Como StorageGRID implementa a API REST S3

StorageGRID software

NetApp
July 23, 2026

Índice

Como StorageGRID implementa a API REST S3

Como a StorageGRID API REST resolve solicitações conflitantes do cliente

Solicitações conflitantes de clientes, como dois clientes escrevendo na mesma chave, são resolvidas com base no princípio "o mais recente vence".

O momento da avaliação "latest-wins" é baseado em quando o sistema StorageGRID conclui uma determinada solicitação, e não em quando os clientes S3 iniciam uma operação.

Como StorageGRID API REST S3 equilibra disponibilidade e consistência

A consistência proporciona um equilíbrio entre a disponibilidade dos objetos e a consistência desses objetos em diferentes Storage Nodes e sites. Você pode alterar a consistência conforme necessário para sua aplicação.

Por padrão, StorageGRID garante a consistência de leitura após gravação para objetos recém-criados. Qualquer GET subsequente a um PUT concluído com sucesso poderá ler os dados recém-gravados. Sobrescritas de objetos existentes, atualizações de metadados e exclusões são eventualmente consistentes.

Se você deseja executar operações de objetos com um nível de consistência diferente, você pode:

- Especifique uma consistência para cada bucket.
- Especifique uma consistência para cada operação de API.
- Altere a consistência padrão de toda a grade executando uma das seguintes tarefas:
  - No Grid Manager, acesse **Configuração** > **Sistema** > **Configurações de armazenamento** > **Consistência padrão do bucket**.
  - .

> Uma alteração na consistência de toda a grade aplica-se apenas aos buckets criados após a alteração da configuração. Para determinar os detalhes de uma alteração, consulte o log de auditoria localizado em `/var/local/log` (pesquise por **consistencyLevel**).

Valores de consistência

A consistência afeta a forma como os metadados que StorageGRID usa para rastrear objetos são distribuídos entre os nós. A consistência afeta a disponibilidade de objetos para solicitações de clientes.

Você pode definir a consistência de um bucket ou de uma operação de API para um dos seguintes valores:

- **Todos**: Todos os nós recebem metadados de objeto imediatamente ou a solicitação falhará.
- **Strong-global**: Garante a consistência de leitura após gravação para todas as solicitações do cliente em todos os sites. Quando a semântica de Quorum está configurada, os seguintes comportamentos se aplicam:

    - Permite tolerância a falha do local para solicitações de clientes quando as grades têm três ou mais sites. Grades com dois sites não terão tolerância a falha do local.
    - As seguintes operações do S3 não serão bem-sucedidas com um site inativo:
        - DeleteBucketEncryption
        - PutBucketBranch
        - PutBucketEncryption
        - PutBucketVersioning
        - PutObjectLegalHold
        - PutObjectLockConfiguration
        - PutObjectRetention

    Se necessário, você pode "configurar a semântica de quorum do StorageGRID para consistência global forte".

- **Strong-site**: os metadados do objeto são distribuídos imediatamente para outros nós no site. Garante a consistência de leitura após gravação para todas as solicitações do cliente dentro de um site.
- **Leitura após nova gravação**: Oferece consistência de leitura após gravação para novos objetos e consistência eventual para atualizações de objetos. Oferece alta disponibilidade e garantias de proteção de dados. Recomendado para a maioria dos casos.
- **Disponível**: Oferece consistência eventual tanto para novos objetos quanto para atualizações de objetos. Para buckets do S3, use somente quando necessário (por exemplo, para um bucket que contém valores de log que são lidos raramente ou para operações HEAD ou GET em chaves que não existem). Não é compatível com buckets do S3 FabricPool.

Use as consistências "Read-after-new-write" e "Available"

Quando uma operação HEAD ou GET utiliza a consistência "Leitura após nova gravação", StorageGRID realiza a pesquisa em várias etapas, da seguinte forma:

- Primeiro, ele busca o objeto usando uma consistência baixa.
- Caso essa busca falhe, ela se repete no próximo valor de consistência até atingir uma consistência equivalente ao comportamento de strong-global.

Se uma operação HEAD ou GET usar a consistência "Read-after-new-write", mas o objeto não existir, a busca pelo objeto sempre atingirá uma consistência equivalente ao comportamento para strong-global. Como essa consistência exige que várias cópias dos metadados do objeto estejam disponíveis em cada site, você pode receber um grande número de erros 500 Internal Server se dois ou mais Storage Nodes no mesmo site estiverem indisponíveis.

A menos que você precise de garantias de consistência semelhantes às da Amazon S3, você pode evitar esses erros para operações HEAD e GET definindo a consistência como "Disponível". Quando uma operação HEAD ou GET usa a consistência "Disponível", StorageGRID fornece apenas consistência eventual. Não tenta novamente uma operação com falha em níveis de consistência crescentes, portanto, não exige que várias cópias dos metadados do objeto estejam disponíveis.

Especifique a consistência para a operação da API

Para definir a consistência de uma operação de API específica, os valores de consistência devem ser suportados pela operação e você deve especificar a consistência no cabeçalho da solicitação. Este exemplo define a consistência como "Strong-site" para uma operação GetObject.

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

Você deve usar a mesma consistência para as operações PutObject e GetObject.

Especifique a consistência para o bucket

Para definir a consistência do bucket, você pode usar a solicitação "Consistência de PUT Bucket" do StorageGRID. Ou você pode "alterar a consistência de um bucket" a partir do Tenant Manager.

Ao definir a consistência de um bucket, esteja ciente do seguinte:

- A configuração de consistência para um bucket determina qual consistência é usada para as operações S3 realizadas nos objetos no bucket ou na configuração do bucket. Ela não afeta as operações no próprio bucket.
- A consistência de uma operação individual da API tem precedência sobre a consistência do bucket.
- Em geral, os buckets devem usar a consistência padrão, "Leitura após nova gravação". Se as solicitações não estiverem funcionando corretamente, altere o comportamento do cliente do aplicativo, se possível. Ou configure o cliente para especificar a consistência para cada solicitação de API. Defina a consistência no nível do bucket somente como último recurso.

Como as regras de consistência e ILM interagem para afetar a proteção de dados

Tanto a sua escolha de consistência quanto a sua regra ILM afetam a forma como os objetos são protegidos. Essas configurações podem interagir.

Por exemplo, a consistência usada quando um objeto é armazenado afeta o posicionamento inicial dos metadados do objeto, enquanto o comportamento de ingestão selecionado para a regra de gerenciamento de ciclo de vida (ILM) afeta o posicionamento inicial das cópias do objeto. Como StorageGRID requer acesso tanto aos metadados quanto aos dados de um objeto para atender às solicitações do cliente, selecionar níveis de proteção correspondentes para a consistência e o comportamento de ingestão pode proporcionar melhor proteção inicial dos dados e respostas mais previsíveis do sistema.

As seguintes "opções de ingestão" estão disponíveis para as regras do ILM:

Commit duplo
: StorageGRID cria imediatamente cópias provisórias do objeto e retorna sucesso ao cliente. Cópias especificadas na regra ILM são feitas quando possível.

Estrito
: Todas as cópias especificadas na regra ILM devem ser feitas antes que a mensagem de sucesso seja retornada ao cliente.

Equilibrado
: StorageGRID tenta criar todas as cópias especificadas na regra ILM durante a ingestão; se isso não for possível, cópias provisórias são criadas e uma mensagem de sucesso é retornada ao cliente. As cópias especificadas na regra ILM são criadas sempre que possível.

Exemplo de como a consistência e a regra de ILM podem interagir

Suponha que você tenha uma grade de três sites com a seguinte regra ILM e a seguinte consistência:

- **Regra ILM**: Crie três cópias do objeto, uma no local e uma em cada local remoto. Use o comportamento de ingestão estrito.
- **Consistência**: Global forte (os metadados do objeto são distribuídos imediatamente para vários sites).

Quando um cliente armazena um objeto na grid, StorageGRID cria as três cópias do objeto e distribui os metadados para vários sites antes de retornar sucesso ao cliente.

O objeto fica totalmente protegido contra perda no momento em que a mensagem de ingestão bem-sucedida é recebida. Por exemplo, se o local remoto for perdido logo após a ingestão, cópias dos dados do objeto e dos metadados do objeto ainda existirão nos locais remotos. O objeto poderá ser recuperado integralmente dos outros locais.

Se, em vez disso, você usasse a mesma regra ILM e a consistência forte do site, o cliente poderia receber uma mensagem de sucesso após os dados do objeto serem replicados para os locais remotos, mas antes que os metadados do objeto fossem distribuídos lá. Nesse caso, o nível de proteção dos metadados do objeto não corresponde ao nível de proteção dos dados do objeto. Se o local remoto for perdido logo após a ingestão, os metadados do objeto serão perdidos. O objeto não pode ser recuperado.

A inter-relação entre as regras de consistência e as regras de ILM pode ser complexa. Entre em contato com a NetApp se você precisar de assistência.

Como StorageGRID usa o versionamento de objetos

Você pode definir o estado de versionamento de um bucket se desejar manter várias versões de cada objeto. Habilitar o versionamento para um bucket pode ajudar a proteger contra a exclusão acidental de objetos e permite que você recupere e restaure versões anteriores de um objeto.

O sistema StorageGRID implementa o versionamento com suporte para a maioria dos recursos, com algumas limitações. StorageGRID suporta até 10.000 versões de cada objeto.

O versionamento de objetos pode ser combinado com o gerenciamento do ciclo de vida das informações (ILM) do StorageGRID ou com a configuração do ciclo de vida do bucket S3. Você deve habilitar explicitamente o versionamento para cada bucket. Quando o versionamento está habilitado para um bucket, cada objeto adicionado ao bucket recebe um ID de versão, que é gerado pelo sistema StorageGRID.

O uso de MFA (autenticação multifator) com a função Delete não é suportado.

> O versionamento só pode ser ativado em buckets criados com StorageGRID versão 10.3 ou posterior.

ILM e versionamento

As políticas de ILM são aplicadas a cada versão de um objeto. Um processo de varredura de ILM examina continuamente todos os objetos e os reavalia em relação à política de ILM atual. Quaisquer alterações que você fizer nas políticas de ILM são aplicadas a todos os objetos previamente ingeridos. Isso inclui versões previamente ingeridas se o versionamento estiver habilitado. A varredura de ILM aplica as novas alterações de ILM aos objetos previamente ingeridos.

Para objetos S3 em buckets com versionamento habilitado, o suporte a versionamento permite que você crie regras ILM que usam "Tempo não atual" como tempo de referência (selecione **Sim** para a pergunta "Aplicar esta regra somente a versões antigas do objeto?" em "Etapa 1 do assistente Criar uma regra ILM"). Quando um objeto é atualizado, suas versões anteriores se tornam não atuais. Usar um filtro de "Tempo não atual" permite que você crie políticas que reduzem o impacto de armazenamento das versões anteriores dos objetos.

> Ao carregar uma nova versão de um objeto usando uma operação de upload multipart, a hora não atual da versão original do objeto reflete o momento em que o upload multipart foi criado para a nova versão, não quando o upload multipart foi concluído. Em casos limitados, a hora não atual da versão original pode ser horas ou dias anterior à hora da versão atual.

Informações relacionadas

- "Como os objetos versionados do S3 são excluídos"
- "Regras e políticas do ILM para objetos versionados do S3 (Exemplo 4)".

Use a API REST S3 para configurar o S3 Object Lock do StorageGRID

Se a configuração global de Bloqueio de Objetos S3 estiver habilitada para o seu sistema StorageGRID, você pode criar buckets com o Bloqueio de Objetos S3 ativado. Você pode especificar o período de retenção padrão para cada bucket ou as configurações de retenção para cada versão de objeto.

Como habilitar o S3 Object Lock para um bucket

Se a configuração global de S3 Object Lock estiver ativada para o seu sistema StorageGRID, você pode, opcionalmente, ativar o S3 Object Lock ao criar cada bucket.

O Bloqueio de Objetos do S3 é uma configuração permanente que só pode ser ativada ao criar um bucket. Você não pode adicionar ou desativar o Bloqueio de Objetos do S3 depois que um bucket é criado.

Para ativar o S3 Object Lock para um bucket, use um destes métodos:

- Crie o bucket usando o Gerenciador de Locatários. Consulte "Criar bucket S3".
- Crie o bucket usando uma solicitação CreateBucket com o cabeçalho de solicitação `x-amz-bucket-object-lock-enabled`. Veja "Operações em buckets".

O S3 Object Lock requer o versionamento do bucket, que é ativado automaticamente quando o bucket é criado. Você não pode suspender o versionamento do bucket. Consulte "Versionamento de objetos".

Configurações de retenção padrão para um bucket

Quando o S3 Object Lock está habilitado para um bucket, você pode opcionalmente habilitar a retenção padrão para o bucket e especificar um modo de retenção padrão e um período de retenção padrão.

Modo de retenção padrão

- No modo COMPLIANCE:

    - O objeto não pode ser excluído até que sua retain-until-date seja atingida.
    - A data de retenção do objeto pode ser aumentada, mas não pode ser diminuída.
    - A data de retenção do objeto não pode ser removida até que essa data seja atingida.
- No modo GOVERNANÇA:
    - Usuários com a `s3:BypassGovernanceRetention` permissão podem usar o `x-amz-bypass-governance-retention: true` cabeçalho da solicitação para ignorar as configurações de retenção.
    - Esses usuários podem excluir uma versão de objeto antes que a data de retenção seja atingida.
    - Esses usuários podem aumentar, diminuir ou remover a retain-until-date de um objeto.

Período de retenção padrão

Cada bucket pode ter um período de retenção padrão especificado em anos ou dias.

Como definir a retenção padrão para um bucket

Para definir a retenção padrão de um bucket, use um destes métodos:

- Gerencie as configurações do bucket a partir do Tenant Manager. Consulte "Crie um bucket S3" e "Atualizar retenção padrão do S3 Object Lock".
- Envie uma solicitação PutObjectLockConfiguration para o bucket para especificar o modo padrão e o número padrão de dias ou anos.

PutObjectLockConfiguration

A solicitação PutObjectLockConfiguration permite que você defina e modifique o modo de retenção padrão e o período de retenção padrão para um bucket com o S3 Object Lock ativado. Você também pode remover as configurações de retenção padrão previamente definidas.

Quando novas versões de objetos são ingeridas no bucket, o modo de retenção padrão é aplicado se `x-amz-object-lock-mode` e `x-amz-object-lock-retain-until-date` não forem especificados. O período de retenção padrão é usado para calcular a data de retenção até se `x-amz-object-lock-retain-until-date` não for especificado.

Se o período de retenção padrão for modificado após a ingestão de uma versão de objeto, a data de retenção da versão do objeto permanecerá a mesma e não será recalculada usando o novo período de retenção padrão.

Você precisa ter a `s3:PutBucketObjectLockConfiguration` permissão, ou ser o root da conta, para concluir esta operação.

O `Content-MD5`cabeçalho da requisição deve ser especificado na solicitação PUT.

Exemplo de solicitação

Este exemplo habilita o S3 Object Lock para um bucket e define o modo de retenção padrão como COMPLIANCE e o período de retenção padrão como 6 anos.

```
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>
```

Como determinar a retenção padrão para um bucket

Para determinar se o S3 Object Lock está ativado para um bucket e para ver o modo de retenção padrão e o período de retenção, use um destes métodos:

- Veja o bucket no Tenant Manager. Consulte "Visualizar buckets do S3".
- Emita uma solicitação GetObjectLockConfiguration.

GetObjectLockConfiguration

A solicitação GetObjectLockConfiguration permite que você determine se o S3 Object Lock está habilitado para um bucket e, se estiver habilitado, veja se há um modo de retenção padrão e um período de retenção configurados para o bucket.

Quando novas versões de objetos são ingeridas no bucket, o modo de retenção padrão é aplicado se `x-amz-object-lock-mode` não for especificado. O período de retenção padrão é usado para calcular a retain-until-date se `x-amz-object-lock-retain-until-date` não for especificado.

Você precisa ter a `s3:GetBucketObjectLockConfiguration` permissão, ou ser account root, para concluir esta operação.

Exemplo de solicitação

```
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
```

Exemplo de resposta

```
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>
```

Como especificar as configurações de retenção para um objeto

Um bucket com o S3 Object Lock ativado pode conter uma combinação de objetos com e sem configurações de retenção do S3 Object Lock.

As configurações de retenção em nível de objeto são especificadas usando a API REST do S3. As configurações de retenção para um objeto substituem quaisquer configurações de retenção padrão para o bucket.

Você pode especificar as seguintes configurações para cada objeto:

- **Modo de retenção**: COMPLIANCE ou GOVERNANÇA.
- **Retain-until-date**: Uma data que especifica por quanto tempo a versão do objeto deve ser retida pelo StorageGRID.

    - No modo COMPLIANCE, se a retain-until-date for futura, o objeto poderá ser recuperado, mas não poderá ser modificado ou excluído. A retain-until-date pode ser aumentada, mas essa data não pode ser diminuída ou removida.
    - No modo GOVERNANÇA, usuários com permissões especiais podem ignorar a configuração de retain-until-date. Eles podem excluir uma versão de objeto antes que seu período de retenção tenha expirado. Eles também podem aumentar, diminuir ou até mesmo remover o retain-until-date.
- **Guarda legal**: Aplicar uma guarda legal a uma versão de objeto bloqueia imediatamente esse objeto. Por exemplo, você pode precisar aplicar uma guarda legal a um objeto relacionado a uma investigação ou disputa legal. Uma guarda legal não tem data de expiração, mas permanece em vigor até ser explicitamente removida.

  A configuração de guarda legal para um objeto é independente do modo de retenção e da data limite de retenção. Se uma versão de um objeto estiver sob guarda legal, ninguém pode excluir essa versão.

Para especificar as configurações de S3 Object Lock ao adicionar uma versão de objeto a um bucket, emita uma solicitação "PutObject", "CopyObject" ou "CreateMultipartUpload".

Você pode usar o seguinte:

- `x-amz-object-lock-mode`, que pode ser COMPLIANCE ou GOVERNANCE (diferencia maiúsculas de minúsculas).

  > Se você especificar `x-amz-object-lock-mode`, você também deve especificar `x-amz-object-lock-retain-until-date`.

- `x-amz-object-lock-retain-until-date`
    - O valor de retain-until-date deve estar no formato `2020-08-10T21:46:00Z`. Frações de segundo são permitidas, mas apenas 3 casas decimais são preservadas (precisão em milissegundos). Outros formatos ISO 8601 não são permitidos.
    - A data de retenção deve estar no futuro.
- `x-amz-object-lock-legal-hold`

  Se guarda legal estiver ON (diferencia maiúsculas de minúsculas), o objeto é colocado sob guarda legal. Se guarda legal estiver OFF, nenhuma guarda legal é aplicada. Qualquer outro valor resulta em um erro 400 Bad Request (InvalidArgument).

Se você usar algum desses cabeçalhos de solicitação, esteja ciente destas restrições:

- O `Content-MD5`cabeçalho da requisição é obrigatório se qualquer `x-amz-object-lock-*`cabeçalho de requisição estiver presente na solicitação PutObject. `Content-MD5`Não é obrigatório para CopyObject ou CreateMultipartUpload.
- Se o bucket não tiver o S3 Object Lock ativado e um `x-amz-object-lock-*` cabeçalho de solicitação estiver presente, será retornado um erro 400 Bad Request (InvalidRequest).
- A solicitação PutObject dá suporte ao uso de `x-amz-storage-class: REDUCED_REDUNDANCY` para corresponder ao comportamento da AWS. No entanto, quando um objeto é ingerido em um bucket com S3 Object Lock ativado, StorageGRID sempre realizará uma ingestão com confirmação dupla.
- Uma resposta GET ou de versão HeadObject subsequente incluirá os cabeçalhos `x-amz-object-lock-mode`, `x-amz-object-lock-retain-until-date` e `x-amz-object-lock-legal-hold`, se configurados e se o remetente da solicitação tiver as permissões corretas de `s3:Get*`.

Você pode usar a `s3:object-lock-remaining-retention-days`chave de condição de política para limitar os períodos de retenção mínimos e máximos permitidos para seus objetos.

Como atualizar as configurações de retenção de um objeto

Se você precisar atualizar as configurações de guarda legal ou de retenção para uma versão de objeto existente, pode executar as seguintes operações de sub-recurso de objeto:

- `PutObjectLegalHold`

  Se o novo valor de guarda legal estiver ATIVADO, o objeto fica sujeito a guarda legal. Se o valor de guarda legal estiver DESATIVADO, a guarda legal é suspensa.

- `PutObjectRetention`
  - O valor do modo pode ser COMPLIANCE ou GOVERNANCE (diferencia maiúsculas de minúsculas).
  - O valor de retain-until-date deve estar no formato `2020-08-10T21:46:00Z`. Frações de segundo são permitidas, mas apenas 3 casas decimais são preservadas (precisão em milissegundos). Outros formatos ISO 8601 não são permitidos.
  - Se uma versão de um objeto já possui uma data de retenção definida, você só pode aumentá-la. O novo valor deve estar no futuro.

Como usar o modo GOVERNANCE

Usuários que têm a permissão `s3:BypassGovernanceRetention` podem ignorar as configurações de retenção ativa de um objeto que utiliza o modo GOVERNANCE. Quaisquer operações DELETE ou PutObjectRetention devem incluir o cabeçalho `x-amz-bypass-governance-retention:true` da solicitação. Esses usuários podem executar as seguintes operações adicionais:

- Execute operações DeleteObject ou DeleteObjects para excluir uma versão de objeto antes que seu período de retenção tenha expirado.

  Objetos que estão sob guarda legal não podem ser excluídos. A guarda legal deve estar DESATIVADA.

- Execute operações PutObjectRetention que alteram o modo de uma versão de objeto de GOVERNANCE para COMPLIANCE antes que o período de retenção tenha expirado.

  A mudança do modo de COMPLIANCE para GOVERNANCE nunca é permitida.

- Realize operações PutObjectRetention para aumentar, diminuir ou remover o período de retenção de uma versão do objeto.

Informações relacionadas

- "Gerencie objetos com S3 Object Lock"
- "Use o S3 Object Lock para reter objetos"
- "Guia do usuário do Amazon Simple Storage Service: bloqueando objetos"

Saiba mais sobre a configuração do ciclo de vida do S3 para StorageGRID

Você pode criar uma configuração de ciclo de vida do S3 para controlar quando objetos

específicos são excluídos do sistema StorageGRID.

O exemplo simples nesta seção ilustra como uma configuração de ciclo de vida do S3 pode controlar quando determinados objetos são excluídos (expirados) de buckets específicos do S3. O exemplo nesta seção é apenas para fins ilustrativos. Para obter detalhes completos sobre a criação de configurações de ciclo de vida do S3, consulte "Guia do usuário do Amazon Simple Storage Service: gerenciamento de ciclo de vida de objetos". Observe que StorageGRID oferece suporte apenas a ações de Expiração; não oferece suporte a ações de Transição.

O que é configuração de ciclo de vida

Uma configuração de ciclo de vida é um conjunto de regras aplicadas aos objetos em buckets específicos do S3. Cada regra especifica quais objetos são afetados e quando esses objetos expirarão (em uma data específica ou após um determinado número de dias).

StorageGRID suporta até 1.000 regras de ciclo de vida em uma configuração de ciclo de vida. Cada regra pode incluir os seguintes elementos XML:

- Expiração: exclua um objeto quando uma data específica for atingida ou quando um número específico de dias for alcançado, a partir do momento em que o objeto foi ingerido.
- NoncurrentVersionExpiration: Exclua um objeto quando um número específico de dias for atingido, a partir do momento em que o objeto deixou de ser atual.
- Filtro (Prefixo, Tag)
- Status
- ID

Cada objeto segue as configurações de retenção de um ciclo de vida do bucket S3 ou de uma política de ILM. Quando um ciclo de vida do bucket S3 é configurado, as ações de expiração do ciclo de vida substituem a política de ILM para objetos que correspondem ao filtro do ciclo de vida do bucket. Objetos que não correspondem ao filtro do ciclo de vida do bucket usam as configurações de retenção da política de ILM. Se um objeto corresponder a um filtro do ciclo de vida do bucket e nenhuma ação de expiração for explicitamente especificada, as configurações de retenção da política de ILM não são usadas e entende-se que as versões do objeto são retidas para sempre. Consulte "Exemplos de prioridades para ciclo de vida de bucket S3 e política ILM".

Como resultado, um objeto pode ser removido da grid mesmo que as instruções de posicionamento em uma regra ILM ainda se apliquem ao objeto. Ou, um objeto pode ser mantido na grid mesmo depois que quaisquer instruções de posicionamento ILM para o objeto tiverem expirado. Para obter detalhes, consulte "Como o ILM opera ao longo do ciclo de vida de um objeto".

> A configuração de ciclo de vida do bucket pode ser usada com buckets que têm o S3 Object Lock ativado, mas a configuração de ciclo de vida do bucket não é compatível com buckets legados Compliant.

StorageGRID suporta o uso das seguintes operações de bucket para gerenciar configurações de ciclo de vida:

- DeleteBucketLifecycle
- GetBucketLifecycleConfiguration
- PutBucketLifecycleConfiguration

Criar configuração de ciclo de vida

Como primeiro passo para criar uma configuração de ciclo de vida, você cria um arquivo JSON que inclui uma ou mais regras. Por exemplo, este arquivo JSON inclui três regras, como segue:

1. A regra 1 aplica-se apenas a objetos que correspondam ao prefixo `category1/` e que tenham um valor de `key2 tag2`. O parâmetro `Expiration` especifica que os objetos que correspondam ao filtro expirarão à meia-noite de 22 de agosto de 2020.
2. A regra 2 aplica-se apenas a objetos que correspondam ao prefixo `category2/`. O `Expiration` parâmetro especifica que objetos que correspondem ao filtro expirarão 100 dias após serem ingeridos.

   > As regras que especificam um número de dias são relativas à data de ingestão do objeto. Se a data atual for posterior à data de ingestão mais o número de dias, alguns objetos podem ser removidos do bucket assim que a configuração do ciclo de vida for aplicada.

3. A regra 3 aplica-se apenas a objetos que correspondam ao prefixo `category3/`. O `Expiration`parâmetro especifica que quaisquer versões não atuais de objetos correspondentes expirarão 50 dias após se tornarem não atuais.

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

Aplicar configuração de ciclo de vida ao bucket

Após criar o arquivo de configuração de ciclo de vida, você o aplica a um bucket emitindo uma solicitação PutBucketLifecycleConfiguration.

Esta solicitação aplica a configuração de ciclo de vida no arquivo de exemplo aos objetos em um bucket chamado `testbucket`.

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

Para validar se uma configuração de ciclo de vida foi aplicada com sucesso ao bucket, emita uma solicitação GetBucketLifecycleConfiguration. Por exemplo:

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

Uma resposta bem-sucedida lista a configuração de ciclo de vida que você acabou de aplicar.

Valide se a expiração do ciclo de vida do bucket se aplica ao objeto

Você pode determinar se uma regra de expiração na configuração de ciclo de vida se aplica a um objeto específico ao emitir uma solicitação PutObject, HeadObject ou GetObject. Se uma regra se aplicar, a resposta incluirá um parâmetro `Expiration` que indica quando o objeto expira e qual regra de expiração foi correspondida.

> ⓘ Como o ciclo de vida do bucket substitui o ILM, a `expiry-date` data exibida é a data real em que o objeto será excluído. Para obter detalhes, consulte "Como a retenção de objeto é determinada".

Por exemplo, esta solicitação PutObject foi emitida em 22 de junho de 2020 e coloca um objeto no bucket `testbucket`.

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

A resposta de sucesso indica que o objeto expirará em 100 dias (01 out 2020) e que corresponde à Regra 2 da configuração do ciclo de vida.

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

Por exemplo, esta solicitação HeadObject foi usada para obter metadados para o mesmo objeto no bucket testbucket.

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

A resposta de sucesso inclui os metadados do objeto e indica que o objeto expirará em 100 dias e que corresponde à Regra 2.

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

Para buckets com controle de versão ativado, o `x-amz-expiration` cabeçalho de resposta se aplica somente às versões atuais dos objetos.

Recomendações para implementar a API REST S3 com StorageGRID

Você deve seguir estas recomendações ao implementar a API REST para uso com StorageGRID.

Recomendações para HEADs para objetos inexistentes

Se sua aplicação verifica rotineiramente se um objeto existe em um caminho onde você não espera que ele exista, você deve usar a consistência "Disponível" "consistência". Por exemplo, você deve usar a consistência "Disponível" se sua aplicação executar um HEAD em um local antes de executar um PUT nele.

Caso contrário, se a operação HEAD não encontrar o objeto, você poderá receber um grande número de erros 500 Internal Server se dois ou mais nós de armazenamento no mesmo site estiverem indisponíveis ou se um local remoto estiver inacessível.

Você pode definir a consistência "Disponível" para cada bucket usando a "Consistência de PUT Bucket" solicitação, ou pode especificar a consistência no cabeçalho da solicitação para uma operação de API individual.

Recomendações para chaves de objeto

Siga estas recomendações para nomes de chaves de objetos, com base em quando o bucket foi criado pela primeira vez.

Buckets criados no StorageGRID 11.4 ou anterior

- Não utilize valores aleatórios como os quatro primeiros caracteres das chaves de objetos. Isso contrasta com a recomendação anterior da AWS para prefixos de chave. Em vez disso, utilize prefixos não aleatórios e não exclusivos, como `image`.
- Se você seguir a recomendação anterior da AWS de usar caracteres aleatórios e exclusivos nos prefixos das chaves, prefixe as chaves dos objetos com um nome de diretório. Ou seja, use este formato:

  `mybucket/mydir/f8e3-image3132.jpg`

  Em vez deste formato:

  `mybucket/f8e3-image3132.jpg`

Buckets criados no StorageGRID 11.4 ou posterior

Não é necessário restringir os nomes das chaves de objeto para atender às melhores práticas de desempenho. Na maioria dos casos, você pode usar valores aleatórios para os quatro primeiros caracteres dos nomes das chaves de objeto.

> Uma exceção a isso é uma carga de trabalho do S3 que remove continuamente todos os objetos após um curto período de tempo. Para minimizar o impacto no desempenho nesse caso de uso, varie a parte inicial do nome da chave a cada alguns milhares de objetos com algo como a data. Por exemplo, suponha que um cliente S3 normalmente grave 2.000 objetos por segundo e que a política de ciclo de vida do ILM ou do bucket remova todos os objetos após três dias. Para minimizar o impacto no desempenho, você pode nomear as chaves usando um padrão como este: `/mybucket/mydir/yyyymmddhhmmss-random_UUID.jpg`

Recomendações para "leituras de intervalo"

Se a "opção global para compactar objetos armazenados" estiver habilitada, os aplicativos cliente do S3 devem evitar realizar operações GetObject que especifiquem um intervalo de bytes a ser retornado. Essas operações de "leitura de intervalo" são ineficientes porque o StorageGRID precisa efetivamente descompactar os objetos para acessar os bytes solicitados. Operações GetObject que solicitam um pequeno intervalo de bytes de um objeto muito grande são especialmente ineficientes; por exemplo, é ineficiente ler um intervalo de 10 MB de um objeto compactado de 50 GB.

Se intervalos forem lidos de objetos compactados, as solicitações do cliente podem expirar.

> Se você precisar compactar objetos e seu aplicativo cliente precisar usar leituras de intervalo, aumente o tempo limite de leitura do aplicativo.

Informações sobre direitos autorais

Informações sobre marcas comerciais