.0.2.0/24`* 주소가 IPv4(IP 버전 4) IP 주소 허용 범위에 속하는지 식별합니다.

`Condition` 블록은 AWS 전체 조건 키인 `NotIpAddress` 및 `aws:SourceIp` 조건 키를 사용합니다. `aws:SourceIp` 조건 키는 퍼블릭 IP 주소 범위에만 사용할 수 있습니다. 이러한 조건 키에 대한 자세한 내용은 Actions, resources, and condition keys for S3 on Outposts를 참조하세요. `aws:SourceIp` IPv4 값은 표준 CIDR 표기법을 사용합니다. 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

> **Warning**
>
> 이 S3 on Outposts 정책을 사용하기 전에 이 예제의 *`192.0.2.0/24`* IP 주소 범위를 사용 사례에 적합한 값으로 바꿉니다. 그렇지 않으면 버킷에 액세스할 수 없습니다.

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

```
    "Id": "S3OutpostsPolicyId1",
    "Statement": [
        {
            "Sid": "IPAllow",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3-outposts:*",
            "Resource": [
                "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
accesspoint/EXAMPLE-ACCESS-POINT-NAME",
                "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/amzn-s3-demo-bucket"
            ],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": "192.0.2.0/24"
                }
            }
        }
    ]
}
```

IPv4 및 IPv6 주소 모두 허용

IPv6 주소를 사용하고자 할 때는 기존 IPv4 범위 외에도 IPv6 주소 범위를 포함하도록 조직의 모든 정책을 업데이트하는 것이 좋습니다. 이렇게 하면 IPv6로 전환하는 중에도 정책이 계속 기능합니다.

다음 S3 on Outposts 버킷 정책 예제에서는 IPv4 및 IPv6 주소 범위를 혼합하여 조직의 유효 IP 주소를 모두 표현하는 방법을 보여 줍니다. 이 정책 예에서는 IP 주소 예제 *192.0.2.1* 및 *2001:DB8:1234:5678::1*에 대한 액세스를 허용하며, 주소 *203.0.113.1* 및 *2001:DB8:1234:5678:ABCD::1*에 대한 액세스는 거부합니다.

`aws:SourceIp` 조건 키는 퍼블릭 IP 주소 범위에만 사용할 수 있습니다. `aws:SourceIp`에 대한 IPv6 값은 표준 CIDR 형식이어야 합니다. IPv6의 경우 0의 범위를 나타내기 위해 `::`을 사용할 수 있습니다 (예: `2001:DB8:1234:5678::/64`). 자세한 내용은 IAM 사용 설명서의 IP 주소 조건 연산자를 참조하세요.

> **Warning**
>
> 이 S3 on Outposts 정책을 사용하기 전에 이 예제의 IP 주소 범위를 사용 사례에 적합한 값으로 바꾸세요. 그렇지 않으면 버킷에 액세스할 수 없습니다.

```
{
    "Id": "S3OutpostsPolicyId2",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIPmix",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3outposts:*",
            "Resource": [
                "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/amzn-s3-demo-bucket",
                        "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-
ID/bucket/amzn-s3-demo-bucket/*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                },
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678:ABCD::/80"
                    ]
                }
            }
        }
    ]
}
```

Amazon S3 on Outposts 버킷 나열

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태

깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

S3 on Outposts에서의 버킷 작업에 대한 자세한 내용은 S3 on Outposts 버킷 작업 단원을 참조하세요.

다음 예제는 AWS Management Console, AWS CLI, AWS SDK for Java를 사용하여 S3 on Outposts 버킷 목록을 반환하는 방법을 보여줍니다.

S3 콘솔 사용

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. Outposts 버킷(Outposts buckets)에서 S3 on Outposts 버킷 목록을 검토합니다.

AWS CLI 사용

다음 AWS CLI 예제에서는 Outpost에 있는 버킷 목록을 가져옵니다. 이 명령을 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 list-regional-buckets를 참조하세요.

```
aws s3control list-regional-buckets --account-id 123456789012 --outpost-
id op-01ac5d28a6a232904
```

Java용 AWS SDK 사용

다음 SDK for Java 예제에서는 Outpost의 버킷 목록을 가져옵니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 ListRegionalBuckets를 참조하세요.

```
import com.amazonaws.services.s3control.model.*;

public void listRegionalBuckets() {

    ListRegionalBucketsRequest reqListBuckets = new ListRegionalBucketsRequest()
            .withAccountId(AccountId)
            .withOutpostId(OutpostId);

    ListRegionalBucketsResult respListBuckets =
 s3ControlClient.listRegionalBuckets(reqListBuckets);
```

```
    System.out.printf("ListRegionalBuckets Response: %s%n",
 respListBuckets.toString());

}
```

AWS CLI 및 Java용 SDK를 사용하여 S3 on Outposts 버킷 가져오기

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

다음 예제에서는 AWS CLI 및 AWS SDK for Java를 사용하여 S3 on Outposts 버킷을 가져오는 방법을 보여줍니다.

> **Note**
>
> AWS CLI 또는 AWS SDK를 통해 Amazon S3 on Outposts에서 작업할 때 버킷 이름 대신 Outpost의 액세스 포인트 ARN을 제공합니다. 액세스 포인트 ARN은 다음과 같은 형식을 취하는데, 여기서 *region*은 Outpost가 위치한 리전의 AWS 리전 코드입니다.
>
> `arn:aws:s3-outposts:`*`region`*`:`*`123456789012`*`:outpost/`*`op-01ac5d28a6a232904`*`/accesspoint/`*`example-outposts-access-point`*
>
> S3 on Outposts ARN에 대한 자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하세요.

AWS CLI 사용

다음 S3 on Outposts 예제에서는 AWS CLI를 사용하여 버킷을 가져옵니다. 이 명령을 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 get-bucket을 참조하세요.

```
aws s3control get-bucket --account-id 123456789012 --bucket "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket"
```

Java용 AWS SDK 사용

다음 S3 on Outposts 예제에서는 Java용 SDK를 사용하여 버킷을 가져옵니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 GetBucket을 참조하세요.

```
import com.amazonaws.services.s3control.model.*;

public void getBucket(String bucketArn) {

    GetBucketRequest reqGetBucket = new GetBucketRequest()
            .withBucket(bucketArn)
            .withAccountId(AccountId);

    GetBucketResult respGetBucket = s3ControlClient.getBucket(reqGetBucket);
    System.out.printf("GetBucket Response: %s%n", respGetBucket.toString());

}
```

Amazon S3 on Outposts 버킷 삭제

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

S3 on Outposts에서의 버킷 작업에 대한 자세한 내용은 S3 on Outposts 버킷 작업 단원을 참조하세요.

버킷은 버킷을 생성하는 AWS 계정이 소유하며 이 계정은 버킷을 삭제할 수 있는 유일한 계정입니다.

> **Note**
>
> - Outposts 버킷을 삭제하려면 버킷이 비어 있어야 합니다.
>
> Amazon S3 콘솔은 S3 on Outposts 객체 작업을 지원하지 않습니다. S3 on Outposts 버킷의 객체를 삭제하려면 REST API, AWS CLI 또는 AWS SDK를 사용해야 합니다.
> - Outposts 버킷을 삭제하려면 먼저 버킷에 대한 Outposts 액세스 포인트를 삭제해야 합니다. 자세한 내용은 액세스 포인트 삭제 단원을 참조하십시오.
> - 삭제한 버킷은 복구할 수 없습니다.

다음 예제에서는 AWS Management Console 및 AWS Command Line Interface(AWS CLI)를 사용하여 S3 on Outposts 버킷을 삭제하는 방법을 보여줍니다.

S3 콘솔 사용

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 삭제하려는 버킷을 선택하고 삭제를 선택합니다.
4. 삭제를 확인합니다.

AWS CLI 사용

다음 예제에서는 AWS CLI를 사용하여 S3 on Outposts 버킷(`s3-outposts:DeleteBucket`)을 삭제합니다. 이 명령을 실행하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
aws s3control delete-bucket --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket
```

Amazon S3 on Outposts 액세스 포인트 작업

Amazon S3 on Outposts 버킷에 액세스하려면 액세스 포인트를 생성하고 구성해야 합니다.

액세스 포인트는 Amazon S3의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 Amazon S3 객체 작업을 수행하는 데 사용할 수 있

는 버킷에 연결된 네트워크 엔드포인트입니다. S3 on Outposts를 통해 Outposts 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 액세스 포인트는 가상 호스트 스타일의 주소 지정만 지원합니다.

> **Note**
>
> Outposts 버킷은 Outposts 버킷을 생성하는 AWS 계정이 소유하며 이 계정은 버킷에 액세스 포인트를 할당할 수 있는 유일한 계정입니다.

다음 단원에서는 S3 on Outposts 버킷에 대한 액세스 포인트를 생성하고 관리하는 방법에 대해 설명합니다.

주제

- S3 on Outposts 액세스 포인트 생성
- S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용
- 액세스 포인트 구성에 대한 정보 보기
- Amazon S3 on Outposts 액세스 포인트 목록 보기
- 액세스 포인트 삭제
- 액세스 포인트 정책 추가 또는 편집
- S3 on Outposts 액세스 포인트에 대한 액세스 포인트 정책 보기

S3 on Outposts 액세스 포인트 생성

Amazon S3 on Outposts 버킷에 액세스하려면 액세스 포인트를 생성하고 구성해야 합니다.

액세스 포인트는 Amazon S3의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 Amazon S3 객체 작업을 수행하는 데 사용할 수 있는 버킷에 연결된 네트워크 엔드포인트입니다. S3 on Outposts를 통해 Outposts 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 액세스 포인트는 가상 호스트 스타일의 주소 지정만 지원합니다.

다음 예제에서는 AWS Management Console, AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 액세스 포인트를 생성하는 방법을 보여줍니다.

> **Note**
>
> Outposts 버킷은 Outposts 버킷을 생성하는 AWS 계정이 소유하며 이 계정은 버킷에 액세스 포인트를 할당할 수 있는 유일한 계정입니다.

S3 콘솔 사용

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 액세스 포인트를 생성하려는 Outposts 버킷을 선택합니다.
4. [Outposts 액세스 포인트(Outposts access points)] 탭을 선택합니다.
5. Outposts 액세스 포인트 단원에서 Outposts 액세스 포인트 생성을 선택합니다.
6. Outposts 액세스 포인트 설정 단원에서 액세스 포인트의 이름을 입력하고 액세스 포인트의 Virtual Private Cloud(VPC)를 선택합니다.
7. 액세스 포인트에 대한 정책을 추가하려면 Outposts 액세스 포인트 정책 단원에 해당 정책을 입력합니다.

   자세한 내용은 S3 on Outposts로 IAM 설정 단원을 참조하십시오.

AWS CLI 사용

Example

다음 AWS CLI 예제에서는 Outposts 버킷에 대한 액세스 포인트를 생성합니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3control create-access-point --account-id 123456789012
 --name example-outposts-access-point --bucket "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket" --vpc-configuration VpcId=example-vpc-12345
```

Java용 AWS SDK 사용

Example

다음 SDK for Java 예제에서는 Outposts 버킷에 대한 액세스 포인트를 생성합니다. 이 예제를 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.services.s3control.model.*;

public String createAccessPoint(String bucketArn, String accessPointName) {

    CreateAccessPointRequest reqCreateAP = new CreateAccessPointRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withName(accessPointName)
            .withVpcConfiguration(new VpcConfiguration().withVpcId("vpc-12345"));

    CreateAccessPointResult respCreateAP =
 s3ControlClient.createAccessPoint(reqCreateAP);
    System.out.printf("CreateAccessPoint Response: %s%n", respCreateAP.toString());

    return respCreateAP.getAccessPointArn();

}
```

S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용

S3 on Outposts를 통해 Outposts 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 버킷에 대한 액세스 포인트를 생성할 때마다 S3 on Outposts는 자동으로 액세스 포인트 별칭을 생성합니다. 모든 데이터 플레인 작업에 액세스 포인트 ARN 대신 이 액세스 포인트 별칭을 사용할 수 있습니다. 예를 들어 액세스 포인트 별칭을 사용하여 PUT, GET, LIST 등과 같은 객체 수준 작업을 수행할 수 있습니다. 해당 작업의 목록은 객체 관리를 위한 Amazon S3 API 작업을(를) 참조하십시오.

다음 예제는 이름이 *my-access-point*인 액세스 포인트에 대한 ARN 및 액세스 포인트 별칭을 보여줍니다.

- 액세스 포인트 ARN - `arn:aws:s3-outposts:`*`region`*`:`*`123456789012`*`:outpost/`*`op-01ac5d28a6a232904`*`/accesspoint/`*`my-access-point`*
- 액세스 포인트 별칭 - *`my-access-po-o01ac5d28a6a232904e8xz5w8ijx1qzlbp3i3kuse10`*`--op-s3`

ARN에 대한 자세한 내용은 AWS 일반 참조의 Amazon 리소스 이름(ARN)을 참조하세요.

액세스 포인트 별칭에 대한 자세한 내용은 다음 항목을 참조하세요.

주제

- 액세스 포인트 별칭
- S3 on Outposts 객체 작업에서 액세스 포인트 별칭 사용
- 제한 사항

액세스 포인트 별칭

액세스 포인트 별칭은 S3 on Outposts 버킷과 동일한 네임스페이스 내에 생성됩니다. 액세스 포인트를 생성하면 S3 on Outposts에서 변경할 수 없는 액세스 포인트 별칭을 자동으로 생성합니다. 액세스 포인트 별칭은 유효한 S3 on Outposts 버킷 이름의 모든 요구 사항을 충족하며 다음 부분으로 구성됩니다.

```
access point name prefix-metadata--op-s3
```

> **Note**
>
> `--op-s3` 접미사는 액세스 포인트 별칭용으로 예약되어 있으므로 버킷 또는 액세스 포인트 이름에는 사용하지 않는 것이 좋습니다. S3 on Outposts 버킷 이름 지정 규칙에 대한 자세한 내용은 S3 on Outposts 버킷 작업 섹션을 참조하세요.

액세스 포인트 별칭 찾기

다음 예제에서는 Amazon S3 콘솔 및 AWS CLI를 사용하여 액세스 포인트 별칭을 찾는 방법을 보여줍니다.

Example : Amazon S3 콘솔에서 액세스 포인트 별칭 찾기 및 복사

콘솔에서 액세스 포인트를 생성한 후 Access Points(액세스 포인트) 목록의 Access Point alias(액세스 포인트 별칭) 열에서 액세스 포인트 별칭을 가져올 수 있습니다.

액세스 포인트 별칭을 복사하려면 다음 중 하나를 수행합니다.

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 액세스 포인트를 선택합니다.
3. 액세스 포인트 별칭을 복사하려면 다음 중 하나를 수행합니다.
   - Access Points(액세스 포인트) 목록에서 액세스 포인트 이름 옆에 있는 옵션 버튼을 선택한 다음 Copy Access Point alias(액세스 포인트 별칭 복사)를 선택합니다.

- 액세스 포인트 이름을 선택합니다. 그런 다음 Outposts access point overview(Outposts 액세스 지점 개요)에서 액세스 포인트 별칭을 복사합니다.

Example : AWS CLI를 사용하여 액세스 포인트를 생성하고 응답에서 액세스 포인트 별칭 찾기

`create-access-point` 명령에 대한 다음 AWS CLI 예제는 액세스 포인트를 생성하고 자동으로 생성된 액세스 포인트 별칭을 반환합니다. 이 명령을 실행하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
aws s3control create-access-point --bucket example-outposts-bucket --name example-
outposts-access-point --account-id 123456789012

{
    "AccessPointArn":
    "arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/
accesspoint/example-outposts-access-point",
    "Alias": "example-outp-o01ac5d28a6a232904e8xz5w8ijx1qzlbp3i3kuse10--op-s3"
}
```

Example : AWS CLI를 사용하여 액세스 포인트 별칭 가져오기

AWS CLI 명령에 대한 다음 `get-access-point` 예제는 지정된 액세스 포인트에 대한 정보를 반환합니다. 이 정보에는 액세스 포인트 별칭이 포함됩니다. 이 명령을 실행하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
aws s3control get-access-point --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --name example-outposts-access-point --account-id 123456789012

{
    "Name": "example-outposts-access-point",
    "Bucket": "example-outposts-bucket",
    "NetworkOrigin": "Vpc",
    "VpcConfiguration": {
        "VpcId": "vpc-01234567890abcdef"
    },
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": true,
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true
```

```
    },
    "CreationDate": "2022-09-18T17:49:15.584000+00:00",
    "Alias": "example-outp-o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3"
}
```

Example : AWS CLI를 사용하여 액세스 포인트 별칭을 찾기 위한 액세스 포인트 나열

`list-access-points` 명령에 대한 다음 AWS CLI 예제는 지정된 액세스 포인트에 대한 정보를 나열합니다. 이 정보에는 액세스 포인트 별칭이 포함됩니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3control list-access-points --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket

{
    "AccessPointList": [
        {
            "Name": "example-outposts-access-point",
            "NetworkOrigin": "Vpc",
            "VpcConfiguration": {
                "VpcId": "vpc-01234567890abcdef"
            },
            "Bucket": "example-outposts-bucket",
            "AccessPointArn": "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point",
            "Alias": "example-outp-o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3"
        }
    ]
}
```

S3 on Outposts 객체 작업에서 액세스 포인트 별칭 사용

액세스 포인트를 채택할 때 광범위한 코드 변경을 하지 않고 액세스 포인트 별칭을 사용할 수 있습니다.

이 AWS CLI 예제에서는 S3 on Outposts 버킷의 `get-object` 작업을 보여줍니다. 이 예제에서는 전체 액세스 포인트 ARN 대신 액세스 포인트 별칭을 `--bucket`의 값으로 사용합니다.

```
aws s3api get-object --bucket my-access-po-
o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3 --key  testkey  sample-object.rtf
```

```
{
    "AcceptRanges": "bytes",
    "LastModified": "2020-01-08T22:16:28+00:00",
    "ContentLength": 910,
    "ETag": "\"00751974dc146b76404bb7290f8f51bb\"",
    "VersionId": "null",
    "ContentType": "text/rtf",
    "Metadata": {}
}
```

제한 사항

- 별칭은 고객이 구성할 수 없습니다.
- 액세스 포인트에서 별칭을 삭제, 수정 또는 비활성화할 수 없습니다.
- Amazon S3 on Outposts 컨트롤 플레인 작업에는 액세스 포인트 별칭을 사용할 수 없습니다. S3 on Outposts 컨트롤 플레인 작업의 목록은 버킷 관리를 위한 Amazon S3 Control API 작업 섹션을 참조하세요
- 별칭은 AWS Identity and Access Management(IAM) 정책에서 사용할 수 없습니다.

액세스 포인트 구성에 대한 정보 보기

액세스 포인트는 Amazon S3의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 Amazon S3 객체 작업을 수행하는 데 사용할 수 있는 버킷에 연결된 네트워크 엔드포인트입니다. S3 on Outposts를 통해 Outposts 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 액세스 포인트는 가상 호스트 스타일의 주소 지정만 지원합니다.

다음 주제는 AWS Management Console, AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 액세스 포인트에 대한 구성 정보를 반환하는 방법을 보여줍니다.

S3 콘솔 사용

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 액세스 포인트를 선택합니다.
3. 구성 세부 정보를 볼 Outposts 액세스 포인트를 선택합니다.
4. Outposts 액세스 포인트 개요(Outposts access point overview)에서 액세스 포인트 구성 세부 정보를 검토합니다.

AWS CLI 사용

다음 AWS CLI 예제에서는 Outposts 버킷에 대한 액세스 포인트를 가져옵니다. *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3control get-access-point --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

Java용 AWS SDK 사용

다음 SDK for Java 예제에서는 Outposts 버킷에 대한 액세스 포인트를 가져옵니다.

```
import com.amazonaws.services.s3control.model.*;

public void getAccessPoint(String accessPointArn) {

    GetAccessPointRequest reqGetAP = new GetAccessPointRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn);

    GetAccessPointResult respGetAP = s3ControlClient.getAccessPoint(reqGetAP);
    System.out.printf("GetAccessPoint Response: %s%n", respGetAP.toString());

}
```

Amazon S3 on Outposts 액세스 포인트 목록 보기

액세스 포인트는 Amazon S3의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 Amazon S3 객체 작업을 수행하는 데 사용할 수 있는 버킷에 연결된 네트워크 엔드포인트입니다. S3 on Outposts를 통해 Outposts 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 액세스 포인트는 가상 호스트 스타일의 주소 지정만 지원합니다.

다음 주제는 AWS Management Console, AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 액세스 포인트 목록을 반환하는 방법을 보여줍니다.

S3 콘솔 사용

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.

2. 왼쪽 탐색 창에서 Outposts 액세스 포인트를 선택합니다.
3. Outposts 액세스 포인트(Outposts access points)에서 S3 on Outposts 액세스 포인트 목록을 검토합니다.

AWS CLI 사용

다음 AWS CLI 예제에서는 Outposts 버킷에 대한 액세스 포인트를 나열합니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3control list-access-points --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket
```

Java용 AWS SDK 사용

다음 SDK for Java 예제에서는 Outposts 버킷에 대한 액세스 포인트를 나열합니다.

```
import com.amazonaws.services.s3control.model.*;

public void listAccessPoints(String bucketArn) {

    ListAccessPointsRequest reqListAPs = new ListAccessPointsRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    ListAccessPointsResult respListAPs = s3ControlClient.listAccessPoints(reqListAPs);
    System.out.printf("ListAccessPoints Response: %s%n", respListAPs.toString());

}
```

액세스 포인트 삭제

액세스 포인트는 Amazon S3의 공유 데이터 세트에 대한 대규모 데이터 액세스 관리를 간소화합니다. 액세스 포인트는 `GetObject` 및 `PutObject` 같은 Amazon S3 객체 작업을 수행하는 데 사용할 수 있는 버킷에 연결된 네트워크 엔드포인트입니다. S3 on Outposts를 통해 Outposts 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 액세스 포인트는 가상 호스트 스타일의 주소 지정만 지원합니다.

다음 예제에서는 AWS Management Console 및 AWS Command Line Interface(AWS CLI)를 사용하여 액세스 포인트를 삭제하는 방법을 보여줍니다.

S3 콘솔 사용

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 액세스 포인트를 선택합니다.
3. Outposts 액세스 포인트 단원에서 삭제할 Outposts 액세스 포인트를 선택합니다.
4. 삭제를 선택합니다.
5. 삭제를 확인합니다.

AWS CLI 사용

다음 AWS CLI 예제에서는 Outposts 액세스 포인트를 삭제합니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3control delete-access-point --account-id 123456789012 --name arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-access-point
```

액세스 포인트 정책 추가 또는 편집

액세스 포인트에는 해당 액세스 포인트를 통해 이루어진 모든 요청에 대해 Amazon S3 on Outposts가 적용하는 고유한 권한 및 네트워크 제어가 있습니다. 각 액세스 포인트는 기본 버킷에 연결된 버킷 정책과 함께 작동하는 사용자 지정 액세스 포인트 정책을 적용합니다. 자세한 내용은 액세스 포인트 단원을 참조하십시오.

다음 주제는 AWS Management Console, AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 액세스 포인트에 대한 액세스 포인트 정책을 추가 또는 편집하는 방법을 보여줍니다.

S3 콘솔 사용

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 액세스 포인트 정책을 편집할 Outposts 버킷을 선택합니다.
4. [Outposts 액세스 포인트(Outposts access points)] 탭을 선택합니다.
5. Outposts 액세스 포인트 단원에서 정책을 편집할 액세스 포인트를 선택하고 정책 편집을 선택합니다.

6. Outposts 액세스 포인트 정책 단원에서 정책을 추가하거나 편집합니다. 자세한 내용은 S3 on Outposts로 IAM 설정 단원을 참조하십시오.

AWS CLI 사용

다음 AWS CLI 예제에서는 Outposts 액세스 포인트에 대한 정책을 적용합니다.

1. 다음 액세스 포인트 정책을 JSON 파일로 저장합니다. 이 예시에서 파일의 이름은 `appolicy1.json`으로 지정됩니다. *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
{
   "Version":"2012-10-17",
   "Id":"exampleAccessPointPolicy",
   "Statement":[
      {
         "Sid":"st1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
         "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point
      }
   ]
}
```

2. `put-access-point-policy` CLI 명령의 일부로 JSON 파일을 제출합니다. *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
aws s3control put-access-point-policy --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --policy file://appolicy1.json
```

Java용 AWS SDK 사용

다음 SDK for Java 예제에서는 Outposts 액세스 포인트에 대한 정책을 적용합니다.

```
import com.amazonaws.services.s3control.model.*;

public void putAccessPointPolicy(String accessPointArn) {

    String policy = "{\"Version\":\"2012-10-17\",\"Id\":\"testAccessPointPolicy\",
\"Statement\":[{\"Sid\":\"st1\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"" +
 AccountId + "\"},\"Action\":\"s3-outposts:*\",\"Resource\":\"" + accessPointArn +
 "\"}]}";

    PutAccessPointPolicyRequest reqPutAccessPointPolicy = new
 PutAccessPointPolicyRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn)
            .withPolicy(policy);

    PutAccessPointPolicyResult respPutAccessPointPolicy =
 s3ControlClient.putAccessPointPolicy(reqPutAccessPointPolicy);
    System.out.printf("PutAccessPointPolicy Response: %s%n",
 respPutAccessPointPolicy.toString());
    printWriter.printf("PutAccessPointPolicy Response: %s%n",
 respPutAccessPointPolicy.toString());

}
```

S3 on Outposts 액세스 포인트에 대한 액세스 포인트 정책 보기

액세스 포인트에는 해당 액세스 포인트를 통해 이루어진 모든 요청에 대해 Amazon S3 on Outposts가 적용하는 고유한 권한 및 네트워크 제어가 있습니다. 각 액세스 포인트는 기본 버킷에 연결된 버킷 정책과 함께 작동하는 사용자 지정 액세스 포인트 정책을 적용합니다. 자세한 내용은 액세스 포인트 단원을 참조하십시오.

S3 on Outposts에서의 액세스 포인트 작업에 대한 자세한 내용은 S3 on Outposts 버킷 작업 단원을 참조하세요.

다음 주제는 AWS Management Console, AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 액세스 포인트 정책을 보는 방법을 보여줍니다.

S3 콘솔 사용

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 액세스 포인트를 선택합니다.

3. 정책을 볼 Outposts 액세스 포인트를 선택합니다.
4. 권한(Permissions) 탭에서 S3 on Outposts 액세스 포인트 정책을 검토합니다.
5. 액세스 포인트 정책을 편집하려면 액세스 포인트 정책 추가 또는 편집 단원을 참조하세요.

AWS CLI 사용

다음 AWS CLI 예제에서는 Outposts 액세스 포인트에 대한 정책을 가져옵니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3control get-access-point-policy --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

Java용 AWS SDK 사용

다음 SDK for Java 예제에서는 Outposts 액세스 포인트에 대한 정책을 가져옵니다.

```
import com.amazonaws.services.s3control.model.*;

public void getAccessPointPolicy(String accessPointArn) {

    GetAccessPointPolicyRequest reqGetAccessPointPolicy = new
 GetAccessPointPolicyRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn);

    GetAccessPointPolicyResult respGetAccessPointPolicy =
 s3ControlClient.getAccessPointPolicy(reqGetAccessPointPolicy);
    System.out.printf("GetAccessPointPolicy Response: %s%n",
 respGetAccessPointPolicy.toString());
    printWriter.printf("GetAccessPointPolicy Response: %s%n",
 respGetAccessPointPolicy.toString());

}
```

Amazon S3 on Outposts 엔드포인트 작업

요청을 Amazon S3 on Outposts 액세스 포인트로 라우팅하려면 S3 on Outposts 엔드포인트를 생성하고 구성해야 합니다. 엔드포인트를 생성하려면 Outposts 홈 리전에 대한 서비스 링크와의 활성 연결이 필요합니다. Outpost에 있는 각 Virtual Private Cloud(VPC)에는 연결된 엔드포인트가 하나씩 있을 수

있습니다. 엔드포인트 할당량에 대한 자세한 내용은 S3 on Outposts 네트워크 요구 사항 단원을 참조하세요. Outposts 버킷에 액세스하고 객체 작업을 수행하려면 엔드포인트를 생성해야 합니다. 자세한 내용은 엔드포인트 단원을 참조하십시오.

엔드포인트를 생성한 후 '상태' 필드를 사용하여 엔드포인트의 상태를 파악할 수 있습니다. Outposts가 오프라인 상태일 경우 CREATE_FAILED가 반환됩니다. 연결이 재개된 후 서비스 링크 연결을 확인하고, 엔드포인트를 삭제하고, 생성 작업을 재시도할 수 있습니다. 추가 오류 코드 목록은 아래를 참조하세요. 자세한 내용은 엔드포인트 단원을 참조하십시오.

API	상태 표시기	실패 이유 오류 코드	메시지 - 실패 이유
`CreateEndpoint`	Create_Failed	OutpostNotReachable	Outposts 홈 리전에 대한 서비스 링크 연결이 끊어졌으므로 엔드포인트를 생성할 수 없습니다. 연결을 확인하고 엔드포인트를 삭제한 후 다시 시도하세요.
`CreateEndpoint`	Create_Failed	InternalError	내부 오류로 인해 엔드포인트를 생성할 수 없습니다. 엔드포인트를 삭제하고 다시 생성하세요.
`DeleteEndpoint`	Delete_Failed	OutpostNotReachable	Outposts 홈 리전에 대한 서비스 링크 연결이 끊어졌으므로 엔드포인트를 삭제할 수 없습니다. 연결을 확인하고 다시 시도하세요.
`DeleteEndpoint`	Delete_Failed	InternalError	내부 오류로 인해 엔드포인트를 삭제할 수 없습니다. 다시 시도하세요.

S3 on Outposts에서의 버킷 작업에 대한 자세한 내용은 S3 on Outposts 버킷 작업 섹션을 참조하세요.

다음 섹션에서는 S3 on Outposts에 대한 엔드포인트를 생성하고 관리하는 방법에 대해 설명합니다.

주제

- Outpost에 엔드포인트 생성
- Amazon S3 on Outposts 엔드포인트 목록 보기
- Amazon S3 on Outposts 엔드포인트 삭제

Outpost에 엔드포인트 생성

요청을 Amazon S3 on Outposts 액세스 포인트로 라우팅하려면 S3 on Outposts 엔드포인트를 생성하고 구성해야 합니다. 엔드포인트를 생성하려면 Outposts 홈 리전에 대한 서비스 링크와의 활성 연결이 필요합니다. Outpost에 있는 각 Virtual Private Cloud(VPC)에는 연결된 엔드포인트가 하나씩 있을 수 있습니다. 엔드포인트 할당량에 대한 자세한 내용은 S3 on Outposts 네트워크 요구 사항 단원을 참조하세요. Outposts 버킷에 액세스하고 객체 작업을 수행하려면 엔드포인트를 생성해야 합니다. 자세한 내용은 엔드포인트 단원을 참조하십시오.

권한

엔드포인트 생성에 필요한 권한에 대한 자세한 내용은 S3 on Outposts 엔드포인트에 대한 권한 단원을 참조하세요.

엔드포인트를 생성하면 S3 on Outposts에서 AWS 계정에 서비스 연결 역할을 생성합니다. 자세한 내용은 Amazon S3 on Outposts에 대한 서비스 연결 역할 사용 단원을 참조하십시오.

다음 예제에서는 AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDK for Java를 사용하여 S3 on Outposts 엔드포인트를 생성하는 방법을 보여줍니다.

S3 콘솔 사용

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 액세스 포인트를 선택합니다.
3. Outposts 엔드포인트(Outposts endpoints) 탭을 선택합니다.
4. Outposts 엔드포인트 생성(Create Outposts endpoint)을 선택합니다.
5. Outpost에서 이 엔드포인트를 생성할 Outpost를 선택합니다.
6. VPC에서 아직 엔드포인트가 없으며 Outposts 엔드포인트 규칙을 준수하는 VPC를 선택합니다.

   Virtual Private Cloud(VPC)를 사용하면 사용자가 정의한 가상 네트워크로 AWS 리소스를 시작할 수 있습니다. 이 가상 네트워크는 AWS의 확장 가능한 인프라를 사용한다는 이점과 함께 고객의 자체 데이터 센터에서 운영하는 기존 네트워크와 매우 유사합니다.

   VPC가 없으면 VPC 생성을 선택합니다. 자세한 내용은 Amazon S3 사용 설명서의 가상 프라이빗 클라우드(VPC)로 제한된 액세스 포인트 만들기를 참조하세요.
7. Outposts 엔드포인트 생성(Create Outposts endpoint)을 선택합니다.

AWS CLI 사용

Example

다음 AWS CLI 예제에서는 VPC 리소스 액세스 유형을 사용하여 Outposts의 엔드포인트를 생성합니다. VPC는 서브넷에서 파생되었습니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1
```

다음 AWS CLI 예제에서는 고객 소유 IP 주소 풀(CoIP 풀) 액세스 유형을 사용하여 Outpost에 대한 엔드포인트를 생성합니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1 --access-type CustomerOwnedIp --
customer-owned-ipv4-pool ipv4pool-coip-12345678901234567
```

Java용 AWS SDK 사용

Example

다음 SDK for Java 예제에서는 Outpost에 대한 엔드포인트를 생성합니다. 이 예제를 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.CreateEndpointRequest;
import com.amazonaws.services.s3outposts.model.CreateEndpointResult;

public void createEndpoint() {
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    CreateEndpointRequest createEndpointRequest = new CreateEndpointRequest()
                .withOutpostId("op-0d79779cef3c30a40")
                .withSubnetId("subnet-8c7a57c5")
                .withSecurityGroupId("sg-ab19e0d1")
                .withAccessType("CustomerOwnedIp")
                .withCustomerOwnedIpv4Pool("ipv4pool-coip-12345678901234567");
    // Use .withAccessType and .withCustomerOwnedIpv4Pool only when the access type is
```

```
    // customer-owned IP address pool (CoIP pool)
    CreateEndpointResult createEndpointResult =
 s3OutpostsClient.createEndpoint(createEndpointRequest);
    System.out.println("Endpoint is created and its ARN is " +
 createEndpointResult.getEndpointArn());
}
```

Amazon S3 on Outposts 엔드포인트 목록 보기

요청을 Amazon S3 on Outposts 액세스 포인트로 라우팅하려면 S3 on Outposts 엔드포인트를 생성하고 구성해야 합니다. 엔드포인트를 생성하려면 Outposts 홈 리전에 대한 서비스 링크와의 활성 연결이 필요합니다. Outpost에 있는 각 Virtual Private Cloud(VPC)에는 연결된 엔드포인트가 하나씩 있을 수 있습니다. 엔드포인트 할당량에 대한 자세한 내용은 S3 on Outposts 네트워크 요구 사항 단원을 참조하세요. Outposts 버킷에 액세스하고 객체 작업을 수행하려면 엔드포인트를 생성해야 합니다. 자세한 내용은 엔드포인트 단원을 참조하십시오.

다음 예제는 AWS Management Console, AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 엔드포인트 목록을 반환하는 방법을 보여줍니다.

S3 콘솔 사용

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 액세스 포인트를 선택합니다.
3. Outposts 액세스 포인트(Outposts access points) 페이지에서 Outposts 엔드포인트(Outposts endpoints) 탭을 선택합니다.
4. Outposts 엔드포인트(Outposts endpoints)에서 S3 on Outposts 엔드포인트 목록을 볼 수 있습니다.

AWS CLI 사용

다음 AWS CLI 예제에서는 사용자 계정과 연결된 AWS Outposts 리소스의 엔드포인트를 나열합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 list-endpoints를 참조하세요.

```
aws s3outposts list-endpoints
```

Java용 AWS SDK 사용

다음 SDK for Java 예제에서는 Outpost의 엔드포인트를 나열합니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 ListEndpoints를 참조하세요.

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.ListEndpointsRequest;
import com.amazonaws.services.s3outposts.model.ListEndpointsResult;

public void listEndpoints() {
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    ListEndpointsRequest listEndpointsRequest = new ListEndpointsRequest();
    ListEndpointsResult listEndpointsResult =
 s3OutpostsClient.listEndpoints(listEndpointsRequest);
    System.out.println("List endpoints result is " + listEndpointsResult);
}
```

Amazon S3 on Outposts 엔드포인트 삭제

요청을 Amazon S3 on Outposts 액세스 포인트로 라우팅하려면 S3 on Outposts 엔드포인트를 생성하고 구성해야 합니다. 엔드포인트를 생성하려면 Outposts 홈 리전에 대한 서비스 링크와의 활성 연결이 필요합니다. Outpost에 있는 각 Virtual Private Cloud(VPC)에는 연결된 엔드포인트가 하나씩 있을 수 있습니다. 엔드포인트 할당량에 대한 자세한 내용은 S3 on Outposts 네트워크 요구 사항 단원을 참조하세요. Outposts 버킷에 액세스하고 객체 작업을 수행하려면 엔드포인트를 생성해야 합니다. 자세한 내용은 엔드포인트 단원을 참조하십시오.

다음 예제는 AWS Management Console, AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 엔드포인트 목록을 삭제하는 방법을 보여줍니다.

S3 콘솔 사용

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 액세스 포인트를 선택합니다.
3. Outposts 액세스 포인트(Outposts access points) 페이지에서 Outposts 엔드포인트(Outposts endpoints) 탭을 선택합니다.
4. Outposts 엔드포인트(Outposts endpoints)에서 삭제할 엔드포인트를 선택하고 삭제(Delete)를 선택합니다.

AWS CLI 사용

다음 AWS CLI 예제에서는 Outpost의 엔드포인트를 삭제합니다. 이 명령을 실행하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws s3outposts delete-endpoint --endpoint-id example-endpoint-id --outpost-
id op-01ac5d28a6a232904
```

Java용 AWS SDK 사용

다음 SDK for Java 예제에서는 Outpost의 엔드포인트를 삭제합니다. 이 예제를 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.arn.Arn;
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.DeleteEndpointRequest;

public void deleteEndpoint(String endpointArnInput) {
    String outpostId = "op-01ac5d28a6a232904";
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    Arn endpointArn = Arn.fromString(endpointArnInput);
    String[] resourceParts = endpointArn.getResource().getResource().split("/");
    String endpointId = resourceParts[resourceParts.length - 1];
    DeleteEndpointRequest deleteEndpointRequest = new DeleteEndpointRequest()
                .withEndpointId(endpointId)
                .withOutpostId(outpostId);
    s3OutpostsClient.deleteEndpoint(deleteEndpointRequest);
    System.out.println("Endpoint with id " + endpointId + " is deleted.");
}
```

S3 on Outposts 객체 작업

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다.

객체는 Amazon S3 on Outposts에 저장되는 기본 개체입니다. 모든 객체는 어떤 버킷에 포함됩니다. Outpost 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 객체 작업에 버킷을 지정할 때 액세스 포인트 이름이 포함된 액세스 포인트 Amazon 리소스 이름(ARN) 또는 액세스 포인트 별칭을 사용합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

다음 예제는 S3 on Outposts 액세스 포인트에 대한 ARN 형식을 보여줍니다. 여기에는 Outpost가 있는 리전의 AWS 리전 코드, AWS 계정 ID, Outpost ID 및 액세스 포인트 이름이 포함됩니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

S3 on Outposts ARN에 대한 자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하세요.

객체 ARN은 Outpost가 위치한 AWS 리전, AWS 계정 ID, Outpost ID, 버킷 이름, 객체 키를 포함한 다음과 같은 형식을 사용합니다.

```
arn:aws:s3-outposts:us-west-2:123456789012:outpost/ op-01ac5d28a6a232904/bucket/amzn-s3-demo-bucket1/object/myobject
```

Amazon S3 on Outposts를 사용할 경우 객체 데이터는 항상 Outpost에 저장됩니다. AWS가 Outpost 랙을 설치하는 경우 데이터 상주 요구 사항을 충족하기 위해 데이터가 Outpost에 로컬로 유지됩니다. 객체는 Outpost에서 벗어나지 않으며 AWS 리전에 있지 않습니다. AWS Management Console이 리전 내에 호스팅되므로 콘솔을 사용하여 Outpost의 객체를 업로드하거나 관리할 수 없습니다. 그러나 REST API, AWS Command Line Interface(AWS CLI), AWS SDK를 사용하여 액세스 포인트를 통해 객체를 업로드하고 관리할 수 있습니다.

주제

- S3 on Outposts 버킷에 객체 업로드
- AWS SDK for Java를 사용하여 Amazon S3 on Outposts 버킷의 객체 복사
- Amazon S3 on Outposts 버킷에서 객체 가져오기
- Amazon S3 on Outposts 버킷의 객체 나열
- Amazon S3 on Outposts 버킷의 객체 삭제
- HeadBucket을 사용하여 S3 on Outposts 버킷이 존재하고 버킷에 액세스할 수 있는 권한이 있는지 확인
- Java용 SDK를 사용하여 멀티파트 업로드 수행 및 관리
- S3 on Outposts에서 미리 서명된 URL 사용
- 로컬 Amazon EMR on Outposts를 사용하는 Amazon S3 on Outposts
- 권한 부여 및 인증 캐시

S3 on Outposts 버킷에 객체 업로드

객체는 Amazon S3 on Outposts에 저장되는 기본 개체입니다. 모든 객체는 어떤 버킷에 포함됩니다. Outpost 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 객체 작업에 버킷을 지정할 때 액세스 포인트 이름이 포함된 액세스 포인트 Amazon 리소스 이름(ARN) 또는 액세스 포인트 별칭을 사용합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

다음 예제는 S3 on Outposts 액세스 포인트에 대한 ARN 형식을 보여줍니다. 여기에는 Outpost가 있는 리전의 AWS 리전 코드, AWS 계정 ID, Outpost ID 및 액세스 포인트 이름이 포함됩니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

S3 on Outposts ARN에 대한 자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하세요.

Amazon S3 on Outposts를 사용할 경우 객체 데이터는 항상 Outpost에 저장됩니다. AWS가 Outpost 랙을 설치하는 경우 데이터 상주 요구 사항을 충족하기 위해 데이터가 Outpost에 로컬로 유지됩니다. 객체는 Outpost에서 벗어나지 않으며 AWS 리전에 있지 않습니다. AWS Management Console이 리전 내에 호스팅되므로 콘솔을 사용하여 Outpost의 객체를 업로드하거나 관리할 수 없습니다. 그러나 REST API, AWS Command Line Interface(AWS CLI), AWS SDK를 사용하여 액세스 포인트를 통해 객체를 업로드하고 관리할 수 있습니다.

다음 AWS CLI 및 AWS SDK for Java 예제에서는 액세스 포인트를 사용하여 S3 on Outposts 버킷에 객체를 업로드하는 방법을 보여줍니다.

AWS CLI

Example

다음 예제에서는 AWS CLI를 사용하여 `sample-object.xml`이라는 객체를 S3 on Outposts 버킷(`s3-outposts:PutObject`)에 배치합니다. 이 명령을 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 put-object를 참조하세요.

```
aws s3api put-object --bucket arn:aws:s3-
outposts:Region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --key sample-object.xml --body sample-object.xml
```

SDK for Java

Example

다음 예제에서는 Java용 SDK를 사용하여 S3 on Outposts 버킷에 객체를 배치합니다. 이 예제를 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectMetadata;
import com.amazonaws.services.s3.model.PutObjectRequest;

import java.io.File;

public class PutObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String stringObjKeyName = "*** String object key name ***";
        String fileObjKeyName = "*** File object key name ***";
        String fileName = "*** Path to file to upload ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
```

```
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Upload a text string as a new object.
            s3Client.putObject(accessPointArn, stringObjKeyName, "Uploaded String
 Object");

            // Upload a file as a new object with ContentType and title specified.
            PutObjectRequest request = new PutObjectRequest(accessPointArn,
 fileObjKeyName, new File(fileName));
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.setContentType("plain/text");
            metadata.addUserMetadata("title", "someTitle");
            request.setMetadata(metadata);
            s3Client.putObject(request);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

AWS SDK for Java를 사용하여 Amazon S3 on Outposts 버킷의 객체 복사

객체는 Amazon S3 on Outposts에 저장되는 기본 개체입니다. 모든 객체는 어떤 버킷에 포함됩니다. Outpost 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 객체 작업에 버킷을 지정할 때 액세스 포인트 이름이 포함된 액세스 포인트 Amazon 리소스 이름(ARN) 또는 액세스 포인트 별칭을 사용합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

다음 예제는 S3 on Outposts 액세스 포인트에 대한 ARN 형식을 보여줍니다. 여기에는 Outpost가 있는 리전의 AWS 리전 코드, AWS 계정 ID, Outpost ID 및 액세스 포인트 이름이 포함됩니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

S3 on Outposts ARN에 대한 자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하세요.

Amazon S3 on Outposts를 사용할 경우 객체 데이터는 항상 Outpost에 저장됩니다. AWS가 Outpost 랙을 설치하는 경우 데이터 상주 요구 사항을 충족하기 위해 데이터가 Outpost에 로컬로 유지됩니다. 객체는 Outpost에서 벗어나지 않으며 AWS 리전에 있지 않습니다. AWS Management Console이 리전 내에 호스팅되므로 콘솔을 사용하여 Outpost의 객체를 업로드하거나 관리할 수 없습니다. 그러나 REST API, AWS Command Line Interface(AWS CLI), AWS SDK를 사용하여 액세스 포인트를 통해 객체를 업로드하고 관리할 수 있습니다.

다음 예제에서는 AWS SDK for Java를 사용하여 S3 on Outposts 버킷의 객체를 나열하는 방법을 보여줍니다.

Java용 AWS SDK 사용

다음 S3 on Outposts 예제에서는 SDK for Java를 사용하여 객체를 동일한 버킷의 새 객체로 복사합니다. 이 예제를 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;

public class CopyObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String sourceKey = "*** Source object key ***";
        String destinationKey = "*** Destination object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Copy the object into a new object in the same bucket.
            CopyObjectRequest copyObjectRequest = new CopyObjectRequest(accessPointArn,
 sourceKey, accessPointArn, destinationKey);
```

```
            s3Client.copyObject(copyObjectRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Amazon S3 on Outposts 버킷에서 객체 가져오기

객체는 Amazon S3 on Outposts에 저장되는 기본 개체입니다. 모든 객체는 어떤 버킷에 포함됩니다. Outpost 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 객체 작업에 버킷을 지정할 때 액세스 포인트 이름이 포함된 액세스 포인트 Amazon 리소스 이름(ARN) 또는 액세스 포인트 별칭을 사용합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

다음 예제는 S3 on Outposts 액세스 포인트에 대한 ARN 형식을 보여줍니다. 여기에는 Outpost가 있는 리전의 AWS 리전 코드, AWS 계정 ID, Outpost ID 및 액세스 포인트 이름이 포함됩니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

S3 on Outposts ARN에 대한 자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하세요.

Amazon S3 on Outposts를 사용할 경우 객체 데이터는 항상 Outpost에 저장됩니다. AWS가 Outpost 랙을 설치하는 경우 데이터 상주 요구 사항을 충족하기 위해 데이터가 Outpost에 로컬로 유지됩니다. 객체는 Outpost에서 벗어나지 않으며 AWS 리전에 있지 않습니다. AWS Management Console이 리전 내에 호스팅되므로 콘솔을 사용하여 Outpost의 객체를 업로드하거나 관리할 수 없습니다. 그러나 REST API, AWS Command Line Interface(AWS CLI), AWS SDK를 사용하여 액세스 포인트를 통해 객체를 업로드하고 관리할 수 있습니다.

다음 예제에서는 AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 객체를 다운로드(가져오기)하는 방법을 보여줍니다.

AWS CLI 사용

다음 예제에서는 AWS CLI를 사용하여 `sample-object.xml`이라는 객체를 S3 on Outposts 버킷(`s3-outposts:GetObject`)에서 가져옵니다. 이 명령을 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 get-object를 참조하세요.

```
aws s3api get-object --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-access-point --key testkey sample-object.xml
```

Java용 AWS SDK 사용

다음 S3 on Outposts 예제에서는 Java용 SDK를 사용하여 객체를 가져옵니다. 이 예제를 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다. 자세한 내용은 Amazon Simple Storage Service API Reference의 GetObject 섹션을 참조하세요.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GetObjectRequest;
import com.amazonaws.services.s3.model.ResponseHeaderOverrides;
import com.amazonaws.services.s3.model.S3Object;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

public class GetObject {
    public static void main(String[] args) throws IOException {
        String accessPointArn = "*** access point ARN ***";
        String key = "*** Object key ***";

        S3Object fullObject = null, objectPortion = null, headerOverrideObject = null;
        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
```

```
                    .build();

            // Get an object and print its contents.
            System.out.println("Downloading an object");
            fullObject = s3Client.getObject(new GetObjectRequest(accessPointArn, key));
            System.out.println("Content-Type: " +
fullObject.getObjectMetadata().getContentType());
            System.out.println("Content: ");
            displayTextInputStream(fullObject.getObjectContent());

            // Get a range of bytes from an object and print the bytes.
            GetObjectRequest rangeObjectRequest = new GetObjectRequest(accessPointArn,
key)
                    .withRange(0, 9);
            objectPortion = s3Client.getObject(rangeObjectRequest);
            System.out.println("Printing bytes retrieved.");
            displayTextInputStream(objectPortion.getObjectContent());

            // Get an entire object, overriding the specified response headers, and
print the object's content.
            ResponseHeaderOverrides headerOverrides = new ResponseHeaderOverrides()
                    .withCacheControl("No-cache")
                    .withContentDisposition("attachment; filename=example.txt");
            GetObjectRequest getObjectRequestHeaderOverride = new
GetObjectRequest(accessPointArn, key)
                    .withResponseHeaders(headerOverrides);
            headerOverrideObject = s3Client.getObject(getObjectRequestHeaderOverride);
            displayTextInputStream(headerOverrideObject.getObjectContent());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        } finally {
            // To ensure that the network connection doesn't remain open, close any
open input streams.
            if (fullObject != null) {
                fullObject.close();
            }
            if (objectPortion != null) {
                objectPortion.close();
```

```
            }
            if (headerOverrideObject != null) {
                headerOverrideObject.close();
            }
        }
    }

    private static void displayTextInputStream(InputStream input) throws IOException {
        // Read the text input stream one line at a time and display each line.
        BufferedReader reader = new BufferedReader(new InputStreamReader(input));
        String line = null;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        System.out.println();
    }
}
```

Amazon S3 on Outposts 버킷의 객체 나열

객체는 Amazon S3 on Outposts에 저장되는 기본 개체입니다. 모든 객체는 어떤 버킷에 포함됩니다. Outpost 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 객체 작업에 버킷을 지정할 때 액세스 포인트 이름이 포함된 액세스 포인트 Amazon 리소스 이름(ARN) 또는 액세스 포인트 별칭을 사용합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

다음 예제는 S3 on Outposts 액세스 포인트에 대한 ARN 형식을 보여줍니다. 여기에는 Outpost가 있는 리전의 AWS 리전 코드, AWS 계정 ID, Outpost ID 및 액세스 포인트 이름이 포함됩니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

S3 on Outposts ARN에 대한 자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하세요.

> **Note**
>
> Amazon S3 on Outposts를 사용할 경우 객체 데이터는 항상 Outpost에 저장됩니다. AWS가 Outpost 랙을 설치하는 경우 데이터 상주 요구 사항을 충족하기 위해 데이터가 Outpost에 로컬로 유지됩니다. 객체는 Outpost에서 벗어나지 않으며 AWS 리전에 있지 않습니다. AWS Management Console이 리전 내에 호스팅되므로 콘솔을 사용하여 Outpost의 객체를 업로드하

거나 관리할 수 없습니다. 그러나 REST API, AWS Command Line Interface(AWS CLI), AWS SDK를 사용하여 액세스 포인트를 통해 객체를 업로드하고 관리할 수 있습니다.

다음 예제에서는 AWS CLI 및 AWS SDK for Java를 사용하여 S3 on Outposts 버킷의 객체를 나열하는 방법을 보여줍니다.

AWS CLI 사용

다음 예제에서는 AWS CLI를 사용하여 S3 on Outposts 버킷(`s3-outposts:ListObjectsV2`)의 객체를 나열합니다. 이 명령을 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 list-objects-v2를 참조하세요.

```
aws s3api list-objects-v2 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

> **Note**
>
> AWS SDK를 통해 Amazon S3 on Outposts에서 이 작업을 사용할 때 버킷 이름 대신 Outposts 액세스 포인트 ARN을 `arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-Outposts-Access-Point` 형식으로 제공합니다. S3 on Outposts ARN에 대한 자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하세요.

Java용 AWS SDK 사용

다음 S3 on Outposts 예제에서는 Java용 SDK를 사용하여 버킷의 객체를 나열합니다. 이 예제를 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다.

> **Important**
>
> 이 예제에서는 `ListObjects` API 작업의 최신 버전인 ListObjectsV2를 사용합니다. 애플리케이션 개발 시 이 개정된 API 작업을 사용하는 것이 좋습니다. 이전 버전과의 호환성을 위해 Amazon S3는 이 API 작업의 이전 버전을 계속 지원합니다.

```
import com.amazonaws.AmazonServiceException;
```

```
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListObjectsV2Request;
import com.amazonaws.services.s3.model.ListObjectsV2Result;
import com.amazonaws.services.s3.model.S3ObjectSummary;

public class ListObjectsV2 {

    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            System.out.println("Listing objects");

            // maxKeys is set to 2 to demonstrate the use of
            // ListObjectsV2Result.getNextContinuationToken()
            ListObjectsV2Request req = new
 ListObjectsV2Request().withBucketName(accessPointArn).withMaxKeys(2);
            ListObjectsV2Result result;

            do {
                result = s3Client.listObjectsV2(req);

                for (S3ObjectSummary objectSummary : result.getObjectSummaries()) {
                    System.out.printf(" - %s (size: %d)\n", objectSummary.getKey(),
 objectSummary.getSize());
                }
                // If there are more than maxKeys keys in the bucket, get a
 continuation token
                // and list the next objects.
                String token = result.getNextContinuationToken();
                System.out.println("Next Continuation Token: " + token);
                req.setContinuationToken(token);
            } while (result.isTruncated());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
```

```
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Amazon S3 on Outposts 버킷의 객체 삭제

객체는 Amazon S3 on Outposts에 저장되는 기본 개체입니다. 모든 객체는 어떤 버킷에 포함됩니다. Outpost 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 객체 작업에 버킷을 지정할 때 액세스 포인트 이름이 포함된 액세스 포인트 Amazon 리소스 이름(ARN) 또는 액세스 포인트 별칭을 사용합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

다음 예제는 S3 on Outposts 액세스 포인트에 대한 ARN 형식을 보여줍니다. 여기에는 Outpost가 있는 리전의 AWS 리전 코드, AWS 계정 ID, Outpost ID 및 액세스 포인트 이름이 포함됩니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

S3 on Outposts ARN에 대한 자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하세요.

Amazon S3 on Outposts를 사용할 경우 객체 데이터는 항상 Outpost에 저장됩니다. AWS가 Outpost 랙을 설치하는 경우 데이터 상주 요구 사항을 충족하기 위해 데이터가 Outpost에 로컬로 유지됩니다. 객체는 Outpost에서 벗어나지 않으며 AWS 리전에 있지 않습니다. AWS Management Console이 리전 내에 호스팅되므로 콘솔을 사용하여 Outpost의 객체를 업로드하거나 관리할 수 없습니다. 그러나 REST API, AWS Command Line Interface(AWS CLI), AWS SDK를 사용하여 액세스 포인트를 통해 객체를 업로드하고 관리할 수 있습니다.

다음 예제에서는 AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 버킷의 단일 객체 또는 여러 객체를 삭제하는 방법을 보여줍니다.

AWS CLI 사용

다음 예제에서는 S3 on Outposts 버킷에서 단일 객체 또는 여러 객체를 삭제하는 방법을 보여줍니다.

delete-object

다음 예제에서는 AWS CLI를 사용하여 S3 on Outposts 버킷(`s3-outposts:DeleteObject`)에서 `sample-object.xml`이라는 객체를 삭제합니다. 이 명령을 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 delete-object를 참조하세요.

```
aws s3api delete-object --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --key sample-object.xml
```

delete-objects

다음 예제에서는 AWS CLI를 사용하여 S3 on Outposts 버킷(`s3-outposts:DeleteObject`)에서 `sample-object.xml` 및 `test1.text`라는 두 객체를 삭제합니다. 이 명령을 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 delete-objects를 참조하세요.

```
aws s3api delete-objects --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --delete file://delete.json

delete.json
{
  "Objects": [
    {
      "Key": "test1.txt"
    },
    {
      "Key": "sample-object.xml"
    }
  ],
  "Quiet": false
}
```

Java용 AWS SDK 사용

다음 예제에서는 S3 on Outposts 버킷에서 단일 객체 또는 여러 객체를 삭제하는 방법을 보여줍니다.

DeleteObject

다음 S3 on Outposts 예제에서는 Java용 SDK를 사용하여 버킷의 객체를 삭제합니다. 이 예제를 사용하려면 Outpost에 대한 액세스 포인트 ARN과 삭제할 객체의 키 이름을 지정합니다. 자세한 내용은 Amazon Simple Storage Service API 참조에서 DeleteObject를 참조하세요.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectRequest;

public class DeleteObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String keyName = "*** key name ****";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            s3Client.deleteObject(new DeleteObjectRequest(accessPointArn, keyName));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

DeleteObjects

다음 S3 on Outposts 예제에서는 Java용 SDK를 사용하여 버킷에 객체를 업로드하고 삭제합니다. 이 예제를 사용하려면 Outpost에 대한 액세스 포인트 ARN을 지정합니다. 자세한 내용은 Amazon Simple Storage Service API 참조에서 DeleteObjects를 참조하세요.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectsRequest;
import com.amazonaws.services.s3.model.DeleteObjectsRequest.KeyVersion;
import com.amazonaws.services.s3.model.DeleteObjectsResult;

import java.util.ArrayList;

public class DeleteObjects {

    public static void main(String[] args) {
       String accessPointArn = "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point";


        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Upload three sample objects.
            ArrayList<KeyVersion> keys = new ArrayList<KeyVersion>();
            for (int i = 0; i < 3; i++) {
                String keyName = "delete object example " + i;
                s3Client.putObject(accessPointArn, keyName, "Object number " + i + "
 to be deleted.");
                keys.add(new KeyVersion(keyName));
            }
            System.out.println(keys.size() + " objects successfully created.");

            // Delete the sample objects.
            DeleteObjectsRequest multiObjectDeleteRequest = new
 DeleteObjectsRequest(accessPointArn)
                    .withKeys(keys)
                    .withQuiet(false);

            // Verify that the objects were deleted successfully.
```

```
            DeleteObjectsResult delObjRes =
 s3Client.deleteObjects(multiObjectDeleteRequest);
            int successfulDeletes = delObjRes.getDeletedObjects().size();
            System.out.println(successfulDeletes + " objects successfully
 deleted.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

HeadBucket을 사용하여 S3 on Outposts 버킷이 존재하고 버킷에 액세스할 수 있는 권한이 있는지 확인

객체는 Amazon S3 on Outposts에 저장되는 기본 개체입니다. 모든 객체는 어떤 버킷에 포함됩니다. Outpost 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. 객체 작업에 버킷을 지정할 때 액세스 포인트 이름이 포함된 액세스 포인트 Amazon 리소스 이름(ARN) 또는 액세스 포인트 별칭을 사용합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

다음 예제는 S3 on Outposts 액세스 포인트에 대한 ARN 형식을 보여줍니다. 여기에는 Outpost가 있는 리전의 AWS 리전 코드, AWS 계정 ID, Outpost ID 및 액세스 포인트 이름이 포함됩니다.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

S3 on Outposts ARN에 대한 자세한 내용은 S3 on Outposts의 리소스 ARN 단원을 참조하세요.

> **Note**
>
> Amazon S3 on Outposts를 사용할 경우 객체 데이터는 항상 Outpost에 저장됩니다. AWS가 Outpost 랙을 설치하는 경우 데이터 상주 요구 사항을 충족하기 위해 데이터가 Outpost에 로컬로 유지됩니다. 객체는 Outpost에서 벗어나지 않으며 AWS 리전에 있지 않습니다. AWS Management Console이 리전 내에 호스팅되므로 콘솔을 사용하여 Outpost의 객체를 업로드하

거나 관리할 수 없습니다. 그러나 REST API, AWS Command Line Interface(AWS CLI), AWS SDK를 사용하여 액세스 포인트를 통해 객체를 업로드하고 관리할 수 있습니다.

다음 AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java 예제에서는 HeadBucket API 작업을 사용하여 Amazon S3 on Outposts 버킷이 존재하고 버킷에 액세스할 수 있는 권한이 있는지를 확인하는 방법을 보여줍니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 HeadBucket를 참조하세요.

AWS CLI 사용

다음 S3 on Outposts AWS CLI 예제에서는 `head-bucket` 명령을 사용하여 버킷이 존재하고 버킷에 액세스할 수 있는 권한이 있는지를 확인합니다. 이 명령을 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 head-bucket을 참조하세요.

```
aws s3api head-bucket --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

Java용 AWS SDK 사용

다음 S3 on Outposts 예제에서는 버킷이 존재하고 버킷에 액세스할 수 있는 권한이 있는지 확인하는 방법을 보여줍니다. 이 예제를 사용하려면 Outpost에 대한 액세스 포인트 ARN을 지정합니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 HeadBucket를 참조하세요.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.HeadBucketRequest;

public class HeadBucket {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
```

```
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            s3Client.headBucket(new HeadBucketRequest(accessPointArn));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Java용 SDK를 사용하여 멀티파트 업로드 수행 및 관리

Amazon S3 on Outposts를 사용하면 AWS Outposts 리소스에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 저장하고 검색할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

다음 예제에서는 AWS SDK for Java와 함께 S3 on Outposts를 사용하여 멀티파트 업로드를 수행하고 관리하는 방법을 보여줍니다.

주제

- S3 on Outposts 버킷에 있는 객체의 멀티파트 업로드 수행
- 멀티파트 업로드를 사용하여 S3 on Outposts 버킷에 있는 대형 객체 복사
- S3 on Outposts 버킷에 있는 객체의 부분 나열
- S3 on Outposts 버킷에서 진행 중인 멀티파트 업로드 목록 검색

S3 on Outposts 버킷에 있는 객체의 멀티파트 업로드 수행

다음 S3 on Outposts 예제에서는 Java용 SDK를 사용하여 버킷에 객체의 멀티파트 업로드를 시작, 업로드 및 완료합니다. 이 예제를 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체

합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 멀티파트 업로드를 사용한 객체 업로드를 참조하세요.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.ArrayList;
import java.util.List;

public class MultipartUploadCopy {
    public static void main(String[] args) {
        String accessPointArn = "*** Source access point ARN ***";
        String sourceObjectKey = "*** Source object key ***";
        String destObjectKey = "*** Target object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(accessPointArn, destObjectKey);
            InitiateMultipartUploadResult initResult =
 s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest(accessPointArn, sourceObjectKey);
            ObjectMetadata metadataResult =
 s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();

            // Copy the object using 5 MB parts.
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
```

```
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(accessPointArn)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(accessPointArn)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and make
the copied object available.
            CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest(
                    accessPointArn,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
```

```
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
```

멀티파트 업로드를 사용하여 S3 on Outposts 버킷에 있는 대형 객체 복사

다음 S3 on Outposts 예제에서는 Java용 SDK를 사용하여 버킷에 객체를 복사합니다. 이 예제를 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.ArrayList;
import java.util.List;

public class MultipartUploadCopy {
    public static void main(String[] args) {
        String accessPointArn = "*** Source access point ARN ***";
        String sourceObjectKey = "*** Source object key ***";
        String destObjectKey = "*** Target object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(accessPointArn, destObjectKey);
            InitiateMultipartUploadResult initResult =
 s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest(accessPointArn, sourceObjectKey);
```

```
            ObjectMetadata metadataResult =
 s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();

            // Copy the object using 5 MB parts.
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(accessPointArn)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(accessPointArn)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and make
 the copied object available.
            CompleteMultipartUploadRequest completeRequest = new
 CompleteMultipartUploadRequest(
                    accessPointArn,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
```

```
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
}
```

S3 on Outposts 버킷에 있는 객체의 부분 나열

다음 S3 on Outposts 예제에서는 Java용 SDK를 사용하여 버킷의 객체 파트를 나열합니다. 이 예제를 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.List;

public class ListParts {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String keyName = "*** Key name ***";
        String uploadId = "*** Upload ID ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();
```

```
            ListPartsRequest listPartsRequest = new ListPartsRequest(accessPointArn,
 keyName, uploadId);
            PartListing partListing = s3Client.listParts(listPartsRequest);
            List<PartSummary> partSummaries = partListing.getParts();

            System.out.println(partSummaries.size() + " multipart upload parts");
            for (PartSummary p : partSummaries) {
                System.out.println("Upload part: Part number = \"" + p.getPartNumber()
 + "\", ETag = " + p.getETag());
            }

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

S3 on Outposts 버킷에서 진행 중인 멀티파트 업로드 목록 검색

다음 S3 on Outposts 예제에서는 Outposts 버킷에서 Java용 SDK를 사용하여 진행 중인 멀티파트 업로드 목록을 검색하는 방법을 보여줍니다. 이 예제를 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListMultipartUploadsRequest;
import com.amazonaws.services.s3.model.MultipartUpload;
import com.amazonaws.services.s3.model.MultipartUploadListing;

import java.util.List;

public class ListMultipartUploads {
    public static void main(String[] args) {
                String accessPointArn = "*** access point ARN ***";
```

```
        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Retrieve a list of all in-progress multipart uploads.
            ListMultipartUploadsRequest allMultipartUploadsRequest = new
 ListMultipartUploadsRequest(accessPointArn);
            MultipartUploadListing multipartUploadListing =
 s3Client.listMultipartUploads(allMultipartUploadsRequest);
            List<MultipartUpload> uploads =
 multipartUploadListing.getMultipartUploads();

            // Display information about all in-progress multipart uploads.
            System.out.println(uploads.size() + " multipart upload(s) in progress.");
            for (MultipartUpload u : uploads) {
                System.out.println("Upload in progress: Key = \"" + u.getKey() + "\",
 id = " + u.getUploadId());
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

S3 on Outposts에서 미리 서명된 URL 사용

버킷 정책을 업데이트하지 않고 Outpost에 로컬로 저장된 객체에 한시적 액세스 권한을 부여하려면 미리 서명된 URL을 사용할 수 있습니다. 버킷 소유자는 미리 서명된 URL을 사용하여 Virtual Private Cloud(VPC)의 사용자와 객체를 공유하거나 이들에게 객체를 업로드 또는 삭제할 수 있는 권한을 부여할 수 있습니다.

AWS SDK 또는 AWS Command Line Interface(AWS CLI)를 사용하여 미리 서명된 URL을 생성하면 URL을 특정 작업과 연결합니다. 또한 최소 1초 및 최대 7일의 사용자 지정 만료 시간을 선택하여 미리 서명된 URL에 대한 한시적 액세스 권한을 부여할 수 있습니다. 미리 서명된 URL을 공유하면 VPC의 사용자가 원래 서명 사용자인 것처럼 URL에 포함된 작업을 수행할 수 있습니다. URL이 만료 시간에 도달하면 URL이 만료되고 더 이상 작동하지 않습니다.

미리 서명된 URL 기능 제한

미리 서명된 URL의 기능은 이를 만든 사용자의 권한에 의해 제한됩니다. 미리 서명된 URL은 기본적으로 이를 소유한 사용자에게 액세스 권한을 부여하는 보유자 토큰입니다. 따라서 이러한 URL은 적절하게 보호하는 것이 좋습니다.

AWS Signature Version 4(SigV4)

사전 서명된 URL 요청이 AWS Signature Version 4(SigV4)를 사용하여 인증될 때 특정 동작을 적용하기 위해 버킷 정책 및 액세스 포인트 정책에서 조건 키를 사용할 수 있습니다. 예를 들어 서명이 10분 이상 지난 경우, `example-outpost-bucket` 버킷의 객체에 대한 Amazon S3 on Outposts 사전 서명된 URL 요청을 거부하는 `s3-outposts:signatureAge` 조건을 사용하는 버킷 정책을 생성할 수 있습니다. 이 예제를 사용하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Deny a presigned URL request if the signature is more than 10
 minutes old",
            "Effect": "Deny",
            "Principal": {"AWS":"444455556666"},
            "Action": "s3-outposts:*",
            "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
            "Condition": {
                "NumericGreaterThan": {"s3-outposts:signatureAge": 600000},
                "StringEquals": {"s3-outposts:authType": "REST-QUERY-STRING"}
            }
        }
    ]
}
```

사전 서명된 URL 요청이 서명 버전 4를 사용하여 인증될 때 특정 동작을 적용하는 데 사용할 수 있는 조건 키 및 추가 예제 정책 목록은 AWS Signature Version 4(SigV4) 인증별 정책 키 섹션을 참조하세요.

네트워크 경로 제한

특정 네트워크 경로에 대해 미리 서명된 URL 및 모든 S3 on Outposts 액세스의 사용을 제한하려는 경우, 특정 네트워크 경로가 필요한 정책을 작성할 수 있습니다. 호출을 실행하는 IAM 보안 주체에 대한 제한을 설정하려면 자격 증명 기반 AWS Identity and Access Management(IAM) 정책(예: 사용자, 그룹 또는 역할 정책)을 사용할 수 있습니다. S3 on Outposts 리소스에 대한 제한을 설정하려면 리소스 기반 정책(예: 버킷 및 액세스 포인트 정책)을 사용할 수 있습니다.

IAM 보안 주체에 대한 네트워크 경로 제한은 해당 보안 인증 정보의 사용자가 지정된 네트워크에서 요청해야 합니다. 버킷 또는 액세스 포인트에 대한 제한은 해당 리소스에 대한 모든 요청이 지정된 네트워크에서 시작되도록 요구합니다. 이러한 제한은 미리 서명된 URL 시나리오 외부에서도 적용됩니다.

사용하는 IAM 전역 조건은 엔드포인트 유형에 따라 달라집니다. S3 on Outposts용 퍼블릭 엔드포인트를 사용하는 경우 `aws:SourceIp`를 사용합니다. S3 on Outposts용 VPC 엔드포인트를 사용하는 경우 `aws:SourceVpc` 또는 `aws:SourceVpce`를 사용합니다.

다음 IAM 정책 설명에서는 보안 주체가 지정된 네트워크 범위에서만 AWS에 액세스해야 합니다. 이 정책 설명을 사용하면 모든 액세스가 해당 범위에서 시작되어야 합니다. 여기에는 S3 on Outposts에 대해 미리 서명된 URL을 사용하는 사례가 포함됩니다. 이 예제를 사용하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

```
{
    "Sid": "NetworkRestrictionForIAMPrincipal",
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
        "NotIpAddressIfExists": {"aws:SourceIp": "IP-address-range"},
        "BoolIfExists": {"aws:ViaAWSService": "false"}
    }
}
```

`aws:SourceIP` AWS 전역 조건 키를 사용하여 S3 on Outposts 버킷에 대한 액세스를 특정 네트워크 범위로 제한하는 버킷 정책 예제는 S3 on Outposts로 IAM 설정 섹션을 참조하세요.

미리 서명된 URL을 생성할 수 있는 사용자

유효한 보안 자격 증명을 가진 사용자는 누구나 미리 서명된 URL을 만들 수 있습니다. 단, VPC의 사용자가 객체에 성공적으로 액세스하려면 미리 서명된 URL의 기반이 되는 작업을 수행할 권한이 있는 사람이 미리 서명된 URL을 생성해야 합니다.

다음 보안 인증 정보를 사용하여 미리 서명된 URL을 만들 수 있습니다.

- IAM 인스턴스 프로파일: 최대 6시간 동안 유효함.
- AWS Security Token Service: AWS 계정 계정 루트 사용자 또는 IAM 사용자의 보안 인증 정보 등의 영구 보안 인증 정보를 통해 서명된 경우 최대 36시간 동안 유효함.
- IAM 사용자: AWS 서명 버전 4를 사용할 경우 최대 7일 동안 유효함.

  최대 7일 동안 유효한 미리 서명된 URL을 생성하려면 먼저 IAM 사용자 보안 인증 정보(액세스 키 및 비밀 키)를 사용 중인 SDK에 위임합니다. 그런 다음 AWS 서명 버전 4를 사용하여 미리 서명된 URL을 생성합니다.

> **Note**
>
> - 임시 토큰을 사용하여 미리 서명된 URL을 생성할 경우, URL의 만료 시간이 토큰 만료 시간보다 이후인 경우에도 토큰이 만료되면 URL도 만료됩니다.
> - 미리 서명된 URL은 해당 URL을 소유한 모든 사람에게 S3 on Outposts 버킷에 대한 액세스 권한을 부여하므로, URL을 적절하게 보호하는 것이 좋습니다. 미리 서명된 URL 보호에 대한 자세한 내용은 미리 서명된 URL 기능 제한 섹션을 참조하세요.

S3 on Outposts는 미리 서명된 URL의 만료 날짜 및 시간을 언제 확인하나요?

HTTP 요청 시 S3 on Outposts는 서명된 URL의 만료 날짜와 시간을 확인합니다. 예를 들어 클라이언트가 만료 시간 직전에 대용량 파일을 다운로드하기 시작한 경우, 다운로드 중에 만료 시간이 경과해도 다운로드는 진행됩니다. 단, 연결이 끊어진 경우 클라이언트가 만료 시간 이후에 다운로드를 다시 시작하는 것은 불가능합니다.

미리 서명된 URL을 사용하여 객체를 공유하거나 업로드하는 방법에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

- 미리 서명된 URL을 사용하여 객체 공유
- 미리 서명된 URL을 생성하여 S3 on Outposts 버킷에 객체 업로드

미리 서명된 URL을 사용하여 객체 공유

버킷 정책을 업데이트하지 않고 Outpost에 로컬로 저장된 객체에 한시적 액세스 권한을 부여하려면 미리 서명된 URL을 사용할 수 있습니다. 버킷 소유자는 미리 서명된 URL을 사용하여 Virtual Private Cloud(VPC)의 사용자와 객체를 공유하거나 이들에게 객체를 업로드 또는 삭제할 수 있는 권한을 부여할 수 있습니다.

AWS SDK 또는 AWS Command Line Interface(AWS CLI)를 사용하여 미리 서명된 URL을 생성하면 URL을 특정 작업과 연결합니다. 또한 최소 1초 및 최대 7일의 사용자 지정 만료 시간을 선택하여 미리 서명된 URL에 대한 한시적 액세스 권한을 부여할 수 있습니다. 미리 서명된 URL을 공유하면 VPC의 사용자가 원래 서명 사용자인 것처럼 URL에 포함된 작업을 수행할 수 있습니다. URL이 만료 시간에 도달하면 URL이 만료되고 더 이상 작동하지 않습니다.

미리 서명된 URL을 생성하면 보안 인증 정보를 제공한 후 다음을 지정해야 합니다.

- Amazon S3 on Outposts 버킷에 대한 액세스 포인트 Amazon 리소스 이름(ARN)
- 객체 키
- HTTP 메서드(객체 다운로드를 위한 `GET`)
- 만료 날짜 및 시간

미리 서명된 URL은 지정된 기간 동안만 유효합니다. 즉, 만료 날짜 및 시간 전에 URL에서 허용하는 작업을 시작해야 합니다. 미리 서명된 URL은 만료 날짜 및 시간까지 여러 번 사용할 수 있습니다. 임시 토큰을 사용하여 미리 서명된 URL을 생성할 경우, URL의 만료 시간이 토큰 만료 시간보다 이후인 경우에도 토큰이 만료되면 URL도 만료됩니다.

미리 서명된 URL에 액세스할 수 있는 Virtual Private Cloud(VPC) 사용자는 객체에 액세스할 수 있습니다. 예를 들어, 버킷에 동영상이 있고 버킷과 객체 모두 비공개인 경우 미리 서명된 URL을 만들어 다른 사용자와 동영상을 공유할 수 있습니다. 미리 서명된 URL은 해당 URL을 소유한 모든 사람에게 S3 on Outposts 버킷에 대한 액세스 권한을 부여하므로, URL을 적절하게 보호하는 것이 좋습니다. 미리 서명된 URL 보호에 대한 자세한 내용은 미리 서명된 URL 기능 제한 섹션을 참조하세요.

유효한 보안 자격 증명을 가진 사용자는 누구나 미리 서명된 URL을 만들 수 있습니다. 단, 미리 서명된 URL은 미리 서명된 URL에서 제공하려는 작업을 수행할 권한이 있는 사용자가 생성해야 합니다. 자세한 내용은 미리 서명된 URL을 생성할 수 있는 사용자 단원을 참조하십시오.

AWS SDK 및 AWS CLI를 사용하여 S3 on Outposts 버킷에 있는 객체를 공유할 미리 서명된 URL을 생성할 수 있습니다. 자세한 정보는 다음 예를 참조하십시오.

AWS SDK 사용

AWS SDK를 사용하여 객체를 검색할 수 있도록 다른 사용자에게 제공할 미리 서명된 URL을 생성할 수 있습니다.

> **Note**
>
> AWS SDK를 사용하여 미리 서명된 URL을 생성할 때 미리 서명된 URL의 최대 만료 시간은 생성 시점으로부터 7일입니다.

Java

Example

다음 예제에서는 S3 on Outposts 버킷에서 객체를 검색할 수 있도록 다른 사용자에게 제공할 미리 서명된 URL을 생성합니다. 자세한 내용은 S3 on Outposts에서 미리 서명된 URL 사용 단원을 참조하십시오. 이 예제를 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.HttpMethod;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GeneratePresignedUrlRequest;

import java.io.IOException;
import java.net.URL;
import java.time.Instant;

public class GeneratePresignedURL {

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

```
        Regions clientRegion = Regions.DEFAULT_REGION;
        String accessPointArn = "*** access point ARN ***";
        String objectKey = "*** object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Set the presigned URL to expire after one hour.
            java.util.Date expiration = new java.util.Date();
            long expTimeMillis = Instant.now().toEpochMilli();
            expTimeMillis += 1000 * 60 * 60;
            expiration.setTime(expTimeMillis);

            // Generate the presigned URL.
            System.out.println("Generating pre-signed URL.");
            GeneratePresignedUrlRequest generatePresignedUrlRequest =
                    new GeneratePresignedUrlRequest(accessPointArn, objectKey)
                            .withMethod(HttpMethod.GET)
                            .withExpiration(expiration);
            URL url = s3Client.generatePresignedUrl(generatePresignedUrlRequest);

            System.out.println("Pre-Signed URL: " + url.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't
 process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Example

다음 예제에서는 S3 on Outposts 버킷에서 객체를 검색할 수 있도록 다른 사용자에게 제공할 미리 서명된 URL을 생성합니다. 자세한 내용은 S3 on Outposts에서 미리 서명된 URL 사용 단원을 참조하십시오. 이 예제를 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;

namespace Amazon.DocSamples.S3
{
    class GenPresignedURLTest
    {
        private const string accessPointArn = "*** access point ARN ***";
        private const string objectKey = "*** object key ***";
        // Specify how long the presigned URL lasts, in hours.
        private const double timeoutDuration = 12;
        // Specify your bucket Region (an example Region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            string urlString = GeneratePreSignedURL(timeoutDuration);
        }
        static string GeneratePreSignedURL(double duration)
        {
            string urlString = "";
            try
            {
                GetPreSignedUrlRequest request1 = new GetPreSignedUrlRequest
                {
                    BucketName = accessPointArn,
                    Key = objectKey,
                    Expires = DateTime.UtcNow.AddHours(duration)
                };
                urlString = s3Client.GetPreSignedURL(request1);
```

```
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            return urlString;
        }
    }
}
```

Python

다음 예제에서는 미리 서명된 URL을 생성하여 SDK for Python(Boto3)을 사용해 객체를 공유합니다. 예를 들어 Boto3 클라이언트와 `generate_presigned_url` 함수를 사용하여 객체를 `GET`할 수 있는 미리 서명된 URL을 생성합니다.

```
import boto3
    url = boto3.client('s3').generate_presigned_url(
    ClientMethod='get_object',
    Params={'Bucket': 'ACCESS_POINT_ARN', 'Key': 'OBJECT_KEY'},
    ExpiresIn=3600)
```

SDK for Python(Boto3)을 사용하여 미리 서명된 URL을 생성하는 방법에 대한 자세한 내용은 AWS SDK for Python (Boto) API 참조의 Python을 참조하세요.

AWS CLI 사용

다음 예제에서 AWS CLI 명령은 S3 on Outposts 버킷에 대해 미리 서명된 URL을 생성합니다. 이 예제를 사용하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

> **Note**
>
> AWS CLI를 사용하여 미리 서명된 URL을 생성할 때 미리 서명된 URL의 최대 만료 시간은 생성 시점으로부터 7일입니다.

```
aws s3 presign s3://arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/example-outpost-access-
point/mydoc.txt --expires-in 604800
```

자세한 내용은 AWS CLI 명령 참조 안내서의 미리 서명을 참조하세요.

미리 서명된 URL을 생성하여 S3 on Outposts 버킷에 객체 업로드

버킷 정책을 업데이트하지 않고 Outpost에 로컬로 저장된 객체에 한시적 액세스 권한을 부여하려면 미리 서명된 URL을 사용할 수 있습니다. 버킷 소유자는 미리 서명된 URL을 사용하여 Virtual Private Cloud(VPC)의 사용자와 객체를 공유하거나 이들에게 객체를 업로드 또는 삭제할 수 있는 권한을 부여할 수 있습니다.

AWS SDK 또는 AWS Command Line Interface(AWS CLI)를 사용하여 미리 서명된 URL을 생성하면 URL을 특정 작업과 연결합니다. 또한 최소 1초 및 최대 7일의 사용자 지정 만료 시간을 선택하여 미리 서명된 URL에 대한 한시적 액세스 권한을 부여할 수 있습니다. 미리 서명된 URL을 공유하면 VPC의 사용자가 원래 서명 사용자인 것처럼 URL에 포함된 작업을 수행할 수 있습니다. URL이 만료 시간에 도달하면 URL이 만료되고 더 이상 작동하지 않습니다.

미리 서명된 URL을 생성하면 보안 인증 정보를 제공한 후 다음을 지정해야 합니다.

- Amazon S3 on Outposts 버킷에 대한 액세스 포인트 Amazon 리소스 이름(ARN)
- 객체 키
- HTTP 메서드(객체 업로드를 위한 PUT)
- 만료 날짜 및 시간

미리 서명된 URL은 지정된 기간 동안만 유효합니다. 즉, 만료 날짜 및 시간 전에 URL에서 허용하는 작업을 시작해야 합니다. 미리 서명된 URL은 만료 날짜 및 시간까지 여러 번 사용할 수 있습니다. 임시 토큰을 사용하여 미리 서명된 URL을 생성할 경우, URL의 만료 시간이 토큰 만료 시간보다 이후인 경우에도 토큰이 만료되면 URL도 만료됩니다.

미리 서명된 URL에서 허용하는 작업이 멀티파트 업로드와 같이 여러 단계로 구성된 경우 만료 시간 전에 모든 단계를 시작해야 합니다. S3 on Outposts가 만료된 URL로 단계를 시작하려고 시도하는 경우 오류가 발생합니다.

미리 서명된 URL에 액세스할 수 있는 Virtual Private Cloud(VPC)의 사용자는 객체를 업로드할 수 있습니다. 예를 들어 미리 서명된 URL에 액세스할 수 있는 VPC의 사용자는 버킷에 객체를 업로드할 수 있습니다. 미리 서명된 URL은 미리 서명된 URL에 액세스할 수 있는 VPC의 모든 사용자에게 S3 on

Outposts 버킷에 대한 액세스 권한을 부여하므로, URL을 적절하게 보호하는 것이 좋습니다. 미리 서명된 URL 보호에 대한 자세한 내용은 미리 서명된 URL 기능 제한 섹션을 참조하세요.

유효한 보안 자격 증명을 가진 사용자는 누구나 미리 서명된 URL을 만들 수 있습니다. 단, 미리 서명된 URL은 미리 서명된 URL에서 제공하려는 작업을 수행할 권한이 있는 사용자가 생성해야 합니다. 자세한 내용은 미리 서명된 URL을 생성할 수 있는 사용자 단원을 참조하십시오.

AWS SDK를 사용하여 S3 on Outposts 객체 작업에 대한 미리 서명된 URL 생성

Java

SDK for Java 2.x

이 예제는 한시적으로 S3 on Outposts 버킷에 객체를 업로드하는 데 사용할 수 있는 미리 서명된 URL을 생성하는 방법을 보여줍니다. 자세한 내용은 S3 on Outposts에서 미리 서명된 URL 사용 단원을 참조하십시오.

```
   public static void signBucket(S3Presigner presigner, String
outpostAccessPointArn, String keyName) {

       try {
           PutObjectRequest objectRequest = PutObjectRequest.builder()
                   .bucket(accessPointArn)
                   .key(keyName)
                   .contentType("text/plain")
                   .build();

           PutObjectPresignRequest presignRequest =
PutObjectPresignRequest.builder()
                   .signatureDuration(Duration.ofMinutes(10))
                   .putObjectRequest(objectRequest)
                   .build();

           PresignedPutObjectRequest presignedRequest =
presigner.presignPutObject(presignRequest);


           String myURL = presignedRequest.url().toString();
           System.out.println("Presigned URL to upload a file to: " +myURL);
           System.out.println("Which HTTP method must be used when uploading a
file: " +
                   presignedRequest.httpRequest().method());
```

```
            // Upload content to the S3 on Outposts bucket by using this URL.
            URL url = presignedRequest.url();

            // Create the connection and use it to upload the new object by using
 the presigned URL.
            HttpURLConnection connection = (HttpURLConnection)
 url.openConnection();
            connection.setDoOutput(true);
            connection.setRequestProperty("Content-Type","text/plain");
            connection.setRequestMethod("PUT");
            OutputStreamWriter out = new
 OutputStreamWriter(connection.getOutputStream());
            out.write("This text was uploaded as an object by using a presigned
 URL.");
            out.close();

            connection.getResponseCode();
            System.out.println("HTTP response code is " +
 connection.getResponseCode());

        } catch (S3Exception e) {
            e.getStackTrace();
        } catch (IOException e) {
            e.getStackTrace();
        }
    }
```

Python

SDK for Python(Boto3)

이 예제는 한시적으로 S3 on Outposts 작업을 수행할 수 있는 미리 서명된 URL을 생성하는 방법을 보여줍니다. 자세한 내용은 S3 on Outposts에서 미리 서명된 URL 사용 단원을 참조하십시오. URL을 사용하여 요청하려면 `Requests` 패키지를 사용하세요.

```
import argparse
import logging
import boto3
from botocore.exceptions import ClientError
import requests

logger = logging.getLogger(__name__)
```

```
def generate_presigned_url(s3_client, client_method, method_parameters,
 expires_in):
    """
    Generate a presigned S3 on Outposts URL that can be used to perform an
 action.

    :param s3_client: A Boto3 Amazon S3 client.
    :param client_method: The name of the client method that the URL performs.
    :param method_parameters: The parameters of the specified client method.
    :param expires_in: The number of seconds that the presigned URL is valid for.
    :return: The presigned URL.
    """
    try:
        url = s3_client.generate_presigned_url(
            ClientMethod=client_method,
            Params=method_parameters,
            ExpiresIn=expires_in
        )
        logger.info("Got presigned URL: %s", url)
    except ClientError:
        logger.exception(
            "Couldn't get a presigned URL for client method '%s'.",
 client_method)
        raise
    return url


def usage_demo():
    logging.basicConfig(level=logging.INFO, format='%(levelname)s: %(message)s')

    print('-'*88)
    print("Welcome to the Amazon S3 on Outposts presigned URL demo.")
    print('-'*88)

    parser = argparse.ArgumentParser()
    parser.add_argument('accessPointArn', help="The name of the S3 on Outposts
 access point ARN.")
    parser.add_argument(
        'key', help="For a GET operation, the key of the object in S3 on
 Outposts. For a "
                    "PUT operation, the name of a file to upload.")
    parser.add_argument(
```

```
        'action', choices=('get', 'put'), help="The action to perform.")
    args = parser.parse_args()

    s3_client = boto3.client('s3')
    client_action = 'get_object' if args.action == 'get' else 'put_object'
    url = generate_presigned_url(
        s3_client, client_action, {'Bucket': args.accessPointArn, 'Key':
 args.key}, 1000)

    print("Using the Requests package to send a request to the URL.")
    response = None
    if args.action == 'get':
        response = requests.get(url)
    elif args.action == 'put':
        print("Putting data to the URL.")
        try:
            with open(args.key, 'r') as object_file:
                object_text = object_file.read()
            response = requests.put(url, data=object_text)
        except FileNotFoundError:
            print(f"Couldn't find {args.key}. For a PUT operation, the key must
 be the "
                  f"name of a file that exists on your computer.")

    if response is not None:
        print("Got response:")
        print(f"Status: {response.status_code}")
        print(response.text)

    print('-'*88)


if __name__ == '__main__':
    usage_demo()
```

로컬 Amazon EMR on Outposts를 사용하는 Amazon S3 on Outposts

Amazon EMR은 AWS에서 Apache Hadoop 및 Apache Spark와 같은 빅 데이터 프레임워크 실행을 단순화하여 방대한 양의 데이터를 처리하고 분석하는 관리형 클러스터 플랫폼입니다. 이러한 프레임워크 및 관련 오픈 소스 프로젝트를 사용하면 분석 목적 및 비즈니스 인텔리전스 워크로드용 데이터를 처

리할 수 있습니다. 또한 Amazon EMR은 대량의 데이터를 다른 AWS 데이터 스토어 및 데이터베이스로 변환하고 이동하는 데 도움이 되며, Amazon S3 on Outposts를 지원합니다. Amazon EMR에 대한 자세한 내용은 Amazon EMR 관리 안내서의 Amazon EMR on Outposts를 참조하세요.

Amazon S3 on Outposts의 경우, Amazon EMR은 버전 7.0.0에서 Apache Hadoop S3A 커넥터를 지원하기 시작했습니다. 이전 버전의 Amazon EMR은 로컬 S3 on Outposts를 지원하지 않으며 EMR 파일 시스템(EMRFS)은 지원되지 않습니다.

지원되는 애플리케이션

Amazon S3 on Outposts를 사용하는 Amazon EMR은 다음과 같은 애플리케이션을 지원합니다.

- Hadoop
- Spark
- Hue
- Hive
- Sqoop
- Pig
- Hudi
- Flink

자세한 내용은 Amazon EMR 릴리스 안내서를 참조하세요.

Amazon S3 on Outposts 버킷 생성 및 구성

Amazon EMR은 Amazon S3 on Outposts와 AWS SDK for Java를 사용하여 입력 데이터와 출력 데이터를 저장합니다. Amazon EMR 로그 파일은 사용자가 선택한 리전 Amazon S3 위치에 저장되며 Outpost에 로컬로 저장되지는 않습니다. 자세한 내용은 Amazon EMR 관리 안내서에서 Amazon EMR 로그를 참조하세요.

Amazon S3 및 DNS 요구 사항을 준수하기 위해 S3 on Outposts 버킷에는 이름 지정 제한 및 제한 사항이 있습니다. 자세한 내용은 S3 on Outposts 버킷 생성 단원을 참조하십시오.

Amazon EMR 버전 7.0.0 이상에서는 Amazon EMR을 S3 on Outposts 및 S3A 파일 시스템과 함께 사용할 수 있습니다.

사전 조건

S3 on Outposts 권한 - Amazon EMR 인스턴스 프로파일을 생성할 때 역할에 S3 on Outposts에 대한 AWS Identity and Access Management(IAM) 네임스페이스가 포함되어야 합니다. S3 on Outposts에는 자체 네임스페이스인 `s3-outposts*`가 있습니다. 이 네임스페이스를 사용하는 정책 예제는 S3 on Outposts로 IAM 설정 섹션을 참조하세요.

S3A 커넥터 - EMR 클러스터가 Amazon S3 on Outposts 버킷의 데이터에 액세스하도록 구성하려면 Apache Hadoop S3A 커넥터를 사용해야 합니다. 커넥터를 사용하려면 모든 S3 URI가 `s3a` 체계를 사용하는지 확인해야 합니다. 그렇지 않은 경우 S3 URI가 S3A 커넥터와 함께 작동하도록 EMR 클러스터에 사용하는 파일 시스템 구현을 구성할 수 있습니다.

S3A 커넥터와 함께 작동하도록 파일 시스템 구현을 구성하려면 *file_scheme*이 보유한 S3 URI 유형에 해당하는 EMR 클러스터의 `fs.`*file_scheme*`.impl` 및 `fs.AbstractFileSystem.`*file_scheme*`.impl` 구성 속성을 사용합니다. 다음 예시를 사용하려면 *user input placeholders*를 실제 정보로 대체하십시오. 예를 들어, 이 `s3` 체계를 사용하는 S3 URI의 파일 시스템 구현을 변경하려면 다음 클러스터 구성 속성을 지정합니다.

```
[
  {
"Classification": "core-site",
    "Properties": {
    "fs.s3.impl": "org.apache.hadoop.fs.s3a.S3AFileSystem",
    "fs.AbstractFileSystem.s3.impl": "org.apache.hadoop.fs.s3a.S3A"
    }
  }
]
```

S3A를 사용하려면 `fs.`*file_scheme*`.impl` 구성 속성을 `org.apache.hadoop.fs.s3a.S3AFileSystem`으로 설정하고 `fs.AbstractFileSystem.`*file_scheme*`.impl` 속성을 `org.apache.hadoop.fs.s3a.S3A`로 설정합니다.

예를 들어 경로 `s3a://`*bucket*`/...`에 액세스하는 경우 `fs.s3a.impl` 속성을 `org.apache.hadoop.fs.s3a.S3AFileSystem`으로 설정하고 `fs.AbstractFileSystem.s3a.impl` 속성을 `org.apache.hadoop.fs.s3a.S3A`로 설정합니다.

Amazon S3 on Outposts를 사용하여 Amazon EMR 사용 시작

다음 주제에서는 Amazon S3 on Outposts를 사용하여 Amazon EMR 사용을 시작하는 방법을 알아봅니다.

주제

- 권한 정책 생성
- 클러스터 생성 및 구성
- 구성 개요
- 고려 사항

권한 정책 생성

Amazon S3 on Outposts를 사용하는 EMR 클러스터를 생성하기 전에 클러스터의 Amazon EC2 인스턴스 프로파일에 연결할 IAM 정책을 생성해야 합니다. 정책에는 S3 on Outposts의 액세스 포인트 Amazon 리소스 이름(ARN)에 대한 액세스 권한이 있어야 합니다. S3 on Outposts용 IAM 정책을 생성하는 방법에 대한 자세한 내용은 S3 on Outposts로 IAM 설정 섹션을 참조하세요.

다음 정책 예제에서는 필요한 권한을 부여하는 방법을 보여줍니다. 정책을 생성한 후에는 the section called “클러스터 생성 및 구성” 섹션의 내용대로 정책을 EMR 클러스터를 생성하는 데 사용하는 인스턴스 프로파일 역할에 연결할 수 있습니다. 이 예제를 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
{
"Version":"2012-10-17",
  "Statement": [
        {
  "Effect": "Allow",
            "Resource": "arn:aws:s3-outposts:us-
west-2:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/access-point-name,
            "Action": [
                "s3-outposts:*"
            ]
        }
    ]

 }
```

클러스터 생성 및 구성

S3 on Outposts를 사용하여 Spark를 실행하는 클러스터를 생성하려면 콘솔에서 다음 단계를 완료하세요.

S3 on Outposts를 사용하여 Spark를 실행하는 클러스터를 만들려면

1. https://console.aws.amazon.com/elasticmapreduce/에서 Amazon EMR 콘솔을 엽니다.
2. 좌측 탐색 창에서 클러스터를 선택합니다.
3. 클러스터 생성을 선택합니다.
4. Amazon EMR 릴리스의 경우 emr-7.0.0 이상을 선택합니다.
5. 애플리케이션 번들의 경우 Spark 대화형을 선택합니다. 그런 다음 클러스터에 포함할 다른 지원 애플리케이션을 선택합니다.
6. Amazon S3 on Outposts를 활성화하려면 구성 설정을 입력합니다.

   샘플 구성 설정

   다음 샘플 구성 설정을 사용하려면 *user input placeholders*를 실제 정보로 대체합니다.

```
[
 {
   "Classification": "core-site",
   "Properties": {
     "fs.s3a.bucket.DOC-EXAMPLE-BUCKET.accesspoint.arn": "arn:aws:s3-outposts:us-
west-2:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/access-point-name"
     "fs.s3a.committer.name": "magic",
     "fs.s3a.select.enabled": "false"
    }
  },
  {
    "Classification": "hadoop-env",
    "Configurations": [
      {
        "Classification": "export",
        "Properties": {
          "JAVA_HOME": "/usr/lib/jvm/java-11-amazon-corretto.x86_64"
          }
       }
     ],
     "Properties": {}
   },
   {
     "Classification": "spark-env",
     "Configurations": [
       {
```

```
          "Classification": "export",
          "Properties": {
            "JAVA_HOME": "/usr/lib/jvm/java-11-amazon-corretto.x86_64"
          }
        }
      ],
      "Properties": {}
    },
    {
      "Classification": "spark-defaults",
      "Properties": {
        "spark.executorEnv.JAVA_HOME": "/usr/lib/jvm/java-11-amazon-
corretto.x86_64",
        "spark.sql.sources.fastS3PartitionDiscovery.enabled": "false"
      }
    }
  ]
```

7. 네트워킹 섹션에서 AWS Outposts 랙에 있는 Virtual Private Cloud(VPC) 및 서브넷을 선택합니다. Amazon EMR on Outposts에 대한 자세한 내용은 Amazon EMR 관리 안내서의 AWS Outposts의 EMR 클러스터를 참조하세요.
8. Amazon EMR용 EC2 인스턴스 프로필 섹션에서 이전에 생성한 권한 정책이 첨부되어 있는 IAM 역할을 선택합니다.
9. 나머지 클러스터 설정을 구성한 다음 클러스터 생성을 선택합니다.

구성 개요

다음 표에서는 Amazon EMR과 함께 S3 on Outposts를 사용하는 클러스터를 설정할 때 S3A 구성과 해당 파라미터에 지정할 값을 설명합니다.

파라미터	기본값	S3 on Outposts의 필수 값	설명
`fs.s3a.aws.credentials.provider`	지정하지 않으면 S3A는 Outposts 버킷 이름을 가진 리전 버킷에서 S3를 찾습니다.	S3 on Outposts 버킷의 액세스 포인트 ARN	Amazon S3 on Outposts는 Outposts 버킷에 액세스할 수 있는 유일한 수단으로 Virtual Private

파라미터	기본값	S3 on Outposts의 필수 값	설명
			Cloud(VPC) 전용 액세스 포인트를 지원합니다.
`fs.s3a.committer.name`	`file`	`magic`	매직 커미터는 S3 on Outposts에 대해 지원되는 유일한 커미터입니다.
`fs.s3a.select.enabled`	`TRUE`	`FALSE`	S3 Select는 Outposts에서 지원되지 않습니다.
`JAVA_HOME`	`/usr/lib/jvm/java-8`	`/usr/lib/jvm/java-11-amazon-corretto.x86_64`	S3A의 S3 on Outposts에는 Java 버전 11이 필요합니다.

다음 표에서는 Amazon EMR과 함께 S3 on Outposts를 사용하는 클러스터를 설정할 때 Spark 구성과 해당 파라미터에 지정할 값을 설명합니다.

파라미터	기본값	S3 on Outposts의 필수 값	설명
`spark.sql.sources.fastS3PartitionDiscovery.enabled`	`TRUE`	`FALSE`	S3 on Outposts는 빠른 파티션을 지원하지 않습니다.
`spark.executorEnv.JAVA_HOME`	`/usr/lib/jvm/java-8`	`/usr/lib/jvm/java-11-amazon-corretto.x86_64`	S3A의 S3 on Outposts에는 Java 버전 11이 필요합니다.

고려 사항

Amazon EMR을 S3 on Outposts 버킷과 통합하는 경우에는 다음 사항을 고려하세요.

- Amazon S3 on Outposts는 Amazon EMR 버전 7.0.0 이상에서 지원됩니다.
- S3 on Outposts를 Amazon EMR과 함께 사용하기 위해서는 S3A 커넥터가 있어야 합니다. S3 on Outposts 버킷과 상호 작용하는 데 필요한 기능은 S3A에만 있습니다. S3A 커넥터 설정 정보는 사전 조건을 참조하세요.
- Amazon S3 on Outposts는 Amazon EMR을 통해 Amazon S3 관리형 키(SSE-S3)를 사용한 서버 측 암호화만 지원합니다. 자세한 내용은 the section called “데이터 암호화” 단원을 참조하십시오.
- Amazon S3 on Outposts는 S3A FileOutputCommitter를 사용한 쓰기를 지원하지 않습니다. S3 on Outposts에서 S3A FileOutputCommitter를 사용하여 쓰면 invalidStorageClass: 지정한 스토리지 클래스가 유효하지 않습니다. 오류가 발생합니다.
- Amazon S3 on Outposts는 Amazon EMR Serverless 또는 Amazon EMR on EKS에서 지원되지 않습니다.
- Amazon EMR 로그는 사용자가 선택한 리전 Amazon S3 위치에 저장되며 S3 on Outposts 버킷에 로컬로 저장되지는 않습니다.

권한 부여 및 인증 캐시

S3 on Outposts는 인증 및 권한 부여 데이터를 Outposts 랙에 로컬로 안전하게 캐시합니다. 캐시는 요청이 있을 때마다 상위 AWS 리전으로의 왕복 이동을 제거합니다. 이렇게 하면 네트워크 왕복으로 인해 발생하는 변동성이 없어집니다. S3 on Outposts의 인증 및 권한 부여 캐시를 사용하면 Outposts와 AWS 리전 간의 연결 지연 시간에 구애받지 않고 일관된 지연 시간을 확보할 수 있습니다.

S3 on Outposts API 요청을 수행하면 인증 및 권한 부여 데이터가 안전하게 캐시됩니다. 그런 다음, 캐시된 데이터를 사용하여 후속 S3 객체 API 요청을 인증합니다. S3 on Outposts는 요청이 Signature Version 4A(SigV4A)를 사용하여 서명된 경우에만 인증 및 권한 부여 데이터를 캐시합니다. 캐시는 S3 on Outposts 서비스 내의 Outposts에 로컬로 저장됩니다. S3 API 요청을 하면 비동기적으로 새로 고쳐집니다. 캐시는 암호화되며 Outposts에는 일반 텍스트 암호화 키가 저장되지 않습니다.

Outpost가 AWS 리전에 연결된 경우 캐시는 최대 10분 동안 유효합니다. S3 on Outposts API 요청을 수행하면 최신 정책이 사용되도록 비동기적으로 새로 고쳐집니다. Outpost와 AWS 리전의 연결이 끊긴 경우 캐시는 최대 12시간 동안 유효합니다.

권한 부여 및 인증 캐시 구성

S3 on Outposts는 SigV4A 알고리즘으로 서명된 요청에 대한 인증 및 권한 부여 데이터를 자동으로 캐시합니다. 자세한 내용은 AWS Identity and Access Management 사용 설명서의 AWS API 요청에 서명을 참조하세요. SigV4A 알고리즘은 AWS SDK의 최신 버전에서 사용 가능합니다. AWS 공용 런타임 (CRT) 라이브러리의 종속성을 통해 얻을 수 있습니다.

최신 버전의 AWS SDK를 사용하고 최신 버전의 CRT를 설치해야 합니다. 예를 들어, `pip install awscrt`를 실행하여 Boto3로 최신 버전의 CRT를 얻을 수 있습니다.

S3 on Outposts는 SigV4 알고리즘으로 서명된 요청에 대한 인증 및 권한 부여 데이터를 캐시하지 않습니다.

SigV4A 서명 검증

요청을 SigV4A로 서명했는지 검증하는 데 AWS CloudTrail을 사용할 수 있습니다. S3 on Outposts를 위해 CloudTrail을 설정하는 방법에 대한 자세한 내용은 AWS CloudTrail 로그로 S3 on Outposts 모니터링 섹션을 참조하세요.

CloudTrail을 구성한 후에는 CloudTrail 로그의 `SignatureVersion` 필드에서 요청이 어떻게 서명되었는지 확인할 수 있습니다. SigV4A로 서명된 요청은 `SignatureVersion`이 `AWS4-ECDSA-P256-SHA256`으로 설정됩니다. SigV4로 서명된 요청은 `SignatureVersion`이 `AWS4-HMAC-SHA256`으로 설정됩니다.

S3 on Outposts의 보안

AWS에서 클라우드 보안은 가장 중요합니다. AWS 고객은 보안에 가장 보안에 민감한 조직의 요구 사항에 부합하도록 구축된 데이터 센터 및 네트워크 아키텍처의 혜택을 누릴 수 있습니다.

보안은 AWS와 사용자의 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드의 보안 - AWS는 AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호합니다. AWS는 또한 안전하게 사용할 수 있는 서비스를 제공합니다. 타사 감사자는 AWS규정 준수 프로그램의 일환으로 보안 효과를 정기적으로 테스트하고 검증합니다. Amazon S3 on Outposts에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 프로그램 제공 AWS 범위 내 서비스를 참조하세요.
- 클라우드 내 보안 – 사용자의 책임은 사용자가 사용하는 AWS 서비스에 의해 결정됩니다. 또한 귀하는 데이터의 민감도, 회사 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 S3 on Outposts 사용 시 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 주제는 보안 및 규정 준수 목표를 충족하도록 S3 on Outposts를 구성하는 방법을 보여줍니다. 또한 S3 on Outposts 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법을 알아봅니다.

주제

- S3 on Outposts로 IAM 설정
- S3 on Outposts의 데이터 암호화
- S3 on Outposts에 대한 AWS PrivateLink
- AWS Signature Version 4(SigV4) 인증별 정책 키
- Amazon S3 on Outposts용 AWS 관리형 정책
- Amazon S3 on Outposts에 대한 서비스 연결 역할 사용

S3 on Outposts로 IAM 설정

AWS Identity and Access Management(IAM)은 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어할 수 있도록 지원하는 AWS 서비스입니다. IAM 관리자는 어떤 사용자가 Amazon S3 on Outposts

리소스를 사용할 수 있도록 인증(로그인)되고 권한이 부여(권한 있음)될 수 있는지 제어합니다. IAM은 추가 비용 없이 사용할 수 있는 AWS 서비스입니다. 기본적으로 사용자는 S3 on Outposts 리소스 및 작업에 대한 권한이 없습니다. S3 on Outposts 리소스 및 API 작업에 대한 액세스 권한을 부여하려면 IAM을 사용하여 사용자, 그룹 또는 역할을 생성하고 권한을 연결할 수 있습니다.

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따릅니다.
- 보안 인증 공급자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 Create a role for a third-party identity provider (federation)의 지침을 따릅니다.
- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 Create a role for an IAM user의 지침을 따릅니다.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

IAM 자격 증명 기반 정책 외에도 S3 on Outposts는 버킷 정책과 액세스 포인트 정책을 모두 지원합니다. 버킷 정책 및 액세스 포인트 정책은 S3 on Outposts 리소스에 연결된 리소스 기반 정책입니다.

- 버킷 정책은 버킷에 연결되며 정책의 요소를 기반으로 버킷 및 버킷에 있는 객체에 대한 요청을 허용하거나 거부합니다.
- 반대로, 액세스 포인트 정책은 액세스 포인트에 연결되어 액세스 포인트에 대한 요청을 허용하거나 거부합니다.

액세스 포인트 정책은 기본 S3 on Outposts 버킷에 연결된 버킷 정책과 함께 작동합니다. 애플리케이션 또는 사용자가 S3 on Outposts 액세스 포인트를 통해 S3 on Outposts 버킷의 객체에 액세스하려면 액세스 포인트 정책과 버킷 정책 모두에서 요청을 허용해야 합니다.

액세스 포인트 정책에 포함시키는 제한은 해당 액세스 포인트를 통해 이루어진 요청에만 적용됩니다. 예를 들어 액세스 포인트가 버킷에 연결된 경우 액세스 포인트 정책을 사용하여 버킷에 직접 전송되는 요청을 허용하거나 거부할 수 없습니다. 단, 버킷 정책에 적용하는 제한 사항은 버킷에 직접 또는 액세스 포인트를 통해 이루어진 요청을 허용하거나 거부할 수 있습니다.

IAM 정책 또는 리소스 기반 정책에서 어떤 S3 on Outposts 작업을 허용 또는 거부할지 정의합니다. S3 on Outposts 작업은 특정 S3 on Outposts API 작업에 해당합니다. S3 on Outposts 작업은 `s3-outposts:` 네임스페이스 접두사를 사용합니다. AWS 리전의 S3 on Outposts 제어 API에 대한 요청과 Outpost의 객체 API 엔드포인트에 대한 요청은 IAM을 사용하여 인증되고 `s3-outposts:` 네임스페이스 접두사에 대해 권한이 부여됩니다. S3 on Outposts로 작업하려면 IAM 사용자를 구성하고 `s3-outposts:` IAM 네임스페이스에 대해 권한을 부여합니다.

자세한 내용은 서비스 권한 부여 참조에서 Amazon S3 on Outposts에 사용되는 작업, 리소스 및 조건 키를 참조하세요.

> **Note**
>
> - 액세스 제어 목록(ACL)은 S3 on Outposts에서 지원되지 않습니다.
> - 버킷의 소유자가 객체에 액세스하거나 객체를 삭제하지 못하는 일이 없도록 S3 on Outposts에서는 기본적으로 버킷 소유자가 객체 소유자로 지정됩니다.
> - 객체가 퍼블릭 액세스 권한을 가질 수 없도록 S3 on Outposts에서는 항상 S3 퍼블릭 액세스 차단이 활성화되어 있습니다.

S3 on Outposts에 대한 IAM 설정 방법을 자세히 알아보려면 다음 주제를 참조하세요.

주제

- S3 on Outposts 정책의 보안 주체
- S3 on Outposts의 리소스 ARN
- S3 on Outposts에 대한 정책 예제
- S3 on Outposts 엔드포인트에 대한 권한
- S3 on Outposts의 서비스 연결 역할

S3 on Outposts 정책의 보안 주체

S3 on Outposts 버킷에 대한 액세스 권한을 부여하는 리소스 기반 정책을 생성할 때, `Principal` 요소를 사용하여 해당 리소스에 대한 작업이나 작업을 요청할 수 있는 사람 또는 애플리케이션을 지정해야 합니다. S3 on Outposts 정책의 경우, 다음 보안 주체 중 하나를 사용할 수 있습니다.

- AWS 계정
- IAM 사용자

- IAM 역할
- 특정 IP 범위에 대한 액세스를 제한하는 `Condition` 요소를 사용하는 정책에서 와일드카드 문자(*)를 지정하여 모든 보안 주체

> **Important**
>
> 정책에 특정 IP 주소 범위에 대한 액세스를 제한하는 `Condition`도 포함하지 않는 한 `Principal` 요소에 와일드카드 문자(*)를 사용하는 Outposts 버킷의 S3에 대한 정책을 작성할 수 없습니다. 이 제한은 Outposts 버킷의 S3에 대한 공개 액세스가 없도록 하는 데 도움이 됩니다. 예시는 S3 on Outposts에 대한 정책 예제에서 확인하세요.

`Principal` 요소에 대한 자세한 내용은 IAM 사용 설명서의 AWS JSON 정책 요소: 보안 주체를 참조하세요.

S3 on Outposts의 리소스 ARN

S3 on Outposts에 대한 Amazon 리소스 이름(ARN)에는 Outpost가 위치한 AWS 리전 외에Outpost ID, AWS 계정 ID, 리소스 이름이 포함됩니다. Outposts 버킷 및 객체에 액세스하고 작업을 수행하려면 다음 테이블에 표시된 ARN 형식 중 하나를 사용해야 합니다.

ARN의 *partition* 값은 AWS 리전 그룹을 나타냅니다. 각 AWS 계정는 하나의 파티션으로 범위가 지정됩니다. 지원되는 파티션은 다음과 같습니다.

- `aws` – AWS 리전
- `aws-us-gov` - AWS GovCloud (US) 리전

다음 테이블에서는 S3 on Outposts ARN 형식을 보여줍니다.

Outposts 기반 Amazon S3 ARN	ARN 형식	예
버킷 ARN	`arn:`*partition*`:s3-outposts:`*region*`:`*account_id*`:outpost/`*outpost_id*`/bucket/`*bucket_name*	`arn:`*aws*`:s3-outposts:`*us-west-2*`:`*123456789012*`:outpost/`*op-01ac5d28a6a232904*`/`

Outposts 기반 Amazon S3 ARN	ARN 형식	예
		bucket/*amzn-s3-demo-bucket1*
액세스 포인트 ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost / *outpost_id* /accesspoint/ *accesspoint_name*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* : outpost/ *op-01ac5d28a6a232904* /accesspoint/ *access-point-name*
객체 ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost / *outpost_id* / bucket/*bucket_name* / object/*object_key*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* : outpost/ *op-01ac5d28a6a232904* / bucket/*amzn-s3-demo-bucket1* /object/*myobject*
S3 on Outposts 액세스 포인트 객체 ARN(정책에 사용)	arn:*partition* :s3-outposts: *region*: *account_id* :outpost / *outpost_id* /accesspoint/ *accesspoint_name* / object/*object_key*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* : outpost/ *op-01ac5d28a6a232904* /accesspoint/ *access-point-name/object/myobject*
Outposts 기반 S3 ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost / *outpost_id*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* : outpost/ *op-01ac5d28a6a232904*

S3 on Outposts에 대한 정책 예제

Example : AWS 계정 보안 주체가 있는 S3 on Outposts 버킷 정책

다음 버킷 정책은 AWS 계정 보안 주체를 사용하여 S3 on Outposts 버킷에 대한 액세스 권한을 부여합니다. 이 버킷 정책을 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
{
   "Version":"2012-10-17",
   "Id":"ExampleBucketPolicy1",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
         "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket"
      }
   ]
}
```

Example : 특정 IP 주소 범위에 대한 액세스를 제한하는 와일드카드 보안 주체(*) 및 조건 키가 있는 Outposts 버킷 정책의 S3

다음 버킷 정책은 특정 IP 범위에 대한 액세스를 제한하는 `aws:SourceIp` 조건과 함께 와일드카드 보안 주체(*)를 사용합니다. 이 버킷 정책을 사용하려면 *user input placeholders*를 사용자의 정보로 대체합니다.

```
{
    "Version": "2012-10-17",
    "Id": "ExampleBucketPolicy2",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": { "AWS" : "*" },
            "Action":"s3-outposts:*",
```

```
            "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket",
            "Condition" : {
                "IpAddress" : {
                    "aws:SourceIp": "192.0.2.0/24"
                },
                "NotIpAddress" : {
                    "aws:SourceIp": "198.51.100.0/24"
                }
            }
        }
    ]
}
```

S3 on Outposts 엔드포인트에 대한 권한

S3 on Outposts 엔드포인트 작업을 관리하기 위해 S3 on Outposts에는 IAM의 고유한 권한이 필요합니다.

> **Note**
>
> - 고객 소유 IP 주소 풀(CoIP 풀) 액세스 유형을 사용하는 엔드포인트의 경우 다음 테이블에 설명된 대로 CoIP 풀의 IP 주소로 작업할 수 있는 권한도 있어야 합니다.
> - AWS Resource Access Manager를 사용하여 S3 on Outposts에 액세스하는 공유 계정의 경우 공유 계정의 사용자는 공유 서브넷에서 자체 엔드포인트를 생성할 수 없습니다. 공유 계정의 사용자가 자체 엔드포인트를 관리하려는 경우 공유 계정은 Outpost에서 자체 서브넷을 생성해야 합니다. 자세한 내용은 the section called “S3 on Outposts 공유” 단원을 참조하십시오.

다음 테이블에서는 S3 on Outposts 엔드포인트 관련 IAM 권한을 보여줍니다.

작업	IAM 권한

`CreateEndpoint`	`s3-outposts:CreateEndpoint`
	
	`ec2:CreateNetworkInterface`

작업	IAM 권한

	`ec2:DescribeNetworkInterfaces`
	
	`ec2:DescribeVpcs`
	
	`ec2:DescribeSecurityGroups`
	
	`ec2:DescribeSubnets`
	
	`ec2:CreateTags`
	
	`iam:CreateServiceLinkedRole`
	
	온프레미스 고객 소유 IP 주소 풀(CoIP 풀) 액세스 유형을 사용하는 엔드포인트의 경우 다음과 같은 추가 권한이 필요합니다.
	
	`s3-outposts:CreateEndpoint`
	
	`ec2:DescribeCoipPools`
	
	`ec2:GetCoipPoolUsage`
	
	`ec2:AllocateAddress`
	
	`ec2:AssociateAddress`
	
	`ec2:DescribeAddresses`
	
	`ec2:DescribeLocalGatewayRouteTableVpcAssociations`

작업	IAM 권한
DeleteEndpoint	s3-outposts:DeleteEndpoint ec2:DeleteNetworkInterface ec2:DescribeNetworkInterfaces 온프레미스 고객 소유 IP 주소 풀(CoIP 풀) 액세스 유형을 사용하는 엔드포인트의 경우 다음과 같은 추가 권한이 필요합니다. s3-outposts:DeleteEndpoint ec2:DisassociateAddress ec2:DescribeAddresses ec2:ReleaseAddress
ListEndpoints	s3-outposts:ListEndpoints

> **Note**
> IAM 정책에서 리소스 태그를 사용하여 권한을 관리할 수 있습니다.

S3 on Outposts의 서비스 연결 역할

S3 on Outposts는 IAM 서비스 연결 역할을 사용하여 사용자를 대신해 일부 네트워크 리소스를 생성합니다. 자세한 내용은 Amazon S3 on Outposts에 대한 서비스 연결 역할 사용 단원을 참조하십시오.

S3 on Outposts의 데이터 암호화

기본적으로, Amazon S3 on Outposts에 저장된 모든 데이터는 Amazon S3 관리형 암호화 키를 통한 서버 측 암호화(SSE-S3)를 사용하여 암호화됩니다. 자세한 내용은 Amazon S3 사용 설명서의 Amazon S3 관리형 키를 사용한 서버 측 암호화(SSE-S3) 사용을 참조하세요.

원한다면 고객 제공 암호화 키(SSE-C)로 서버 측 암호화를 사용할 수도 있습니다. SSE-C를 사용하려면 객체 API 요청의 일부로 암호화 키를 지정합니다. 서버 측 암호화는 객체 메타데이터가 아닌 객체 데이터만 암호화합니다. 자세한 내용은 Amazon S3 사용 설명서의 Using server-side encryption with customer-provided keys를 참조하세요.

> **Note**
>
> S3 on Outposts는 AWS Key Management Service(AWS KMS) 키(SSE-KMS)를 사용한 서버 측 암호화를 지원하지 않습니다.

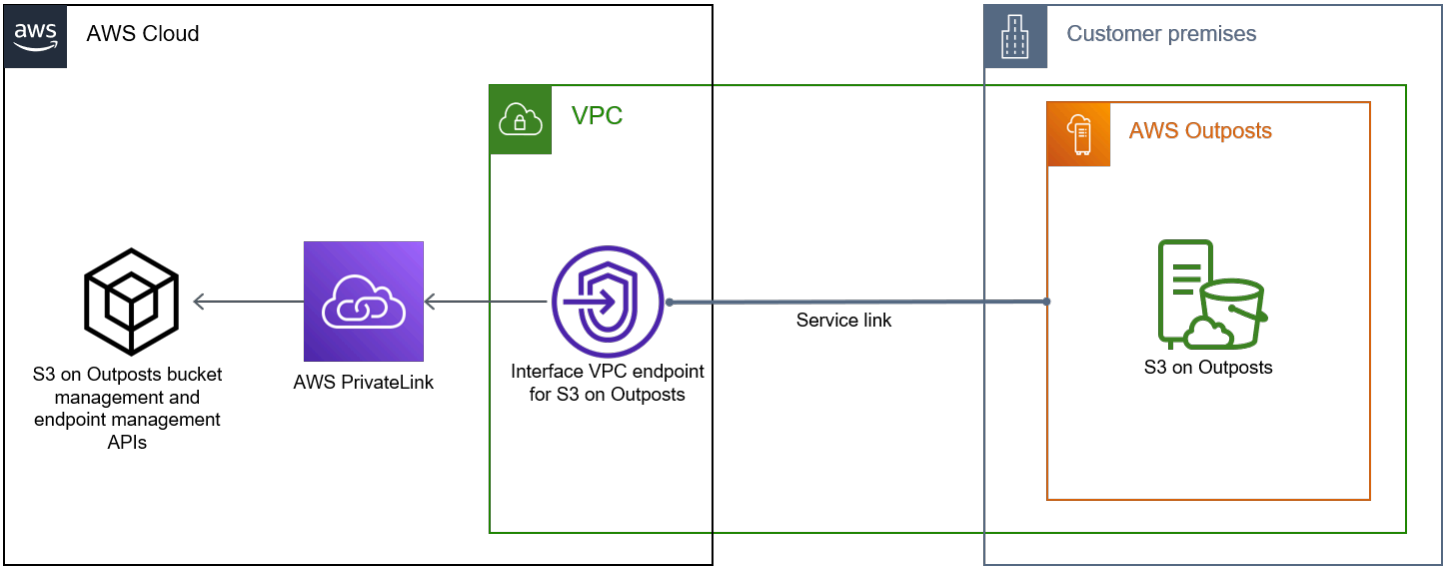
S3 on Outposts에 대한 AWS PrivateLink

S3 on Outposts는 가상 프라이빗 네트워크 내의 프라이빗 엔드포인트를 통해 S3 on Outposts 스토리지에 대한 직접 관리 액세스를 제공하는 AWS PrivateLink를 지원합니다. 이를 통해 내부 네트워크 아키텍처를 간소화하고 가상 프라이빗 클라우드(VPC)의 프라이빗 IP 주소를 사용하여 Outposts 객체 스토리지에 대한 관리 작업을 수행할 수 있습니다. AWS PrivateLink를 사용하면 퍼블릭 IP 주소나 프록시 서버를 사용할 필요가 없습니다.

Amazon S3 on Outposts에 대한 AWS PrivateLink를 사용하면 가상 프라이빗 클라우드(VPC)에서 인터페이스 VPC 엔드포인트를 프로비저닝하여 S3 on Outposts 버킷 관리 및 엔드포인트 관리 API에 액세스할 수 있습니다. 인터페이스 VPC 엔드포인트는 가상 프라이빗 네트워크(VPN) 또는 AWS Direct Connect를 통해 VPC 또는 온프레미스에 배포된 애플리케이션에서 직접 액세스할 수 있습니다. 버킷 및 엔드포인트 관리 API는 AWS PrivateLink를 통해 액세스할 수 있습니다. AWS PrivateLink는 GET, PUT 및 이와 유사한 API와 같은 데이터 전송 API 작업을 지원하지 않습니다. 이러한 작업은 이미 S3 on Outposts 엔드포인트 및 액세스 포인트 구성을 통해 비공개로 전송됩니다. 자세한 내용은 S3 on Outposts에 대한 네트워킹 단원을 참조하십시오.

인터페이스 엔드포인트는 VPC의 서브넷에서 프라이빗 IP 주소가 할당된 하나 이상의 탄력적 네트워크 인터페이스(ENI)로 표시됩니다. S3 on Outposts에 대한 엔드포인트 인터페이스 요청은 AWS 네트워크의 S3 on Outposts 버킷 및 엔드포인트 관리 API로 자동으로 라우팅됩니다. 또한, AWS Direct Connect 또는 AWS Virtual Private Network(AWS VPN)을 통해 온프레미스 애플리케이션에서 VPC의 인터페이스 엔드포인트에 액세스할 수 있습니다. VPC를 온프레미스 네트워크에 연결하는 방법에 대한 자세한 내용은 AWS Direct Connect 사용 설명서 및 AWS Site-to-Site VPN 사용 설명서를 참조하십시오.

인터페이스 엔드포인트는 다음 다이어그램과 같이 AWS 네트워크를 통해 그리고 AWS PrivateLink를 통해 S3 on Outposts 버킷 및 엔드포인트 관리 API에 대한 요청을 라우팅합니다.



인터페이스 엔드포인트에 대한 일반적인 정보는 AWS PrivateLink 가이드의 인터페이스 VPC 엔드포인트(AWS PrivateLink)를 참조하십시오.

주제

- 규제 및 제한
- S3 on Outposts 인터페이스 엔드포인트 액세스
- 온프레미스 DNS 구성 업데이트
- S3 on Outposts에 대한 VPC 엔드포인트 생성
- S3 on Outposts에 대한 버킷 정책 및 VPC 엔드포인트 정책 생성

규제 및 제한

AWS PrivateLink를 통해 S3 on Outposts 버킷 및 엔드포인트 관리 API에 액세스하는 경우 VPC 제한이 적용됩니다. 자세한 내용은 AWS PrivateLink 가이드의 인터페이스 엔드포인트 속성 및 제한 및 AWS PrivateLink 할당량을 참조하세요.

또한 AWS PrivateLink는 다음을 지원하지 않습니다.

- Federal Information Processing Standard(FIPS) 엔드포인트
- S3 on Outposts 데이터 전송 API(예: GET, PUT 및 유사한 객체 API 작업)
- 프라이빗 DNS

S3 on Outposts 인터페이스 엔드포인트 액세스

AWS PrivateLink를 사용하여 S3 on Outposts 버킷 및 엔드포인트 관리 API에 액세스하려면 엔드포인트별 DNS 이름을 사용하도록 애플리케이션을 업데이트해야 합니다. 인터페이스 엔드포인트를 생성할 때 AWS PrivateLink에서는 2가지 유형의 엔드포인트별 S3 on Outposts 이름인 리전별과 영역별을 생성합니다.

- 리전별 DNS 이름 - 고유한 VPC 엔드포인트 ID, 서비스 식별자, AWS 리전, `vpce.amazonaws.com`(예: *`vpce-1a2b3c4d-5e6f`*`.s3-outposts.us-east-1.vpce.amazonaws.com`)이 포함됩니다.
- 영역별 DNS 이름 - 고유한 VPC 엔드포인트 ID, 가용 영역, 서비스 식별자, AWS 리전, `vpce.amazonaws.com`(예: *`vpce-1a2b3c4d-5e6f`*`-`**`us-east-1a`**`.s3-outposts.us-east-1.vpce.amazonaws.com`)이 포함됩니다. 아키텍처가 가용 영역을 분리하는 경우 이 옵션을 사용할 수 있습니다. 예를 들어, 오류를 제한하거나 리전별 데이터 전송 비용을 줄이는 데 영역별 DNS 이름을 사용할 수 있습니다.

> **Important**
>
> S3 on Outposts 인터페이스 엔드포인트는 퍼블릭 DNS 도메인에서 확인됩니다. S3 on Outposts는 프라이빗 DNS를 지원하지 않습니다. 모든 버킷 및 엔드포인트 관리 API에 `--endpoint-url` 파라미터를 사용합니다.

AWS CLI 예제

`--region` 및 `--endpoint-url` 파라미터를 사용하여 S3 on Outposts 인터페이스 엔드포인트를 통해 버킷 관리 및 엔드포인트 관리 API에 액세스합니다.

Example : 엔드포인트 URL을 사용하여 S3 제어 API가 있는 버킷 나열

다음 예제에서 리전 *`us-east-1`*, VPC 엔드포인트 URL *`vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com`* 및 계정 ID *`111122223333`*을 해당하는 정보로 바꿉니다.

```
aws s3control list-regional-buckets --region us-east-1 --endpoint-url
 https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com --account-
id 111122223333
```

AWS SDK 예제

SDK를 최신 버전으로 업데이트하고, S3 on Outposts 인터페이스 엔드포인트에 대한 S3 제어 API에 액세스하기 위해 엔드포인트 URL을 사용하도록 클라이언트를 구성합니다.

SDK for Python (Boto3)

Example : 엔드포인트 URL을 사용하여 S3 제어 API에 액세스

다음 예제에서 리전 *us-east-1* 및 VPC 엔드포인트 URL *vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com*을 해당하는 정보로 바꿉니다.

```
control_client = session.client(
service_name='s3control',
region_name='us-east-1',
endpoint_url='https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com'
)
```

자세한 정보는 Boto3 개발자 안내서의 AWS PrivateLink for Amazon S3를 참조하세요.

SDK for Java 2.x

Example : 엔드포인트 URL을 사용하여 S3 제어 API에 액세스

다음 예제에서 VPC 엔드포인트 URL *vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com* 및 리전 *Region.US_EAST_1*을 해당하는 정보로 바꿉니다.

```
// control client
Region region = Region.US_EAST_1;
s3ControlClient = S3ControlClient.builder().region(region)

 .endpointOverride(URI.create("https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com"))
                                 .build()
```

자세한 내용을 알아보려면 AWS SDK for Java API 참조의 `S3ControlClient` 섹션을 참조하십시오.

온프레미스 DNS 구성 업데이트

엔드포인트별 DNS 이름을 사용하여 S3 on Outposts 버킷 관리 및 엔드포인트 관리 API용 인터페이스 엔드포인트에 액세스할 때는 온프레미스 DNS 확인자를 업데이트할 필요가 없습니다. 퍼블릭 S3 on

Outposts DNS 도메인에서 인터페이스 엔드포인트의 프라이빗 IP 주소로 엔드포인트별 DNS 이름을 확인할 수 있습니다.

S3 on Outposts에 대한 VPC 엔드포인트 생성

S3 on Outposts에 대한 VPC 인터페이스 엔드포인트를 생성하려면 AWS PrivateLink 설명서의 VPC 엔드포인트 생성을 참조하세요.

S3 on Outposts에 대한 버킷 정책 및 VPC 엔드포인트 정책 생성

S3 on Outposts에 대한 액세스를 제어하는 VPC 엔드포인트에 엔드포인트 정책을 연결할 수 있습니다. S3 on Outposts 버킷 정책의 `aws:sourceVpce` 조건을 사용하여 특정 VPC 엔드포인트의 특정 버킷에 대한 액세스를 제한할 수도 있습니다. VPC 엔드포인트 정책을 사용하면 S3 on Outposts 버킷 관리 API 및 엔드포인트 관리 API에 대한 액세스를 제어할 수 있습니다. 버킷 정책을 사용하면 S3 on Outposts 버킷 관리 API에 대한 액세스를 제어할 수 있습니다. 그러나 `aws:sourceVpce`를 사용하여 S3 on Outposts의 객체 작업에 대한 액세스를 관리할 수는 없습니다.

S3 on Outposts 액세스 정책은 다음과 같은 정보를 지정합니다.

- 이 작업을 허용하거나 거부하는 AWS Identity and Access Management(IAM) 보안 주체
- 허용되거나 거부되는 S3 제어 작업
- 작업이 허용되거나 거부되는 S3 on Outposts 리소스

다음 예에서는 버킷 또는 엔드포인트에 대한 액세스를 제한하는 정책을 보여줍니다. VPC 연결에 대한 자세한 내용은 AWS 백서, Amazon Virtual Private Cloud(VPC) 연결 옵션에서 네트워크와 VPC 연결 옵션을 참조하십시오.

> **Important**
>
> - 이 섹션에서 설명하는 VPC 엔드포인트에 대한 예제 정책을 적용할 경우 의도치 않게 버킷에 대한 액세스를 차단할 수 있습니다. VPC 엔드포인트에서 시작하는 연결에 대한 버킷 액세스를 제한하기 위한 버킷 권한은 버킷에 대한 모든 연결을 차단할 수 있습니다. 이 문제를 해결하는 방법에 대한 자세한 내용은 버킷 정책의 VPC 또는 VPC 엔드포인트 ID가 올바르지 않습니다.를 참조하십시오. 버킷에 액세스할 수 있도록 정책을 수정하는 방법은 무엇입니까?(지원 지식 센터)를 참조하십시오.
> - 다음 예제 버킷 정책을 사용하기 전에 VPC 엔드포인트 ID를 사용 사례에 적합한 값으로 바꾸세요. 그렇지 않으면 버킷에 액세스할 수 없습니다.

- 정책에서 특정 VPC 엔드포인트의 S3 on Outposts 버킷에 대한 액세스만 허용하는 경우 콘솔 요청이 지정된 VPC 엔드포인트에서 발생하지 않으므로 해당 버킷에 대한 콘솔 액세스가 비활성화됩니다.

주제

- 예제: VPC 엔드포인트에서 특정 버킷에 대한 액세스 제한
- 예: S3 on Outposts 버킷 정책의 특정 VPC 엔드포인트에서 액세스 거부

예제: VPC 엔드포인트에서 특정 버킷에 대한 액세스 제한

특정 S3 on Outposts 버킷에 대한 액세스만 제한하는 엔드포인트 정책을 생성할 수 있습니다. 다음 정책은 GetBucketPolicy 작업에 대한 액세스를 *example-outpost-bucket*으로만 제한합니다. 이 정책을 사용하려면 예제 값을 사용자의 값으로 바꾸어야 합니다.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1415115909151",
  "Statement": [
    { "Sid": "Access-to-specific-bucket-only",
      "Principal": {"AWS":"111122223333"},
      "Action": "s3-outposts:GetBucketPolicy",
      "Effect": "Allow",
      "Resource": "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outpost-
bucket"
    }
  ]
}
```

예: S3 on Outposts 버킷 정책의 특정 VPC 엔드포인트에서 액세스 거부

다음 S3 on Outposts 버킷 정책은 *vpce-1a2b3c4d* VPC 엔드포인트를 통해 *example-outpost-bucket* 버킷에서 GetBucketPolicy에 대한 액세스를 거부합니다.

`aws:sourceVpce` 조건은 엔드포인트를 지정하며, VPC 엔드포인트 리소스에 대한 Amazon 리소스 이름(ARN)을 필요로 하지 않고 엔드포인트 ID만 필요로 합니다. 이 정책을 사용하려면 예제 값을 사용자의 값으로 바꾸어야 합니다.

```
{
    "Version": "2012-10-17",
    "Id": "Policy1415115909152",
    "Statement": [
        {
            "Sid": "Deny-access-to-specific-VPCE",
            "Principal": {"AWS":"111122223333"},
            "Action": "s3-outposts:GetBucketPolicy",
            "Effect": "Deny",
            "Resource": "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outpost-
bucket",
            "Condition": {
                "StringEquals": {"aws:sourceVpce": "vpce-1a2b3c4d"}
            }
        }
    ]
}
```

AWS Signature Version 4(SigV4) 인증별 정책 키

다음 표는 Amazon S3 on Outposts와 함께 사용할 수 있는 AWS Signature Version 4(SigV4) 인증과 관련된 조건 키를 보여줍니다. 버킷 정책에서 이러한 조건을 추가하여 서명 버전 4를 사용하여 요청이 인증될 때 특정 동작을 적용할 수 있습니다. 예시 정책은 서명 버전 4 관련 조건 키를 사용하는 버킷 정책 예제 섹션을 참조하세요. 서명 버전 4를 사용한 요청 인증에 대한 자세한 내용은 Amazon Simple Storage Service API 참조의 요청 인증(AWS 서명 버전 4)을 참조하세요.

적용 가능한 키	설명

`s3-outposts:authType`	S3 on Outposts는 다양한 인증 방법을 지원합니다. 특정 인증 방법을 사용하도록 수신 요청을 제한하려면 이 선택적 조건 키를 사용할 수 있습니다. 예를 들어 이 조건 키를 사용하여 요청 인증에 HTTP `Authorization` 헤더만 사용하도록 허용할 수 있습니다.
	
	유효한 값:
	
	`REST-HEADER`
	
	`REST-QUERY-STRING`

적용 가능한 키	설명
`s3-outposts:signatureAge`	인증된 요청에서 서명이 유효한 시간(밀리초)입니다. 이 조건은 미리 서명된 URL에만 적용됩니다. Signature Version 4에서는 서명 키가 최대 7일간 유효합니다. 따라서 서명도 최대 7일간 유효합니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 서명 요청 소개를 참조하세요. 이 조건을 사용하여 서명 연령을 추가로 제한할 수 있습니다. 예제 값: `600000`
`s3-outposts:x-amz-content-sha256`	이 조건 키를 사용하여 버킷에서 서명되지 않은 콘텐츠를 허용하지 않을 수 있습니다. 서명 버전 4를 사용할 때 `Authorization` 헤더를 사용하는 요청에 대해 서명 계산에 `x-amz-content-sha256` 헤더를 추가한 다음 해당 값을 해시 페이로드로 설정합니다. 버킷 정책에서 이 조건 키를 사용하여 페이로드가 서명되지 않은 업로드를 거부할 수 있습니다. 예: • `Authorization` 헤더를 사용하여 요청을 인증하지만 페이로드에 서명하지 않는 업로드를 거부합니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 단일 청크로 페이로드 전송(Transferring payload in a single chunk)을 참조하세요. • 미리 서명된 URL을 사용하는 업로드를 거부합니다. 미리 서명된 URL에는 항상 `UNSIGNED_PAYLOAD` 가 있습니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 인증 요청 및 인증 방법을 참조하세요. 유효한 값: `UNSIGNED-PAYLOAD`

서명 버전 4 관련 조건 키를 사용하는 버킷 정책 예제

다음 예제를 사용하려면 *`user input placeholders`*를 사용자의 정보로 대체합니다.

Example : **s3-outposts:signatureAge**

다음 버킷 정책은 서명이 10분 이상 지난 경우 example-outpost-bucket의 객체에 대한 URL 요청이 사전 서명된 S3 on Outposts를 거부합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Deny a presigned URL request if the signature is more than 10
 minutes old",
            "Effect": "Deny",
            "Principal": {"AWS":"444455556666"},
            "Action": "s3-outposts:*",
            "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
            "Condition": {
                "NumericGreaterThan": {"s3-outposts:signatureAge": 600000},
                "StringEquals": {"s3-outposts:authType": "REST-QUERY-STRING"}
            }
        }
    ]
}
```

Example : **s3-outposts:authType**

다음 버킷 정책은 요청 인증에 Authorization 헤더를 사용하는 요청만 허용합니다. 미리 서명된 URL은 쿼리 매개변수를 사용하여 요청 및 인증 정보를 제공하기 때문에 미리 서명된 URL 요청은 거부됩니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 인증 방법을 참조하세요.

```
{
   "Version": "2012-10-17",
   "Statement": [
         {
               "Sid": "Allow only requests that use the Authorization header for
 request authentication. Deny presigned URL requests.",
               "Effect": "Deny",
               "Principal": {"AWS":"111122223333"},
               "Action": "s3-outposts:*",
               "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
```

```
                "Condition": {
                      "StringNotEquals": {
                            "s3-outposts:authType": "REST-HEADER"
                      }
                }
          }
    ]
}
```

Example : **`s3-outposts:x-amz-content-sha256`**

다음 버킷 정책은 사전 서명된 URL을 사용하는 업로드와 같이 서명되지 않은 페이로드가 있는 모든 업로드를 거부합니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 인증 요청 및 인증 방법을 참조하세요.

```
{
   "Version": "2012-10-17",
   "Statement": [
         {
                "Sid": "Deny uploads with unsigned payloads.",
                "Effect": "Deny",
                "Principal": {"AWS":"111122223333"},
                "Action": "s3-outposts:*",
                "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
                "Condition": {
                      "StringEquals": {
                            "s3-outposts:x-amz-content-sha256": "UNSIGNED-PAYLOAD"
                      }
                }
          }
    ]
}
```

Amazon S3 on Outposts용 AWS 관리형 정책

AWS 관리형 정책은 AWS에 의해 생성되고 관리되는 독립 실행형 정책입니다. AWS 관리형 정책은 사용자, 그룹 및 역할에 권한 할당을 시작할 수 있도록 많은 일반 사용 사례에 대한 권한을 제공하도록 설계되었습니다.

AWS 관리형 정책은 모든 AWS 고객이 사용할 수 있기 때문에 특정 사용 사례에 대해 최소 권한을 부여하지 않을 수 있습니다. 사용 사례에 고유한 고객 관리형 정책을 정의하여 권한을 줄이는 것이 좋습니다.

AWS 관리형 정책에서 정의한 권한은 변경할 수 없습니다. 만약 AWS가 AWS 관리형 정책에 정의된 권한을 업데이트할 경우 정책이 연결되어 있는 모든 보안 주체 엔터티(사용자, 그룹 및 역할)에도 업데이트가 적용됩니다. 새로운 AWS 서비스를 시작하거나 새로운 API 작업을 기존 서비스에 이용하는 경우 AWS가 AWS 관리형 정책을 업데이트할 가능성이 높습니다.

자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책을 참조하세요.

AWS 관리형 정책: AWSS3OnOutpostsServiceRolePolicy

서비스 연결 역할인 `AWSServiceRoleForS3OnOutposts`의 일부로서 네트워크 리소스를 관리하는 데 도움이 됩니다.

이 정책의 권한을 보려면 AWSS3OnOutpostsServiceRolePolicy를 참조하세요.

AWS 관리형 정책에 대한 S3 on Outposts 업데이트

이 서비스가 이러한 변경 내용을 추적하기 시작한 이후부터 S3 on Outposts의 AWS 관리형 정책 업데이트에 대한 세부 정보를 봅니다.

변경 사항	설명	날짜
S3 on Outposts에 `AWSS3OnOutpostsServiceRolePolicy` 추가	S3 on Outposts에 서비스 연결 역할 `AWSServiceRoleForS3OnOutposts` 의 일부로 `AWSS3OnOutpostsServiceRolePolicy` 가 추가되어 네트워크 리소스를 관리하는 데 도움이 됩니다.	2023년 10월 3일
S3 on Outposts에서 변경 사항 추적 시작	S3 on Outposts에서 AWS 관리형 정책에 대한 변경 사항 추적이 시작되었습니다.	2023년 10월 3일

Amazon S3 on Outposts에 대한 서비스 연결 역할 사용

Amazon S3 on Outposts는 AWS Identity and Access Management(IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 S3 on Outposts에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 S3 on Outposts에서 사전 정의하며 서비스에서 다른 AWS 서비스를 자동으로 호출하기 위해 필요한 모든 권한을 포함합니다.

필요한 권한을 수동으로 추가할 필요가 없으므로 서비스 연결 역할을 통해 S3 on Outposts를 더 쉽게 설정할 수 있습니다. S3 on Outposts에서 서비스 연결 역할 권한을 정의하므로, 달리 정의되지 않은 한 S3 on Outposts만 해당 역할을 맡을 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

먼저 관련 리소스를 삭제한 후에만 서비스 연결 역할을 삭제할 수 있습니다. 이렇게 하면 리소스에 대한 액세스 권한을 부주의로 삭제할 수 없기 때문에 S3 on Outposts 리소스가 보호됩니다.

서비스 연결 역할을 지원하는 기타 서비스에 대한 자세한 내용을 알아보려면 AWS IAM으로 작업하는 서비스를 참조하고 서비스 연결 역할 열에 예가 표시된 서비스를 찾으십시오. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예 링크를 선택합니다.

S3 on Outposts에 대한 서비스 연결 역할 권한

S3 on Outposts는 AWSServiceRoleForS3OnOutposts라는 서비스 연결 역할을 사용하여 네트워크 리소스를 관리하는 데 도움을 줍니다.

`AWSServiceRoleForS3OnOutposts` 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- `s3-outposts.amazonaws.com`

이름이 `AWSS3OnOutpostsServiceRolePolicy`인 역할 권한 정책은 S3 on Outposts가 지정된 리소스에서 다음 작업을 완료하도록 허용합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeNetworkInterfaces",
```

```
                "ec2:DescribeVpcs",
                "ec2:DescribeCoipPools",
                "ec2:GetCoipPoolUsage",
                "ec2:DescribeAddresses",
                "ec2:DescribeLocalGatewayRouteTableVpcAssociations"
            ],
            "Resource": "*",
            "Sid": "DescribeVpcResources"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:security-group/*"
            ],
            "Sid": "CreateNetworkInterface"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "CreateTagsForCreateNetworkInterface"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AllocateAddress"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:ipv4pool-ec2/*"
            ],
            "Sid": "AllocateIpAddress"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AllocateAddress"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:elastic-ip/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "CreateTagsForAllocateIpAddress"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:CreateNetworkInterfacePermission",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteNetworkInterfacePermission",
                "ec2:DisassociateAddress",
                "ec2:ReleaseAddress",
                "ec2:AssociateAddress"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "ReleaseVpcResources"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
```

```
                        "CreateNetworkInterface",
                        "AllocateAddress"
                    ],
                    "aws:RequestTag/CreatedBy": [
                        "S3 On Outposts"
                    ]
                }
            },
            "Sid": "CreateTags"
        }
    ]
}
```

IAM 엔터티(역할 등)가 서비스 연결 역할을 작성하고 편집하거나 삭제할 수 있도록 권한을 구성해야 합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하십시오.

S3 on Outposts에 대한 서비스 연결 역할 생성

서비스 링크 역할은 수동으로 생성할 필요가 없습니다. AWS Management Console, AWS CLI 또는 AWS API에서 S3 on Outposts 엔드포인트를 생성하면 S3 on Outposts에서 서비스 연결 역할이 자동으로 생성됩니다.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다. 또한 S3 on Outposts 엔드포인트를 생성하면 S3 on Outposts에서 서비스 연결 역할이 자동으로 생성됩니다.

또한 IAM 콘솔을 사용해 S3 on Outposts 사용 사례로 서비스 연결 역할을 생성할 수도 있습니다. AWS CLI 또는 AWS API에서 `s3-outposts.amazonaws.com` 서비스 이름의 서비스 연결 역할을 생성합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 생성을 참조하십시오. 이 서비스 연결 역할을 삭제하면 동일한 프로세스를 사용하여 역할을 다시 생성할 수 있습니다.

S3 on Outposts에 대한 서비스 연결 역할 편집

S3 on Outposts에서는 `AWSServiceRoleForS3OnOutposts` 서비스 연결 역할을 편집하도록 허용하지 않습니다. 다양한 엔터티티가 역할을 참조할 수 있기 때문에 역할 이름이 포함됩니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하십시오.

S3 on Outposts에 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 링크 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

> **Note**
>
> 리소스를 삭제하려 할 때 S3 on Outposts 서비스가 역할을 사용 중이면 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AWSServiceRoleForS3OnOutposts 역할에 사용된 S3 on Outposts 리소스를 삭제하려면

1. 모든 AWS 리전 전체의 AWS 계정에서 S3 on Outposts 엔드포인트 삭제.
2. IAM을 사용하여 서비스 연결 역할을 삭제합니다.

   IAM 콘솔, AWS CLI 또는 AWS API를 사용하여 `AWSServiceRoleForS3OnOutposts` 서비스 연결 역할을 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스에 연결 역할 삭제를 참조하십시오.

S3 on Outposts 서비스 연결 역할에 대해 지원되는 리전

S3 on Outposts는 서비스가 제공되는 모든 AWS 리전에서 서비스 연결 역할 사용을 지원합니다. 자세한 내용은 S3 on Outposts 리전 및 엔드포인트를 참조하세요.

S3 on Outposts 스토리지 관리

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

Amazon S3 on Outposts 스토리지 용량의 관리 및 공유에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

- S3 on Outposts 버킷의 S3 버전 관리에 대한 관리
- Amazon S3 on Outposts 버킷에 대한 수명 주기 구성 생성 및 관리
- S3 on Outposts에 대한 객체 복제
- AWS RAM 사용을 통해 S3 on Outposts 공유
- S3 on Outposts를 사용하는 다른 AWS 서비스

S3 on Outposts 버킷의 S3 버전 관리에 대한 관리

S3 버전 관리를 활성화하면 동일 버킷 내에 여러 개의 개별 객체 복제본을 저장합니다. S3 버전 관리를 사용하여 Outposts 버킷에 저장된 모든 버전의 객체를 모두 보존, 검색 및 복원할 수 있습니다. S3 버전 관리는 의도치 않은 사용자 작업 및 애플리케이션 장애로부터 복구하는 데 도움이 됩니다.

Amazon S3 on Outposts 버킷에는 다음과 같이 세 가지 버전 관리 상태가 있습니다.

- Unversioned(버전이 관리되지 않음) – 버킷에서 S3 버전 관리를 활성화하거나 일시 중지한 적이 없으면 버전이 지정되지 않고 S3 버전 관리 상태를 반환하지 않습니다. S3 버전 관리에 대한 자세한 내용은 S3 on Outposts 버킷의 S3 버전 관리에 대한 관리 섹션을 참조하십시오.
- Enabled(활성화됨) - 버킷의 객체에 대해 S3 버전 관리를 활성화합니다. 버킷에 추가된 모든 객체는 고유한 버전 ID를 받습니다. 버전 관리를 사용 설정할 때 버킷에 이미 존재하는 객체에는 `null`의 버

전 ID가 있습니다. 이러한(또는 다른) 객체를 PutObject와 같은 기타 작업으로 수정하는 경우 새 객체가 고유한 버전 ID를 가집니다.

- Suspended(일시 중지됨) - 버킷의 객체에 대해 S3 버전 관리를 일시 중지합니다. 버전 관리가 일시 중단된 후 버킷에 추가된 모든 객체는 버전 `null`을 수신합니다. 자세한 내용은 Amazon S3 사용 설명서의 버전 관리가 일시 중지된 버킷에 객체 추가를 참조하세요.

S3 on Outposts 버킷에 대해 S3 버전 관리를 활성화한 후에는 버전이 관리되지 않은 상태로 돌아갈 수 없습니다. 그러나 버전 관리를 일시 중지할 수는 있습니다. S3 버전 관리에 대한 자세한 내용은 S3 on Outposts 버킷의 S3 버전 관리에 대한 관리 섹션을 참조하십시오.

버킷의 각 객체에 대해 현재 버전과 0개 이상의 이전 버전이 있습니다. 스토리지 비용을 줄이려면 지정된 기간 후에 최신 버전이 아닌 버전을 만료하도록 버킷 S3 수명 주기 규칙을 구성할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts 버킷에 대한 수명 주기 구성 생성 및 관리 단원을 참조하십시오.

다음 예에서는 AWS Management Console 및 AWS Command Line Interface(AWS CLI)를 사용하여 기존 S3 on Outposts 버킷에 대한 버전 관리를 활성화하거나 일시 중지하는 방법을 보여줍니다. S3 버전 관리가 활성화된 S3 버킷을 생성하려면 S3 on Outposts 버킷 생성 섹션을 참조하세요.

> **Note**
>
> 버킷은 버킷을 생성하는 AWS 계정이 소유하며 이 계정은 버킷에 작업을 커밋할 수 있는 유일한 계정입니다. 버킷에는 Outpost, 태그, 기본 암호화, 액세스 포인트 설정 등의 구성 속성이 있습니다. 액세스 포인트 설정에는 버킷의 객체에 액세스하기 위한 Virtual Private Cloud(VPC) 및 액세스 포인트 정책 그리고 기타 메타데이터가 포함됩니다. 자세한 내용은 S3 on Outposts 사양 단원을 참조하십시오.

S3 콘솔 사용

버킷의 S3 버전 관리 설정을 편집하려면 다음과 같이 하세요.

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. S3 버전 관리를 활성화할 Outposts 버킷을 선택합니다.
4. 속성(Properties) 탭을 선택합니다.
5. 버킷 버전 관리(Bucket Versioning)에서 편집을 선택합니다.

6. 다음 옵션 중 하나를 선택하여 버킷의 S3 버전 관리 설정을 편집합니다.
   - S3 버전 관리를 일시 중지하고 새 객체 버전 생성을 중지하려면 Suspend(일시 중지)를 선택합니다.
   - S3 버전 관리를 활성화하고 객체별로 여러 개의 개별 복제본을 저장하려면 Enable(활성화)을 선택합니다.
7. Save changes(변경 사항 저장)를 선택합니다.

AWS CLI 사용

AWS CLI를 사용하여 버킷에 대한 S3 버전 관리를 활성화하거나 일시 중지하려면 다음 예제와 같이 `put-bucket-versioning` 명령을 사용합니다. 이러한 예제를 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다.

자세한 내용은 AWS CLI 참조의 put-bucket-versioning을 참조하세요.

Example : S3 버전 관리를 활성화하는 경우

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --versioning-configuration Status=Enabled
```

Example : S3 버전 관리를 일시 중지하는 경우

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --versioning-configuration Status=Suspended
```

Amazon S3 on Outposts 버킷에 대한 수명 주기 구성 생성 및 관리

S3 수명 주기를 사용하여 Amazon S3 on Outposts의 스토리지 용량을 최적화할 수 있습니다. 객체가 오래되거나 더 최신 버전으로 교체되면 객체를 만료시키도록 수명 주기 규칙을 만들 수 있습니다. 수명 주기 규칙을 생성, 사용, 사용 중지 또는 삭제할 수 있습니다.

S3 수명 주기에 대한 자세한 내용은 Amazon S3 on Outposts 버킷에 대한 수명 주기 구성 생성 및 관리 섹션을 참조하세요.

> **Note**
> 버킷은 버킷을 생성하는 AWS 계정이 소유하며 이 계정은 수명 주기 규칙을 생성, 사용, 사용 중지 또는 삭제할 수 있는 유일한 계정입니다.

S3 on Outposts 버킷에 대한 수명 주기 구성을 생성하고 관리하려면 다음 주제를 참조하세요.

주제

- AWS Management Console을 사용하여 수명 주기 규칙 생성 및 관리
- AWS CLI 및 Java용 SDK를 사용하여 수명 주기 구성 생성 및 관리

AWS Management Console을 사용하여 수명 주기 규칙 생성 및 관리

S3 수명 주기를 사용하여 Amazon S3 on Outposts의 스토리지 용량을 최적화할 수 있습니다. 객체가 오래되거나 더 최신 버전으로 교체되면 객체를 만료시키도록 수명 주기 규칙을 만들 수 있습니다. 수명 주기 규칙을 생성, 사용, 사용 중지 또는 삭제할 수 있습니다.

S3 수명 주기에 대한 자세한 내용은 Amazon S3 on Outposts 버킷에 대한 수명 주기 구성 생성 및 관리 섹션을 참조하세요.

> **Note**
> 버킷은 버킷을 생성하는 AWS 계정이 소유하며 이 계정은 수명 주기 규칙을 생성, 사용, 사용 중지 또는 삭제할 수 있는 유일한 계정입니다.

AWS Management Console을 사용하여 S3 on Outposts에 대한 수명 주기 규칙을 생성 및 관리하려면 다음 주제를 참조하세요.

주제

- 수명 주기 규칙 생성
- 수명 주기 규칙 사용
- 수명 주기 규칙 편집
- 수명 주기 규칙 삭제

수명 주기 규칙 생성

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 수명 주기 규칙을 생성하려는 Outposts 버킷을 선택합니다.
4. 관리(Management) 탭을 선택하고 수명 주기 규칙 생성(Create lifecycle rule)을 선택합니다.
5. Lifecycle rule name(수명 주기 규칙 이름)에 값을 입력합니다.
6. Rule scope(규칙 범위)에서 다음 옵션 중 하나를 선택합니다.
   - 특정 필터로 범위를 제한하려면 Limit the scope of this rule using one or more filters(하나 이상의 필터를 사용하여 이 규칙의 범위 제한)를 선택합니다. 그런 다음 접두사 필터, 태그 또는 객체 크기를 추가합니다.
   - 버킷의 모든 객체에 이 수명 주기 규칙을 적용하려면 Apply to all objects in the bucket(버킷의 모든 객체에 적용)을 선택합니다.
7. Lifecycle rule actions(수명 주기 규칙 작업)에서 다음 옵션 중 하나를 선택합니다.
   - Expire current versions of objects(객체의 현재 버전 만료) – 버전 관리가 활성화된 버킷의 경우 S3 on Outposts는 삭제 마커를 추가하고 객체를 이전 버전으로 유지합니다. S3 버전 관리를 사용하지 않는 버킷의 경우 S3 on Outposts는 객체를 영구적으로 삭제합니다.
   - Permanently delete noncurrent versions of objects(객체의 이전 버전 영구 삭제) – S3 on Outposts는 이전 버전의 객체를 영구적으로 삭제합니다.
   - Delete expired object delete markers or incomplete multipart uploads(만료된 객체 삭제 마커 또는 불완전한 멀티파트 업로드 삭제) — S3 on Outposts는 만료된 객체 삭제 마커 또는 불완전한 멀티파트 업로드를 영구적으로 삭제합니다.

     객체 태그를 사용하여 수명 주기 규칙의 범위를 제한하는 경우 Delete expired object delete markers(만료된 객체 삭제 마커 삭제)를 선택할 수 없습니다. Expire current object versions(객체의 현재 버전 만료)를 선택한 경우 Delete expired object delete markers(만료된 객체 삭제 마커 삭제)도 선택할 수 없습니다.

> **Note**
>
> 크기 기반 필터는 삭제 마커 및 불완전한 멀티파트 업로드와 함께 사용할 수 없습니다.

8. Expire current versions of objects(객체의 현재 버전 만료) 또는 Permanently delete noncurrent versions of objects(객체의 이전 버전 영구 삭제)를 선택한 경우 특정 날짜 또는 객체의 수명을 기반으로 규칙 트리거를 구성합니다.
9. Delete expired object delete markers(만료된 객체 삭제 마커 삭제)를 선택한 경우 만료된 객체 삭제 마커 삭제를 확인하려면 Delete expired object delete markers(만료된 객체 삭제 마커 삭제)를 선택합니다.
10. Timeline Summary(타임라인 요약)에서 수명 주기 규칙을 검토하고 Create rule(규칙 생성)을 선택합니다.

수명 주기 규칙 사용

버킷 수명 주기 규칙 사용 설정 또는 사용 중지

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 수명 주기 규칙을 사용 설정 또는 사용 중지할 Outposts 버킷을 선택합니다.
4. Management(관리) 탭을 선택한 다음 Lifecycle rule(수명 주기 규칙)에서 활성화하거나 비활성화할 규칙을 선택합니다.
5. 동작에 대해 규칙 사용 설정 또는 사용 중지를 선택합니다.

수명 주기 규칙 편집

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 수명 주기 규칙을 편집하려는 Outposts 버킷을 선택합니다.
4. 관리(Management) 탭을 선택하고 편집할 수명 주기 규칙을 선택합니다.
5. (선택 사항) Lifecycle rule name(수명 주기 규칙 이름) 값을 업데이트합니다.
6. Rule scope(규칙 범위)에서 필요에 따라 다음과 같이 범위를 편집합니다.
   - 특정 필터로 범위를 제한하려면 Limit the scope of this rule using one or more filters(하나 이상의 필터를 사용하여 이 규칙의 범위 제한)를 선택합니다. 그런 다음 접두사 필터, 태그 또는 객체 크기를 추가합니다.
   - 버킷의 모든 객체에 이 수명 주기 규칙을 적용하려면 Apply to all objects in the bucket(버킷의 모든 객체에 적용)을 선택합니다.

7. Lifecycle rule actions(수명 주기 규칙 작업)에서 다음 옵션 중 하나를 선택합니다.

   - Expire current versions of objects(객체의 현재 버전 만료) – 버전 관리가 활성화된 버킷의 경우 S3 on Outposts는 삭제 마커를 추가하고 객체를 이전 버전으로 유지합니다. S3 버전 관리를 사용하지 않는 버킷의 경우 S3 on Outposts는 객체를 영구적으로 삭제합니다.
   - Permanently delete noncurrent versions of objects(객체의 이전 버전 영구 삭제) – S3 on Outposts는 이전 버전의 객체를 영구적으로 삭제합니다.
   - Delete expired object delete markers or incomplete multipart uploads(만료된 객체 삭제 마커 또는 불완전한 멀티파트 업로드 삭제) — S3 on Outposts는 만료된 객체 삭제 마커 또는 불완전한 멀티파트 업로드를 영구적으로 삭제합니다.

     객체 태그를 사용하여 수명 주기 규칙의 범위를 제한하는 경우 Delete expired object delete markers(만료된 객체 삭제 마커 삭제)를 선택할 수 없습니다. Expire current object versions(객체의 현재 버전 만료)를 선택한 경우 Delete expired object delete markers(만료된 객체 삭제 마커 삭제)도 선택할 수 없습니다.

   > **Note**
   >
   > 크기 기반 필터는 삭제 마커 및 불완전한 멀티파트 업로드와 함께 사용할 수 없습니다.

8. Expire current versions of objects(객체의 현재 버전 만료) 또는 Permanently delete noncurrent versions of objects(객체의 이전 버전 영구 삭제)를 선택한 경우 특정 날짜 또는 객체 수명을 기반으로 규칙 트리거를 구성합니다.
9. Delete expired object delete markers(만료된 객체 삭제 마커 삭제)를 선택한 경우 만료된 객체 삭제 마커 삭제를 확인하려면 Delete expired object delete markers(만료된 객체 삭제 마커 삭제)를 선택합니다.
10. Save(저장)를 선택합니다.

수명 주기 규칙 삭제

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. 수명 주기 규칙을 삭제하려는 Outposts 버킷을 선택합니다.
4. Management(관리) 탭을 선택한 다음 Lifecycle rule(수명 주기 규칙)에서 삭제할 규칙을 선택합니다.

5. Delete(삭제)를 선택합니다.

AWS CLI 및 Java용 SDK를 사용하여 수명 주기 구성 생성 및 관리

S3 수명 주기를 사용하여 Amazon S3 on Outposts의 스토리지 용량을 최적화할 수 있습니다. 객체가 오래되거나 더 최신 버전으로 교체되면 객체를 만료시키도록 수명 주기 규칙을 만들 수 있습니다. 수명 주기 규칙을 생성, 사용, 사용 중지 또는 삭제할 수 있습니다.

S3 수명 주기에 대한 자세한 내용은 Amazon S3 on Outposts 버킷에 대한 수명 주기 구성 생성 및 관리 섹션을 참조하세요.

> **Note**
> 버킷은 버킷을 생성하는 AWS 계정이 소유하며 이 계정은 수명 주기 규칙을 생성, 사용, 사용 중지 또는 삭제할 수 있는 유일한 계정입니다.

AWS Command Line Interface(AWS CLI) 및 AWS SDK for Java를 사용하여 S3 on Outposts 버킷에 대한 수명 주기 구성을 생성 및 관리하려면 다음 예제를 참조하세요.

주제

- 수명 주기 구성 적용
- S3 on Outposts 버킷에 대한 수명 주기 구성 가져오기

수명 주기 구성 적용

AWS CLI

다음 AWS CLI 예제에서는 수명 주기 구성 정책을 Outpost 버킷에 적용합니다. 이 정책은 플래그가 지정된 접두사(*myprefix*)와 10일 후에 만료되는 태그가 포함된 모든 객체를 지정합니다. 이 예제를 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체합니다.

1. 수명 주기 구성 정책을 JSON 파일로 저장합니다. 이 예시에서 파일의 이름은 `lifecycle1.json`으로 지정됩니다.

```
{
    "Rules": [
        {
```

```
            "ID": "id-1",
            "Filter": {
                "And": {
                    "Prefix": "myprefix",
                    "Tags": [
                        {
                            "Value": "mytagvalue1",
                            "Key": "mytagkey1"
                        },
                        {
                            "Value": "mytagvalue2",
                            "Key": "mytagkey2"
                        }
                    ],
                    "ObjectSizeGreaterThan": 1000,
                    "ObjectSizeLessThan": 5000
                }
            },
            "Status": "Enabled",
            "Expiration": {
                "Days": 10
            }
        }
    ]
}
```

2. `put-bucket-lifecycle-configuration` CLI 명령의 일부로 JSON 파일을 제출합니다. 이 명령을 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 AWS CLI 참조의 put-bucket-lifecycle-configuration을 참조하세요.

```
aws s3control put-bucket-lifecycle-configuration --account-id 123456789012 --
bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/
bucket/example-outposts-bucket --lifecycle-configuration file://lifecycle1.json
```

SDK for Java

다음 SDK for Java 예제에서는 수명 주기 구성을 Outpost 버킷에 적용합니다. 이 수명 주기 구성은 플래그가 지정된 접두사(*`myprefix`*)와 10일 후에 만료되는 태그가 포함된 모든 객체를 지정합니다. 이 예제를 사용하려면 각 *`user input placeholder`*를 사용자의 정보로 대체합니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 PutBucketLifecycleConfiguration을 참조하세요.

```
import com.amazonaws.services.s3control.model.*;

public void putBucketLifecycleConfiguration(String bucketArn) {

    S3Tag tag1 = new S3Tag().withKey("mytagkey1").withValue("mytagkey1");
    S3Tag tag2 = new S3Tag().withKey("mytagkey2").withValue("mytagkey2");

    LifecycleRuleFilter lifecycleRuleFilter = new LifecycleRuleFilter()
            .withAnd(new LifecycleRuleAndOperator()
                    .withPrefix("myprefix")
                    .withTags(tag1, tag2))
                    .withObjectSizeGreaterThan(1000)
                    .withObjectSizeLessThan(5000);

    LifecycleExpiration lifecycleExpiration = new LifecycleExpiration()
            .withExpiredObjectDeleteMarker(false)
            .withDays(10);

    LifecycleRule lifecycleRule = new LifecycleRule()
            .withStatus("Enabled")
            .withFilter(lifecycleRuleFilter)
            .withExpiration(lifecycleExpiration)
            .withID("id-1");

    LifecycleConfiguration lifecycleConfiguration = new LifecycleConfiguration()
            .withRules(lifecycleRule);

    PutBucketLifecycleConfigurationRequest reqPutBucketLifecycle = new
 PutBucketLifecycleConfigurationRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withLifecycleConfiguration(lifecycleConfiguration);

    PutBucketLifecycleConfigurationResult respPutBucketLifecycle =
 s3ControlClient.putBucketLifecycleConfiguration(reqPutBucketLifecycle);
    System.out.printf("PutBucketLifecycleConfiguration Response: %s%n",
 respPutBucketLifecycle.toString());

}
```

S3 on Outposts 버킷에 대한 수명 주기 구성 가져오기

AWS CLI

다음 AWS CLI 예제에서는 수명 주기 구성 정책을 Outpost 버킷에 가져옵니다. 이 명령을 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체합니다. 이 명령에 대한 자세한 내용은 Amazon Simple Storage Service API 참조의 get-bucket-lifecycle-configuration을 참조하세요.

```
aws s3control get-bucket-lifecycle-configuration --account-id 123456789012 --bucket
 arn:aws:s3-outposts:<your-region>:123456789012:outpost/op-01ac5d28a6a232904/
bucket/example-outposts-bucket
```

SDK for Java

다음 SDK for Java 예제에서는 수명 주기 구성을 Outpost 버킷에 가져옵니다. 자세한 내용은 Amazon Simple Storage Service API 참조에서 GetBucketLifecycleConfiguration을 참조하세요.

```
import com.amazonaws.services.s3control.model.*;

public void getBucketLifecycleConfiguration(String bucketArn) {

    GetBucketLifecycleConfigurationRequest reqGetBucketLifecycle = new
 GetBucketLifecycleConfigurationRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    GetBucketLifecycleConfigurationResult respGetBucketLifecycle =
 s3ControlClient.getBucketLifecycleConfiguration(reqGetBucketLifecycle);
    System.out.printf("GetBucketLifecycleConfiguration Response: %s%n",
 respGetBucketLifecycle.toString());

}
```

S3 on Outposts에 대한 객체 복제

AWS Outposts에서 S3 복제를 사용하면 다른 Outposts 또는 동일한 Outpost에 있는 버킷 간에 S3 객체를 자동으로 복제하도록 Amazon S3 on Outposts를 구성할 수 있습니다. Outposts에서 S3 복제를 사용하면 데이터 상주 요구 사항을 충족하는 데 도움이 되도록 같거나 다른 Outposts 또는 여러 계정에 걸쳐 데이터의 여러 복제본을 유지할 수 있습니다. Outposts에서 S3 복제를 사용하면 규정을 준수하는 스토리지 요구 사항을 충족하고 계정 간의 데이터 공유를 강화할 수 있습니다. 복제본이 소스 데이터와

동일해야 한다면 Outposts에서 S3 복제를 사용하여 원래 객체 생성 시간, 태그, 버전 ID 등의 모든 메타데이터가 보존되는 객체의 복제본을 만들 수 있습니다.

또한 Outposts에서 S3 복제 기능은 버킷 간 객체 복제 상태를 모니터링하기 위한 상세한 지표 및 알림을 제공합니다. Amazon CloudWatch를 사용하여 복제 보류 중인 바이트, 복제 보류 중인 작업, 소스 및 대상 버킷 간의 복제 지연 시간을 추적하여 복제 진행 상황을 모니터링할 수 있습니다. 구성 문제를 신속하게 진단하고 수정하려면 복제 객체 실패에 대한 알림을 수신하도록 Amazon EventBridge를 설정할 수도 있습니다. 자세한 내용은 복제 관리을 참조하십시오.

주제

- 복제 구성
- Outposts에서 S3 복제 기능의 요구 사항
- 복제 가능한 객체
- 복제 불가능한 객체
- Outposts에서 S3 복제 기능이 지원하지 않는 것은 무엇인가요?
- 복제 설정
- 복제 관리

복제 구성

S3 on Outposts는 복제 구성을 XML로 저장합니다. 복제 구성 XML 파일에서 AWS Identity and Access Management(IAM) 규칙과 하나 이상의 규칙을 지정합니다.

```
<ReplicationConfiguration>
    <Role>IAM-role-ARN</Role>
    <Rule>
        ...
    </Rule>
    <Rule>
         ...
    </Rule>
     ...
</ReplicationConfiguration>
```

S3 on Outposts는 사용자가 부여한 권한 없이 객체를 복제할 수 없습니다. 복제 구성에서 지정한 IAM 역할을 사용하여 S3 on Outposts에 권한을 부여합니다. S3 on Outposts는 사용자를 대신하여 객체를

복제하기 위해 IAM 역할을 맡습니다. 복제를 시작하기 전에 IAM 역할에 필요한 권한을 부여해야 합니다. S3 on Outposts에 대한 이러한 권한과 관련한 자세한 내용은 IAM 역할 생성 섹션을 참조하세요.

다음 시나리오에서는 복제 구성에 한 가지 규칙을 추가합니다.

- 모든 객체를 복제하려는 경우
- 객체의 하위 집합 하나를 복제하려는 경우. 규칙에 필터를 추가하여 객체 하위 집합을 식별하는 경우 필터에서 객체 키 접두사, 태그 또는 이 두 가지를 모두 지정하여 객체에서 규칙이 적용될 하위 집합을 식별합니다.

다른 객체의 하위 집합을 복제하려면 복제 구성에 여러 규칙을 추가할 수 있습니다. 각 규칙에서 다른 객체 하위 집합을 선택하는 필터를 지정합니다. 예를 들어 `tax/` 또는`document/` 키 접두사를 갖는 객체를 복제하도록 선택할 수 있습니다. 이렇게 하려면 두 가지 규칙을 추가합니다. 하나는 `tax/` 키 접두사 필터를 지정하고 다른 하나는 `document/` 키 접두사를 지정합니다.

S3 on Outposts 복제 구성 및 복제 규칙에 대한 자세한 내용은 Amazon Simple Storage Service API 참조의 ReplicationConfiguration을 참조하세요.

Outposts에서 S3 복제 기능의 요구 사항

복제 요구 사항은 다음과 같습니다.

- 대상 Outpost CIDR 범위는 소스 Outpost 서브넷 테이블에 연결되어야 합니다. 자세한 내용은 복제 규칙 생성을 위한 사전 조건 단원을 참조하십시오.
- 소스 버킷과 대상 버킷 모두에서 S3 버전 관리를 활성화해야 합니다. 버전 관리에 대한 자세한 내용은 S3 on Outposts 버킷의 S3 버전 관리에 대한 관리 섹션을 참조하십시오.
- Amazon S3 on Outposts가 사용자를 대신해서 대상 버킷에 소스 버킷의 객체를 복제할 권한을 가지고 있어야 합니다. 즉, 사용자가 S3 on Outposts에 `GET` 및 `PUT` 권한을 위임하는 서비스 역할을 생성해야 합니다.
  1. 서비스 역할을 생성하기 전에 소스 버킷에 대한 `GET` 권한과 대상 버킷에 대한 `PUT` 권한이 있어야 합니다.
  2. S3 on Outposts에 권한을 위임하는 서비스 역할을 생성하려면 먼저 IAM 엔터티(사용자 또는 역할)가 `iam:CreateRole` 및 `iam:PassRole` 작업을 수행할 수 있도록 권한을 구성해야 합니다. 그런 다음, IAM 엔터티가 서비스 역할을 생성하도록 허용합니다. S3 on Outposts가 사용자를 대신하여 서비스 역할을 맡고 `GET` 및 `PUT` 권한을 S3 on Outposts에 위임하도록 하려면 필요한 신뢰 정책 및 권한 정책을 역할에 할당해야 합니다. S3 on Outposts에 대한 이러한 권한과 관련한 자

세한 내용은 IAM 역할 생성 섹션을 참조하세요. 서비스 역할 생성에 대한 자세한 내용은 서비스 역할 생성을 참조하세요.

복제 가능한 객체

기본적으로 S3 on Outposts는 다음을 복제합니다.

- 복제 구성을 추가한 후에 생성된 객체입니다.
- 원본 객체에서 복제본으로 가는 객체 메타데이터입니다. 복제본에서 소스 객체로 메타데이터를 복제하는 방법에 대한 자세한 내용은 Outposts에서 Amazon S3 복제본 수정 동기화가 활성화된 경우 복제 상태 섹션을 참조하세요.
- 객체 태그(있는 경우).

삭제 작업이 복제에 미치는 영향

원본 버킷에서 객체를 삭제하는 경우 기본적으로 다음 작업이 이루어집니다.

- 객체 버전 ID를 지정하지 않고 DELETE 요청을 수행하면 S3 on Outposts는 삭제 마커를 추가합니다. S3 on Outposts는 다음과 같이 삭제 마커를 처리합니다.
  - S3 on Outposts는 기본적으로 삭제 마커를 복제하지 않습니다.
  - 그러나 태그 기반이 아닌 규칙에도 삭제 마커 복제를 추가할 수 있습니다. 복제 구성에서 삭제 마커 복제를 활성화하는 방법에 대한 자세한 내용은 S3 콘솔 사용 섹션을 참조하세요.
- 사용자가 DELETE 요청에서 삭제할 객체 버전 ID를 지정할 경우, S3 on Outposts는 소스 버킷에서 해당 객체 버전을 영구적으로 삭제합니다. 하지만 대상 버킷에서는 삭제를 복제하지 않습니다. 즉, 대상 버킷에서 동일한 객체 버전을 삭제하지 않습니다. 이 동작은 악의적 삭제로부터 데이터를 보호합니다.

복제 불가능한 객체

기본적으로 S3 on Outposts는 다음을 복제하지 않습니다.

- 원본 버킷의 객체는 다른 복제 규칙에 따라 생성된 복제본입니다. 예를 들어, 버킷 A가 소스이고 버킷 B가 대상인 복제를 구성하는 경우. 이제 버킷 B가 원본이고 버킷 C가 대상인 다른 복제 구성을 추가한다고 가정합니다. 이 경우, 버킷 A 객체의 복제본인 버킷 B의 객체는 버킷 C로 복제되지 않습니다.

- 다른 대상에 이미 복제된 원본 버킷의 객체입니다. 예를 들어, 기존 복제 구성에서 대상 버킷을 변경하더라도 S3 on Outposts가 해당 객체를 다시 복제하지 않습니다.
- 고객 제공 암호화 키(SSE-C)를 통한 서버 측 암호화로 생성된 객체.
- 버킷 레벨 하위 리소스에 대한 업데이트.

  예를 들어, 수명 주기 구성을 변경하거나 원본 버킷에 알림 구성을 추가할 경우 이러한 변경 사항은 대상 버킷에 적용되지 않습니다. 이 기능을 사용하여 소스 버킷과 대상 버킷에 서로 다른 버킷 구성을 할 수 있습니다.
- 수명 주기 구성에 의해 수행되는 작업.

  예를 들어, 소스 버킷에서만 수명 주기 구성을 활성화하고 만료 작업을 구성하면 S3 on Outposts는 소스 버킷에서 만료된 객체에 대해 삭제 마커를 만들지만 그 마커를 대상 버킷에 복제하지는 않습니다. 소스 버킷과 대상 버킷에 동일한 수명 주기 구성을 적용하려는 경우 두 버킷 모두에서 동일한 수명 주기를 사용 설정합니다. 수명 주기 구성에 대한 자세한 내용은 Amazon S3 on Outposts 버킷에 대한 수명 주기 구성 생성 및 관리 섹션을 참조하십시오.

Outposts에서 S3 복제 기능이 지원하지 않는 것은 무엇인가요?

다음 S3 복제 기능은 현재 S3 on Outposts에서 지원되지 않습니다.

- S3 Replication Time Control(S3 RTC) Outposts에서 S3 복제의 객체 트래픽이 온프레미스 네트워크(로컬 게이트웨이)를 통해 이동하기 때문에 S3 RTC는 지원되지 않습니다. 로컬 게이트웨이에 대한 자세한 내용은 AWS Outposts 사용 설명서의 로컬 게이트웨이 작업을 참조하세요.
- 배치 작업을 위한 S3 복제.

복제 설정

> **Note**
>
> 복제를 설정하기 전에 버킷에 있던 객체는 자동으로 복제되지 않습니다. 즉, Amazon S3 on Outposts는 소급하여 객체를 복제하지 않습니다. 복제 구성 전에 생성된 객체를 복제하려면 `CopyObject` API 작업을 사용하여 동일한 버킷에 객체를 복사할 수 있습니다. 객체가 복사된 후에는 버킷에 '새' 객체로 나타나고 복제 구성이 해당 객체에 적용됩니다. 객체 복사에 대한 자세한 내용은 Amazon Simple Storage Service API 참조의 AWS SDK for Java를 사용하여 Amazon S3 on Outposts 버킷의 객체 복사 및 CopyObject 섹션을 참조하세요.

Outposts에서 S3 복제를 활성화하려면 소스 Outposts 버킷에 복제 규칙을 추가하세요. 복제 규칙은 지정된 대로 객체를 복제하도록 S3 on Outposts에 지시합니다. 복제 규칙에서는 다음을 제공해야 합니다.

- 소스 Outposts 버킷 액세스 포인트 - S3 on Outposts가 객체를 복사하도록 할 버킷의 액세스 포인트 Amazon 리소스 이름(ARN) 또는 액세스 포인트 별칭. 액세스 포인트 별칭 사용에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용을 참조하세요.
- 복제할 객체 - 소스 Outposts 버킷의 모든 객체 또는 하위 집합을 복제할 수 있습니다. 구성에 키 이름 접두사, 하나 이상의 객체 태그, 또는 둘 모두를 제공하여 하위 집합을 식별합니다.

  예를 들어, 키 이름 접두사 `Tax/`가 포함된 객체만 복제할 복제 규칙을 구성할 경우 S3 on Outposts는 `Tax/doc1` 또는 `Tax/doc2`와 같은 키가 있는 객체를 복제합니다. 그러나 `Legal/doc3` 키가 있는 객체는 복제하지 않습니다. 접두사와 하나 이상의 태그를 함께 지정할 경우 S3 on Outposts는 특정 키 접두사와 태그가 있는 객체만 복제합니다.
- 대상 Outposts 버킷 - S3 on Outposts가 객체를 복사하도록 할 버킷의 ARN 또는 액세스 포인트 별칭.

REST API, AWS SDK, AWS Command Line Interface(AWS CLI) 또는 Amazon S3 콘솔을 사용하여 복제 규칙을 구성할 수 있습니다.

또한 S3 on Outposts는 복제 규칙 설정을 지원하기 위해 API 작업을 제공합니다. 자세한 내용은 Amazon Simple Storage Service API 참조의 다음 주제들을 참조하십시오.

- PutBucketReplication
- GetBucketReplication
- DeleteBucketReplication

주제

- 복제 규칙 생성을 위한 사전 조건
- Outposts에서 복제 규칙 생성

복제 규칙 생성을 위한 사전 조건

주제

- 소스 및 대상 Outpost 서브넷 연결

- IAM 역할 생성

소스 및 대상 Outpost 서브넷 연결

복제 트래픽이 로컬 게이트웨이를 통해 소스 Outpost에서 대상 Outpost로 이동하도록 하려면 새 라우팅을 추가하여 네트워킹을 설정해야 합니다. 액세스 포인트의 Classless Inter-Domain Routing(CIDR) 네트워킹 범위를 함께 연결해야 합니다. 각 액세스 포인트 쌍에 대해 이 연결을 한 번만 설정하면 됩니다.

연결 설정을 위한 일부 단계는 액세스 포인트와 연결된 Outposts 엔드포인트의 액세스 유형에 따라 달라집니다. 엔드포인트의 액세스 유형은 프라이빗(AWS Outposts용 직접 Virtual Private Cloud(VPC) 라우팅) 또는 고객 소유 IP(온프레미스 네트워크 내의 고객 소유 IP 주소 풀(CoIP 풀))입니다.

1단계: 소스 Outposts 엔드포인트의 CIDR 범위 찾기

소스 액세스 포인트와 연결된 소스 엔드포인트의 CIDR 범위를 찾는 방법

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. Outposts 버킷 목록에서 복제할 소스 버킷을 선택합니다.
4. Outposts 액세스 포인트 탭을 선택하고 복제 규칙의 소스 버킷에 사용할 Outposts 액세스 포인트를 선택합니다.
5. Outposts 엔드포인트를 선택합니다.
6. 5단계에서 사용할 서브넷 ID를 복사합니다.
7. 소스 Outposts 엔드포인트의 CIDR 범위를 찾는 데 사용하는 방법은 엔드포인트의 액세스 유형에 따라 달라집니다.

   Outposts 엔드포인트 개요 섹션에서 액세스 유형을 참조하세요.

   - 액세스 유형이 프라이빗인 경우 6단계에서 사용할 Classless Inter-Domain Routing(CIDR) 값을 복사합니다.
   - 액세스 유형이 고객 소유 IP인 경우 다음을 수행합니다.
     1. 고객 소유 IPv4 풀 값을 복사하여 나중에 주소 풀의 ID로 사용합니다.
     2. https://console.aws.amazon.com/outposts/에서 AWS Outposts 콘솔을 엽니다.
     3. 탐색 창에서 로컬 게이트웨이 라우팅 테이블을 선택합니다.
     4. 소스 Outpost의 로컬 게이트웨이 라우팅 테이블 ID 값을 선택합니다.

5. 세부 정보 창에서 CoIP 풀 탭을 선택합니다. 이전에 복사한 CoIP 풀 ID의 값을 검색 상자에 붙여 넣습니다.
6. 일치하는 CoIP 풀의 경우 소스 Outposts 엔드포인트의 해당 CIDR 값을 복사하여 6단계에서 사용합니다.

2단계: 대상 Outposts 엔드포인트의 서브넷 ID 및 CIDR 범위 찾기

대상 액세스 포인트와 연결된 대상 엔드포인트의 서브넷 ID 및 CIDR 범위를 찾으려면 1단계의 하위 단계와 동일한 단계를 따르고 해당 하위 단계를 적용할 때 소스 Outposts 엔드포인트를 대상 Outposts 엔드포인트로 변경합니다. 6단계에서 사용할 대상 Outposts 엔드포인트의 서브넷 ID 값을 복사합니다. 5단계에서 사용할 대상 Outposts 엔드포인트의 CIDR 값을 복사합니다.

3단계: 소스 Outpost의 로컬 게이트웨이 ID 찾기

소스 Outpost의 로컬 게이트웨이 ID를 찾는 방법

1. https://console.aws.amazon.com/outposts/에서 AWS Outposts 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 로컬 게이트웨이를 선택합니다.
3. 로컬 게이트웨이 페이지에서 복제에 사용할 소스 Outpost의 Outpost ID를 찾습니다.
4. 5단계에서 사용할 소스 Outpost의 로컬 게이트웨이 ID 값을 복사합니다.

로컬 게이트웨이에 대한 자세한 내용은 AWS Outposts 사용 설명서의 로컬 게이트웨이를 참조하세요.

3단계: 대상 Outpost의 로컬 게이트웨이 ID 찾기

대상 Outpost의 로컬 게이트웨이 ID를 찾으려면 3단계의 하위 단계와 동일한 단계를 수행하되, 소스가 아닌 대상 Outpost의 Outpost ID를 찾습니다. 6단계에서 사용할 대상 Outpost의 로컬 게이트웨이 ID 값을 복사합니다.

5단계: 소스 Outpost 서브넷에서 대상 Outpost 서브넷으로의 연결 설정

소스 Outpost 서브넷에서 대상 Outpost 서브넷으로 연결하는 방법

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 서브넷을 선택합니다.
3. 검색 상자에 1단계에서 찾은 소스 Outposts 엔드포인트의 서브넷 ID를 입력합니다. 일치하는 서브넷 ID가 있는 서브넷을 선택합니다.

4. 일치하는 서브넷 항목에 대해 이 서브넷의 라우팅 테이블 값을 선택합니다.
5. 선택한 라우팅 테이블이 있는 페이지에서 작업을 선택한 다음 라우팅 편집을 선택합니다.
6. 라우팅 편집 페이지에서 라우팅 추가를 선택합니다.
7. 대상(Destination)에서 2단계에서 찾은 대상 Outposts 엔드포인트의 CIDR 범위를 입력합니다.
8. 대상(Target)에서 Outpost 로컬 게이트웨이를 선택하고 3단계에서 찾은 소스 Outpost의 로컬 게이트웨이 ID를 입력합니다.
9. Save changes(변경 사항 저장)를 선택합니다.
10. 라우팅의 상태가 활성이어야 합니다.

6단계: 대상 Outpost 서브넷에서 소스 Outpost 서브넷으로의 연결 설정

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 서브넷을 선택합니다.
3. 검색 상자에 2단계에서 찾은 대상 Outposts 엔드포인트의 서브넷 ID를 입력합니다. 일치하는 서브넷 ID가 있는 서브넷을 선택합니다.
4. 일치하는 서브넷 항목에 대해 이 서브넷의 라우팅 테이블 값을 선택합니다.
5. 선택한 라우팅 테이블이 있는 페이지에서 작업을 선택한 다음 라우팅 편집을 선택합니다.
6. 라우팅 편집 페이지에서 라우팅 추가를 선택합니다.
7. 대상(Destination)에서 1단계에서 찾은 소스 Outposts 엔드포인트의 CIDR 범위를 입력합니다.
8. 대상(Target)에서 Outpost 로컬 게이트웨이를 선택하고 4단계에서 찾은 대상 Outpost의 로컬 게이트웨이 ID를 입력합니다.
9. Save changes(변경 사항 저장)를 선택합니다.
10. 라우팅의 상태가 활성이어야 합니다.

소스 및 대상 액세스 포인트의 CIDR 네트워킹 범위를 연결한 후에는 AWS Identity and Access Management(IAM) 역할을 생성해야 합니다.

IAM 역할 생성

기본적으로 S3 on Outposts 리소스인 버킷, 객체 및 관련 하위 리소스는 모두 비공개이며 리소스 소유자만 리소스에 액세스할 수 있습니다. S3 on Outposts는 소스 Outposts 버킷에서 객체를 읽고 복제할 수 있는 권한이 필요합니다. IAM 서비스 역할을 생성하고 복제 구성에서 이 역할을 지정하여 이러한 권한을 부여합니다.

이 섹션에서는 신뢰 정책과 필요한 최소한의 권한 정책을 설명합니다. 연습 예제가 IAM 역할을 생성하는 단계별 지침을 제공합니다. 자세한 내용은 Outposts에서 복제 규칙 생성 단원을 참조하십시오. IAM 역할에 대한 자세한 내용은 IAM 사용 설명서의 IAM 역할 단원을 참조하세요.

- 다음 예시는 역할을 맡을 수 있는 서비스 보안 주체로서 S3 on Outposts를 식별하는 신뢰 정책을 보여줍니다.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3-outposts.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

- 다음 예제는 역할에 사용자 대신 복제 작업을 수행할 권한을 부여하는 액세스 정책을 보여줍니다. S3 on Outposts가 이 역할을 맡으면 이 정책에 지정된 권한을 보유하게 됩니다. 이 정책을 사용하려면 *user input placeholders*를 실제 정보로 대체합니다. 소스 및 대상 Outpost 버킷의 Outpost ID와 소스 및 대상 Outposts 버킷의 버킷 이름 및 액세스 포인트 이름으로 교체해야 합니다.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:GetObjectVersionForReplication",
            "s3-outposts:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
bucket/SOURCE-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
accesspoint/SOURCE-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
      },
      {
```

```
            "Effect":"Allow",
            "Action":[
               "s3-outposts:ReplicateObject",
               "s3-outposts:ReplicateDelete"
            ],
            "Resource":[
               "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/
bucket/DESTINATION-OUTPOSTS-BUCKET/object/*",
               "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/
accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
            ]
         }
      ]
}
```

액세스 정책은 다음 작업을 수행할 권한을 부여합니다.

- `s3-outposts:GetObjectVersionForReplication` - 모든 객체에 이 작업에 대한 권한이 부여되어 S3 on Outposts가 각 객체와 연결된 특정 객체 버전을 가져올 수 있습니다.

- `s3-outposts:GetObjectVersionTagging` - *SOURCE-OUTPOSTS-BUCKET* 버킷(소스 버킷)의 객체에 이 작업 권한이 있으면 S3 on Outposts가 복제를 위해 객체 태그를 읽을 수 있습니다. 자세한 내용은 S3 on Outposts 버킷에 대한 태그 추가 단원을 참조하십시오. 이러한 권한이 없을 경우 S3 on Outposts는 객체를 복제하지만 객체 태그는 복제하지 않습니다.
- `s3-outposts:ReplicateObject` 및 `s3-outposts:ReplicateDelete` - *DESTINATION-OUTPOSTS-BUCKET* 버킷(대상 버킷)의 모든 객체에 이 작업 권한이 있으면 S3 on Outposts가 대상 버킷에 객체 또는 삭제 마커를 복제할 수 있습니다. 삭제 마커에 대한 자세한 내용은 삭제 작업이 복제에 미치는 영향을 참조하십시오.

> **Note**
>
> - *DESTINATION-OUTPOSTS-BUCKET* 버킷(대상 버킷)에 `s3-outposts:ReplicateObject` 작업에 대한 권한이 있으면 객체 태그의 복제도 허용합니다. 따라서 `s3-outposts:ReplicateTags` 작업에 대한 권한을 명시적으로 부여할 필요가 없습니다.
> - 크로스 계정 복제의 경우 대상 Outposts 버킷의 소유자가 버킷 정책을 업데이트하여 *DESTINATION-OUTPOSTS-BUCKET*에 `s3-outposts:ReplicateObject` 작업에 대한 권한을 부여해야 합니다. `s3-outposts:ReplicateObject` 작업은 S3 on Outposts가 대상 Outposts 버킷에 객체 및 객체 태그를 복제하도록 허용합니다.

S3 on Outposts 작업 목록은 S3 on Outposts에서 정의한 작업을 참조하세요.

> **Important**
>
> IAM 역할을 소유한 AWS 계정은 해당 IAM 역할에 부여된 작업을 수행할 권한이 있어야 합니다.
>
> 예를 들어, 소스 Outposts 버킷에 다른 AWS 계정이 소유한 객체가 포함되어 있다고 가정하겠습니다. 객체 소유자는 버킷 정책 및 액세스 포인트 정책을 통해 IAM 역할을 소유한 AWS 계정에 필요한 권한을 명시적으로 부여해야 합니다. 그러지 않으면 S3 on Outposts는 객체에 액세스할 수 없으므로 객체의 복제가 실패합니다.
>
> 여기에서 설명하는 권한은 최소 복제 구성과 관련됩니다. 선택적 복제 구성을 추가하려면 S3 on Outposts에 추가 권한을 부여해야 합니다.

소스 및 대상 Outposts 버킷을 서로 다른 AWS 계정에서 소유할 경우 권한 부여

소스 및 대상 Outposts 버킷을 동일한 계정에서 소유하지 않는 경우, 대상 Outposts 버킷의 소유자가 대상 버킷에 대한 버킷과 액세스 포인트 정책을 업데이트해야 합니다. 이러한 정책은 다음 정책 예시와 같이 소스 Outposts 버킷의 소유자와 IAM 서비스 역할에 복제 작업을 수행할 수 있는 권한을 부여해야 합니다. 그러지 않으면 복제가 실패합니다. 이 정책에서는 *DESTINATION-OUTPOSTS-BUCKET*가 대상 버킷입니다. 이러한 정책 예시를 사용하려면 *user input placeholders*를 실제 정보로 대체하세요.

IAM 서비스 역할을 수동으로 생성하는 경우 다음 정책 예시에 나온 것처럼 역할 경로를 `role/service-role/`로 설정합니다. 자세한 내용을 알아보려면 IAM 사용 설명서의 IAM ARN을 참조하세요.

```
{
   "Version":"2012-10-17",
   "Id":"PolicyForDestinationBucket",
   "Statement":[
      {
         "Sid":"Permissions on objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action":[
```

```
            "s3-outposts:ReplicateDelete",
            "s3-outposts:ReplicateObject"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:DestinationBucket-account-
ID:outpost/DESTINATION-OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET/object/*"
         ]
      }

   ]
}
```

```
{
"Version":"2012-10-17",
   "Id":"PolicyForDestinationAccessPoint",
   "Statement":[
      {
         "Sid":"Permissions on objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action":[
            "s3-outposts:ReplicateDelete",
            "s3-outposts:ReplicateObject"
         ],
         "Resource" :[
            "arn:aws:s3-outposts:region:DestinationBucket-account-
ID:outpost/DESTINATION-OUTPOST-ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/
object/*"
         ]
      }
   ]
}
```

> **Note**
>
> 소스 Outposts 버킷에 있는 객체에 태그가 지정된 경우, 다음에 주의하세요.
> 소스 Outposts 버킷 소유자가 객체 태그를 복사할 수 있도록 `s3-outposts:GetObjectVersionTagging` 및 `s3-outposts:ReplicateTags` 작업 권한을

(IAM 역할을 통해) S3 on Outposts에 부여하는 경우, Amazon S3는 해당 객체와 함께 태그를 복제합니다. IAM 역할에 대한 상세 정보는 IAM 역할 생성 단원을 참조하십시오.

Outposts에서 복제 규칙 생성

Outposts에서의 S3 복제는 동일하거나 서로 다른 AWS Outposts의 버킷 간에 객체가 비동기식으로 자동 복사되는 것을 말합니다. 복제는 새로 생성된 객체 및 객체 업데이트를 소스 Outposts 버킷에서 대상 Outposts 버킷으로 복사합니다. 자세한 내용은 S3 on Outposts에 대한 객체 복제 단원을 참조하십시오.

> **Note**
>
> 복제를 설정하기 전에 소스 Outposts 버킷에 있던 객체는 자동으로 복제되지 않습니다. 즉, S3 on Outposts는 소급하여 객체를 복제하지 않습니다. 복제 구성 전에 생성된 객체를 복제하려면 `CopyObject` API 작업을 사용하여 동일한 버킷에 객체를 복사할 수 있습니다. 객체가 복사된 후에는 버킷에 '새' 객체로 나타나고 복제 구성이 해당 객체에 적용됩니다. 객체 복사에 대한 자세한 내용은 Amazon Simple Storage Service API 참조의 AWS SDK for Java를 사용하여 Amazon S3 on Outposts 버킷의 객체 복사 및 CopyObject 섹션을 참조하세요.

복제를 구성할 때는 소스 Outposts 버킷에 복제 규칙을 추가합니다. 복제 규칙은 복제할 소스 Outposts 버킷 객체와 복제된 객체가 저장될 대상 Outposts 버킷을 정의합니다. 특정 키 이름 접두사, 하나 이상의 객체 태그 또는 이 두 가지를 모두 포함하는 버킷 또는 객체 하위 집합에 있는 모든 객체를 복제하는 규칙을 작성할 수 있습니다. 대상 Outposts 버킷은 소스 Outposts 버킷과 동일한 Outpost에 있거나 다른 Outpost에 존재할 수 있습니다.

S3 on Outposts 복제 규칙의 경우 소스 및 대상 Outposts 버킷 이름 대신 소스 Outposts 버킷의 액세스 포인트 Amazon 리소스 이름(ARN)과 대상 Outposts 버킷의 액세스 포인트 ARN을 모두 제공해야 합니다.

삭제할 객체 버전 ID를 지정하는 경우, S3 on Outposts가 소스 Outposts 버킷에서 해당 객체 버전을 삭제합니다. 하지만 대상 Outposts 버킷에는 삭제를 복제하지 않습니다. 즉, 대상 Outposts 버킷에서 동일한 객체 버전을 삭제하지 않습니다. 이 동작은 악의적 삭제로부터 데이터를 보호합니다.

Outposts 버킷에 복제 규칙을 추가하면 이 규칙이 기본적으로 활성화되므로 이 규칙을 저장하면 그 즉시 적용됩니다.

이 예시에서는 소스 및 대상 Outposts 버킷이 서로 다른 Outposts에 존재하지만 동일한 AWS 계정에서 소유한 상황에서 복제를 설정합니다. Amazon S3 콘솔, AWS Command Line Interface(AWS CLI), AWS SDK for Java 및 AWS SDK for .NET을 사용하는 예제가 제공됩니다. Outposts에서의 크로스 계정 S3 복제 권한에 대한 자세한 내용은 소스 및 대상 Outposts 버킷을 서로 다른 AWS 계정에서 소유할 경우 권한 부여 섹션을 참조하세요.

S3 on Outposts 복제 규칙을 설정하기 위한 사전 요구 사항은 복제 규칙 생성을 위한 사전 조건 섹션을 참조하세요.

S3 콘솔 사용

대상 Amazon S3 on Outposts 버킷이 소스 Outposts 버킷과 다른 Outpost에 있는 경우 복제 규칙을 구성하려면 다음 단계를 따릅니다.

대상 Outposts 버킷이 소스 Outposts 버킷과 다른 계정에 있는 경우, 소스 Outposts 버킷 계정의 소유자에게 대상 Outposts 버킷의 객체를 복제할 수 있는 권한을 부여하려면 대상 Outposts 버킷에 하나의 버킷 정책을 추가해야 합니다.

복제 규칙을 만드는 방법

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 버킷 목록에서 소스 버킷으로 사용할 버킷의 이름을 선택합니다.
3. 관리 탭을 선택하고 복제 규칙 섹션까지 아래로 스크롤한 다음 복제 규칙 생성을 선택합니다.
4. 복제 규칙 이름에 나중에 규칙을 쉽게 식별할 수 있는 규칙 이름을 입력합니다. 이름은 필수 항목이며 버킷 내에서 고유해야 합니다.
5. 상태 아래의 활성화됨이 기본적으로 선택됩니다. 사용 설정된 규칙은 저장하는 즉시 적용되기 시작합니다. 나중에 규칙을 활성화하고 싶다면 비활성화됨을 선택하세요.
6. 우선 순위에서는 규칙의 우선 순위 값에 따라 중복되는 규칙이 있는 경우 적용할 규칙이 결정됩니다. 객체가 여러 복제 규칙의 범위에 포함된 경우 S3 on Outposts는 이 우선 순위 값을 사용하여 충돌을 피합니다. 기본적으로 새 규칙은 가장 높은 우선 순위로 복제 구성에 추가됩니다. 숫자가 클수록 우선 순위가 높아집니다.

   규칙의 우선 순위를 변경하려면 규칙을 저장한 후 복제 규칙 목록에서 규칙 이름을 선택하고 작업을 선택한 다음 우선 순위 편집을 선택합니다.
7. 소스 버킷에는 복제 소스를 설정하기 위한 다음과 같은 옵션이 있습니다.

   - 전체 버킷을 복제하려면 버킷의 모든 객체에 적용을 선택합니다.

- 복제 소스에 접두사 또는 태그 필터링을 적용하려면 하나 이상의 필터를 사용하여 이 규칙의 범위 제한을 선택합니다. 접두사와 태그를 결합할 수도 있습니다.
  - 동일한 접두사를 가진 모든 객체를 복제하려면 접두사에서 상자에 접두사를 입력합니다. 접두사 필터를 사용하면 이름이 동일한 문자열(예: `pictures`)로 시작하는 모든 객체로 복제가 제한됩니다.

    폴더의 이름에 해당하는 접두사를 입력할 경우, /(슬래시)를 마지막 문자로 사용해야 합니다(예: `pictures/`).
  - 하나 이상의 동일한 객체 태그가 있는 모든 객체를 복제하려면 태그 추가를 선택한 다음, 상자에 키 값 페어를 입력합니다. 다른 태그를 추가하려면 이 절차를 반복합니다. 객체 태그에 대한 자세한 내용은 S3 on Outposts 버킷에 대한 태그 추가 섹션을 참조하십시오.

8. 복제를 위해 S3 on Outposts 소스 버킷에 액세스하려면 소스 액세스 포인트 이름에서 소스 버킷에 연결된 액세스 포인트를 선택합니다.

9. 대상에서 S3 on Outposts가 객체를 복사하도록 할 Outposts 버킷킷의 액세스 포인트 ARN을 선택합니다. 대상 Outposts 버킷은 소스 Outposts 버킷과 동일하거나 다른 AWS 계정에 있을 수 있습니다.

   대상 버킷이 소스 Outposts 버킷과 다른 계정에 있는 경우, 소스 Outposts 버킷 계정의 소유자에게 대상 Outposts 버킷에 객체를 복제할 수 있는 권한을 부여하려면 대상 Outposts 버킷에 하나의 버킷 정책을 추가해야 합니다. 자세한 내용은 소스 및 대상 Outposts 버킷을 서로 다른 AWS 계정에서 소유할 경우 권한 부여 단원을 참조하십시오.

   > **Note**
   >
   > 대상 Outposts 버킷에서 버전 관리가 활성화되어 있지 않을 경우, 버전 관리 활성화 버튼이 포함된 경고가 나타납니다. 이 버튼을 선택하면 버킷의 버전 관리가 사용 설정됩니다.

10. S3 on Outposts가 사용자 대신 객체를 복제하기 위해 수임할 수 있는 AWS Identity and Access Management(IAM) 서비스 역할을 설정합니다.

    IAM 역할을 설정하려면 IAM 역할에서 다음 중 하나를 수행합니다.

    - S3 on Outposts에서 복제 구성을 위한 새 IAM 역할을 생성하도록 하려면 기존 IAM 역할에서 선택을 선택한 다음, 새 역할 생성을 선택합니다. 규칙을 저장하면 선택한 소스 및 대상 Outposts 버킷과 일치하는 IAM 역할에 대해 새 정책이 생성됩니다. 새 역할 생성을 선택하는 것이 좋습니다.

- 기존 IAM 역할을 사용하는 방법도 있습니다. 이 경우 복제에 필요한 권한을 S3 on Outposts에 부여하는 역할을 선택해야 합니다. 이 역할이 복제 규칙을 따를 수 있는 충분한 권한을 S3 on Outposts에 부여하지 않으면 복제가 실패합니다.

  기존 역할을 선택하려면 기존 IAM 역할에서 선택을 선택한 다음, 드롭다운 메뉴에서 역할을 선택합니다. IAM 역할 ARN 입력을 선택한 다음 IAM 역할의 Amazon 리소스 이름(ARN)을 입력할 수도 있습니다.

> **Important**
>
> S3 on Outposts 버킷에 복제 규칙을 추가할 때 S3 on Outposts에 복제 권한을 부여하는 IAM 역할을 만들어 전달할 수 있는 `iam:CreateRole` 및 `iam:PassRole` 권한이 있어야 합니다. 자세한 내용은 IAM 사용 설명서에서 사용자에게 AWS 서비스에 역할을 전달할 권한 부여를 참조하세요.

11. Outposts 버킷에 있는 모든 객체는 기본적으로 암호화됩니다. S3 on Outposts 암호화에 대한 자세한 내용은 S3 on Outposts의 데이터 암호화 섹션을 참조하세요. Amazon S3 관리형 키(SSE-S3)를 통한 서버 측 암호화를 사용하여 암호화된 객체만 복제할 수 있습니다. AWS Key Management Service(AWS KMS) 키(SSE-KMS)를 통한 서버 측 암호화 또는 고객 제공 암호화 키(SSE-C)를 통한 서버 측 암호화로 암호화된 객체에는 복제가 지원되지 않습니다.
12. 필요에 따라 복제 규칙 구성을 설정할 때 다음과 같은 추가 옵션을 활성화하세요.
    - 복제 구성에서 S3 on Outposts 복제 지표를 활성화하려면 복제 지표를 선택합니다. 자세한 내용은 복제 지표로 진행 상태 모니터링 단원을 참조하십시오.
    - 복제 구성에서 삭제 마커 복제를 사용 설정하려면 삭제 마커 복제(Delete marker replication)를 선택합니다. 자세한 내용은 삭제 작업이 복제에 미치는 영향 단원을 참조하십시오.
    - 복제본에 대한 메타데이터 변경 내용을 소스 객체에 다시 복제하려면 복제본 수정 동기화를 선택합니다. 자세한 내용은 Outposts에서 Amazon S3 복제본 수정 동기화가 활성화된 경우 복제 상태 단원을 참조하십시오.
13. 완료하려면 규칙 생성을 선택합니다.

규칙을 저장하면 규칙을 편집, 활성화, 비활성화하거나 삭제할 수 있습니다. 이렇게 하려면 소스 Outposts 버킷의 관리 탭으로 이동하여 아래로 스크롤하여 복제 규칙 섹션에서 규칙을 선택한 다음 규칙 편집을 선택합니다.

AWS CLI 사용

소스 및 대상 Outposts 버킷을 동일한 AWS 계정에서 소유한 경우 AWS CLI를 사용하여 복제를 설정하려면 다음을 수행합니다.

- 소스 및 대상 Outposts 버킷 생성
- 두 버킷의 버전 관리를 활성화합니다.
- S3 on Outposts에 객체 복제 권한을 제공하는 IAM 역할을 생성합니다.
- 소스 Outposts 버킷에 복제 구성을 추가합니다.

설정을 확인하려면 테스트합니다.

소스 및 대상 Outposts 버킷을 동일한 AWS 계정에서 소유한 경우 복제를 설정하는 방법

1. AWS CLI의 자격 증명 프로필을 설정합니다. 이 예제에서는 프로필 이름 `acctA`를 사용합니다. 보안 인증 프로파일 설정에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 명명된 프로파일을 참조하세요.

   > **Important**
   > 이 연습에 사용하는 프로파일에 필요한 권한이 있어야 합니다. 예를 들어 복제 구성에서 S3 on Outposts가 맡을 수 있는 IAM 서비스 역할을 지정합니다. 사용하는 프로필에 `iam:CreateRole` 및 `iam:PassRole` 권한이 있을 경우에만 이 작업을 수행할 수 있습니다. 자세한 내용은 IAM 사용 설명서에서 사용자에게 AWS 서비스에 역할을 전달할 권한 부여를 참조하세요. 관리자 보안 인증 정보를 사용하여 명명된 프로파일을 생성할 경우 명명된 프로파일에 모든 작업을 수행하는 데 필요한 권한이 부여됩니다.

2. 원본 버킷을 생성하고 버킷에서 버전 관리를 사용 설정합니다. 다음 `create-bucket` 명령은 미국 동부(버지니아 북부)(`us-east-1`) 리전에 *`SOURCE-OUTPOSTS-BUCKET`* 버킷을 생성합니다. 이 명령을 사용하려면 *`user input placeholders`*를 실제 정보로 대체합니다.

   ```
   aws s3control create-bucket --bucket SOURCE-OUTPOSTS-BUCKET --outpost-id SOURCE-
   OUTPOST-ID --profile acctA --region us-east-1
   ```

   다음 `put-bucket-versioning` 명령은 *`SOURCE-OUTPOSTS-BUCKET`* 버킷의 버전 관리를 활성화합니다. 이 명령을 사용하려면 *`user input placeholders`*를 실제 정보로 대체합니다.

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/bucket/SOURCE-OUTPOSTS-
BUCKET --versioning-configuration Status=Enabled --profile acctA
```

3. 대상 버킷을 생성하고 버킷에서 버전 관리를 사용 설정합니다. 다음 `create-bucket` 명령은 미국 서부(오레곤)(`us-west-2`) 리전에 *`DESTINATION-OUTPOSTS-BUCKET`* 버킷을 생성합니다. 이 명령을 사용하려면 *`user input placeholders`*를 실제 정보로 대체합니다.

> **Note**
>
> 소스 및 대상 Outposts 버킷이 모두 동일한 AWS 계정에 있을 때 복제 구성을 설정하려면 동일한 명명된 프로파일을 사용합니다. 이 예제에서는 `acctA`를 사용합니다. 두 버킷을 서로 다른 AWS 계정에서 소유한 경우의 복제 구성을 테스트하려면 버킷마다 각각 다른 프로파일을 지정합니다.

```
aws s3control create-bucket --bucket DESTINATION-OUTPOSTS-BUCKET --create-bucket-
configuration LocationConstraint=us-west-2 --outpost-id DESTINATION-OUTPOST-ID --
profile acctA --region us-west-2
```

다음 `put-bucket-versioning` 명령은 *`DESTINATION-OUTPOSTS-BUCKET`* 버킷의 버전 관리를 활성화합니다. 이 명령을 사용하려면 *`user input placeholders`*를 실제 정보로 대체합니다.

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/bucket/DESTINATION-
OUTPOSTS-BUCKET --versioning-configuration Status=Enabled --profile acctA
```

4. IAM 서비스 역할을 생성합니다. 복제 구성 과정에서 이 서비스 역할을 *`SOURCE-OUTPOSTS-BUCKET`* 버킷에 추가하게 됩니다. S3 on Outposts는 사용자를 대신하여 객체를 복제하기 위해 이 역할을 맡습니다. IAM 역할은 다음의 두 단계로 생성합니다.

   a. IAM 역할을 생성합니다.

      i. 다음 신뢰 정책을 복사하여 로컬 컴퓨터의 현재 디렉터리에 `s3-on-outposts-role-trust-policy.json`이라는 이름의 파일로 저장합니다. 이 정책은 서비스 역할을 맡을 권한을 S3 on Outposts 서비스 보안 주체에 부여합니다.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3-outposts.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

ii. 다음 명령을 실행해 역할을 생성합니다. *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws iam create-role --role-name replicationRole --assume-role-policy-
document file://s3-on-outposts-role-trust-policy.json --profile acctA
```

b. 서비스 역할에 권한 정책을 연결합니다.

i. 다음 권한 정책을 복사하여 로컬 컴퓨터의 현재 디렉터리에 `s3-on-outposts-role-permissions-policy.json` 파일로 저장합니다. 이 정책은 다양한 S3 on Outposts 버킷 및 객체 작업에 대한 권한을 부여합니다. 이 정책을 사용하려면 *user input placeholders*를 실제 정보로 대체합니다.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:GetObjectVersionForReplication",
            "s3-outposts:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-
OUTPOST-ID/bucket/SOURCE-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-
OUTPOST-ID/accesspoint/SOURCE-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
```

```
        },
        {
           "Effect":"Allow",
           "Action":[
              "s3-outposts:ReplicateObject",
              "s3-outposts:ReplicateDelete"
           ],
           "Resource":[
              "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-
OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET/object/*",
              "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-
OUTPOST-ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
           ]
        }
    ]
}
```

ii. 다음 명령을 실행하여 정책을 생성하고 이를 역할에 연결합니다. *user input placeholders*를 사용자의 정보로 대체합니다.

```
aws iam put-role-policy --role-name replicationRole --policy-
document file://s3-on-outposts-role-permissions-policy.json --policy-
name replicationRolePolicy --profile acctA
```

5. *SOURCE-OUTPOSTS-BUCKET* 버킷에 복제 구성을 추가합니다.

   a. S3 on Outposts API는 XML 형식의 복제 구성을 요구하지만 AWS CLI는 JSON으로 지정된 복제 구성을 요구합니다. 다음 JSON을 로컬 컴퓨터의 현재 디렉터리에 `replication.json` 파일로 저장합니다. 이 구성을 사용하려면 *user input placeholders*를 실제 정보로 대체합니다.

```
{
  "Role": "IAM-role-ARN",
  "Rules": [
    {
      "Status": "Enabled",
      "Priority": 1,
      "DeleteMarkerReplication": { "Status": "Disabled" },
      "Filter" : { "Prefix": "Tax"},
      "Destination": {
        "Bucket":
        "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-
ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT"
```

```
      }
    }
  ]
}
```

b. 다음 `put-bucket-replication` 명령을 실행하여 소스 Outposts 버킷에 복제 구성을 추가합니다. 이 명령을 사용하려면 *`user input placeholders`*를 실제 정보로 대체합니다.

```
aws s3control put-bucket-replication --account-id 123456789012 --
bucket arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-
ID/bucket/SOURCE-OUTPOSTS-BUCKET --replication-configuration file://
replication.json --profile acctA
```

c. 복제 구성을 검색하려면 `get-bucket-replication` 명령을 사용합니다. 이 명령을 사용하려면 *`user input placeholders`*를 실제 정보로 대체합니다.

```
aws s3control get-bucket-replication --account-id 123456789012 --bucket
 arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
bucket/SOURCE-OUTPOSTS-BUCKET --profile acctA
```

6. Amazon S3 콘솔에서 다음과 같이 설정을 테스트합니다.

a. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.

b. *`SOURCE-OUTPOSTS-BUCKET`* 버킷에서 이름이 `Tax`인 폴더를 생성합니다.

c. *`SOURCE-OUTPOSTS-BUCKET`* 버킷의 `Tax` 폴더에 샘플 객체를 추가합니다.

d. *`DESTINATION-OUTPOSTS-BUCKET`* 버킷에서 다음을 확인합니다.

- S3 on Outposts가 객체를 복제했습니다.

> **Note**
>
> S3 on Outposts가 객체를 복제하는 데 걸리는 시간은 객체 크기에 따라 다릅니다. 복제 상태를 확인하는 방법에 대한 자세한 내용은 복제 상태 정보 가져오기 단원을 참조하십시오.

- 객체의 속성에서 복제 상태가 복제본으로 설정됩니다(이 객체를 복제본 객체로 식별함).

복제 관리

이 섹션에서는 S3 on Outposts에서 사용할 수 있는 추가 복제 구성 옵션, 복제 상태 확인 방법 및 복제 문제 해결 방법을 설명합니다. 핵심 복제 구성에 대한 자세한 내용은 복제 설정단원을 참조하십시오.

주제

- 복제 지표로 진행 상태 모니터링
- 복제 상태 정보 가져오기
- 복제 문제 해결
- Outposts에서 S3 복제 기능에 EventBridge 사용

복제 지표로 진행 상태 모니터링

Outposts에서의 S3 복제는 복제 구성의 복제 규칙에 대한 세부 지표를 제공합니다. 복제 지표를 통해 복제 보류 중인 바이트, 복제 지연, 보류 중인 작업을 추적하여 5분 간격으로 복제 진행 상황을 모니터링할 수 있습니다. 구성 문제 해결에 도움이 되도록 Amazon EventBridge를 설정하여 복제 실패에 대한 알림을 수신할 수 있습니다.

복제 지표가 활성화되면 Outposts에서의 S3 복제가 다음 지표를 Amazon CloudWatch에 게시합니다.

- 복제 보류 중인 바이트 - 지정된 복제 규칙에 대해 복제 보류 중인 객체의 총 바이트 수입니다.
- 복제 지연 시간 - 지정된 복제 규칙에 대해 복제 대상 버킷이 소스 버킷보다 늦어지는 최대 시간(초)입니다.
- 복제 보류 중인 작업 - 지정된 복제 규칙에 대해 복제 보류 중인 작업 수입니다. 작업에는 객체, 삭제 마커 및 태그가 포함됩니다.

> **Note**
>
> Outposts에서의 S3 복제 지표는 CloudWatch 사용자 지정 지표와 동일한 요금으로 청구됩니다. 자세한 내용은 CloudWatch 요금을 참조하십시오.

복제 상태 정보 가져오기

복제 상태는 Amazon S3 on Outposts가 복제 중인 객체의 현재 상태를 확인하는 데 도움이 될 수 있습니다. 원본 객체의 복제 상태는 `PENDING`, `COMPLETED`, `FAILED` 중 하나를 반환합니다. 복제본의 복제 상태가 `REPLICA`을(를) 반환합니다.

복제 상태 개요

복제 시나리오에는 복제를 구성하는 소스 버킷과 S3 on Outposts가 객체를 복제하는 대상 버킷이 있습니다. 이러한 버킷에서 객체(`GetObject` 사용)나 객체 메타데이터(`HeadObject` 사용)를 요청하면 S3 on Outposts는 응답으로 다음과 같이 `x-amz-replication-status` 헤더를 반환합니다.

- 소스 버킷에서 객체를 요청하면 요청한 객체가 복제에 적합한 경우 S3 on Outposts가 `x-amz-replication-status` 헤더를 반환합니다.

  예를 들어 복제 구성에 객체 접두사 `TaxDocs`를 지정해 키 이름 접두사가 `TaxDocs`인 객체만 복제하도록 S3 on Outposts에 지시한다고 가정해 봅시다. 이 키 이름 접두사(예: `TaxDocs/document1.pdf`)를 갖는 객체를 업로드하면 모두 복제됩니다. 이 키 이름 접두사를 사용한 객체 요청에 대해 S3 on Outposts는 객체 복제 상태를 표시하는 값 `PENDING`, `COMPLETED` 또는 `FAILED`가 포함된 `x-amz-replication-status` 헤더를 반환합니다.

  > **Note**
  >
  > 객체를 업로드한 후 객체 복제가 실패할 경우 복제를 재시도할 수 없습니다. 객체를 다시 업로드해야 합니다. 복제 역할 권한 누락 또는 버킷 권한 누락과 같은 문제의 경우 객체가 `FAILED` 상태로 전환됩니다. 버킷 또는 Outpost를 사용할 수 없는 등 일시적인 실패의 경우 복제 상태가 `FAILED`로 전환되지 않고 `PENDING` 상태 그대로 유지됩니다. 리소스가 다시 온라인 상태가 되면 S3 on Outposts는 해당 객체 복제를 재개합니다.

- 대상 버킷으로부터 객체를 요청하면 요청의 객체가 S3 on Outposts에서 생성한 복제본인 경우 S3 on Outposts는 값이 `REPLICA`인 `x-amz-replication-status` 헤더를 반환합니다.

> **Note**
>
> 복제가 사용 설정된 원본 버킷에서 객체를 삭제하려면, 먼저 해당 객체의 복제 상태를 통해 복제가 완료되었는지 확인해야 합니다.

Outposts에서 Amazon S3 복제본 수정 동기화가 활성화된 경우 복제 상태

복제 규칙에서 S3 on Outposts 복제본 수정 동기화를 활성화하면 복제본은 `REPLICA` 이외의 상태를 보고할 수 있습니다. 메타데이터 변경 사항이 복제 과정에 있는 경우 복제본의 `x-amz-replication-status` 헤더가 `PENDING`을 반환합니다. 복제본 수정 동기화가 메타데이터 복제에

실패하면 복제본의 헤더가 `FAILED`를 반환합니다. 메타데이터가 올바르게 복제되면 복제본의 헤더가 `REPLICA` 값을 반환합니다.

복제 문제 해결

복제를 구성한 후 대상 Amazon S3 on Outposts 버킷에 객체 복제본이 표시되지 않는다면 다음 문제 해결 도움말을 사용하여 문제를 해결하세요.

- S3 on Outposts가 객체를 복제하는 데 걸리는 시간은 소스 및 대상 Outposts 간의 거리, 객체의 크기를 비롯한 다양한 요소에 따라 달라집니다.

  소스 객체의 복제 상태를 확인할 수 있습니다. 객체 복제 상태가 `PENDING`이면 S3 on Outposts가 복제를 완료하지 않은 것입니다. 객체 복제 상태가 `FAILED`이면 소스 버킷에서 설정된 복제 구성을 확인하세요.
- 원본 버킷의 복제 구성에서 다음을 확인합니다.
  - 대상 버킷의 액세스 포인트 Amazon 리소스 이름(ARN)이 정확해야 합니다.
  - 키 이름 접두사가 정확합니다. 예를 들어, 접두사 `Tax`로 객체 복제를 구성한 경우라면 `Tax/document1` 또는 `Tax/document2`와 같이 키 이름이 포함된 객체만 복제됩니다. 키 이름이 `document3`인 객체는 복제되지 않습니다.
  - 상태가 `Enabled`입니다.
- 버전 관리가 일시 중단된 버킷이 없는지 확인합니다. 소스 버킷과 대상 버킷 모두에서 버전 관리를 활성화해야 합니다.
- 다른 AWS 계정이 대상 버킷을 소유한 경우, 해당 대상 버킷에 소스 버킷 소유자의 객체 복제를 허용하는 버킷 정책이 있는지 확인합니다. 예시는 소스 및 대상 Outposts 버킷을 서로 다른 AWS 계정에서 소유할 경우 권한 부여을 확인하세요.
- 객체 복제본이 대상 버킷에 나타나지 않는 경우, 다음 문제로 복제가 차단될 수 있습니다.
  - S3 on Outposts는 소스 버킷에서 다른 복제 구성이 생성한 복제본을 복제하지 않습니다. 예를 들어 버킷 A에서 버킷 B로, 버킷 B에서 버킷 C로 복제 구성을 설정하면 S3 on Outposts는 버킷 B에 있는 객체 복제본을 버킷 C로 복제하지 않습니다.

    버킷 A의 객체를 버킷 B와 버킷 C로 복제하려면 소스 버킷 복제 구성의 서로 다른 복제 규칙에서 여러 버킷 대상을 설정합니다. 예를 들어, 소스 버킷 A에 두 개의 복제 규칙을 생성하여 하나의 규칙은 대상 버킷 B에 복제하도록 하고 다른 규칙은 대상 버킷 C에 복제하도록 할 수 있습니다.
  - 소스 버킷 소유자는 다른 AWS 계정에 객체 업로드 권한을 부여할 수 있습니다. 기본적으로 원본 버킷 소유자는 다른 계정에서 생성한 객체에 대해 권한이 없습니다. 복제 구성은 원본 버킷 소유자가 액세스 권한을 가진 객체만 복제합니다. 복제 문제를 예방하기 위해 소스 버킷 소유자는 다른

AWS 계정에 해당 객체에 대한 명시적 액세스 권한을 조건부로 요구하는 객체 생성 권한을 부여할 수 있습니다.

- 복제 구성에서 객체 중 특정 태그를 갖는 하위 집합을 복제하는 규칙을 추가한다고 가정해 봅시다. 이 경우 S3 on Outposts가 객체를 복제하도록 하려면 객체를 생성할 때 특정 태그 키 및 값을 할당해야 합니다. 먼저 객체를 생성한 후 해당 기존 객체에 태그를 추가할 경우 S3 on Outposts는 해당 객체를 복제하지 않습니다.
- 버킷 정책이 다음 작업 중 하나에 대해 복제 역할에 대한 액세스를 거부하면 복제가 실패합니다.

  원본 버킷:

  ```
  "s3-outposts:GetObjectVersionForReplication",
  "s3-outposts:GetObjectVersionTagging"
  ```

  대상 버킷:

  ```
  "s3-outposts:ReplicateObject",
  "s3-outposts:ReplicateDelete",
  "s3-outposts:ReplicateTags"
  ```

- Amazon EventBridge는 객체가 대상 Outposts로 복제되지 않을 때 사용자에게 알림을 보낼 수 있습니다. 자세한 내용은 Outposts에서 S3 복제 기능에 EventBridge 사용 단원을 참조하십시오.

Outposts에서 S3 복제 기능에 EventBridge 사용

Amazon S3 on Outposts는 Amazon EventBridge와 통합되며 `s3-outposts` 네임스페이스를 사용합니다. EventBridge는 애플리케이션을 다양한 소스의 데이터와 연결하는 데 사용할 수 있는 서버리스 이벤트 버스 서비스입니다. 자세한 내용은 Amazon EventBridge 사용 설명서의 Amazon EventBridge란? 섹션을 참조하세요.

복제 구성 문제 해결에 도움이 되도록 Amazon EventBridge를 설정하여 복제 실패 이벤트에 대한 알림을 수신할 수 있습니다. EventBridge는 객체가 대상 Outposts로 복제되지 않을 경우 사용자에게 알림을 보낼 수 있습니다. 복제 중인 객체의 현재 상태에 대한 자세한 내용은 복제 상태 개요 섹션을 참조하세요.

Outposts 버킷에서 특정 이벤트가 발생할 때마다 S3 on Outposts는 EventBridge에 이벤트를 보낼 수 있습니다. 다른 대상과 달리 전송할 이벤트 유형을 선택할 필요가 없습니다. 또한 EventBridge 규칙을 사용하여 이벤트를 추가 대상으로 라우팅할 수 있습니다. EventBridge가 활성화되면 S3 on Outposts는 다음 이벤트를 모두 EventBridge로 전송합니다.

이벤트 유형	설명	네임스페이스
`Operation FailedReplication`	복제 규칙 내의 객체 복제가 실패했습니다. Outposts에서 S3 복제 실패 이유에 대한 자세한 내용은 EventBridge를 사용하여 Outposts에서 S3 복제 실패 이유 보기 섹션을 참조하세요.	`s3-outposts`

EventBridge를 사용하여 Outposts에서 S3 복제 실패 이유 보기

다음 테이블에는 Outposts에서 S3 복제 실패 이유가 나열되어 있습니다. EventBridge 규칙을 구성하여 Amazon Simple Queue Service(Amazon SQS), Amazon Simple Notification Service(Amazon SNS), AWS Lambda 또는 Amazon CloudWatch Logs를 통해 실패 이유를 게시하고 볼 수 있습니다. EventBridge에서 이러한 리소스 사용에 필요한 권한에 대한 자세한 내용은 EventBridge에 리소스 기반 정책 사용을 참조하세요.

복제 실패 이유	설명
`AssumeRoleNotPermitted`	S3 on Outposts가 복제 구성에 지정된 AWS Identity and Access Management(IAM) 역할을 맡을 수 없습니다.
`DstBucketNotFound`	S3 on Outposts가 복제 구성에 지정된 대상 버킷을 찾을 수 없습니다.
`DstBucketUnversioned`	Outposts 대상 버킷에서 버전 관리가 활성화되지 않았습니다. Outposts에서 S3 복제로 객체를 복제하려면 대상 버킷에서 버전 관리를 활성화해야 합니다.
`DstDelObjNotPermitted`	S3 on Outposts가 대상 버킷에 삭제된 객체를 복제할 수 없습니다. 대상 버킷에 대한 `s3-outposts:ReplicateDelete` 권한이 누락되었을 수 있습니다.
`DstMultipartCompleteNotPermitted`	S3 on Outposts가 대상 버킷에 있는 객체의 멀티파트 업로드를 완료할 수 없습니다. 대

복제 실패 이유	설명
	상 버킷에 대한 `s3-outposts:ReplicateObject` 권한이 누락되었을 수 있습니다.
`DstMultipartInitNotPermitted`	S3 on Outposts가 대상 버킷에 있는 객체의 멀티파트 업로드를 시작할 수 없습니다. 대상 버킷에 대한 `s3-outposts:ReplicateObject` 권한이 누락되었을 수 있습니다.
`DstMultipartPartUploadNotPermitted`	S3 on Outposts가 대상 버킷에 있는 멀티파트 업로드 객체를 업로드할 수 없습니다. 대상 버킷에 대한 `s3-outposts:ReplicateObject` 권한이 누락되었을 수 있습니다.
`DstOutOfCapacity`	대상 Outpost에 S3 스토리지 용량이 부족하기 때문에 S3 on Outposts가 대상 Outpost에 복제할 수 없습니다.
`DstPutObjNotPermitted`	S3 on Outposts가 대상 버킷에 객체를 복제할 수 없습니다. 대상 버킷에 대한 `s3-outposts:ReplicateObject` 권한이 누락되었을 수 있습니다.
`DstPutTaggingNotPermitted`	S3 on Outposts가 대상 버킷에 객체 태그를 복제할 수 없습니다. 대상 버킷에 대한 `s3-outposts:ReplicateObject` 권한이 누락되었을 수 있습니다.
`DstVersionNotFound`	S3 on Outposts가 대상 버킷에서 해당 객체 버전의 메타데이터를 복제하는 데 필요한 객체 버전을 찾을 수 없습니다.

복제 실패 이유	설명
`SrcBucketReplicationConfigMissing`	S3 on Outposts는 소스 Outposts 버킷과 연결된 액세스 포인트에 대한 복제 구성을 찾을 수 없습니다.
`SrcGetObjNotPermitted`	S3 on Outposts가 복제를 위해 소스 버킷의 객체에 액세스할 수 없습니다. 소스 버킷에 대한 `s3-outposts:GetObjectVersionForReplication` 권한이 누락되었을 수 있습니다.
`SrcGetTaggingNotPermitted`	S3 on Outposts가 소스 버킷의 객체 태그 정보에 액세스할 수 없습니다. 소스 버킷에 대한 `s3-outposts:GetObjectVersionTagging` 권한이 누락되었을 수 있습니다.
`SrcHeadObjectNotPermitted`	S3 on Outposts가 소스 버킷에서 객체 메타데이터를 검색할 수 없습니다. 소스 버킷에 대한 `s3-outposts:GetObjectVersionForReplication` 권한이 누락되었을 수 있습니다.
`SrcObjectNotEligible`	객체를 복제에 사용할 수 없습니다. 객체 또는 객체 태그가 복제 구성과 일치하지 않습니다.

복제 문제 해결에 대한 자세한 내용은 다음 주제를 참조하세요.

- IAM 역할 생성
- 복제 문제 해결

CloudWatch를 사용하여 EventBridge 모니터링

모니터링을 위해 Amazon EventBridge가 Amazon CloudWatch와 통합됩니다. EventBridge는 지표를 1분마다 CloudWatch에 전송합니다. 이러한 지표에는 규칙과 일치하는 이벤트 수 및 규칙에 의해 대상이 호출된 횟수가 포함됩니다. EventBridge에서 규칙이 실행되면 이 규칙과 연관된 모든 대상이 호출됩니다. CloudWatch를 통해 다음과 같은 방법으로 EventBridge를 모니터링할 수 있습니다.

- CloudWatch 대시보드에서 EventBridge 규칙에 사용할 수 있는 EventBridge 지표를 모니터링할 수 있습니다. 그런 다음 CloudWatch 경보와 같은 CloudWatch 기능을 사용하여 특정 지표에 대한 경보를 설정할 수 있습니다. 이러한 지표가 경보에서 지정한 사용자 지정 임계값에 도달하면 알림을 받고 그에 따라 조치를 취할 수 있습니다.
- Amazon CloudWatch Logs를 EventBridge 규칙의 대상으로 설정할 수 있습니다. 그런 다음 EventBridge는 로그 스트림을 생성하고 CloudWatch Logs는 이벤트의 텍스트를 로그 항목으로 저장합니다. 자세한 내용은 EventBridge 및 CloudWatch Logs를 참조하세요.

EventBridge 이벤트 전달 디버깅 및 이벤트 아카이빙에 대한 자세한 내용은 다음 주제를 참조하세요.

- 이벤트 재시도 정책 및 DLQ(Dead Letter Queue) 사용
- EventBridge 이벤트 아카이빙

AWS RAM 사용을 통해 S3 on Outposts 공유

Amazon S3 on Outposts는 AWS Resource Access Manager(AWS RAM) 사용을 통해 조직 내 여러 계정에서 S3 용량 공유를 지원합니다. S3 on Outposts 공유를 사용하면 다른 사용자가 Outpost에서 버킷, 엔드포인트 및 액세스 포인트를 생성하고 관리할 수 있도록 허용할 수 있습니다.

이 주제에서는 AWS RAM을 사용하여 S3 on Outposts 및 관련 리소스를 AWS 조직의 다른 AWS 계정과 공유하는 방법을 보여줍니다.

사전 조건

- Outpost 소유자 계정에는 AWS Organizations에 조직이 구성되어 있습니다. 자세한 내용은 AWS Organizations 사용 설명서의 조직 생성을 참조하세요.
- 조직에는 S3 on Outposts 용량을 공유하려는 AWS 계정이 포함됩니다. 자세한 내용은 AWS Organizations 사용 설명서의 AWS 계정에 초대 전송 섹션을 참조하세요.
- 공유하고자 하는 다음 옵션 중 하나를 선택합니다. 엔드포인트에 액세스할 수 있도록 두 번째 리소스(서브넷 또는 Outposts)를 선택해야 합니다. 엔드포인트는 S3 on Outposts에 저장된 데이터에 액세스하기 위한 네트워킹 요구 사항입니다.

옵션 1	옵션 2
S3 on Outposts	S3 on Outposts

옵션 1	옵션 2
사용자가 Outposts 및 액세스 포인트에 버킷을 생성하고 해당 버킷에 객체를 추가할 수 있습니다. 서브넷 사용자가 VPC(Virtual Private Cloud)와 서브넷과 연결된 엔드포인트를 사용할 수 있도록 허용합니다.	사용자가 Outposts 및 액세스 포인트에 버킷을 생성하고 해당 버킷에 객체를 추가할 수 있습니다. Outposts 사용자가 S3 용량 차트 및 AWS Outposts 콘솔 홈 페이지를 확인할 수 있습니다. 또한 사용자가 공유 Outposts에 서브넷을 생성하고 엔드포인트를 생성할 수 있습니다.

절차

1. Outpost를 소유한 AWS 계정을 사용하여 AWS Management Console에 로그인하고, https://console.aws.amazon.com/ram/home에서 AWS RAM 콘솔을 엽니다.
2. AWS RAM에서 AWS Organizations 대상 공유를 활성화했는지 확인합니다. 자세한 내용은 AWS RAM 사용 설명서에서 AWS Organizations 내에서 리소스 공유 사용을 참조하세요.
3. 사전 조건에서 옵션 1이나 옵션 2를 사용하여 리소스 공유를 생성합니다. S3 on Outposts 리소스가 여러 개 있는 경우 공유하려는 리소스의 Amazon 리소스 이름(ARN)을 선택합니다. 엔드포인트를 활성화하려면 서브넷 또는 Outpost를 공유합니다.

   리소스 공유를 생성하는 방법에 대한 자세한 내용은 AWS RAM 사용 설명서의 리소스 공유 생성을 참조하세요.
4. 리소스를 공유하는 AWS 계정에서 이제 S3 on Outposts를 사용할 수 있어야 합니다. 사전 조건에서 선택한 옵션에 따라 계정 사용자에게 다음 정보를 제공합니다.

옵션 1	옵션 2
Outpost ID VPC ID 서브넷 ID	Outpost ID

옵션 1	옵션 2
보안 그룹 ID	

> **Note**
>
> 사용자는 AWS RAM 콘솔, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 사용하여 리소스가 공유되었는지 확인할 수 있습니다. 사용자는 get-resource-shares CLI 명령을 사용하여 기존 리소스 공유를 볼 수 있습니다.

사용 예제:

S3 on Outposts 리소스를 다른 계정과 공유한 후 해당 계정에서 Outpost의 버킷과 객체를 관리할 수 있습니다. 서브넷 리소스를 공유하는 경우 해당 계정에서 생성한 엔드포인트를 사용할 수 있습니다. 다음 예제에서는 사용자가 AWS CLI를 사용하여 리소스를 공유한 후 Outpost와 상호 작용하는 방법을 보여줍니다.

Example : 버킷 생성

다음 예제에서는 Outpost *op-01ac5d28a6a232904*에서 *amzn-s3-demo-bucket1*이라는 버킷을 생성합니다. 이 명령을 사용하기 전에 각 *user input placeholder*를 사용 사례에 적합한 값으로 바꿉니다.

```
aws s3control create-bucket --bucket amzn-s3-demo-bucket1 --outpost-
id op-01ac5d28a6a232904
```

이 명령에 대한 자세한 내용은 AWS CLI 참조의 create-bucket을 참조하세요.

Example : 액세스 포인트 생성

다음 예제에서는 아래 테이블의 예제 파라미터를 사용하여 Outpost에 액세스 포인트를 생성합니다. 이 명령을 사용하기 전에 *user input placeholder* 값과 AWS 리전 코드를 사용 사례에 적합한 값으로 바꿉니다.

파라미터	값
계정 ID	*111122223333*

파라미터	값
액세스 포인트 이름	*example-outpost-access-point*
Outpost ID	*op-01ac5d28a6a232904*
Outpost 버킷 이름	*amzn-s3-demo-bucket1*
VPC ID	*vpc-1a2b3c4d5e6f7g8h9*

> **Note**
> 계정 ID 파라미터는 공유 사용자인 버킷 소유자의 AWS 계정 ID여야 합니다.

```
aws s3control create-access-point --account-id 111122223333 --name example-outpost-
access-point \
--bucket arn:aws:s3-outposts:us-east-1:111122223333:outpost/op-01ac5d28a6a232904/
bucket/amzn-s3-demo-bucket1 \
--vpc-configuration VpcId=vpc-1a2b3c4d5e6f7g8h9
```

이 명령에 대한 자세한 내용은 AWS CLI 참조의 create-access-point를 참조하세요.

Example : 객체 업로드

다음 예제에서는 사용자의 로컬 파일 시스템에서 AWS 계정 *111122223333*에서 소유하는 Outpost *op-01ac5d28a6a232904*에 있는 액세스 포인트 *example-outpost-access-point*를 통해 이름이 *images/my_image.jpg*인 객체로 *my_image.jpg* 파일을 업로드합니다. 이 명령을 사용하기 전에 *user input placeholder* 값과 AWS 리전 코드를 사용 사례에 적합한 값으로 바꿉니다.

```
aws s3api put-object --bucket arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/example-outpost-access-
point \
--body my_image.jpg --key images/my_image.jpg
```

이 명령에 대한 자세한 내용은 AWS CLI 참조의 put-object를 참조하세요.

> **Note**
>
> 이 작업으로 인해 리소스를 찾을 수 없음 오류가 발생하거나 응답하지 않는 경우 VPC에 공유 엔드포인트가 없을 수 있습니다.
> 공유 엔드포인트가 있는지 확인하려면 list-shared-endpoints AWS CLI명령을 사용하세요. 공유 엔드포인트가 없는 경우 Outpost 소유자와 협력하여 엔드포인트를 생성하세요. 자세한 내용은 Amazon Simple Storage Service API 참조의 ListSharedEndpoints를 참조하세요.

Example : 엔드포인트 생성

다음 예제에서는 Outpost에 대한 엔드포인트를 생성합니다. 이 명령을 사용하기 전에 Outpost ID, 서브넷 ID 및 보안 그룹 ID의 *user input placeholder* 값을 사용 사례에 적합한 값으로 바꿉니다.

> **Note**
>
> 리소스 공유에 Outpost 리소스가 포함된 경우 사용자가 이 작업만을 수행할 수 있습니다.

```
aws s3outposts create-endpoint --outposts-id op-01ac5d28a6a232904 --subnet-id XXXXXX --
security-group-id XXXXXXX
```

이 명령에 대한 자세한 내용은 AWS CLI 참조의 create-endpoint를 참조하세요.

S3 on Outposts를 사용하는 다른 AWS 서비스

AWS Outposts에서 로컬로 실행되는 다른 AWS 서비스도 Amazon S3 on Outposts 용량을 사용할 수 있습니다. Amazon CloudWatch에서 `S3Outposts` 네임스페이스는 S3 on Outposts 내의 버킷에 대한 상세 지표를 보여주지만, 이러한 지표에는 다른 AWS 서비스에 대한 사용량이 포함되지 않습니다. 다른 AWS 서비스에서 사용하는 S3 on Outposts 용량을 관리하려면 다음 표의 정보를 참조하세요.

AWS 서비스	설명	자세히 알아보기
Amazon S3	직접적인 S3 on Outposts 사용량에는 모두 일치하는 계정과 버킷 CloudWatch 지표가 있습니다.	지표 보기

AWS 서비스	설명	자세히 알아보기
Amazon Elastic Block Store(Amazon EBS)	Amazon EBS on Outposts의 경우 AWS Outpost를 스냅샷 대상으로 선택하고 S3 on Outpost에 로컬로 저장할 수 있습니다.	자세히 알아보기
Amazon Relational Database Service(Amazon RDS)	Amazon RDS 로컬 백업을 사용하여 RDS 백업을 Outpost에 로컬로 저장할 수 있습니다.	자세히 알아보기

S3 on Outposts 모니터링

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

Amazon S3 on Outposts 스토리지 용량의 모니터링에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

- Amazon CloudWatch 지표를 사용한 S3 on Outposts 용량 관리
- Amazon CloudWatch Events를 사용하여 S3 on Outposts 이벤트 알림 수신
- AWS CloudTrail 로그로 S3 on Outposts 모니터링

Amazon CloudWatch 지표를 사용한 S3 on Outposts 용량 관리

Outpost의 고정된 S3 용량을 관리하기 위해 스토리지 사용률이 특정 임계값을 초과할 때 알려주는 CloudWatch 알림을 생성할 것을 권장합니다. S3 on Outposts용 CloudWatch 지표에 대한 자세한 내용은 CloudWatch 지표 섹션을 참조하세요. Outpost에 객체를 저장할 공간이 충분하지 않으면 API는 ICE(Insufficient Capacity Exception)를 반환합니다. 공간을 확보하려면 명시적 데이터 삭제를 트리거하는 CloudWatch 경보를 생성하거나 수명 주기 만료 정책을 사용하여 객체를 만료시킬 수 있습니다. 데이터를 삭제하기 전에 데이터를 저장하려면 AWS DataSync를 사용하여 Amazon S3 on Outposts 버킷에서 AWS 리전의 S3 버킷으로 데이터를 복사할 수 있습니다. DataSync 사용에 대한 자세한 내용은 AWS DataSync 사용 설명서의 AWS DataSync 시작하기를 참조하세요.

CloudWatch 지표

`S3Outposts` 네임스페이스에는 Amazon S3 on Outposts 버킷에 대한 다음 지표가 포함됩니다. 프로비저닝된 총 S3 on Outposts 바이트 수, 객체에 사용 가능한 총 바이트 수 및 지정된 버킷에 있는 모든 객체의 총 크기를 모니터링할 수 있습니다. 모든 직접 S3 사용에 대한 버킷 또는 계정 관련 지표가 존

재합니다. Amazon Elastic Block Store 로컬 스냅샷 또는 Amazon Relational Database Service 백업을 Outpost에 저장하는 것과 같은 간접적인 S3 사용은 S3 용량을 소비하지만 버킷 또는 계정 관련 지표에는 포함되지 않습니다. Amazon EBS 로컬 스냅샷에 대한 자세한 내용은 Outposts의 Amazon EBS 로컬 스냅샷을 참조하세요. Amazon EBS 비용 보고서를 보려면 https://console.aws.amazon.com/costmanagement/을 방문하세요.

> **Note**
>
> S3 on Outposts는 다음 지표만 지원하며 다른 Amazon S3 지표는 지원하지 않습니다.
> S3 on Outposts는 용량 한도가 고정되어 있으므로 스토리지 사용률이 특정 임계값을 초과할 경우 알림을 제공하는 CloudWatch 경보를 생성할 것을 권장합니다.

지표	설명	기간	단위	유형
`Outpost` `talByte:`	Outpost에 대해 프로비저닝된 총 용량(바이트)입니다.	5분	바이트	S3 on Outposts
`Outpostl` `eeBytes`	고객 데이터를 저장하기 위해 Outpost에서 사용할 수 있는 여유 바이트 수입니다.	5분	바이트	S3 on Outposts
`BucketU:` `dBytes`	지정된 버킷에 있는 모든 객체의 총 크기	5분	바이트	S3 on Outposts. 직접적인 S3 사용만 가능합니다.
`Accountl` `edBytes`	지정된 Outposts 계정에 대한 모든 객체의 전체 크기입니다.	5분	바이트	S3 on Outposts. 직접적인 S3 사용만 가능합니다.
`BytesPe` `ingRepl:` `ation`	지정된 복제 규칙에 대해 복제 보류 중인 객체의 총 바이트 수입니다. 복제 지표를 활성화하는 방법에 대한 자세한 내용은 Outposts 간 복제 규칙 생성을 참조하세요.	5분	바이트	선택 사항. Outposts에 대한 S3 복제의 경우.
`Operati` `sPendin`	지정된 복제 규칙에 대해 복제 보류 중인 총 작업 수입니다. 복제 지표를 활성화하는 방법에	5분	개수	선택 사항. Outposts에 대한 S3 복제의 경우.

지표	설명	기간	단위	유형
eplicat: n	대한 자세한 내용은 Outposts 간 복제 규칙 생성을 참조하세요.			
Replica onLaten	지정된 복제 규칙에 대해 복제 대상 버킷이 소스 버킷보다 늦은 현재 기간(초)입니다. 복제 지표를 활성화하는 방법에 대한 자세한 내용은 Outposts 간 복제 규칙 생성을 참조하세요.	5분	초	선택 사항. Outposts에 대한 S3 복제의 경우.

Amazon CloudWatch Events를 사용하여 S3 on Outposts 이벤트 알림 수신

CloudWatch를 사용하여 Amazon S3 on Outposts API 이벤트에 대한 규칙을 생성할 수 있습니다. 규칙을 생성할 때 Amazon Simple Queue Service(Amazon SQS), Amazon Simple Notification Service(Amazon SNS), AWS Lambda 등 지원되는 모든 CloudWatch 대상을 통해 알림을 받도록 선택할 수 있습니다. 자세한 내용은 Amazon CloudWatch Events 사용 설명서에서 CloudWatch Events의 대상이 될 수 있는 AWS 서비스 목록을 참조하세요. S3 on Outposts와 함께 사용할 대상 서비스를 선택하려면 Amazon CloudWatch Events 사용 설명서에서 AWS CloudTrail을 사용하여 AWS API 호출에서 트리거하는 CloudWatch Events 규칙 생성을 참조하세요.

> **Note**
>
> S3 on Outposts 객체 작업의 경우, CloudTrail에서 전송한 AWS API 호출 이벤트는 해당 이벤트를 수신하도록 구성된 추적(선택적으로 이벤트 선택기 사용)이 있는 경우에만 규칙과 매칭됩니다. 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail 로그 파일 작업을 참조하세요.

Example

다음은 `DeleteObject` 작업에 대한 샘플 규칙입니다. 이 샘플 규칙을 사용하려면 *`amzn-s3-demo-bucket1`*을 S3 on Outposts 버킷의 이름으로 대체합니다.

```
{
```

```
  "source": [
    "aws.s3-outposts"
  ],
  "detail-type": [
    "AWS API call through CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "s3-outposts.amazonaws.com"
    ],
    "eventName": [
      "DeleteObject"
    ],
    "requestParameters": {
      "bucketName": [
        "amzn-s3-demo-bucket1"
      ]
    }
  }
}
```

AWS CloudTrail 로그로 S3 on Outposts 모니터링

Amazon S3 on Outposts는 S3 on Outposts에서 사용자, 역할 또는 AWS 서비스가 수행한 작업에 대한 레코드를 제공하는 서비스인 AWS CloudTrail과 통합됩니다. AWS CloudTrail을 사용하여 S3 on Outposts 버킷 수준 및 객체 수준 요청에서 정보를 가져와서 S3 on Outposts 이벤트 활동을 감사하고 로깅할 수 있습니다.

모든 Outposts 버킷 또는 특정 Outposts 버킷 목록에 CloudTrail 데이터 이벤트를 활성화하려면 CloudTrail에서 수동으로 추적을 생성해야 합니다. CloudTrail 로그 파일 항목에 대한 자세한 내용은 S3 on Outposts 로그 파일 항목을 참조하세요.

S3 on Outposts의 CloudTrail 데이터 이벤트 전체 목록은 Amazon S3 사용 설명서의 CloudTrail의 Amazon S3 데이터 이벤트를 참조하세요.

> **Note**
>
> - AWS CloudTrail 데이터 이벤트 Outposts 버킷에 대한 수명 주기 정책을 생성하는 것이 모범 사례입니다. 감사에 필요하다고 생각되는 기간이 지나면 로그 파일을 주기적으로 제거하도록 수명 주기 정책을 구성합니다. 그러면 Amazon Athena에서 쿼리마다 분석하는 데이터의

양이 줄어듭니다. 자세한 내용은 Amazon S3 on Outposts 버킷에 대한 수명 주기 구성 생성 및 관리 단원을 참조하십시오.

- CloudTrail 로그를 쿼리하는 방법의 예제는 AWS 빅 데이터 블로그 AWS CloudTrail 및 Amazon Athena를 사용하여 보안, 규정 준수 및 운영 활동 분석을 참조하십시오.

S3 on Outposts 버킷의 객체에 대해 CloudTrail 로깅 활성화

Amazon S3 콘솔에서 AWS CloudTrail 추적을 구성하여 Amazon S3 on Outposts 버킷의 객체에 대한 데이터 이벤트를 로깅할 수 있습니다. CloudTrail은 `GetObject`, `DeleteObject`, `PutObject` 같은 S3 on Outposts 객체 수준 API 작업 로깅을 지원합니다. 이 이벤트를 데이터 이벤트라고 합니다.

기본적으로 CloudTrail 추적은 데이터 이벤트를 로깅하지 않습니다. 그러나 지정한 S3 on Outposts 버킷에 대해 데이터 이벤트를 로깅하거나 AWS 계정에 있는 모든 S3 on Outposts 버킷에 대해 데이터 이벤트를 로깅하도록 추적을 구성할 수 있습니다.

CloudTrail은 CloudTrail 이벤트 기록의 데이터 이벤트를 채우지 않습니다. 또한 모든 S3 on Outposts 버킷 수준 API 작업이 CloudTrail 이벤트 기록에 채워지는 것은 아닙니다. CloudTrail 로그를 쿼리하는 방법에 대한 자세한 내용은 AWS 지식 센터에서 Amazon CloudWatch Logs 필터 패턴 및 Amazon Athena를 사용하여 CloudTrail 로그 쿼리를 참조하세요.

S3 on Outposts 버킷에 대한 데이터 이벤트를 로깅하도록 추적을 구성하기 위해 AWS CloudTrail 콘솔 또는 Amazon S3 콘솔을 사용할 수 있습니다. AWS 계정의 모든 S3 on Outposts 버킷에 대해 데이터 이벤트를 로깅하도록 추적을 구성하는 경우 CloudTrail 콘솔을 사용하는 것이 더 편리합니다. CloudTrail 콘솔을 사용하여 S3 on Outposts 데이터 이벤트를 로깅하도록 추적을 구성하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서에서 데이터 이벤트를 참조하세요.

> **Important**
>
> 데이터 이벤트에는 추가 요금이 적용됩니다. 자세한 내용은 AWS CloudTrail 요금을 참조하십시오.

다음 절차는 Amazon S3 콘솔을 사용하여 CloudTrail 추적이 S3 on Outposts 버킷에 대한 데이터 이벤트를 로깅할 수 있도록 구성하는 방법을 보여줍니다.

> **Note**
>
> 버킷을 생성하는 AWS 계정이 버킷을 소유하며 이 계정은 AWS CloudTrail로 전송할 S3 on Outposts 데이터 이벤트를 구성할 수 있는 유일한 계정입니다.

S3 on Outposts 버킷의 객체에 대해 CloudTrail 데이터 이벤트 로깅 활성화

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 Outposts 버킷을 선택합니다.
3. CloudTrail을 사용하여 로깅하려는 데이터 이벤트가 있는 Outposts 버킷의 이름을 선택합니다.
4. 속성을 선택합니다.
5. AWS CloudTrail 데이터 이벤트 섹션으로 이동하고 CloudTrail에서 구성을 선택합니다.

   AWS CloudTrail 콘솔이 열립니다.

   새 CloudTrail 추적을 생성하거나 기존 추적을 재사용하고 추적에 로깅되도록 S3 on Outposts 데이터 이벤트를 구성할 수 있습니다.
6. CloudTrail 콘솔 대시보드 페이지에서 추적 생성을 선택합니다.
7. 1단계 추적 속성 선택 페이지에서 추적의 이름을 제공하고 추적 로그를 저장할 S3 버킷을 선택하고 원하는 기타 설정을 지정한 후, 다음을 선택합니다.
8. 2단계 로그 이벤트 선택 페이지의 이벤트 유형에서 데이터 이벤트를 선택합니다.

   데이터 이벤트 유형에서 S3 Outposts를 선택합니다. Next(다음)를 선택합니다.

   > **Note**
   >
   > - 추적을 생성하고 S3 on Outposts에 대한 데이터 이벤트 로깅을 구성할 때 데이터 이벤트 유형을 올바르게 지정해야 합니다.
   > - CloudTrail 콘솔을 사용하는 경우 데이터 이벤트 유형으로 S3 Outposts를 선택하세요. CloudTrail 콘솔에서 추적을 생성하는 자세한 방법은 AWS CloudTrail 사용 설명서에서 콘솔을 사용하여 추적 생성 및 업데이트를 참조하십시오. CloudTrail 콘솔을 사용하여 S3 on Outposts 데이터 이벤트 로깅을 구성하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서에서 Amazon S3 객체의 데이터 이벤트 로깅을 참조하세요.

- AWS Command Line Interface(AWS CLI) 또는 AWS SDK를 사용하는 경우 `resources.type` 필드를 `AWS::S3Outposts::Object`로 설정합니다. AWS CLI를 사용하여 S3 on Outposts 데이터 이벤트를 로깅하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 S3 on Outposts 이벤트 로깅을 참조하세요.
- CloudTrail 콘솔이나 Amazon S3 콘솔을 사용하여 S3 on Outposts 버킷에 대해 데이터 이벤트를 로깅하도록 추적을 구성하면 Amazon S3 콘솔에는 해당 버킷에 대해 객체 수준 로깅이 활성화된 것으로 표시됩니다.

9. 3단계 검토 및 생성 페이지에서 구성한 추적 속성과 로그 이벤트를 검토합니다. 추적 생성을 선택합니다.

S3 on Outposts 버킷의 객체에 대해 CloudTrail 데이터 이벤트 로깅 비활성화

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/cloudtrail/에서 CloudTrail 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 추적을 선택합니다.
3. S3 on Outposts 버킷에 이벤트를 로깅하기 위해 생성한 추적의 이름을 선택합니다.
4. 추적의 세부 정보 페이지에서 오른쪽 상단의 로깅 중지를 선택합니다.
5. 표시되는 대화 상자에서 로깅 중지를 선택합니다.

Amazon S3 on Outposts AWS CloudTrail 로그 파일 항목

Amazon S3 on Outposts 관리 이벤트는 AWS CloudTrail을 통해 사용할 수 있습니다. 필요하면 AWS CloudTrail에서 데이터 이벤트에 대한 로깅을 사용할 수도 있습니다.

추적이란 지정한 리전의 S3 버킷에 이벤트를 로그 파일로 입력할 수 있도록 하는 구성입니다. Outposts 버킷에 대한 CloudTrail 로그에는 지정된 버킷이 있는 Outpost를 식별하는 `edgeDeviceDetails`라는 새 필드가 포함됩니다.

추가 로그 필드에는 요청된 작업, 작업 날짜 및 시간과 요청 파라미터가 포함됩니다. CloudTrail 로그 파일은 퍼블릭 API 호출에 대한 순서 지정된 스택 추적이 아니기 때문에 특정 순서로 표시되지 않습니다.

다음 예제에서는 `s3-outposts`의 PutObject 작업을 보여주는 CloudTrail 로그 항목을 확인할 수 있습니다.

```
{
      "eventVersion": "1.08",
```

```
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "111122223333",
        "arn": "arn:aws:iam::111122223333:user/yourUserName",
        "accountId": "222222222222",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "yourUserName"
      },
      "eventTime": "2020-11-30T15:44:33Z",
      "eventSource": "s3-outposts.amazonaws.com",
      "eventName": "PutObject",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "26.29.66.20",
      "userAgent": "aws-cli/1.18.39 Python/3.4.10 Darwin/18.7.0 botocore/1.15.39",
      "requestParameters": {
        "expires": "Wed, 21 Oct 2020 07:28:00 GMT",
        "Content-Language": "english",
        "x-amz-server-side-encryption-customer-key-MD5": "wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
        "ObjectCannedACL": "BucketOwnerFullControl",
        "x-amz-server-side-encryption": "Aes256",
        "Content-Encoding": "gzip",
        "Content-Length": "10",
        "Cache-Control": "no-cache",
        "Content-Type": "text/html; charset=UTF-8",
        "Content-Disposition": "attachment",
        "Content-MD5": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY",
        "x-amz-storage-class": "Outposts",
        "x-amz-server-side-encryption-customer-algorithm": "Aes256",
        "bucketName": "amzn-s3-demo-bucket1",
        "Key": "path/upload.sh"
      },
      "responseElements": {
        "x-amz-server-side-encryption-customer-key-MD5": "wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
        "x-amz-server-side-encryption": "Aes256",
        "x-amz-version-id": "001",
        "x-amz-server-side-encryption-customer-algorithm": "Aes256",
        "ETag": "d41d8cd98f00b204e9800998ecf8427f"
      },
      "additionalEventData": {
        "CipherSuite": "ECDHE-RSA-AES128-SHA",
        "bytesTransferredIn": 10,
```

```
        "x-amz-id-2": "29xXQBV20
+xOHKItvzY1suLv1i6A52E0zOX159fpfsItYd58JhXwKxXAXI4IQkp6",
        "SignatureVersion": "SigV4",
        "bytesTransferredOut": 20,
        "AuthenticationMethod": "AuthHeader"
      },
      "requestID": "8E96D972160306FA",
      "eventID": "ee3b4e0c-ab12-459b-9998-0a5a6f2e4015",
      "readOnly": false,
      "resources": [
        {
          "accountId": "222222222222",
          "type": "AWS::S3Outposts::Object",
          "ARN": "arn:aws:s3-outposts:us-east-1:YYY:outpost/op-01ac5d28a6a232904/
bucket/path/upload.sh"
        },
        {
          "accountId": "222222222222",
          "type": "AWS::S3Outposts::Bucket",
          "ARN": "arn:aws:s3-outposts:us-east-1:YYY:outpost/op-01ac5d28a6a232904/
bucket/"
        }
      ],
      "eventType": "AwsApiCall",
      "managementEvent": false,
      "recipientAccountId": "444455556666",
      "sharedEventID": "02759a4c-c040-4758-b84b-7cbaaf17747a",
      "edgeDeviceDetails": {
        "type": "outposts",
        "deviceId": "op-01ac5d28a6a232904"
      },
      "eventCategory": "Data"
    }
```

Amazon S3 on Outposts로 개발

Amazon S3 on Outposts를 사용하면 AWS Outposts에서 S3 버킷을 생성하고 로컬 데이터 액세스, 로컬 데이터 처리 및 데이터 레지던시가 필요한 애플리케이션을 위해 온프레미스에서 객체를 쉽게 저장하고 검색할 수 있습니다. S3 on Outposts는 S3 Outposts(OUTPOSTS)라는 새로운 스토리지 클래스를 제공합니다. 이 클래스는 Amazon S3 API를 사용하며 AWS Outposts의 여러 디바이스와 서버에 데이터를 이중화된 방식으로 안정적으로 저장하도록 설계되었습니다. Virtual Private Cloud(VPC)를 통한 액세스 포인트 및 엔드포인트 연결을 사용하여 Outpost 버킷과 통신합니다. 액세스 정책, 암호화, 태깅을 포함하여 Amazon S3 버킷에서와 같이 Outpost 버킷에서 동일한 API 및 기능을 사용할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS SDK 또는 REST API를 통해 S3 on Outposts를 사용할 수 있습니다. 자세한 내용은 Amazon S3 on Outposts란 무엇인가요? 섹션을 참조하세요.

다음 주제에서는 S3 on Outposts로 개발하는 방법에 대한 정보를 제공합니다.

주제

- S3 on Outposts 지원 리전
- S3 on Outposts API 작업
- Java용 SDK를 사용하여 S3 on Outposts용 S3 제어 클라이언트 구성
- IPv6를 통해 S3 on Outposts에 요청

S3 on Outposts 지원 리전

S3 on Outposts는 다음 AWS 리전에서 지원됩니다.

- 미국 동부(버지니아 북부)(us-east-1)
- 미국 동부(오하이오)(us-east-2)
- 미국 서부(캘리포니아 북부) (us-west-1)
- 미국 서부(오레곤)(us-west-2)
- 아프리카(케이프타운)(af-south-1)
- 아시아 태평양(자카르타)(ap-southeast-3)
- 아시아 태평양(뭄바이)(ap-south-1)
- 아시아 태평양(오사카) (ap-northeast-3)
- 아시아 태평양(서울)(ap-northeast-2)

- 아시아 태평양(싱가포르)(ap-southeast-1)
- 아시아 태평양(시드니)(ap-southeast-2)
- 아시아 태평양(도쿄)(ap-northeast-1)
- 캐나다(중부)(ca-central-1)
- 유럽(프랑크푸르트)(eu-central-1)
- 유럽(아일랜드)(eu-west-1)
- 유럽(런던) (eu-west-2)
- 유럽(밀라노) (eu-south-1)
- 유럽(파리) (eu-west-3)
- 유럽(스톡홀름) (eu-north-1)
- 이스라엘(텔아비브) (il-central-1)
- 중동(바레인)(me-south-1)
- 남아메리카(상파울루) (sa-east-1)
- AWS GovCloud(미국 동부)(us-gov-east-1)
- AWS GovCloud(미국 서부)(us-gov-west-1)

S3 on Outposts API 작업

이 주제에서는 Amazon S3 on Outposts와 함께 사용할 수 있는 Amazon S3, Amazon S3 Control 및 Amazon S3 on Outposts API 작업을 나열합니다.

주제

- 객체 관리를 위한 Amazon S3 API 작업
- 버킷 관리를 위한 Amazon S3 Control API 작업
- Outposts 관리를 위한 S3 on Outposts API 작업

객체 관리를 위한 Amazon S3 API 작업

S3 on Outposts는 Amazon S3와 동일한 객체 API 작업을 사용하도록 설계되었습니다. Outpost 버킷의 객체에 액세스하려면 액세스 포인트를 사용해야 합니다. S3 on Outposts와 함께 객체 API 작업을 사용할 때 Outposts 액세스 포인트 Amazon 리소스 이름(ARN) 또는 액세스 포인트 별칭을 제공합니다. 액세스 포인트 별칭에 대한 자세한 내용은 S3 on Outposts 버킷 액세스 포인트에 버킷 스타일 별칭 사용 섹션을 참조하세요.

Amazon S3 on Outposts는 다음과 같은 Amazon S3 API 작업을 지원합니다.

- AbortMultipartUpload
- CompleteMultipartUpload
- CopyObject
- CreateMultipartUpload
- DeleteObject
- DeleteObjects
- DeleteObjectTagging
- GetObject
- GetObjectTagging
- HeadBucket
- HeadObject
- ListMultipartUploads
- ListObjects
- ListObjectsV2
- ListObjectVersions
- ListParts
- PutObject
- PutObjectTagging
- UploadPart
- UploadPartCopy

버킷 관리를 위한 Amazon S3 Control API 작업

S3 on Outposts는 버킷 작업을 위해 다음과 같은 Amazon S3 Control API 작업을 지원합니다.

- CreateAccessPoint
- CreateBucket
- DeleteAccessPoint
- DeleteAccessPointPolicy
- DeleteBucket

- DeleteBucketLifecycleConfiguration
- DeleteBucketPolicy
- DeleteBucketReplication
- DeleteBucketTagging
- GetAccessPoint
- GetAccessPointPolicy
- GetBucket
- GetBucketLifecycleConfiguration
- GetBucketPolicy
- GetBucketReplication
- GetBucketTagging
- GetBucketVersioning
- ListAccessPoints
- ListRegionalBuckets
- PutAccessPointPolicy
- PutBucketLifecycleConfiguration
- PutBucketPolicy
- PutBucketReplication
- PutBucketTagging
- PutBucketVersioning

Outposts 관리를 위한 S3 on Outposts API 작업

S3 on Outposts는 엔드포인트 관리를 위해 다음과 같은 Amazon S3 on Outposts API 작업을 지원합니다.

- CreateEndpoint
- DeleteEndpoint
- ListEndpoints
- ListOutpostsWithS3
- ListSharedEndpoints

Java용 SDK를 사용하여 S3 on Outposts용 S3 제어 클라이언트 구성

다음 예제에서는 AWS SDK for Java를 사용하여 Amazon S3 on Outposts용 Amazon S3 제어 클라이언트를 구성합니다. 이 예제를 사용하려면 각 *user input placeholder*를 사용자의 정보로 대체합니다.

```
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;

public AWSS3Control createS3ControlClient() {

    String accessKey = AWSAccessKey;
    String secretKey = SecretAccessKey;
    BasicAWSCredentials awsCreds = new BasicAWSCredentials(accessKey, secretKey);

    return AWSS3ControlClient.builder().enableUseArnRegion()
            .withCredentials(new AWSStaticCredentialsProvider(awsCreds))
            .build();

}
```

IPv6를 통해 S3 on Outposts에 요청

Amazon S3 on Outposts 및 S3 on Outposts 듀얼 스택 엔드포인트는 IPv6 또는 IPv4를 사용하는 S3 on Outposts 버킷 요청을 지원합니다. S3 on Outposts에 대한 IPv6 지원으로, IPv6 네트워크를 사용하는 S3 on Outposts API를 통해 버킷 및 컨트롤 플레인 리소스를 액세스 및 운영할 수 있습니다.

> **Note**
>
> S3 on Outposts 객체 작업(예: `PutObject` 또는`GetObject`) 은 IPv6 네트워크에서는 지원되지 않습니다.

IPv6 네트워크를 통한 S3 on Outposts 액세스에는 추가 요금이 청구되지 않습니다. S3 on Outposts에 대한 자세한 내용은 S3 on Outposts 요금을 참조하세요.

주제

- IPv6 시작하기
- 듀얼 스택 엔드포인트를 사용하여 IPv6 네트워크를 통해 요청
- IAM 정책에서 IPv6 주소 사용
- IP 주소 호환성 테스트
- AWS PrivateLink로 IPv6 사용
- S3 on Outposts 듀얼 스택 엔드포인트 사용

IPv6 시작하기

IPv6를 통해 S3 on Outposts 버킷에 요청하려면 듀얼 스택 엔드포인트를 사용해야 합니다. 다음 단원에서는 듀얼 스택 엔드포인트를 사용하여 IPv6을 통해 요청하는 방법을 설명합니다.

다음은 IPv6를 통해 S3 on Outposts 버킷 액세스를 시도하기 전에 고려해야 할 중요한 사항입니다.

- 버킷에 액세스하는 클라이언트와 네트워크가 IPv6을 사용하도록 설정되어 있어야 합니다.
- 가상 호스팅 방식과 경로 방식 요청이 모두 IPv6 액세스를 지원합니다. 자세한 내용은 S3 on Outposts 듀얼 스택 엔드포인트 사용 단원을 참조하십시오.
- AWS Identity and Access Management(IAM) 사용자 또는 S3 on Outposts 버킷 정책에서 소스 IP 주소 필터링을 사용하는 경우, IPv6 주소 범위를 포함하도록 정책을 업데이트해야 합니다.

  > **Note**
  >
  > 이 요구 사항은 IPv6 네트워크 전반의 S3 on Outposts 버킷 작업 및 컨트롤 플레인 리소스에만 적용됩니다. Amazon S3 on Outposts 객체 작업은 IPv6 네트워크에서는 지원되지 않습니다.

- IPv6을 사용하는 경우, 서버 액세스 로그 파일이 IPv6 형식으로 IP 주소를 출력합니다. IPv6 형식의 원격 IP 주소를 구문 분석할 수 있도록 S3 on Outposts 로그 파일을 구문 분석하는 데 사용하는 기존 도구, 스크립트 및 소프트웨어를 업데이트해야 합니다. 그러면 업데이트된 도구, 스크립트 및 소프트웨어가 IPv6 형식의 원격 IP 주소를 올바르게 구문 분석합니다.

듀얼 스택 엔드포인트를 사용하여 IPv6 네트워크를 통해 요청

IPv6를 통해 S3 on Outposts API 호출을 요청하려면 AWS CLI 또는 AWS SDK를 통해 듀얼 스택 엔드포인트를 사용하면 됩니다. Amazon S3 컨트롤 API 작업과 S3 on Outposts API 작업은 S3 on

Outposts 액세스에 IPv6 프로토콜을 사용하든 IPv4 프로토콜을 사용하든 동일한 방식으로 작동합니다. 하지만 S3 on Outposts 객체 작업(예: `PutObject` 또는 `GetObject`)은 IPv6 네트워크에서는 지원되지 않습니다.

AWS Command Line Interface(AWS CLI) 및 AWS SDK를 사용하는 경우, 파라미터 또는 플래그를 사용하여 듀얼 스택 엔드포인트를 변경할 수 있습니다. 구성 파일의 S3 on Outposts 엔드포인트를 재정의하여 듀얼 스택 엔드포인트를 직접 지정할 수도 있습니다.

듀얼 스택 엔드포인트를 사용하여 다음에서 IPv6를 통해 S3 on Outposts 버킷에 액세스할 수 있습니다.

- AWS CLI에 대한 내용은 AWS CLI의 듀얼 스택 엔드포인트 사용를 참조하십시오.
- AWS SDK(AWS SDK의 S3 on Outposts 듀얼 스택 엔드포인트 사용 참조).

IAM 정책에서 IPv6 주소 사용

IPv6 프로토콜을 사용하여 S3 on Outposts 버킷에 액세스하기 전에 IP 주소 필터링에 사용되는 IAM 사용자 또는 S3 on Outposts 버킷 정책이 IPv6 주소 범위를 포함하도록 업데이트되었는지 확인합니다. IP 주소 필터링 정책이 IPv6 주소를 처리하도록 업데이트되지 않은 경우, IPv6 프로토콜을 사용하려고 하는 동안 S3 on Outposts 버킷에 액세스하지 못하게 될 수 있습니다.

IP 주소를 필터링하는 IAM 정책은 IP 주소 조건 연산자를 사용합니다. 다음 S3 on Outposts 버킷 정책은 IP 주소 조건 연산자를 사용하여 허용되는 IPv4 주소의 54.240.143.* IP 범위를 식별합니다. 이 범위를 벗어난 모든 IP 주소는 S3 on Outposts 버킷(`DOC-EXAMPLE-BUCKET`)에 대한 액세스가 거부됩니다. 모든 IPv6 주소가 허용되는 범위 밖에 있으므로 이 정책은 IPv6 주소가 `DOC-EXAMPLE-BUCKET`에 액세스하지 못하도록 합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3outposts:*",
      "Resource": "arn:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/DOC-EXAMPLE-BUCKET/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"}
```

```
        }
      }
    ]
}
```

다음 예제와 같이 IPv4(`54.240.143.0/24`) 및 IPv6(`2001:DB8:1234:5678::/64`) 주소 범위를 모두 허용하도록 S3 on Outposts 버킷 정책의 `Condition` 요소를 수정할 수 있습니다. 예제에 나와 있는 동일한 `Condition` 블록 유형을 사용하여 IAM 사용자와 버킷 정책 모두를 업데이트할 수 있습니다.

```
        "Condition": {
          "IpAddress": {
             "aws:SourceIp": [
               "54.240.143.0/24",
                "2001:DB8:1234:5678::/64"
              ]
           }
         }
```

IPv6을 사용하기 전에 IP 주소 필터링을 사용하는 관련된 모든 IAM 사용자와 버킷 정책이 IPv6 주소 범위를 허용하도록 업데이트해야 합니다. 기존 IPv4 주소 범위 외에도 조직의 IPv6 주소 범위를 포함하도록 IAM 정책을 업데이트하는 것이 좋습니다. IPv6 및 IPv4 모두를 통한 액세스를 허용하는 버킷 정책의 예는 액세스를 특정 IP 주소로 제한 단원을 참조하세요.

https://console.aws.amazon.com/iam/의 IAM 콘솔을 사용하여 IAM 사용자 정책을 검토할 수 있습니다. IAM에 대한 자세한 내용은 IAM 사용 설명서를 참조하세요. S3 on Outposts 버킷 정책에 대한 자세한 내용은 Amazon S3 on Outposts 버킷의 버킷 정책 추가 또는 편집 섹션을 참조하세요.

IP 주소 호환성 테스트

Linux 또는 Unix 인스턴스 또는 macOS X 플랫폼을 사용하는 경우 IPv6를 통한 듀얼 스택 엔드포인트 액세스를 테스트할 수 있습니다. 예를 들어, IPv6를 통한 Amazon S3 on Outposts 엔드포인트 연결을 테스트하려면 다음 `dig` 명령을 사용합니다.

```
dig s3-outposts.us-west-2.api.aws AAAA +short
```

IPv6 네트워크를 통한 듀얼 스택 엔드포인트가 제대로 설정된 경우 `dig` 명령은 연결된 IPv6 주소를 반환합니다. 예:

```
dig s3-outposts.us-west-2.api.aws AAAA +short
```

```
2600:1f14:2588:4800:b3a9:1460:159f:ebce

2600:1f14:2588:4802:6df6:c1fd:ef8a:fc76

2600:1f14:2588:4801:d802:8ccf:4e04:817
```

AWS PrivateLink로 IPv6 사용

S3 on Outposts는 AWS PrivateLink 서비스 및 엔드포인트에 IPv6 프로토콜을 지원합니다. AWS PrivateLink에서 IPv6 프로토콜을 지원므로 온프레미스 또는 기타 프라이빗 연결에서 IPv6 네트워크를 통해 VPC 내의 서비스 엔드포인트에 연결할 수 있습니다. 또한 S3 on Outposts용 AWS PrivateLink의 IPv6 지원을 통해 AWS PrivateLink를 듀얼 스택 엔드포인트와 통합할 수 있습니다. AWS PrivateLink용 IPv6 활성화 방법에 대한 단계는 Expedite your IPv6 adoption with AWS PrivateLink services and endpoints를 참조하세요.

> **Note**
>
> 지원되는 IP 주소 유형을 IPv4에서 IPv6로 업데이트하려면 AWS PrivateLink 사용 설명서의 지원되는 IP 주소 유형 수정을 참조하세요.

AWS PrivateLink로 IPv6 사용

AWS PrivateLink를 IPv6와 함께 사용하는 경우 IPv6 또는 듀얼 스택 VPC 인터페이스 엔드포인트를 생성해야 합니다. AWS Management Console을 사용하여 VPC 엔드포인트를 생성하는 방법에 대한 일반적인 단계는 AWS PrivateLink 사용 안내서의 인터페이스 VPC 엔드포인트를 사용하여 AWS 서비스 액세스를 참조하세요.

AWS Management Console

다음 절차에 따라 S3 on Outposts에 연결하는 인터페이스 VPC 엔드포인트를 생성합니다.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 엽니다.
2. 탐색 창에서 엔드포인트를 선택합니다.
3. Create endpoint(엔드포인트 생성)을 선택합니다.
4. 서비스 범주(Service category)에서 AWS 서비스를 선택합니다.

5. 서비스 이름의 경우 S3 on Outposts 서비스(com.amazonaws.us-east-1.s3-outposts)를 선택합니다.
6. VPC에서 S3 on Outposts에 액세스하는 데 사용할 VPC를 선택합니다.
7. 서브넷의 경우 S3 on Outposts 액세스를 시작할 서브넷을 가용 영역당 하나만 선택합니다. 동일한 가용 영역에서 여러 서브넷을 선택할 수 없습니다. 선택한 각 서브넷에서 새 엔드포인트 네트워크 인터페이스가 생성됩니다. 기본적으로 서브넷 IP 주소 범위의 IP 주소를 선택하고 엔드포인트 네트워크 인터페이스에 할당합니다. 엔드포인트 네트워크 인터페이스에 대한 IP 주소를 지정하려면 IP 주소 지정을 선택하고 서브넷 주소 범위의 IPv6 주소를 입력합니다.
8. IP 주소 유형의 경우 듀얼 스택을 선택합니다. 엔드포인트 네트워크 인터페이스에 IPv4 및 IPv6 주소를 모두 할당합니다. 이 옵션은 선택한 모든 서브넷에 IPv4 및 IPv6 주소 범위가 모두 있는 경우에만 지원됩니다.
9. 보안 그룹의 경우 VPC 엔드포인트의 엔드포인트 네트워크 인터페이스에 연결할 보안 그룹을 선택합니다. 기본적으로 기본 보안 그룹이 VPC에 연결됩니다.
10. 정책에서 모든 액세스를 선택하여 VPC 엔드포인트를 통한 모든 리소스에 대한 모든 보안 주체의 모든 작업을 허용합니다. 또는 사용자 지정을 선택하여 VPC 엔드포인트를 통해 리소스에 대한 작업을 수행하기 위해 보안 주체에 필요한 권한을 제어하는 VPC 엔드포인트 정책을 연결합니다. 이 옵션은 서비스에서 VPC 엔드포인트 정책을 지원하는 경우에만 사용할 수 있습니다. 자세한 내용은 엔드포인트 정책을 참조하세요.
11. (선택 사항) 태그를 추가하려면 새 태그 추가를 선택하고 태그 키와 태그 값을 입력합니다.
12. Create endpoint(엔드포인트 생성)을 선택합니다.

Example - S3 on Outposts 버킷 정책

S3 on Outposts가 VPC 엔드포인트와 상호 작용하도록 허용하려면 다음과 같이 S3 on Outposts 정책을 업데이트하면 됩니다.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3-outposts:*",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

AWS CLI

> **Note**
>
> VPC 엔드포인트에서 IPv6 네트워크를 활성화하려면 S3 on Outposts에 대한 `SupportedIpAddressType` 필터에 `IPv6`를 설정해야 합니다.

다음 예제에서는 `create-vpc-endpoint` 명령을 사용하여 새 듀얼 스택 인터페이스 엔드포인트를 생성합니다.

```
aws ec2 create-vpc-endpoint \
--vpc-id vpc-12345678 \
--vpc-endpoint-type Interface \
--service-name com.amazonaws.us-east-1.s3-outposts \
--subnet-id subnet-12345678 \
--security-group-id sg-12345678 \
--ip-address-type dualstack \
--dns-options "DnsRecordIpType=dualstack"
```

AWS PrivateLink 서비스 구성에 따라 새로 생성된 엔드포인트 연결은 VPC 엔드포인트 서비스 공급자가 수락해야 사용할 수 있습니다. 자세한 내용은 AWS PrivateLink 사용 설명서의 엔드포인트 연결 요청 수락 및 거부를 참조하세요.

다음 예제에서는 `modify-vpc-endpoint` 명령을 사용하여 IPv 전용 VPC 엔드포인트를 듀얼 스택 엔드포인트로 업데이트합니다. 듀얼 스택 엔드포인트는 IPv4 및 IPv6 네트워크 모두에 대한 액세스를 허용합니다.

```
aws ec2 modify-vpc-endpoint \
--vpc-endpoint-id vpce-12345678 \
--add-subnet-ids subnet-12345678 \
--remove-subnet-ids subnet-12345678 \
--ip-address-type dualstack \
--dns-options "DnsRecordIpType=dualstack"
```

AWS PrivateLink용 IPv6 네트워크 활성화 방법에 대한 자세한 내용은 Expedite your IPv6 adoption with AWS PrivateLink services and endpoints를 참조하세요.

S3 on Outposts 듀얼 스택 엔드포인트 사용

S3 on Outposts 듀얼 스택 엔드포인트는 IPv6 및 IPv4를 통한 S3 on Outposts 버킷 요청을 지원합니다. 이 섹션에서는 S3 on Outposts 듀얼 스택 엔드포인트를 사용하는 방법을 설명합니다.

주제

- S3 on Outposts 듀얼 스택 엔드포인트
- AWS CLI의 듀얼 스택 엔드포인트 사용
- AWS SDK의 S3 on Outposts 듀얼 스택 엔드포인트 사용

S3 on Outposts 듀얼 스택 엔드포인트

듀얼 스택 엔드포인트에 요청하는 경우, S3 on Outposts 버킷 URL이 IPv6 또는 IPv4 주소로 확인됩니다. IPv6를 통해 S3 on Outposts 버킷에 액세스하는 방법의 자세한 내용은 IPv6를 통해 S3 on Outposts에 요청 섹션을 참조하세요.

듀얼 스택 엔드포인트를 통해 S3 on Outposts 버킷에 액세스하려면 경로 방식 엔드포인트 이름을 사용합니다. S3 on Outposts는 리전 듀얼 스택 엔드포인트 이름만 지원합니다. 이는 이름의 일부로 리전을 지정해야 함을 뜻합니다.

FIPS 듀얼 스택 엔드포인트에는 라는 명명 규칙이 사용됩니다.

```
s3-outposts-fips.region.api.aws
```

FIPS 듀얼 스택 엔드포인트에는 라는 명명 규칙이 사용됩니다.

```
s3-outposts.region.api.aws
```

> **Note**
>
> 가상 호스팅 스타일 엔드포인트 이름은 S3 on Outposts에서 지원되지 않습니다.

AWS CLI의 듀얼 스택 엔드포인트 사용

이 단원에서는 듀얼 스택 엔드포인트에 요청하는 데 사용되는 AWS CLI 명령의 예를 보여 줍니다. AWS CLI를 설치하는 지침은 AWS CLI 및 Java용 SDK를 사용하여 시작하기 단원을 참조하십시오.

AWS Config 파일의 프로파일에서 구성 값 `use_dualstack_endpoint`를 `true`로 설정하여 `s3` 및 `s3api` AWS CLI 명령의 모든 Amazon S3 요청을 지정된 리전의 듀얼 스택 엔드포인트로 보냅니다. `--region` 옵션을 사용하여 구성 파일 또는 명령에 리전을 지정합니다.

AWS CLI에서 듀얼 스택 엔드포인트를 사용하는 경우, `path` 주소 형식만 지원됩니다. 구성 파일에 설정된 주소 스타일은 버킷 이름이 호스트 이름에 있는지 URL에 있는지 제어합니다. 자세한 내용은 AWS CLI 사용 설명서의 s3outposts를 참조하십시오.

AWS CLI를 통해 듀얼 스택 엔드포인트를 사용하려면 `s3control` 또는 `s3outposts` 명령에 `http://s3.dualstack.`*`region`*`.amazonaws.com` 또는 `https://s3-outposts-fips.`*`region`*`.api.aws` 엔드포인트와 함께 `--endpoint-url` 파라미터를 사용합니다.

예:

```
$  aws s3control list-regional-buckets --endpoint-url https://s3-
outposts.region.api.aws
```

AWS SDK의 S3 on Outposts 듀얼 스택 엔드포인트 사용

이 섹션에서는 AWS SDK를 사용하여 듀얼 스택 엔드포인트에 액세스하는 방법의 예제를 보여줍니다.

AWS SDK for Java 2.x 듀얼 스택 엔드포인트 예제

다음 예제에서는 AWS SDK for Java 2.x로 S3 on Outposts 클라이언트 생성 시 `S3ControlClient` 및 `S3OutpostsClient` 클래스를 사용하여 듀얼 스택 엔드포인트를 활성화하는 방법을 보여줍니다. Amazon S3용 실제 Java 예제를 작성 및 테스트하는 방법에 대한 자세한 내용은 AWS CLI 및 Java용 SDK를 사용하여 시작하기 섹션을 참조하세요.

Example - 듀얼 스택 엔드포인트가 활성화된 **`S3ControlClient`** 클래스 생성

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.ListRegionalBucketsRequest;
import software.amazon.awssdk.services.s3control.model.ListRegionalBucketsResponse;
import software.amazon.awssdk.services.s3control.model.S3ControlException;


public class DualStackEndpointsExample1 {

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

```
        Region clientRegion = Region.of("us-east-1");
        String accountId = "111122223333";
        String navyId = "9876543210";

        try {
            // Create an S3ControlClient with dual-stack endpoints enabled.
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                                                            .region(clientRegion)
                                                            .dualstackEnabled(true)
                                                            .build();
            ListRegionalBucketsRequest listRegionalBucketsRequest =
 ListRegionalBucketsRequest.builder()

       .accountId(accountId)

       .outpostId(navyId)

       .build();

            ListRegionalBucketsResponse listBuckets =
 s3ControlClient.listRegionalBuckets(listRegionalBucketsRequest);
            System.out.printf("ListRegionalBuckets Response: %s%n",
 listBuckets.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 on Outposts
 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        }
        catch (S3ControlException e) {
            // Unknown exceptions will be thrown as an instance of this type.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 on Outposts couldn't be contacted for a response, or the
 client
            // couldn't parse the response from Amazon S3 on Outposts.
            e.printStackTrace();
        }
    }
}
```

Example - 듀얼 스택 엔드포인트가 활성화된 **S3OutpostsClient** 생성

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3outposts.S3OutpostsClient;
import software.amazon.awssdk.services.s3outposts.model.ListEndpointsRequest;
import software.amazon.awssdk.services.s3outposts.model.ListEndpointsResponse;
import software.amazon.awssdk.services.s3outposts.model.S3OutpostsException;


public class DualStackEndpointsExample2 {

    public static void main(String[] args) {
        Region clientRegion = Region.of("us-east-1");

        try {
            // Create an S3OutpostsClient with dual-stack endpoints enabled.
            S3OutpostsClient s3OutpostsClient = S3OutpostsClient.builder()
                                                                .region(clientRegion)
                                                                .dualstackEnabled(true)
                                                                .build();
            ListEndpointsRequest listEndpointsRequest =
 ListEndpointsRequest.builder().build();

            ListEndpointsResponse listEndpoints =
 s3OutpostsClient.listEndpoints(listEndpointsRequest);
            System.out.printf("ListEndpoints Response: %s%n",
 listEndpoints.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 on Outposts
 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        }
        catch (S3OutpostsException e) {
            // Unknown exceptions will be thrown as an instance of this type.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 on Outposts couldn't be contacted for a response, or the
 client
            // couldn't parse the response from Amazon S3 on Outposts.
            e.printStackTrace();
        }
```

```
    }
}
```

Windows에서 AWS SDK for Java 2.x를 사용하는 경우, 다음과 같은 Java 가상 머신(JVM) 속성을 설정해야 할 수 있습니다.

```
java.net.preferIPv6Addresses=true
```