StorageGRID 实现 S3 REST API

StorageGRID software

NetApp
July 23, 2026

目录

StorageGRID 实现 S3 REST API

StorageGRID S3 REST API 如何解决冲突的客户端请求

冲突的客户端请求（例如两个客户端写入同一密钥）将以"最新获胜"为原则进行解决。

"最新获胜"评估的时间取决于 StorageGRID 系统何时完成给定请求，而不是 S3 客户端何时开始操作。

StorageGRID S3 REST API 如何平衡可用性和一致性

一致性提供了对象的可用性和这些对象在不同存储节点和站点之间的一致性之间的平衡。您可以根据应用程序的要求更改一致性。

默认情况下，StorageGRID 保证新创建的对象的写后读一致性。成功完成 PUT 后的任何 GET 都可以读取新写入的数据。现有对象的覆盖、元数据更新和删除最终是一致的。

如果要以不同的一致性执行对象操作，则可以：

- 请指定 每个存储桶 的一致性。
- 为 每个 API 操作 指定一致性。
- 通过执行以下任务之一更改默认网格范围的一致性：
  - 在网格管理器中，转到*配置* > 系统 > 存储设置 > 默认存储桶一致性。
  - 。

> 对网格范围一致性的更改仅适用于更改设置后创建的存储桶。要确定更改的详细信息，请参见位于 `/var/local/log`的审计日志（搜索 **consistencyLevel**）。

一致性值

一致性会影响StorageGRID用于跟踪对象的元数据在节点之间的分布方式。一致性会影响客户端请求对象的可用性。

可以将存储桶或 API 操作的一致性设置为以下值之一：

- 所有：所有节点立即接收对象元数据，否则请求将失败。
- **Strong-global**：保证所有站点上所有客户端请求的写后读一致性。配置 Quorum 语义时，适用以下行为：
  - 当网格具有三个或更多站点时，允许客户端请求的站点容错。双站点网格不具有站点故障容限。
  - 如果一个站点关闭，以下 S3 操作将不会成功：
    - DeleteBucketEncryption
    - PutBucketBranch
    - PutBucketEncryption
    - PutBucketVersioning

- PutObjectLegalHold
- PutObjectLockConfiguration
- PutObjectRetention

如有需要，您可以 "配置 StorageGRID 仲裁语义以实现强全局一致性"。

- **Strong-site**：对象元数据会立即分发到站点上的其他节点。保证站点内所有客户端请求的写后读一致性。
- **Read-after-new-write**：为新对象提供写后读一致性，并为对象更新提供最终一致性。提供高可用性和数据保护保证。建议在大多数情况下使用。
- 可用：为新对象和对象更新提供最终一致性。对于 S3 存储桶，仅根据需要使用（例如，对于包含很少读取的日志值的存储桶，或者对于不存在的键的 HEAD 或 GET 操作）。不支持 S3 FabricPool 存储桶。

使用"新写后读"和"可用"一致性

当 HEAD 或 GET 操作使用"新写后读"一致性时，StorageGRID 将按以下步骤执行查找：

- 它首先使用低一致性查找对象。
- 如果查找失败，它会重复查找下一个一致性值，直到达到与 strong-global 行为等效的一致性。

如果 HEAD 或 GET 操作使用"新写后读"一致性，但对象不存在，则对象查找将始终达到与 strong-global 行为等效的一致性。由于这种一致性要求每个站点提供多个对象元数据副本，因此如果同一站点上的两个或多个存储节点不可用，则可能会收到大量 500 Internal Server 错误。

除非您需要类似于 Amazon S3 的一致性保证，否则可以通过将一致性设置为"可用"来防止 HEAD 和 GET 操作出现这些错误。当 HEAD 或 GET 操作使用"可用"一致性时，StorageGRID 仅提供最终一致性。它不会以增加一致性的方式重试失败的操作，因此不需要提供多个对象元数据副本。

指定 API 操作的一致性

要为单个 API 操作设置一致性，必须为该操作支持一致性值，并且必须在请求标头中指定一致性。此示例将 GetObject 操作的一致性设置为"Strong-site"。

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

您必须对PutObject和GetObject操作使用相同的一致性。

指定bucket的一致性

要设置存储桶的一致性，您可以使用 StorageGRID "PUT Bucket 一致性" 请求。或者，您可以从租户管理器 "更改存储桶的一致性"。

在设置存储区的一致性时，请注意以下几点：

- 设置存储桶的一致性决定了对存储桶中的对象或存储桶配置执行的 S3 操作所使用的一致性。它不影响存储桶本身的操作。
- 单个 API 操作的一致性将覆盖存储桶的一致性。
- 一般来说，存储桶应使用默认的一致性，"Read-after-new-write."如果请求无法正常工作，请尽可能更改应用程序客户端行为。或者，配置客户端以指定每个 API 请求的一致性。仅作为最后手段，在存储桶级别设置一致性。

一致性和ILM规则如何相互作用以影响数据保护

您选择的一致性和 ILM 规则都会影响对象的保护方式。这些设置可以相互影响。

例如，存储对象时使用的一致性会影响对象元数据的初始放置，而为 ILM 规则选择的摄取行为会影响对象副本的初始放置。由于 StorageGRID 需要访问对象的元数据及其数据来满足客户端请求，因此为一致性和摄取行为选择匹配的保护级别可以提供更好的初始数据保护和更可预测的系统响应。

以下"载入选项"可用于 ILM 规则：

双提交
: StorageGRID 立即生成对象的临时副本，并将成功返回给客户端。ILM 规则中指定的副本将在可能时创建。

严格
: 在成功返回到客户端之前，必须创建 ILM 规则中指定的所有副本。

平衡
: StorageGRID 尝试在导入时生成 ILM 规则中指定的所有副本；如果无法实现，则生成临时副本并将成功状态返回给客户端。ILM 规则中指定的副本将在可能时生成。

一致性和 **ILM** 规则如何相互作用的示例

假设您有一个具有以下 ILM 规则和以下一致性的三站点网格：

- ILM 规则：创建三个对象副本，一个在本地站点，一个在每个远程站点。使用严格的载入行为。
- 一致性：强全局（对象元数据立即分发到多个站点）。

当客户端将对象存储到网格中时，StorageGRID 会创建所有三个对象副本并将元数据分发到多个站点，然后再将成功状态返回给客户端。

对象在接收成功消息时受到完全保护，不会丢失。例如，如果本地站点在载入后不久丢失，则对象数据和对象元数据的副本仍然存在于远程站点。该对象可从其他站点完全检索。

如果改为使用相同的 ILM 规则和强站点一致性，则在对象数据复制到远程站点但对象元数据分发到远程站点之前，客户端可能会收到一条成功消息。在这种情况下，对象元数据的保护级别与对象数据的保护级别不匹配。如果本地站点在载入后不久丢失，则对象元数据将丢失。无法检索对象。

一致性和 ILM 规则之间的相互关系可能很复杂。如需帮助，请联系 NetApp。

StorageGRID 如何使用对象版本控制

如果要保留每个对象的多个版本，可以设置存储桶的版本控制状态。启用存储桶的版本控

制可以帮助防止意外删除对象，并使您能够检索和还原对象的早期版本。

StorageGRID 系统实现了支持大多数功能的版本控制，但有一些限制。StorageGRID 最多支持每个对象的 10,000 个版本。

对象版本控制可以与 StorageGRID 信息生命周期管理 (ILM) 相结合，也可以与 S3 存储桶生命周期配置相结合。必须为每个存储桶显式启用版本控制。当对存储桶启用版本控制时，添加到存储桶的每个对象都会被分配一个版本 ID，该 ID 由 StorageGRID 系统生成。

不支持使用 MFA（多因素身份验证）删除。

> 只能在使用 StorageGRID 10.3 或更高版本创建的存储桶上启用版本控制。

ILM 和版本控制

ILM 策略应用于对象的每个版本。ILM 扫描过程会持续扫描所有对象，并根据当前 ILM 策略对其进行重新评估。对 ILM 策略所做的任何更改都将应用于以前接收的所有对象。如果启用了版本控制，这包括先前接收的版本。ILM 扫描将新的 ILM 更改应用于先前接收的对象。

对于启用版本控制的存储桶中的 S3 对象，版本控制支持允许您创建使用"非当前时间"作为参考时间的 ILM 规则（对于 "创建 ILM 规则向导的步骤 1" 中的"仅将此规则应用于较旧的对象版本？"问题，请选择*是*）。更新对象时，其以前的版本将变为非当前版本。使用"非当前时间"筛选器允许您创建策略，以减少以前版本的对象对存储的影响。

> 使用多部分上传操作上传对象的新版本时，对象原始版本的非当前时间反映的是为新版本创建多部分上传的时间，而不是完成多部分上传的时间。在某些情况下，原始版本的非当前时间可能比当前版本早数小时或数天。

相关信息

- "如何删除S3版本控制对象"
- "S3 版本化对象的 ILM 规则和策略（示例 4）"。

使用 S3 REST API 配置 StorageGRID S3 对象锁定

如果为您的 StorageGRID 系统启用了全局 S3 对象锁定设置，则可以创建启用 S3 对象锁定的存储桶。您可以为每个存储桶指定默认保留，或为每个对象版本指定保留设置。

如何为存储桶启用 S3 对象锁定

如果为 StorageGRID 系统启用了全局 S3 对象锁定设置，则可以选择在创建每个存储桶时启用 S3 对象锁定。

S3 Object Lock 是一个永久设置，只能在创建存储桶时启用。创建存储桶后，无法添加或禁用 S3 Object Lock。

要为存储桶启用 S3 Object Lock，请使用以下任一方法：

- 使用租户管理器创建存储桶。请参阅"创建 S3 存储桶"。
- 使用带有 `x-amz-bucket-object-lock-enabled`请求标头的 CreateBucket 请求创建存储桶。请参

阅"${post_edited_translations.segment}"。

S3 Object Lock 需要存储桶版本控制，在创建存储桶时自动启用。您不能暂停存储桶的版本控制。请参阅 "对象版本控制"。

存储桶的默认保留设置

当为存储桶启用 S3 Object Lock 时，您可以选择为存储桶启用默认保留，并指定默认保留模式和默认保留期限。

默认保留模式

- 在合规模式下：
  - 在达到其保留截止日期之前，无法删除此对象。
  - 对象的保留截止日期可以延长，但不能缩短。
  - 在达到该日期之前，无法删除对象的 retain-until-date。
- 在治理模式下：
  - 具有 `s3:BypassGovernanceRetention`权限的用户可以使用 `x-amz-bypass-governance-retention: true` 请求标头绕过保留设置。
  - 这些用户可以在达到保留截止日期之前删除对象版本。
  - 这些用户可以增加、减少或删除对象的保留截止日期。

默认保留期限

每个存储桶可以具有以年或天为单位指定的默认保留期限。

如何设置存储区的默认保留时间

要设置存储区的默认保留，请使用以下任一方法：

- 从租户管理器管理存储桶设置。请参阅 "${post_edited_translations.segment}" 和 "更新 S3 对象锁定默认保留"。
- 向存储桶发出 PutObjectLockConfiguration 请求，以指定默认模式和默认天数或年数。

PutObjectLockConfiguration

该 PutObjectLockConfiguration 请求允许您为启用了 S3 对象锁的存储桶设置和修改默认保留模式和默认保留期。您还可以删除以前配置的默认保留设置。

将新对象版本接收到存储桶时，如果未指定 `x-amz-object-lock-mode`和 `x-amz-object-lock-retain-until-date`，则应用默认保留模式。如果未指定 `x-amz-object-lock-retain-until-date`，则使用默认保留期限来计算保留截止日期。

如果在摄取对象版本后修改了默认保留期限，则对象版本的 retain-until-date 保持不变，不会使用新的默认保留期限重新计算。

您必须具有 `s3:PutBucketObjectLockConfiguration`权限，或者是 account root，才能完成此操作。

必须在 PUT 请求中指定 `Content-MD5`请求标头。

请求示例

此示例为存储桶启用 S3 Object Lock，并将默认保留模式设置为 COMPLIANCE，将默认保留期限设置为 6 年。

```
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>
```

如何确定存储桶的默认保留时间

要确定是否已为存储桶启用 S3 Object Lock 并查看默认保留模式和保留期限，请使用以下任一方法：

- 在租户管理器中查看存储桶。请参阅"查看 S3 存储桶"。
- 发出 GetObjectLockConfiguration 请求。

GetObjectLockConfiguration

该 GetObjectLockConfiguration 请求允许您确定是否已为存储桶启用 S3 对象锁定，如果已启用，请查看是否为该存储桶配置了默认保留模式和保留期。

将新对象版本载入存储桶时，如果未指定 `x-amz-object-lock-mode`，则应用默认保留模式。如果未指定 `x-amz-object-lock-retain-until-date`，则使用默认保留期限来计算保留截止日期。

您必须具有 `s3:GetBucketObjectLockConfiguration` 权限或是账户根用户，才能完成此操作。

请求示例

```
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
```

响应示例

```
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>
```

如何指定对象的保留设置

启用了 S3 Object Lock 的存储桶可以包含具有和不具有 S3 Object Lock 保留设置的对象的组合。

对象级别保留设置使用 S3 REST API 指定。对象的保留设置将覆盖存储桶的任何默认保留设置。

可以为每个对象指定以下设置：

- **Retention mode**：COMPLIANCE 或 GOVERNANCE。
- **Retain-until-date**：指定对象版本必须由 StorageGRID 保留多长时间的日期。
  - 在合规模式下，如果保留截止日期在未来，则可以检索对象，但不能对其进行修改或删除。保留截止日期可以延长，但不能缩短或删除。
  - 在 GOVERNANCE 模式下，具有特殊权限的用户可以绕过 retain-until-date 设置。他们可以在保留期结

束之前删除对象版本。他们还可以增加、减少甚至取消 retain-until-date。

- 法律冻结：对对象版本应用法律冻结会立即锁定该对象。例如，您可能需要对与调查或法律纠纷相关的对象进行法律冻结。法律冻结没有到期日期，但在明确移除之前会一直存在。

  对象的法律冻结设置与保留模式和保留截止日期无关。如果对象版本处于法定冻结状态，任何人都无法删除该版本。

要在将对象版本添加到存储桶时指定 S3 对象锁定设置，请发出 "放置对象"、"CopyObject" 或 "CreateMultipartUpload" 请求。

可以使用以下项：

- `x-amz-object-lock-mode`，可以是 COMPLIANCE 或 GOVERNANCE（区分大小写）。

  > 如果指定 `x-amz-object-lock-mode`，则还必须指定 `x-amz-object-lock-retain-until-date`。

- `x-amz-object-lock-retain-until-date`
  - retain-until-date 值必须采用以下格式 `2020-08-10T21:46:00Z`。允许使用小数秒，但仅保留 3 位小数（精度为毫秒）。不允许使用其他 ISO 8601 格式。
  - 保留截止日期必须在未来。
- `x-amz-object-lock-legal-hold`

  如果法律保留已开启（区分大小写），则对象将被置于法律保留状态。如果法律保留已关闭，则不会进行法律保留。任何其他值都会导致 400 Bad Request (InvalidArgument) 错误。

如果使用以下任意请求标头，请留意以下限制：

- 如果在 PutObject 请求中存在任何 `x-amz-object-lock-*` 请求标头，则必须提供 `Content-MD5` 请求标头。对于 CopyObject 或 CreateMultipartUpload，不需要 `Content-MD5`。
- 如果存储桶未启用 S3 Object Lock 且存在 `x-amz-object-lock-*`请求标头，则返回 400 Bad Request (InvalidRequest) 错误。
- PutObject 请求支持使用 `x-amz-storage-class: REDUCED_REDUNDANCY`来匹配 AWS 行为。但是，当对象被摄取到启用了 S3 对象锁定的存储桶中时，StorageGRID 将始终执行双提交摄取。
- 后续的 GET 或 HeadObject 版本响应将包括标头 `x-amz-object-lock-mode`、 `x-amz-object-lock-retain-until-date` 和 `x-amz-object-lock-legal-hold`（如果已配置并且请求发送者具有正确的 `s3:Get*`权限）。

您可以使用 `s3:object-lock-remaining-retention-days`策略条件键来限制对象的最短和最长允许保留期限。

如何更新对象的保留设置

如果需要更新现有对象版本的法律保留或保留设置，可以执行以下对象子资源操作：

- `PutObjectLegalHold`

  如果新的法定冻结值为 ON，则该对象将被置于法定冻结状态。如果法定冻结值为 OFF，则解除法定冻结。

- `PutObjectRetention`
  - 模式值可以是 COMPLIANCE 或 GOVERNANCE（区分大小写）。
  - retain-until-date 值必须采用以下格式 `2020-08-10T21:46:00Z`。允许使用小数秒，但仅保留 3 位小数（精度为毫秒）。不允许使用其他 ISO 8601 格式。
  - 如果对象版本具有现有的 retain-until-date，则只能增加它。新值必须是将来的日期。

如何使用 **GOVERNANCE** 模式

拥有 `s3:BypassGovernanceRetention`权限的用户可以绕过使用治理模式的对象的活动保留设置。任何 DELETE 或 PutObjectRetention 操作都必须包含 `x-amz-bypass-governance-retention:true`请求标头。这些用户可以执行以下附加操作：

- 执行 DeleteObject 或 DeleteObjects 操作以在对象版本的保留期结束之前将其删除。

  处于合法保留状态的对象无法删除。必须关闭合法保留。

- 在对象的保留期限到期之前，执行将对象版本的模式从 GOVERNANCE 更改为 COMPLIANCE 的 PutObjectRetention 操作。

  绝不允许将模式从 COMPLIANCE 更改为 GOVERNANCE。

- 执行 PutObjectRetention 操作以增加、减少或删除对象版本的保留期限。

相关信息

- "使用 S3 Object Lock 管理对象"
- "使用 S3 Object Lock 保留对象"
- "Amazon Simple Storage Service 用户指南：锁定对象"

了解 **StorageGRID** 的 **S3** 生命周期配置

您可以创建 S3 生命周期配置，以控制何时从 StorageGRID 系统中删除特定对象。

本节中的简单示例说明了 S3 生命周期配置如何控制从特定 S3 存储桶删除（过期）某些对象的时间。本节中的示例仅用于说明目的。有关创建 S3 生命周期配置的完整详细信息，请参阅 "Amazon Simple Storage Service 用户指南：对象生命周期管理"。请注意，StorageGRID 仅支持 Expiration 操作；不支持 Transition 操作。

什么是生命周期配置

生命周期配置是应用于特定 S3 存储桶中的对象的一组规则。每个规则都指定受影响的对象以及这些对象何时过期（在特定日期或若干天后）。

StorageGRID 在生命周期配置中最多支持 1,000 个生命周期规则。每个规则可以包含以下 XML 元素：

- 过期：当达到指定日期或从摄取对象时起达到指定天数时，删除对象。
- NoncurrentVersionExpiration：当对象变为非当前版本后，达到指定天数时删除该对象。
- 筛选（前缀、标签）

- 状态
- ID

每个对象都遵循 S3 存储桶生命周期或 ILM 策略的保留设置。配置 S3 存储桶生命周期后，生命周期到期操作将覆盖与存储桶生命周期筛选器匹配的对象的 ILM 策略。与存储桶生命周期筛选器不匹配的对象使用 ILM 策略的保留设置。如果对象与存储桶生命周期筛选器匹配，并且未明确指定过期操作，则不会使用 ILM 策略的保留设置，这意味着对象版本将永久保留。请参阅 "S3 存储桶生命周期和 ILM 策略的示例优先级"。

因此，即使ILM规则中的放置指令仍然适用于对象，也可能从网格中删除对象。或者，即使在针对对象的任何ILM放置指令失效后，也可能会将对象保留在网格上。有关详细信息，请参阅 "ILM在对象整个生命周期中的运作方式"。

> Bucket 生命周期配置可用于已启用 S3 Object Lock 的 Bucket，但旧版兼容 Bucket 不支持 Bucket 生命周期配置。

StorageGRID 支持使用以下存储桶操作来管理生命周期配置：

- DeleteBucketLifecycle
- GetBucketLifecycleConfiguration
- PutBucketLifecycleConfiguration

创建生命周期配置

创建生命周期配置的第一步是创建包含一个或多个规则的 JSON 文件。例如，此 JSON 文件包括三个规则，如下所示：

1. 规则 1 仅适用于匹配前缀 `category1/`且 `key2`值为 `tag2`的对象。该 `Expiration`参数指定与筛选器匹配的对象将于 2020 年 8 月 22 日午夜到期。
2. 规则2仅适用于匹配前缀 `category2/`的对象。该 `Expiration`参数指定与筛选器匹配的对象将在摄入后100天过期。

   > 指定天数的规则与摄取对象的时间有关。如果当前日期超过摄取日期加上天数，则某些对象可能会在应用生命周期配置后立即从存储桶中删除。

3. 规则3仅适用于与前缀 `category3/`匹配的对象。该 `Expiration`参数指定匹配对象的任何非当前版本将在其变为非当前版本后50天过期。

```
{
    "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"
        }
    ]
}
```

将生命周期配置应用于存储桶

创建生命周期配置文件后，您可以通过发出 PutBucketLifecycleConfiguration 请求将其应用到存储桶。

此请求将示例文件中的生命周期配置应用于名为 `testbucket`的存储桶中的对象。

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

要验证生命周期配置是否已成功应用于存储桶，请发出 GetBucketLifecycleConfiguration 请求。例如：

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

成功的响应会列出您刚刚应用的生命周期配置。

验证存储桶生命周期过期是否适用于对象

您可以确定在发出 PutObject、HeadObject 或 GetObject 请求时，生命周期配置中的过期规则是否适用于特定对象。如果规则适用，则响应包含一个 `Expiration`参数，该参数指示对象何时过期以及匹配了哪个过期规则。

> 由于存储桶生命周期会覆盖 ILM，因此 `expiry-date`显示的是对象将被删除的实际日期。有关详细信息，请参见"如何确定对象保留"。

例如，此 PutObject 请求于 2020 年 6 月 22 日发出，并将对象放入 `testbucket`存储桶中。

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

成功响应指示对象将在 100 天后（2020 年 10 月 1 日）过期，并且它与生命周期配置的规则 2 匹配。

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

例如，此 HeadObject 请求用于获取 testbucket 存储桶中相同对象的元数据。

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

成功响应包括对象的元数据，并指示对象将在 100 天后过期，且与规则 2 匹配。

```
{
      "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": {}
}
```

对于启用了版本控制的存储桶， `x-amz-expiration` 响应标头仅适用于当前版本的对象。

使用 StorageGRID 实施 S3 REST API 的建议

在实施与 StorageGRID 配合使用的 S3 REST API 时，应遵循以下建议。

针对不存在对象的 HEAD 建议

如果您的应用程序定期检查对象是否存在于您不希望对象实际存在的路径上，则应使用"可用""一致性"。例如，如果您的应用程序在 PUT 某个位置之前对该位置执行 HEAD 操作，则应使用"可用"一致性。

否则，如果 HEAD 操作未找到对象，当同一站点上的两个或多个存储节点不可用或无法访问远程站点时，您可能会收到大量 500 内部服务器错误。

您可以使用"PUT Bucket 一致性"请求为每个存储桶设置"可用"一致性，也可以在单个 API 操作的请求标头中指定一致性。

对象键的建议

根据首次创建存储桶的时间，遵循以下有关对象键名称的建议。

在 **StorageGRID 11.4** 或更早版本中创建的存储桶

- 请勿将随机值用作对象键的前四个字符。这与之前 AWS 对密钥前缀的建议形成鲜明对比。而是使用非随机、非唯一的前缀，例如 `image`。
- 如果您确实遵循以前的 AWS 建议，在密钥前缀中使用随机和唯一字符，请使用目录名称作为对象密钥的前缀。也就是说，使用此格式：

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

代替此格式：

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

在 **StorageGRID 11.4** 或更高版本中创建的存储桶

不需要限制对象键名称以满足性能最佳实践。在大多数情况下，您可以为对象键名称的前四个字符使用随机值。

> 一个例外是 S3 工作负载在短时间内连续删除所有对象。为了最大限度地减少此用例的性能影响，请每隔几千个对象更改密钥名称的前导部分，例如日期。例如，假设 S3 客户端通常每秒写入 2,000 个对象，而 ILM 或存储桶生命周期策略会在三天后删除所有对象。为了最大限度地减少对性能的影响，您可以使用以下模式命名键： `/mybucket/mydir/yyyymmddhhmmss-random_UUID.jpg`

"范围读取"的建议

如果启用了"用于压缩存储对象的全局选项"，S3 客户端应用程序应避免执行指定返回字节范围的 GetObject 操作。这些"范围读取"操作效率低下，因为 StorageGRID 必须有效地解压缩对象才能访问所请求的字节。从非常大的对象请求小范围字节的 GetObject 操作效率尤其低下；例如，从 50 GB 压缩对象读取 10 MB 范围是低效的。

如果从压缩对象读取范围，则客户端请求可能会超时。

> 如果需要压缩对象，并且客户端应用程序必须使用范围读取，请增加应用程序的读取超时时间。