StorageGRID에서 S3 REST API를 구현하는 방법

StorageGRID

NetApp
March 12, 2025

목차

StorageGRID에서 S3 REST API를 구현하는 방법

클라이언트 요청 충돌

동일한 키에 쓰는 두 클라이언트 등의 충돌하는 클라이언트 요청은 "최신 성공" 기준으로 해결됩니다.

"Latest-WINS" 평가 시기는 S3 클라이언트가 작업을 시작할 때가 아니라 StorageGRID 시스템이 지정된 요청을 완료하는 시점을 기준으로 합니다.

일관성 값

정합성 보장은 서로 다른 스토리지 노드 및 사이트에서 객체의 가용성과 객체 일관성 간의 균형을 제공합니다. 애플리케이션에 필요한 만큼 일관성을 변경할 수 있습니다.

기본적으로 StorageGRID는 새로 생성된 개체에 대해 쓰기 후 읽기 일관성을 보장합니다. 성공적으로 완료된 PUT를 팔로우하면 새로 작성된 데이터를 읽을 수 있습니다. 기존 오브젝트, 메타데이터 업데이트 및 삭제를 덮어쓰는 것은 결국 일관성이 유지됩니다. 덮어쓰기는 일반적으로 전파되는 데 몇 초 또는 몇 분이 걸리지만 최대 15일이 소요될 수 있습니다.

개체 작업을 다른 일관성으로 수행하려는 경우 다음을 수행할 수 있습니다.

- 에 대한 일관성을 각 버킷지정합니다.
- 에 대한 일관성을 각 API 작동지정합니다.
- 다음 작업 중 하나를 수행하여 그리드 전체의 기본 일관성을 변경합니다.
  - 그리드 관리자에서 * 구성 * > * 시스템 * > * 스토리지 설정 * > * 기본 일관성 * 로 이동합니다.
  - ..

> 그리드 전체의 일관성에 대한 변경은 설정이 변경된 후에 생성된 버킷에만 적용됩니다. 변경에 대한 세부 정보를 확인하려면 에 있는 감사 로그를 참조하십시오 `/var/local/log`(* consistencyLevel* 검색).

일관성 값

일관성은 StorageGRID이 오브젝트를 추적하는 데 사용하는 메타데이터가 노드 간에 분산되는 방식과 클라이언트 요청을 위한 개체의 가용성에 영향을 줍니다.

버킷 또는 API 작업의 정합성을 다음 값 중 하나로 설정할 수 있습니다.

- * ALL *: 모든 노드가 즉시 데이터를 수신하거나 요청이 실패합니다.
- * 강력한 글로벌 *: 모든 사이트에서 모든 클라이언트 요청에 대해 쓰기 후 읽기 일관성을 보장합니다.
- * 강력한 사이트 *: 사이트 내의 모든 클라이언트 요청에 대해 쓰기 후 읽기 일관성을 보장합니다.
- *Read-after-new-write *: (기본값) 새 개체에 대해 읽기-쓰기 후 일관성을 제공하고 개체 업데이트에 대한 최종 일관성을 제공합니다. 고가용성 및 데이터 보호 보장 제공 대부분의 경우에 권장됩니다.

- * 사용 가능 *: 새 객체 및 객체 업데이트 모두에 대한 최종 일관성을 제공합니다. S3 버킷의 경우 필요한 경우에만 사용하십시오(예: 거의 읽지 않는 로그 값이 포함된 버킷의 경우 또는 존재하지 않는 키의 헤드 또는 GET 작업의 경우). S3 FabricPool 버킷은 지원되지 않습니다.

"Read-after-new-write" 및 "Available" 정합성 보장을 사용합니다

HEAD 또는 GET 작업에서 "Read-after-new-write" 일관성을 사용하는 경우 StorageGRID는 다음과 같이 여러 단계로 조회를 수행합니다.

- 먼저 낮은 일관성을 사용하여 오브젝트를 찾습니다.
- 이 조회가 실패하면 다음 일관성 값에서 조회를 반복하여 강력한 글로벌 동작과 동일한 일관성을 유지합니다.

HEAD 또는 GET 작업에서 "Read-after-new-write" 일관성을 사용하지만 객체가 존재하지 않는 경우 객체 조회는 항상 강력한 글로벌 동작과 동일한 일관성을 유지합니다. 이 일관성을 유지하기 위해서는 각 사이트에서 개체 메타데이터의 여러 복사본을 사용할 수 있어야 하므로, 같은 사이트에 있는 두 개 이상의 스토리지 노드를 사용할 수 없는 경우 500개의 내부 서버 오류가 발생할 수 있습니다.

Amazon S3와 유사한 일관성 보장이 필요하지 않은 경우 일관성을 "사용 가능"으로 설정하여 헤드 및 가져오기 작업에 대한 이러한 오류를 방지할 수 있습니다. 두부 또는 GET 작업에서 "사용 가능한" 일관성을 사용하는 경우 StorageGRID는 최종 일관성을 제공합니다. 일관성 향상을 위해 실패한 작업을 다시 시도하지 않으므로 개체 메타데이터의 여러 복사본을 사용할 필요가 없습니다.

API 작업에 대한 일관성을 지정합니다

개별 API 작업에 대한 일관성을 설정하려면 작업에 대해 정합성 보장 값을 지원해야 하며 요청 헤더에서 일관성을 지정해야 합니다. 이 예제에서는 GetObject 작업의 일관성을 "Strong-site"로 설정합니다.

```
GET /bucket/object HTTP/1.1
Date: date
Authorization: authorization name
Host: host
Consistency-Control: strong-site
```

PutObject 및 GetObject 작업 모두에 대해 동일한 일관성을 사용해야 합니다.

버킷의 일관성을 지정합니다

버킷의 일관성을 설정하려면 StorageGRID 요청을 사용할 수 "버킷 일관성을 유지합니다"있습니다. 또는 Tenant Manager에서 수행할 수 "버킷의 일관성을 변경합니다"있습니다.

버킷의 일관성을 설정할 때 다음 사항에 유의하십시오.

- 버킷의 일관성을 설정하면 버킷의 오브젝트나 버킷 구성에 수행되는 S3 작업에 사용되는 일관성이 결정됩니다. 버킷 자체의 작동에는 영향을 미치지 않습니다.
- 개별 API 작업의 일관성이 버킷의 일관성을 재정의합니다.
- 일반적으로 버킷은 "Read-after-new-write"라는 기본 일관성을 사용해야 합니다. 요청이 올바르게 작동하지 않는 경우 가능한 경우 응용 프로그램 클라이언트 동작을 변경합니다. 또는 각 API 요청의 일관성을 지정하도록

클라이언트를 구성합니다. 버킷 수준의 일관성을 마지막 수단으로 설정합니다.

일관성 및 **ILM** 규칙이 데이터 보호에 영향을 미치는 방식

일관성을 선택하고 ILM 규칙을 따르는 것은 오브젝트가 보호되는 방식에 영향을 미칩니다. 이러한 설정은 상호 작용할 수 있습니다.

예를 들어, 오브젝트가 저장될 때 사용되는 일관성은 오브젝트 메타데이터의 초기 배치에 영향을 주고, ILM 규칙에 대해 선택된 수집 동작은 오브젝트 복사본의 초기 배치에 영향을 미칩니다. StorageGRID에서는 클라이언트 요청을 이행하기 위해 오브젝트의 메타데이터와 해당 데이터에 모두 액세스해야 하므로 일관성 및 수집 동작에 대해 일치하는 보호 수준을 선택하면 초기 데이터 보호 수준을 높이고 시스템 응답을 보다 예측 가능하게 할 수 있습니다.

ILM 규칙에 사용할 수 있는 항목은 다음과 "수집 옵션"같습니다.

이중 커밋

StorageGRID는 즉시 개체의 중간 복사본을 만들고 클라이언트에 성공을 반환합니다. ILM 규칙에 지정된 복사본은 가능한 경우 만들어집니다.

엄격한

ILM 규칙에 지정된 모든 복제본이 클라이언트에 반환되기 전에 만들어져야 합니다.

균형

StorageGRID는 수집 시 ILM 규칙에 지정된 모든 복제본을 만들려고 합니다. 이 작업이 불가능할 경우 중간 복제본이 만들어지고 성공이 클라이언트에 반환됩니다. ILM 규칙에 지정된 복사본은 가능한 경우 만들어집니다.

일관성과 **ILM** 규칙이 상호 작용하는 방법의 예

다음과 같은 ILM 규칙과 다음과 같은 일관성이 있는 2개 사이트 그리드가 있다고 가정합니다.

- * ILM 규칙 *: 로컬 사이트와 원격 사이트에 각각 하나씩, 두 개의 오브젝트 복사본을 만듭니다. 엄격한 수집 동작을 사용합니다.
- * Consistency *: 강력한 글로벌(오브젝트 메타데이터는 모든 사이트에 즉시 배포됨).

클라이언트가 오브젝트를 그리드에 저장할 때 StorageGRID는 오브젝트 복사본을 둘 다 만들고 메타데이터를 두 사이트에 분산한 다음 클라이언트에 성공을 반환합니다.

수집 성공 메시지가 표시된 시점에 객체가 손실로부터 완벽하게 보호됩니다. 예를 들어, 수집 직후 로컬 사이트가 손실되면 오브젝트 데이터와 오브젝트 메타데이터의 복사본이 원격 사이트에 계속 존재합니다. 개체를 완전히 검색할 수 있습니다.

대신 동일한 ILM 규칙과 강력한 사이트 일관성을 사용한 경우 개체 데이터가 원격 사이트에 복제된 후 개체 메타데이터가 이 사이트에 배포되기 전에 클라이언트에서 성공 메시지를 받을 수 있습니다. 이 경우 오브젝트 메타데이터의 보호 수준이 오브젝트 데이터의 보호 수준과 일치하지 않습니다. 수집 후 곧바로 로컬 사이트가 손실되면 오브젝트 메타데이터가 손실됩니다. 개체를 검색할 수 없습니다.

일관성과 ILM 규칙 간의 상호 관계는 복잡할 수 있습니다. 도움이 필요하면 NetApp에 문의하십시오.

오브젝트 버전 관리

각 오브젝트의 여러 버전을 유지하려면 버킷의 버전 관리 상태를 설정할 수 있습니다. 버킷에

대한 버전 관리를 사용하면 실수로 개체가 삭제되지 않도록 보호하고 이전 버전의 개체를 검색 및 복원할 수 있습니다.

StorageGRID 시스템은 대부분의 기능을 지원하는 버전 관리를 구현하지만 몇 가지 제한 사항이 있습니다. StorageGRID는 각 오브젝트의 버전을 최대 10,000개까지 지원합니다.

오브젝트 버전 관리를 StorageGRID ILM(정보 라이프사이클 관리) 또는 S3 버킷 라이프사이클 구성과 결합할 수 있습니다. 각 버킷에 대해 버전 관리를 명시적으로 설정해야 합니다. 버킷에 대해 버전 관리를 사용하도록 설정하면 버킷에 추가된 각 오브젝트에 버전 ID가 할당되며, StorageGRID 시스템에서 생성됩니다.

MFA(다중 요소 인증) 삭제 사용은 지원되지 않습니다.

> ⓘ 버전 관리는 StorageGRID 버전 10.3 이상으로 생성된 버킷에서만 사용할 수 있습니다.

ILM 및 버전 관리

ILM 정책은 개체의 각 버전에 적용됩니다. ILM 스캔 프로세스는 모든 개체를 지속적으로 스캔하고 현재 ILM 정책에 대해 다시 평가합니다. ILM 정책에 대한 모든 변경 사항은 이전에 수집된 모든 개체에 적용됩니다. 여기에는 버전 관리가 활성화된 경우 이전에 수집된 버전이 포함됩니다. ILM 스캐닝은 이전에 수집된 개체에 새로운 ILM 변경 사항을 적용합니다.

버전 관리 지원 버킷의 S3 객체의 경우 버전 관리 지원을 통해 "현재 시간"을 참조 시간으로 사용하는 ILM 규칙을 만들 수 있습니다(의 "이전 객체 버전에만 이 규칙을 적용하시겠습니까?"라는 질문에 대해 * 예 * 를 선택"ILM 규칙 만들기 마법사의 1단계"). 개체가 업데이트되면 이전 버전은 업데이트되지 않습니다. "비현재 시간" 필터를 사용하면 이전 버전의 객체가 스토리지에 미치는 영향을 줄이는 정책을 만들 수 있습니다.

> ⓘ 다중 파트 업로드 작업을 사용하여 새 버전의 개체를 업로드할 때 개체의 원래 버전에 대한 비현재 시간은 다중 파트 업로드가 완료될 때가 아닌 새 버전에 대해 다중 파트 업로드가 생성된 시점을 반영합니다. 제한된 경우 원래 버전의 비현재 시간이 현재 버전의 시간보다 몇 시간 또는 며칠 빨라질 수 있습니다.

관련 정보

- "S3 버전 오브젝트 삭제 방법"
- "S3 버전 오브젝트 ILM 규칙 및 정책(예 4)"..

S3 REST API를 사용하여 S3 오브젝트 잠금을 구성합니다

StorageGRID 시스템에서 글로벌 S3 오브젝트 잠금 설정이 활성화된 경우 S3 오브젝트 잠금이 활성화된 버킷을 생성할 수 있습니다. 각 오브젝트 버전에 대해 각 버킷의 기본 보존 또는 보존 설정을 지정할 수 있습니다.

버킷에 대해 S3 오브젝트 잠금을 활성화하는 방법

StorageGRID 시스템에 대해 글로벌 S3 오브젝트 잠금 설정이 활성화된 경우 각 버킷을 생성할 때 선택적으로 S3 오브젝트 잠금을 활성화할 수 있습니다.

S3 오브젝트 잠금은 버킷을 생성할 때만 활성화할 수 있는 영구 설정입니다. 버킷을 생성한 후에는 S3 오브젝트 잠금을 추가하거나 비활성화할 수 없습니다.

버킷에 대해 S3 오브젝트 잠금을 설정하려면 다음 방법 중 하나를 사용하십시오.

- 테넌트 관리자를 사용하여 버킷을 생성합니다. 을 "S3 버킷을 생성합니다"참조하십시오.
- 요청 헤더가 있는 CreateBucket 요청을 사용하여 버킷을 `x-amz-bucket-object-lock-enabled` 만듭니다. 을 "버킷 작업"참조하십시오.

S3 오브젝트 잠금에서는 버킷 버전 관리가 필요하며, 이 버전은 버킷을 생성할 때 자동으로 활성화됩니다. 버킷의 버전 관리는 일시 중단할 수 없습니다. 을 "오브젝트 버전 관리"참조하십시오.

버킷의 기본 보존 설정입니다

버킷에 대해 S3 오브젝트 잠금이 활성화된 경우 버킷에 대한 기본 보존을 선택적으로 설정하고 기본 보존 모드 및 기본 보존 기간을 지정할 수 있습니다.

기본 보존 모드

- 규정 준수 모드:
  - 보존 기한 에 도달할 때까지 개체를 삭제할 수 없습니다.
  - 오브젝트의 보존 기한 을 늘릴 수 있지만 줄일 수는 없습니다.
  - 개체의 보존 기한 은 해당 날짜에 도달할 때까지 제거할 수 없습니다.
- 거버넌스 모드:
  - 있는 사용자는 `s3:BypassGovernanceRetention`권한이 요청 헤더를 사용하여 보존 설정을 무시할 수 `x-amz-bypass-governance-retention: true` 있습니다.
  - 이러한 사용자는 보존 기한이 되기 전에 개체 버전을 삭제할 수 있습니다.
  - 이러한 사용자는 개체의 보존 기간(Retain-until-date)을 증가, 감소 또는 제거할 수 있습니다.

기본 보존 기간

각 버킷에는 년 또는 일 단위로 지정된 기본 보존 기간이 있을 수 있습니다.

버킷의 기본 보존 설정 방법

버킷의 기본 보존을 설정하려면 다음 방법 중 하나를 사용합니다.

- 테넌트 관리자에서 버킷 설정을 관리합니다. "S3 버킷을 생성합니다"및 을 "S3 오브젝트 잠금 기본 보존 업데이트 "참조하십시오.
- 버킷에 대한 PutObjectLockConfiguration 요청을 실행하여 기본 모드와 기본 일 또는 년 수를 지정합니다.

PutObjectLockConfiguration 을 참조하십시오

PutObjectLockConfiguration 요청을 사용하면 S3 오브젝트 잠금이 설정된 버킷의 기본 보존 모드 및 기본 보존 기간을 설정하고 수정할 수 있습니다. 이전에 구성한 기본 보존 설정을 제거할 수도 있습니다.

새 오브젝트 버전이 버킷에 수집되면 및 `x-amz-object-lock-retain-until-date` 이 지정되지 않은 경우 기본 보존 모드가 `x-amz-object-lock-mode` 적용됩니다. 이 지정되지 않은 경우 기본 보존 기간은 유지 종료 날짜를 계산하는 데 `x-amz-object-lock-retain-until-date` 사용됩니다.

오브젝트 버전을 수집한 후 기본 보존 기간을 수정하면 오브젝트 버전의 보존 기한은 그대로 유지되고 새 기본 보존 기간을 사용하여 다시 계산되지 않습니다.

이 작업을 완료하려면 권한이 있거나 계정 루트여야 `s3:PutBucketObjectLockConfiguration` 합니다.

`Content-MD5`PUT 요청에 요청 헤더를 지정해야 합니다.

요청 예

이 예에서는 버킷에 대해 S3 Object Lock을 설정하고 기본 보존 모드를 규정 준수 로 설정하고 기본 보존 기간을 6년으로 설정합니다.

```
PUT /bucket?object-lock HTTP/1.1
Accept-Encoding: identity
Content-Length: 308
Host: host
Content-MD5: request header
User-Agent: s3sign/1.0.0 requests/2.24.0 python/3.8.2
X-Amz-Date: date
X-Amz-Content-SHA256: authorization-string
Authorization: authorization-string

<ObjectLockConfiguration>
    <ObjectLockEnabled>Enabled</ObjectLockEnabled>
    <Rule>
        <DefaultRetention>
            <Mode>COMPLIANCE</Mode>
            <Years>6</Years>
        </DefaultRetention>
    </Rule>
</ObjectLockConfiguration>
```

버킷의 기본 보존 결정 방법

버킷에 대해 S3 오브젝트 잠금이 설정되었는지 확인하고 기본 보존 모드 및 보존 기간을 확인하려면 다음 방법 중 하나를 사용하십시오.

- 테넌트 관리자에서 버킷을 확인합니다. 을 "S3 버킷을 봅니다"참조하십시오.
- GetObjectLockConfiguration 요청을 실행합니다.

GetObjectLockConfiguration 을 참조하십시오

GetObjectLockConfiguration 요청을 사용하면 버킷에 대해 S3 오브젝트 잠금이 설정되어 있는지 확인하고, 사용하도록 설정되어 있는 경우 버킷에 대해 구성된 기본 보존 모드 및 보존 기간이 있는지 확인할 수 있습니다.

새 오브젝트 버전이 버킷에 수집되면 이 지정되지 않은 경우 기본 보존 모드가 `x-amz-object-lock-mode`

적용됩니다. 이 지정되지 않은 경우 기본 보존 기간은 유지 종료 날짜를 계산하는 데 `x-amz-object-lock-retain-until-date` 사용됩니다.

이 작업을 완료하려면 권한이 있거나 계정 루트여야 `s3:GetBucketObjectLockConfiguration` 합니다.

요청 예

```
GET /bucket?object-lock HTTP/1.1
Host: host
Accept-Encoding: identity
User-Agent: aws-cli/1.18.106 Python/3.8.2 Linux/4.4.0-18362-Microsoft
botocore/1.17.29
x-amz-date: date
x-amz-content-sha256: authorization-string
Authorization: authorization-string
```

응답 예

```
HTTP/1.1 200 OK
x-amz-id-2:
iVmcB7OXXJRkRH1FiVq1151/T24gRfpwpuZrEG11Bb9ImOMAAe98oxSpXlknabA0LTvBYJpSIX
k=
x-amz-request-id: B34E94CACB2CEF6D
Date: Fri, 04 Sep 2020 22:47:09 GMT
Transfer-Encoding: chunked
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<ObjectLockConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <ObjectLockEnabled>Enabled</ObjectLockEnabled>
    <Rule>
        <DefaultRetention>
            <Mode>COMPLIANCE</Mode>
            <Years>6</Years>
        </DefaultRetention>
    </Rule>
</ObjectLockConfiguration>
```

개체의 보존 설정을 지정하는 방법

S3 오브젝트 잠금이 활성화된 버킷에는 S3 오브젝트 잠금 보존 설정이 있는 오브젝트와 없는 오브젝트의 조합이 포함될 수 있습니다.

오브젝트 레벨의 보존 설정은 S3 REST API를 사용하여 지정됩니다. 객체에 대한 보존 설정은 버킷의 기본 보존 설정보다 우선합니다.

각 개체에 대해 다음 설정을 지정할 수 있습니다.

- * 보존 모드 *: 규정 준수 또는 거버넌스 중 하나입니다.
- * Retain-until-date *: StorageGRID에서 개체 버전을 유지해야 하는 기간을 지정하는 날짜입니다.
  - 준수 모드에서 보존 기한이 미래인 경우 오브젝트를 검색할 수 있지만 수정하거나 삭제할 수 없습니다. 보관 기한 을 늘릴 수 있지만 이 날짜는 감소 또는 제거할 수 없습니다.
  - 거버넌스 모드에서 특별 권한이 있는 사용자는 보존 기한 설정을 무시할 수 있습니다. 보존 기간이 경과하기 전에 객체 버전을 삭제할 수 있습니다. 또한 보존 기간을 늘리거나 줄이거나 제거할 수도 있습니다.
- * 법적 증거 자료 보관 *: 개체 버전에 법적 증거 자료 보관 기능을 적용하면 해당 개체가 즉시 잠깁니다. 예를 들어 조사 또는 법적 분쟁과 관련된 객체에 법적 보류를 지정해야 할 수 있습니다. 법적 보류는 만료 날짜가 없지만 명시적으로 제거될 때까지 유지됩니다.

  개체에 대한 법적 보류 설정은 보존 모드 및 보존 기한 과 무관합니다. 개체 버전이 법적 증거 자료 보관 중인 경우 해당 버전을 삭제할 수 없습니다.

오브젝트 버전을 버킷에 추가할 때 S3 오브젝트 잠금 설정을 지정하려면"PutObject 를 선택합니다", 를 실행하거나 "CopyObject 를 선택합니다" "CreateMultptUpload 를 클릭합니다"요청을 실행하십시오.

다음을 사용할 수 있습니다.

- `x-amz-object-lock-mode`규정 준수 또는 거버넌스일 수 있습니다(대소문자 구분).

  > 을 지정한 `x-amz-object-lock-mode` 경우에는 도 지정해야 `x-amz-object-lock-retain-until-date`합니다.

- `x-amz-object-lock-retain-until-date`
  - 유지 기한 값은 형식이어야 `2020-08-10T21:46:00Z`합니다. 소수 자릿수는 허용되지만 소수점 이하 자릿수는 3자리만 유지됩니다(밀리초 단위). 다른 ISO 8601 형식은 허용되지 않습니다.
  - 보존 종료 날짜는 미래여야 합니다.
- `x-amz-object-lock-legal-hold`

  법적 증거 자료 보관(대소문자 구분)이 켜져 있는 경우, 해당 물체는 법적 증거 자료 보관 하에 배치됩니다. 법적 증거 자료 보관 기능이 꺼져 있는 경우 법적 증거 자료 보관 작업이 없습니다. 다른 값을 사용하면 400개의 잘못된 요청(InvalidArgument) 오류가 발생합니다.

이러한 요청 헤더를 사용하는 경우 다음과 같은 제한 사항에 유의하십시오.

- `Content-MD5`요청 헤더가 `PutObject` 요청에 있는 경우 요청 `x-amz-object-lock-*` 헤더가 필요합니다. `Content-MD5` CopyObject 또는 CreateMultipartUpload에는 필요하지 않습니다.
- 버킷에 S3 오브젝트 잠금이 설정되어 있지 않고 요청 헤더가 있는 경우 `x-amz-object-lock-*` 400 Bad Request(InvalidRequest) 오류가 반환됩니다.
- PutObject 요청에서는 AWS 동작을 일치시키기 위해 의 사용을 `x-amz-storage-class: REDUCED_REDUNDANCY` 지원합니다. 하지만 오브젝트가 S3 오브젝트 잠금이 설정된 버킷으로 수집되면 StorageGRID는 항상 이중 커밋 수집을 수행합니다.
- 이후의 Get 또는 HeadObject 버전 응답에는 헤더 `x-amz-object-lock-mode`, `x-amz-object-lock-retain-until-date` 및 `x-amz-object-lock-legal-hold`, 구성된 경우 요청 보낸 사람에게 올바른

권한이 있는지 여부가 `s3:Get*` 포함됩니다.

정책 조건 키를 사용하여 개체에 대해 허용되는 최소 및 최대 보존 기간을 제한할 수 `s3:object-lock-remaining-retention-days` 있습니다.

개체의 보존 설정을 업데이트하는 방법

기존 개체 버전에 대한 법적 증거 자료 보관 또는 보존 설정을 업데이트해야 하는 경우 다음 개체 하위 리소스 작업을 수행할 수 있습니다.

- `PutObjectLegalHold`

  새 법적 증거 자료 보관 값이 켜져 있으면 해당 개체는 법적 증거 자료 보관 아래에 배치됩니다. 법적 증거 자료 보관 가치가 없는 경우 법적 구속이 해제됩니다.

- `PutObjectRetention`
  - 모드 값은 규정 준수 또는 거버넌스(대/소문자 구분)일 수 있습니다.
  - 유지 기한 값은 형식이어야 `2020-08-10T21:46:00Z`합니다. 소수 자릿수는 허용되지만 소수점 이하 자릿수는 3자리만 유지됩니다(밀리초 단위). 다른 ISO 8601 형식은 허용되지 않습니다.
  - 개체 버전에 기존 보존 기한이 있는 경우 개체 버전을 늘릴 수만 있습니다. 새 값은 미래여야 합니다.

거버넌스 모드 사용 방법

권한이 있는 사용자는 `s3:BypassGovernanceRetention` 거버넌스 모드를 사용하는 개체의 활성 보존 설정을 무시할 수 있습니다. 삭제 또는 PutObjectRetention 작업은 요청 헤더를 포함해야 `x-amz-bypass-governance-retention:true` 합니다. 이러한 사용자는 다음과 같은 추가 작업을 수행할 수 있습니다.

- DeleteObject 또는 DeleteObjects 작업을 수행하여 보존 기간이 경과하기 전에 개체 버전을 삭제합니다.

  법적 증거 자료 보관 중인 객체는 삭제할 수 없습니다. 법적 증거 자료 보관 기능을 해제해야 합니다.

- 개체의 보존 기간이 경과하기 전에 개체 버전의 모드를 거버넌스에서 규정 준수로 변경하는 PutObjectRetention 작업을 수행합니다.

  규정 준수 모드를 거버넌스로 변경하는 것은 허용되지 않습니다.

- PutObjectRetention 작업을 수행하여 개체 버전의 보존 기간을 증가, 감소 또는 제거합니다.

관련 정보

- "S3 오브젝트 잠금으로 오브젝트 관리"
- "S3 오브젝트 잠금을 사용하여 오브젝트를 보존합니다"
- "Amazon Simple Storage Service 사용자 가이드: 오브젝트 잠금"

S3 라이프사이클 구성을 생성합니다

S3 라이프사이클 구성을 생성하여 StorageGRID 시스템에서 특정 오브젝트 삭제 시기를 제어할 수 있습니다.

이 섹션의 간단한 예는 S3 라이프사이클 구성에서 특정 S3 버킷에서 특정 객체가 삭제(만료)되는 시기를 제어하는 방법을 보여줍니다. 이 섹션의 예제는 설명을 위한 것입니다. S3 라이프사이클 구성 생성에 대한 자세한 내용은 를 참조하십시오 "Amazon Simple Storage Service 사용자 가이드: 객체 수명 주기 관리". StorageGRID는 만료 작업만 지원하며 전환 작업은 지원하지 않습니다.

문서 수정 상태 설정은 무엇입니까

라이프사이클 구성은 특정 S3 버킷의 오브젝트에 적용되는 규칙 세트입니다. 각 규칙은 영향을 받는 개체와 해당 개체가 만료되는 시기(특정 날짜 또는 특정 일 수 이후)를 지정합니다.

StorageGRID는 수명 주기 구성에서 최대 1,000개의 수명 주기 규칙을 지원합니다. 각 규칙에는 다음 XML 요소가 포함될 수 있습니다.

- 만료: 지정된 날짜에 도달하거나 지정된 일 수에 도달할 때 개체를 인제스트할 때로부터 개체를 삭제합니다.
- NoncurrentVersionExpiration: 지정된 일 수에 도달할 때 개체가 비전류가 되었을 때부터 개체를 삭제합니다.
- 필터(접두사, 태그)
- 상태
- ID입니다

각 오브젝트는 S3 버킷 라이프사이클 또는 ILM 정책의 보존 설정을 따릅니다. S3 버킷 라이프사이클이 구성되면 라이프사이클 만료 작업이 버킷 라이프사이클 필터와 일치하는 오브젝트에 대한 ILM 정책을 재정의합니다. 버킷 수명 주기 필터와 일치하지 않는 객체는 ILM 정책의 보존 설정을 사용합니다. 객체가 버킷 수명 주기 필터와 일치하고 만료 작업이 명시적으로 지정되지 않은 경우 ILM 정책의 보존 설정이 사용되지 않으며 객체 버전이 영구적으로 보존됩니다. 을 "S3 버킷 라이프사이클 및 ILM 정책의 우선순위 예"참조하십시오.

따라서 ILM 규칙의 배치 지침이 개체에 계속 적용되더라도 그리드에서 개체를 제거할 수 있습니다. 또는 개체에 대한 ILM 배치 지침이 만료된 후에도 개체가 그리드에 남아 있을 수 있습니다. 자세한 내용은 을 참조하십시오 "ILM이 개체 수명 전반에 걸쳐 작동하는 방식".

> 버킷 수명 주기 구성은 S3 오브젝트 잠금이 활성화된 버킷과 함께 사용할 수 있지만 버킷 수명 주기 구성은 레거시 준수 버킷에서 지원되지 않습니다.

StorageGRID는 다음 버킷 작업을 사용하여 라이프사이클 구성을 관리합니다.

- DeleteBucketLifecycle
- GetBuckLifecycleConfiguration 을 참조하십시오
- PutBucketLifecycleConfiguration을 참조하십시오

문서 수정 상태 설정 작성

라이프사이클 구성을 만드는 첫 번째 단계에서는 하나 이상의 규칙이 포함된 JSON 파일을 만듭니다. 예를 들어 이 JSON 파일에는 다음과 같은 세 가지 규칙이 포함되어 있습니다.

1. 규칙 1은 접두사/ `key2` 와 일치하고 값이 인 `tag2` 개체에만 `category1` 적용됩니다. `Expiration`매개 변수는 필터와 일치하는 개체가 2020년 8월 22일 자정에 만료되도록 지정합니다.
2. 규칙 2는 접두사/ 과 일치하는 개체에만 `category2` 적용됩니다. `Expiration`매개 변수는 필터와 일치하는 개체가 수집된 후 100일 후에 만료되도록 지정합니다.

일 수를 지정하는 규칙은 오브젝트가 수집된 시점을 기준으로 합니다. 현재 날짜가 수집 날짜와 일 수를 더한 값을 초과하면 라이프사이클 구성이 적용되는 즉시 일부 객체가 버킷에서 제거될 수 있습니다.

3. 규칙 3은 접두사/ 과 일치하는 개체에만 `category3` 적용됩니다. `Expiration`매개 변수는 일치하는 개체의 현재 버전이 아닌 경우 50일 후에 만료되도록 지정합니다.

```
{
    "Rules": [
        {
            "ID": "rule1",
            "Filter": {
                "And": {
                    "Prefix": "category1/",
                    "Tags": [
                        {
                            "Key": "key2",
                            "Value": "tag2"
                        }
                    ]
                }
            },
            "Expiration": {
                "Date": "2020-08-22T00:00:00Z"
            },
            "Status": "Enabled"
        },
        {
            "ID": "rule2",
            "Filter": {
                "Prefix": "category2/"
            },
            "Expiration": {
                "Days": 100
            },
            "Status": "Enabled"
        },
        {
            "ID": "rule3",
            "Filter": {
                "Prefix": "category3/"
            },
            "NoncurrentVersionExpiration": {
                "NoncurrentDays": 50
            },
            "Status": "Enabled"
        }
    ]
}
```

버킷에 라이프사이클 구성을 적용합니다

문서 수정 상태 구성 파일을 만든 후 PutBucketLifecycleConfiguration 요청을 실행하여 버킷에 적용합니다.

이 요청은 예제 파일의 수명주기 구성을 이름이 인 버킷의 객체에 `testbucket` 적용합니다.

```
aws s3api --endpoint-url <StorageGRID endpoint> put-bucket-lifecycle-
configuration
--bucket testbucket --lifecycle-configuration file://bktjson.json
```

수명 주기 구성이 버킷에 성공적으로 적용되었는지 확인하려면 GetBucketLifecycleConfiguration 요청을 실행합니다. 예를 들면 다음과 같습니다.

```
aws s3api --endpoint-url <StorageGRID endpoint> get-bucket-lifecycle-
configuration
 --bucket testbucket
```

성공적으로 응답하면 방금 적용한 문서 수정 상태 설정이 나열됩니다.

버킷 수명 주기 만료가 객체에 적용되는지 확인합니다

PutObject, HeadObject 또는 GetObject 요청을 실행할 때 수명 주기 구성의 만료 규칙이 특정 개체에 적용되는지 여부를 확인할 수 있습니다. 규칙이 적용되는 경우 응답에는 `Expiration` 개체가 만료되는 시점과 일치하는 만료 규칙을 나타내는 매개 변수가 포함됩니다.

> 버킷 수명 주기가 ILM을 재정의하기 때문에 `expiry-date` 표시된 날짜는 오브젝트가 삭제될 실제 날짜입니다. 자세한 내용은 을 참조하십시오 "개체 보존이 결정되는 방식".

예를 들어, 이 PutObject 요청은 2020년 6월 22일에 발행되었으며 버킷에 객체를 `testbucket` 배치합니다.

```
aws s3api --endpoint-url <StorageGRID endpoint> put-object
--bucket testbucket --key obj2test2 --body bktjson.json
```

성공 응답은 개체가 100일(2020년 10월 1일) 내에 만료되고 라이프사이클 구성의 규칙 2와 일치함을 나타냅니다.

```
{
      *"Expiration": "expiry-date=\"Thu, 01 Oct 2020 09:07:49 GMT\", rule-
id=\"rule2\"",
      "ETag": "\"9762f8a803bc34f5340579d4446076f7\""
}
```

예를 들어, 이 HeadObject 요청은 testbucket의 동일한 객체에 대한 메타데이터를 가져오는 데 사용되었습니다.

```
aws s3api --endpoint-url <StorageGRID endpoint> head-object
--bucket testbucket --key obj2test2
```

성공 응답에는 개체의 메타데이터가 포함되며 개체가 100일 후에 만료되고 규칙 2와 일치함을 나타냅니다.

```
{
      "AcceptRanges": "bytes",
      *"Expiration": "expiry-date=\"Thu, 01 Oct 2020 09:07:48 GMT\", rule-
id=\"rule2\"",
      "LastModified": "2020-06-23T09:07:48+00:00",
      "ContentLength": 921,
      "ETag": "\"9762f8a803bc34f5340579d4446076f7\""
      "ContentType": "binary/octet-stream",
      "Metadata": {}
}
```

버전 관리를 사용하는 버킷의 경우 `x-amz-expiration` 응답 헤더는 현재 버전의 객체에만 적용됩니다.

S3 REST API 구현을 위한 권장사항

StorageGRID와 함께 사용할 S3 REST API를 구현할 때는 다음 권장 사항을 따라야 합니다.

존재하지 않는 객체에 대한 헤드 권장 사항

응용 프로그램에서 개체가 실제로 존재할 것으로 예상하지 않는 경로에 개체가 있는지 정기적으로 확인하는 경우 "사용 가능"을 사용해야 "정합성"합니다. 예를 들어, 응용 프로그램이 해당 위치에 배치하기 전에 해당 위치를 헤딩하는 경우 "사용 가능한" 일관성을 사용해야 합니다.

그렇지 않으면 헤드 작업에서 객체를 찾지 못할 경우 같은 사이트에 있는 두 개 이상의 스토리지 노드를 사용할 수 없거나 원격 사이트에 연결할 수 없는 경우 500개의 내부 서버 오류가 발생할 수 있습니다.

요청을 사용하여 각 버킷에 대해 "사용 가능한" 일관성을 설정하거나 개별 API 작업에 대한 요청 헤더에서 일관성을 지정할 수 "버킷 일관성을 유지합니다"있습니다.

개체 키에 대한 권장 사항

버킷이 처음 생성된 시점을 기준으로 오브젝트 키 이름에 대한 다음 권장 사항을 따르십시오.

StorageGRID 11.4 또는 이전 버전에서 생성된 버킷

- 개체 키의 처음 네 문자로 임의 값을 사용하지 마십시오. 이는 이전 AWS에서 권장하는 키 접두사와 다릅니다. 대신 와 같이 고유하지 않은 임의 접두사를 사용합니다 `image`.
- 이전 AWS 권장 사항에 따라 키 접두사에 랜덤 및 고유 문자를 사용하려면 오브젝트 키에 디렉토리 이름이 접두사로 지정됩니다. 즉, 다음 형식을 사용합니다.

```
mybucket/mydir/f8e3-image3132.jpg
```

이 형식 대신:

```
mybucket/f8e3-image3132.jpg
```

StorageGRID 11.4 이상에서 생성된 버킷

성능 모범 사례에 맞게 개체 키 이름을 제한하는 것은 필요하지 않습니다. 대부분의 경우 개체 키 이름의 처음 4개 문자에 임의의 값을 사용할 수 있습니다.

> 단, 짧은 시간 내에 모든 오브젝트를 지속적으로 제거하는 S3 워크로드가 예외입니다. 이 사용 사례에 대한 성능 영향을 최소화하려면 키와 같은 1000개의 오브젝트마다 주요 이름의 앞부분을 다르게 지정해야 합니다. 예를 들어, S3 클라이언트가 일반적으로 초당 2,000개의 오브젝트를 기록하고 ILM 또는 버킷 라이프사이클 정책에 따라 3일 후에 모든 오브젝트를 제거한다고 가정해 보겠습니다. 성능에 미치는 영향을 최소화하기 위해 다음과 같은 패턴을 사용하여 키의 이름을 지정할 수 있습니다.
>
> `/mybucket/mydir/yyyymmddhhmmss-random_UUID.jpg`

"범위 읽기"에 대한 권장 사항

이 설정된 경우 "저장된 개체를 압축하는 전역 옵션"S3 클라이언트 응용 프로그램은 반환되는 바이트 범위를 지정하는 GetObject 작업을 수행하지 않아야 합니다. 이러한 "범위 읽기" 작업은 StorageGRID에서 요청된 바이트에 액세스하기 위해 개체의 압축을 효과적으로 해제해야 하기 때문에 비효율적입니다. 매우 큰 개체에서 작은 범위의 바이트를 요청하는 GetObject 작업은 특히 비효율적입니다. 예를 들어, 50GB의 압축된 개체에서 10MB 범위를 읽는 것은 비효율적입니다.

압축된 개체에서 범위를 읽으면 클라이언트 요청이 시간 초과될 수 있습니다.

> 개체를 압축해야 하고 클라이언트 응용 프로그램에서 범위 읽기를 사용해야 하는 경우 응용 프로그램의 읽기 시간 초과를 늘리십시오.