API を使用する

StorageGRID software

NetApp
July 23, 2026

目次

API を使用する

StorageGRID Grid Management APIを使用する

グリッドマネージャのユーザインターフェイスの代わりに、グリッド管理 REST API を使用してシステム管理タスクを実行できます。たとえば、API を使用して操作を自動化したり、ユーザなどの複数のエンティティをより迅速に作成したりすることができます。

最上位のリソース

グリッド管理APIは、以下のトップレベルリソースを提供します。

- `/grid`：Grid Managerユーザーへのアクセスに限定されており、設定されたグループ権限に基づいています。
- `/org`：アクセスは、テナントアカウントのローカルまたはフェデレーションLDAPグループに属するユーザーに限定されます。詳細については、"テナントアカウントを使用する"を参照してください。
- `/private`：アクセスはGrid Managerユーザーに限定されており、設定されたグループ権限に基づいています。プライベートAPIは予告なく変更される場合があります。StorageGRIDプライベートエンドポイントは、リクエストのAPIバージョンも無視します。

API リクエストを発行する

グリッド管理APIは、オープンソースAPIプラットフォームであるSwaggerを使用しています。Swaggerは、開発者と非開発者がAPIを使用してStorageGRIDでリアルタイム操作を実行できる直感的なユーザーインターフェースを提供します。

Swagger のユーザーインターフェイスは、各 API 操作に関する完全な詳細情報とドキュメントを提供します。

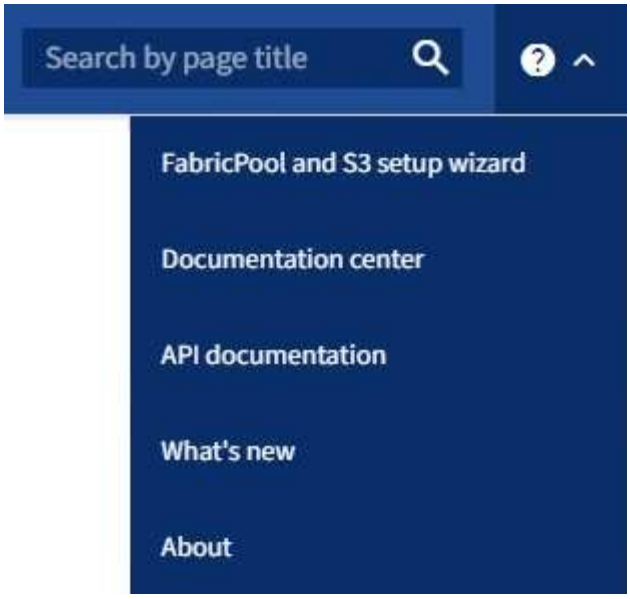
開始する前に

- "対応ウェブブラウザ" を使用して Grid Manager にサインインしています。
- "特定のアクセス権限"があります。

> APIドキュメントのウェブページを使用して実行するAPI操作はすべて、ライブ操作となります。設定データやその他のデータを誤って作成、更新、または削除しないように注意してください。

手順

1. グリッドマネージャーのヘッダーからヘルプアイコンを選択し、「API documentation」を選択します。



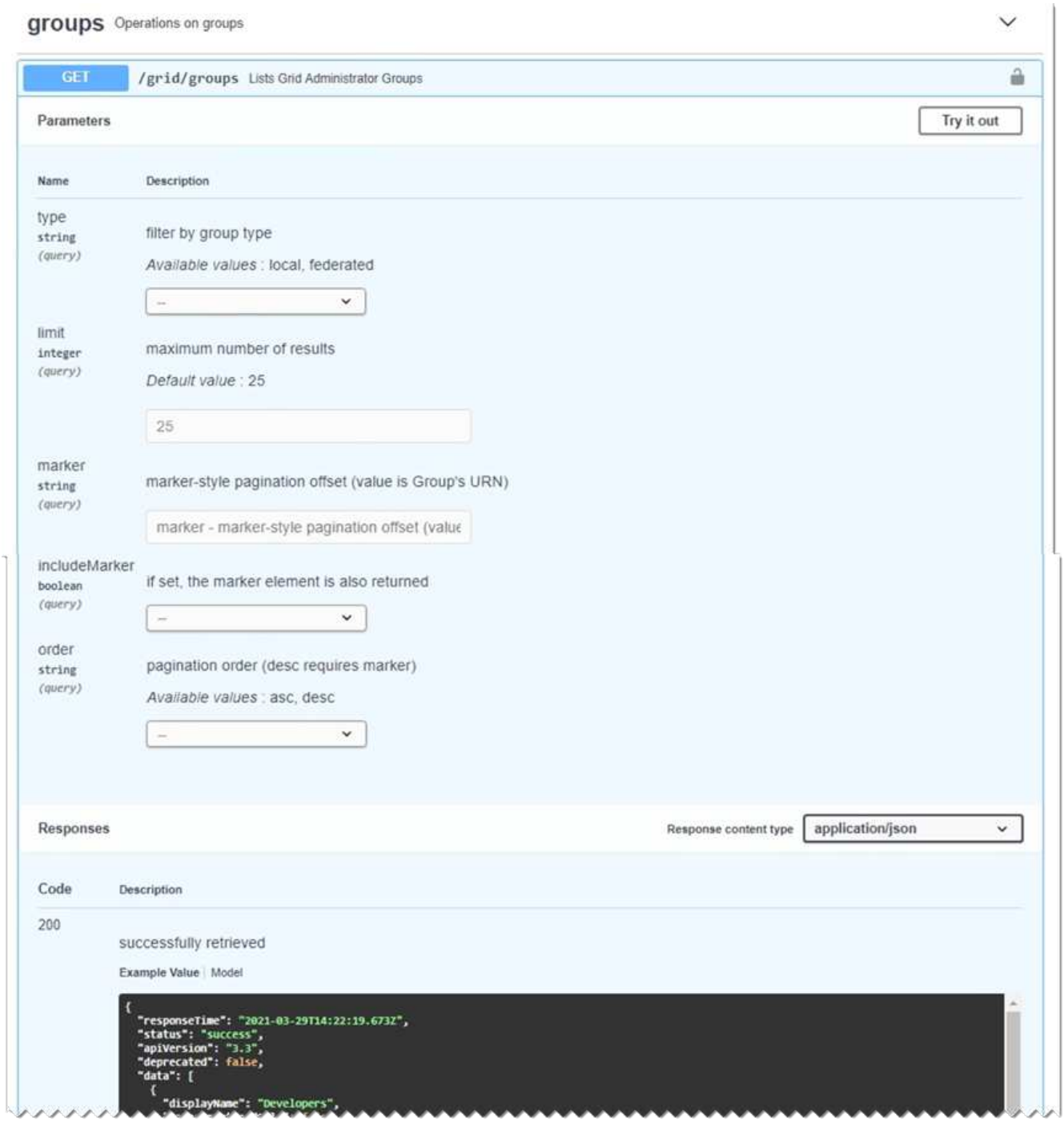
2. プライベート API で操作を実行するには、StorageGRID 管理 API ページで プライベート **API** ドキュメントへ移動 を選択します。

   プライベートAPIは予告なく変更される場合があります。StorageGRIDプライベートエンドポイントは、リクエストのAPIバージョンも無視します。

3. 希望する操作を選択してください。

   API操作を展開すると、GET、PUT、UPDATE、DELETEなどの利用可能なHTTPアクションが表示されます。

4. HTTPアクションを選択すると、エンドポイントURL、必須またはオプションのパラメーター一覧、リクエストボディの例（必要な場合）、および考えられるレスポンスなど、リクエストの詳細が表示されます。



5. リクエストにグループIDやユーザーIDなどの追加パラメータが必要かどうかを判断します。次に、これらの値を取得します。必要な情報を取得するには、まず別のAPIリクエストを発行する必要がある場合があります。
6. サンプルリクエストボディを修正する必要があるかどうかを判断してください。必要な場合は、*モデル*を選択して各フィールドの要件を確認できます。
7. 「Try it out」を選択します。
8. 必要なパラメータを指定するか、必要に応じてリクエスト本文を変更してください。
9. 「実行」を選択します。
10. リクエストが成功したかどうかを判断するには、レスポンスコードを確認してください。

StorageGRID のグリッド管理 API 処理

グリッド管理APIは、利用可能な操作を以下のセクションに分類しています。

このリストには、公開APIで利用可能な操作のみが含まれています。

- アカウント：ストレージテナントアカウントを管理するための操作。これには、新しいアカウントの作成や、特定のアカウントのストレージ使用状況の取得などが含まれます。
- **alert-history**：解決済みのアラートに対する操作。
- **alert-receivers**：アラート通知受信者（電子メール）に対する操作。
- **alert-rules**：アラートルールに対する操作。
- **alert-silences**：アラートのサイレンスに対する操作。
- アラート：アラートに対する操作。
- 監査：監査設定を一覧表示および更新する操作。
- **auth**：ユーザーセッション認証を実行するための操作。

  グリッド管理APIは、ベアラートークン認証方式をサポートしています。サインインするには、認証リクエストの JSON ボディにユーザー名とパスワードを指定します（つまり、 `POST /api/v3/authorize` ）。ユーザー認証が成功すると、セキュリティトークンが返されます。このトークンは、以降のAPIリクエストのヘッダーに含めて指定する必要があります（「Authorization: Bearer *token*」）。トークンは16時間後に期限切れになります。

  StorageGRID システムでシングル サインオンが有効になっている場合は、認証のために異なる手順を実行する必要があります。「シングル サインオンが有効な場合のAPIへの認証」を参照してください。

  認証セキュリティの向上については、「クロスサイトリクエストフォージェリからの保護」を参照してください。

- クライアント証明書：クライアント証明書を設定して、外部監視ツールを使用して StorageGRID に安全にアクセスできるようにするための操作。
- **config**: Grid Management API の製品リリースおよびバージョンに関連する操作。製品のリリースバージョンと、そのリリースでサポートされている Grid Management API のメジャーバージョンを一覧表示したり、非推奨バージョンの API を無効にしたりできます。
- **deactivated-features**：無効化された可能性のある機能を表示するための操作。
- **dns-servers**：設定済みの外部 DNS サーバーを一覧表示および変更する操作。
- **drive-details**：特定のストレージアプライアンスモデルのドライブに対する操作。
- **endpoint-domain-names**：S3エンドポイント ドメイン名をリスト表示および変更する操作。
- **erasure-coding**：消去符号化プロファイルに対する操作。
- 展開：展開に対する操作（プロシージャレベル）。
- **expansion-nodes**：展開に関する操作（ノードレベル）。
- **expansion-sites**: 拡張に関する作業（サイトレベル）。

- **grid-networks**：グリッドネットワークリストを一覧表示および変更する操作。
- **grid-passwords**：グリッドパスワード管理のための操作。
- グループ：ローカルのグリッド管理者グループを管理し、外部LDAPサーバからフェデレーションされたグリッド管理者グループを取得する操作。
- **identity-source**： 外部IDソースを設定し、フェデレーションされたグループとユーザー情報を手動で同期するための操作。
- **ilm**: 情報ライフサイクル管理（ILM）に対する操作。
- **in-progress-procedures**：現在進行中のメンテナンス手順を取得します。
- ライセンス：StorageGRID ライセンスを取得および更新するための操作。
- **logs**：ログファイルの収集とダウンロードの操作。v
- メトリクス：StorageGRIDメトリクスに対する操作には、ある時点における即時メトリクスクエリと、一定期間にわたる範囲メトリクスクエリが含まれます。グリッド管理APIは、バックエンドのデータソースとしてPrometheusシステム監視ツールを使用します。Prometheusクエリの構築方法については、PrometheusのWebサイトを参照してください。

> 名前に*`private`*が含まれる指標は、内部使用のみを目的としています。これらの指標は、予告なしに StorageGRID リリース間で変更される可能性があります。

- **node-details**：ノードの詳細に対する操作。
- **node-health**：ノードの健全性状態に関する操作。
- **node-storage-state**：ノードストレージの状態に関する操作。
- **ntp-servers**：外部ネットワークタイムプロトコル（NTP）サーバーを一覧表示または更新する操作。
- オブジェクト：オブジェクトおよびオブジェクトメタデータに対する操作。
- リカバリ：リカバリ手順のための操作。
- **recovery-package**：リカバリパッケージをダウンロードするための操作。
- **regions**：リージョンを表示および作成するための操作。
- **s3-object-lock**：グローバル S3 オブジェクト ロック設定に対する操作。
- **server-certificate**: Grid Managerサーバー証明書を表示および更新する操作。
- **snmp**：現在のSNMP設定に対する操作。
- **storage-watermarks**: ストレージノードのウォーターマーク。
- **traffic-classes**：トラフィック分類ポリシーの操作。
- **untrusted-client-network**：信頼されていないクライアント ネットワーク構成に対する操作。
- **users**：Grid Managerユーザーを表示および管理するための操作。

StorageGRID におけるグリッド管理 API のバージョン管理

グリッド管理APIは、バージョン管理を利用して、システムへの影響を最小限に抑えたアップグレードをサポートします。

例えば、このリクエスト URL は API のバージョン 4 を指定しています。

`https://hostname_or_ip_address/api/v4/authorize`

APIのメジャーバージョンは、古いバージョンと互換性のない変更が行われた場合、更新されます。APIのマイナーバージョンは、古いバージョンと互換性のある変更が行われた場合、更新されます。互換性のある変更には、新しいエンドポイントまたは新しいプロパティの追加が含まれます。

以下の例は、行われた変更の種類に基づいてAPIバージョンがどのように更新されるかを示しています。

APIに対する変更の種類	旧バージョン	新バージョン
旧バージョンとの互換性あり	2.1	2.2
旧バージョンとの互換性はありません	2.1	3.0

StorageGRID ソフトウェアを初めてインストールする場合、最新バージョンの API のみが有効になります。ただし、StorageGRID の新しい機能リリースにアップグレードした場合でも、少なくとも 1 つの StorageGRID 機能リリースの間は、古い API バージョンに引き続きアクセスできます。

> ⓘ 対応バージョンを設定できます。詳細については、"グリッド管理API" の Swagger API ドキュメントの **config** セクションを参照してください。すべての API クライアントを新しいバージョンに更新した後、古いバージョンのサポートを無効にする必要があります。

古いリクエストは、以下の方法で非推奨としてマークされます。

- レスポンスヘッダーは「Deprecated: true」です。
- JSON レスポンスボディには「deprecated」：true が含まれています
- 非推奨の警告がnms.logに追加されます。例：

```
Received call to deprecated v2 API at POST "/api/v2/authorize"
```

現在のリリースでサポートされている**API**バージョンを確認する

`GET /versions`API リクエストを使用して、サポートされている API のメジャーバージョンのリストを返します。このリクエストは、Swagger API ドキュメントの *config* セクションに記載されています。

```
GET https://{{IP-Address}}/api/versions
{
  "responseTime": "2023-06-27T22:13:50.750Z",
  "status": "success",
  "apiVersion": "4.0",
  "data": [
    2,
    3,
    4
  ]
}
```

リクエストのAPIバージョンを指定します

パスパラメータ（`/api/v4`）またはヘッダー（`Api-Version: 4`）を使用して API バージョンを指定できます。両方の値を指定した場合、ヘッダーの値がパスの値を上書きします。

```
curl https://[IP-Address]/api/v4/grid/accounts

curl -H "Api-Version: 4" https://[IP-Address]/api/grid/accounts
```

StorageGRID でクロスサイトリクエストフォージェリ（CSRF）から保護する

CSRFトークンを使用してCookieを使用する認証を強化することで、StorageGRIDに対するクロスサイトリクエストフォージェリ（CSRF）攻撃からの保護に役立てることができます。Grid ManagerとTenant Managerはこのセキュリティ機能を自動的に有効にします。その他のAPIクライアントは、サインイン時に有効にするかどうかを選択できます。

別のサイトへのリクエスト（HTTPフォームPOSTなど）をトリガーできる攻撃者は、ログインしているユーザーのCookieを使用して特定のリクエストを実行させることができます。

StorageGRID は、CSRF トークンを使用することで、CSRF 攻撃に対する保護に役立ちます。有効にした場合、特定のクッキーの内容は、特定のヘッダーまたは特定の POST ボディパラメータの内容と一致する必要があります。

この機能を有効にするには、認証中に `csrfToken`パラメータを `true`に設定します。デフォルトは `false`です。

```
curl -X POST --header "Content-Type: application/json" --header "Accept:
application/json" -d "{
  \"username\": \"MyUserName\",
  \"password\": \"MyPassword\",
  \"cookie\": true,
  \"csrfToken\": true
}" "https://example.com/api/v3/authorize"
```

true の場合、 `GridCsrfToken` クッキーには Grid Manager へのサインイン用のランダムな値が設定され、 `AccountCsrfToken` クッキーには Tenant Manager へのサインイン用のランダムな値が設定されます。

Cookieが存在する場合、システムの状態を変更できるすべてのリクエスト（POST、PUT、PATCH、DELETE）には、以下のいずれかを含める必要があります。

- `X-Csrf-Token`ヘッダー。ヘッダーの値は、CSRFトークンクッキーの値に設定されます。
- フォームエンコードされたボディを受け入れるエンドポイントの場合：A `csrfToken`フォームエンコードされたリクエストボディパラメータ。

その他の例や詳細については、オンラインAPIドキュメントを参照してください。

> CSRFトークンCookieが設定されているリクエストは、CSRF攻撃に対する追加の保護として、JSONリクエストボディを想定するすべてのリクエストに対して「Content-Type: application/json」ヘッダーを強制的に適用します。

シングル サインオンが有効な場合のAPIの使用

StorageGRID API を使用する場合、 Single Sign-On が有効になっている（Active Directory）

"設定および有効化されたシングル サインオン（SSO）"を使用しており、Active Directory を SSO プロバイダーとして使用している場合は、Grid Management API または Tenant Management API で有効な認証トークンを取得するために、一連の API リクエストを発行する必要があります。

シングル サインオンが有効になっている場合にAPIにサインインする

これらの手順は、SSOのIDプロバイダーとしてActive Directoryを使用している場合に適用されます。

開始する前に

- StorageGRID ユーザーグループに属するフェデレーションユーザーの SSO ユーザー名とパスワードを把握しています。
- テナント管理APIにアクセスするには、テナントアカウントIDを知っている必要があります。

タスク概要

認証トークンを取得するには、以下のいずれかの例を使用できます。

- `storagegrid-ssoauth.py` Python スクリプトは、StorageGRID インストールファイルディレクトリ（(`./rpms` RHEL の場合、 `./debs` Ubuntu または Debian の場合、 `./vsphere` VMware の場合）にあります。
- curlリクエストのワークフロー例。

  curlワークフローは、実行速度が遅すぎるとタイムアウトする可能性があります。次のようなエラーが表示される場合があります： `A valid SubjectConfirmation was not found on this Response`。

  > この例のcurlワークフローでは、パスワードが他のユーザーに見られるのを防ぐことはできません。

URLエンコードに問題がある場合、次のようなエラーが表示されることがあります： `Unsupported SAML version`。

手順

1. 認証トークンを取得するには、以下のいずれかの方法を選択してください。
   - `storagegrid-ssoauth.py`Python スクリプトを使用します。ステップ 2 に進みます。
   - curl リクエストを使用します。ステップ 3 に進みます。
2. `storagegrid-ssoauth.py`スクリプトを使用する場合は、Pythonインタープリタにスクリプトを渡して実行してください。

   指示が表示されたら、以下の引数に値を入力してください。
   - SSO方式。ADFS または adfs と入力してください。
   - SSOユーザ名
   - StorageGRID がインストールされているドメイン
   - StorageGRID のアドレス
   - テナント管理APIにアクセスする場合は、テナントアカウントIDが必要です。

   ```
   python3 storagegrid-ssoauth.py
   sso_method: adfs
   saml_user: my-sso-username
   saml_domain: my-domain
   sg_address: storagegrid.example.com
   tenant_account_id: 12345
   Enter the user's SAML password:
   ********************************************************************************

   ********************************************************************************
   StorageGRID Auth Token: 56eb07bf-21f6-40b7-afob-5c6cacfb25e7
   ```

   StorageGRID 認証トークンが出力に含まれています。これで、SSO を使用していない場合と同様に、トークンを他のリクエストに使用できます。
3. curlリクエストを使用する場合は、以下の手順に従ってください。

a. ログインに必要な変数を宣言します。

```
export SAMLUSER='my-sso-username'
export SAMLPASSWORD='my-password'
export SAMLDOMAIN='my-domain'
export TENANTACCOUNTID='12345'
export STORAGEGRID_ADDRESS='storagegrid.example.com'
export AD_FS_ADDRESS='adfs.example.com'
```

ⓘ グリッド管理APIにアクセスするには、0を `TENANTACCOUNTID`として使用します。

b. 署名付き認証URLを受け取るには、 `/api/v3/authorize-saml`にPOSTリクエストを発行し、レスポンスから余分なJSONエンコーディングを削除します。

この例は、 `TENANTACCOUNTID`の署名付き認証 URL に対する POST リクエストを示しています。結果は `python -m json.tool`に渡され、JSON エンコーディングが削除されます。

```
curl -X POST "https://$STORAGEGRID_ADDRESS/api/v3/authorize-saml" \
  -H "accept: application/json" -H  "Content-Type: application/json"
\
  --data "{\"accountId\": \"$TENANTACCOUNTID\"}" | python -m
json.tool
```

この例に対する応答には、URLエンコードされた署名付きURLが含まれていますが、追加のJSONエンコード層は含まれていません。

```
{
    "apiVersion": "3.0",
    "data":
"https://adfs.example.com/adfs/ls/?SAMLRequest=fZHLbsIwEEV%2FJTuv7...
sSl%2BfQ33cvfwA%3D&RelayState=12345",
    "responseTime": "2018-11-06T16:30:23.355Z",
    "status": "success"
}
```

c. 応答から取得した `SAMLRequest`を、後続のコマンドで使用するために保存します。

```
export SAMLREQUEST='fZHLbsIwEEV%2FJTuv7...sSl%2BfQ33cvfwA%3D'
```

d. AD FSからクライアントリクエストIDを含む完全なURLを取得します。

一つの方法として、前回のレスポンスのURLを使用してログインフォームをリクエストする方法があります。

```
curl "https://$AD_FS_ADDRESS/adfs/ls/?SAMLRequest=
$SAMLREQUEST&RelayState=$TENANTACCOUNTID" | grep 'form method="post"
id="loginForm"'
```

レスポンスにはクライアントリクエスト ID が含まれます：

```
<form method="post" id="loginForm" autocomplete="off"
novalidate="novalidate" onKeyPress="if (event && event.keyCode == 13)
Login.submitLoginRequest();" action="/adfs/ls/?
SAMLRequest=fZHRToMwFIZfhb...UJikvo77sXPw%3D%3D&RelayState=12345&clie
nt-request-id=00000000-0000-0000-ee02-0080000000de" >
```

e. レスポンスからクライアントリクエストIDを保存します。

```
export SAMLREQUESTID='00000000-0000-0000-ee02-0080000000de'
```

f. 前回のレスポンスで指定したフォームアクションに認証情報を送信してください。

```
curl -X POST "https://$AD_FS_ADDRESS
/adfs/ls/?SAMLRequest=$SAMLREQUEST&RelayState=$TENANTACCOUNTID&client
-request-id=$SAMLREQUESTID" \
--data "UserName=$SAMLUSER@$SAMLDOMAIN&Password=
$SAMLPASSWORD&AuthMethod=FormsAuthentication" --include
```

AD FSは、ヘッダーに追加情報を含めた302リダイレクトを返します。

> SSOシステムで多要素認証（MFA）が有効になっている場合、フォーム送信には2つ目のパスワードまたはその他の認証情報も含まれます。

```
HTTP/1.1 302 Found
Content-Length: 0
Content-Type: text/html; charset=utf-8
Location:
https://adfs.example.com/adfs/ls/?SAMLRequest=fZHRToMwFIZfhb...UJikvo
77sXPw%3D%3D&RelayState=12345&client-request-id=00000000-0000-0000-
ee02-0080000000de
Set-Cookie: MSISAuth=AAEAADAvsHpXk6ApV...pmP0aEiNtJvWY=; path=/adfs;
HttpOnly; Secure
Date: Tue, 06 Nov 2018 16:55:05 GMT
```

g. レスポンスから `MSISAuth`クッキーを保存します。

```
export MSISAuth='AAEAADAvsHpXk6ApV...pmP0aEiNtJvWY='
```

h. 認証POSTリクエストから取得したCookieを使用して、指定された場所にGETリクエストを送信します。

```
curl "https://$AD_FS_ADDRESS/adfs/ls/?SAMLRequest=
$SAMLREQUEST&RelayState=$TENANTACCOUNTID&client-request-
id=$SAMLREQUESTID" \
--cookie "MSISAuth=$MSISAuth" --include
```

応答ヘッダーには、後のログアウト時に使用するための AD FS セッション情報が含まれ、応答本文には非表示のフォームフィールドに SAMLResponse が含まれます。

```
HTTP/1.1 200 OK
Cache-Control: no-cache,no-store
Pragma: no-cache
Content-Length: 5665
Content-Type: text/html; charset=utf-8
Expires: -1
Server: Microsoft-HTTPAPI/2.0
P3P: ADFS doesn't have P3P policy, please contact your site's admin
for more details
Set-Cookie:
SamlSession=a3dpbnRlcnMtUHJpbWFyeS1BZG1pbi0xNzgmRmFsc2Umcng4NnJDZmFKV
XFxVWx3bkl1MnFuUSUzZCUzZCYmJiYmXzE3MjAyZTA5LThmMDgtNDRkZC04Yzg5LTQ3ND
UxYzA3ZjkzYw==; path=/adfs; HttpOnly; Secure
Set-Cookie: MSISAuthenticated=MTEvNy8yMDE4IDQ6MzI6NTkgUE0=;
path=/adfs; HttpOnly; Secure
Set-Cookie: MSISLoopDetectionCookie=MjAxOC0xMS0wNzoxNjozMjo1OVpcMQ==;
path=/adfs; HttpOnly; Secure
Date: Wed, 07 Nov 2018 16:32:59 GMT

<form method="POST" name="hiddenform"
action="https://storagegrid.example.com:443/api/saml-response">
  <input type="hidden" name="SAMLResponse"
value="PHNhbWxwOlJlc3BvbnN...1scDpSZXNwb25zZT4=" /><input
type="hidden" name="RelayState" value="12345" />
```

i. 非表示フィールドから `SAMLResponse`を保存します：

```
export SAMLResponse='PHNhbWxwOlJlc3BvbnN...1scDpSZXNwb25zZT4='
```

j. 保存した `SAMLResponse`を使用して、StorageGRID`/api/saml-response`リクエストを作成し、StorageGRID認証トークンを生成します。

`RelayState`には、テナントアカウントIDを使用するか、Grid Management APIにサインインする場合は0を使用してください。

```
curl -X POST "https://$STORAGEGRID_ADDRESS:443/api/saml-response" \
  -H "accept: application/json" \
  --data-urlencode "SAMLResponse=$SAMLResponse" \
  --data-urlencode "RelayState=$TENANTACCOUNTID" \
  | python -m json.tool
```

レスポンスには認証トークンが含まれています。

```
{
    "apiVersion": "3.0",
    "data": "56eb07bf-21f6-40b7-af0b-5c6cacfb25e7",
    "responseTime": "2018-11-07T21:32:53.486Z",
    "status": "success"
}
```

a. レスポンス内の認証トークンを `MYTOKEN`として保存します。

```
export MYTOKEN="56eb07bf-21f6-40b7-af0b-5c6cacfb25e7"
```

これで、SSO を使用しない場合と同様に、 `MYTOKEN`を他のリクエストに使用できます。

シングル サインオンが有効になっている場合にAPIからサインアウトする

シングル サインオン（SSO）が有効になっている場合は、Grid Management APIまたはTenant Management APIからサインアウトするために、一連のAPIリクエストを発行する必要があります。これらの手順は、SSOのIDプロバイダーとしてActive Directoryを使用している場合に適用されます。

タスク概要

必要に応じて、組織のシングル ログアウト ページからログアウトすることで、StorageGRID API からサインアウトできます。または、StorageGRID から シングル ログアウト（SLO）をトリガーすることもできます。その場合、有効な StorageGRID ベアラートークンが必要です。

手順

1. 署名付きログアウトリクエストを生成するには、`cookie "sso=true"` を SLO API に渡します。

```
curl -k -X DELETE "https://$STORAGEGRID_ADDRESS/api/v3/authorize" \
-H "accept: application/json" \
-H "Authorization: Bearer $MYTOKEN" \
--cookie "sso=true" \
| python -m json.tool
```

ログアウト用のURLが返されます：

```
{
    "apiVersion": "3.0",
    "data":
"https://adfs.example.com/adfs/ls/?SAMLRequest=fZDNboMwEIRfhZ...HcQ%3D%3
D",
    "responseTime": "2018-11-20T22:20:30.839Z",
    "status": "success"
}
```

2. ログアウトURLを保存してください。

```
export LOGOUT_REQUEST
='https://adfs.example.com/adfs/ls/?SAMLRequest=fZDNboMwEIRfhZ...HcQ%3D%
3D'
```

3. ログアウト URL にリクエストを送信して SLO をトリガーし、StorageGRID にリダイレクトします。

```
curl --include "$LOGOUT_REQUEST"
```

302レスポンスが返されます。リダイレクト先は、APIのみのログアウトには適用されません。

```
HTTP/1.1 302 Found
Location: https://$STORAGEGRID_ADDRESS:443/api/saml-
logout?SAMLResponse=fVLLasMwEPwVo7ss%...%23rsa-sha256
Set-Cookie: MSISSignoutProtocol=U2FtbA==; expires=Tue, 20 Nov 2018
22:35:03 GMT; path=/adfs; HttpOnly; Secure
```

4. StorageGRID ベアラートークンを削除します。

StorageGRID ベアラートークンの削除は、SSO を使用しない場合と同じように機能します。`cookie "sso=true" が指定されていない場合、ユーザーは SSO の状態に影響を与えることなく StorageGRID からログアウトされます。

```
curl -X DELETE "https://$STORAGEGRID_ADDRESS/api/v3/authorize" \
-H "accept: application/json" \
-H "Authorization: Bearer $MYTOKEN" \
--include
```

A `204 No Content`応答は、ユーザーがサインアウトしたことを示します。

```
HTTP/1.1 204 No Content
```

シングル サインオン が有効になっている場合は **StorageGRID API** を使用します（**Entra ID**）

"設定および有効化されたシングル サインオン（SSO）"をお持ちで、Entra ID をSSOプロバイダーとして使用している場合は、2つのサンプルスクリプトを使用して、Grid Management APIまたはTenant Management APIで有効な認証トークンを取得できます。

Entra ID のシングル サインオンが有効になっている場合は、**API** にサインインします

これらの手順は、SSOのIDプロバイダーとしてEntra IDを使用している場合に適用されます。

開始する前に

- StorageGRID ユーザーグループに属するフェデレーションユーザーの SSO メールアドレスとパスワードを知っている必要があります。
- テナント管理APIにアクセスするには、テナントアカウントIDを知っている必要があります。

タスク概要

認証トークンを取得するには、以下のサンプルスクリプトを使用できます。

- この `storagegrid-ssoauth-azure.py` Python スクリプト
- The `storagegrid-ssoauth-azure.js` Node.js スクリプト

両方のスクリプトは、StorageGRID インストールファイルディレクトリ（`./rpms` は RHEL の場合、`./debs` は Ubuntu または Debian の場合、 `./vsphere` は VMware の場合）にあります。

Entra ID との独自の API 統合を作成するには、 `storagegrid-ssoauth-azure.py`スクリプトを参照してください。Python スクリプトは StorageGRID に対して直接 2 つのリクエストを行い（最初に SAMLRequest を取得し、その後認証トークンを取得します）、さらに Node.js スクリプトを呼び出して Entra ID とやり取りし、SSO 操作を実行します。

SSO操作は一連のAPIリクエストを使用して実行できますが、その方法は簡単ではありません。Puppeteer Node.jsモジュールは、Entra ID SSOインターフェースのスクレイピングに使用されます。

URLエンコードに問題がある場合、次のようなエラーが表示されることがあります： `Unsupported SAML version`。

手順

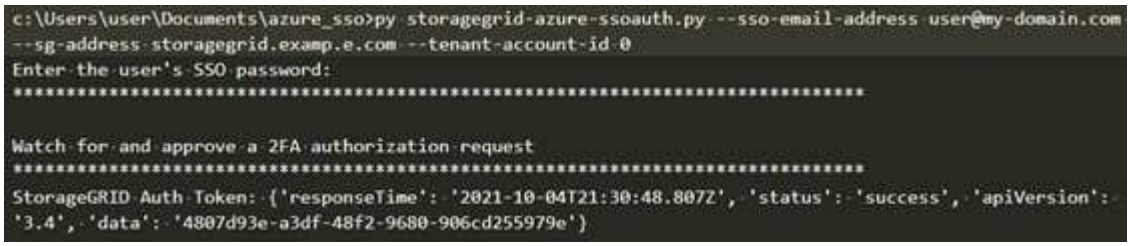
1. 必要な依存関係を以下の手順でインストールします：
   a. Node.jsをインストールします（ "https://nodejs.org/en/download/"を参照）。
   b. 必要なNode.jsモジュール（puppeteerとjsdom）をインストールします：

      ```
      npm install -g <module>
      ```

2. PythonスクリプトをPythonインタープリタに渡して実行します。

   次に、Pythonスクリプトが対応するNode.jsスクリプトを呼び出し、Entra ID SSOのやり取りを実行します。

3. プロンプトが表示されたら、以下の引数に値を入力してください（または、パラメータを使用して渡してください）：
   - Entra ID へのサインインに使用される SSO メールアドレス
   - StorageGRID のアドレス
   - テナント管理APIにアクセスする場合は、テナントアカウントID
4. 指示が表示されたらパスワードを入力し、必要に応じて Entra ID への MFA 認証を提供する準備をしてください。



> このスクリプトは、多要素認証（MFA）がMicrosoft Authenticatorを使用して行われることを前提としています。他の形式の多要素認証（テキストメッセージで受信したコードを入力するなど）をサポートするには、スクリプトを修正する必要がある場合があります。

StorageGRID 認証トークンが出力に含まれています。これで、SSO を使用していない場合と同様に、トークンを他のリクエストに使用できます。

シングル サインオンが有効になっている場合は **StorageGRID API** を使用します（**PingFederate**）

"設定および有効化されたシングル サインオン（SSO）"を使用しており、SSOプロバイダーとしてPingFederateを使用している場合は、Grid Management APIまたはTenant Management APIで有効な認証トークンを取得するために、一連のAPIリクエストを発行する必要があります。

シングル サインオンが有効になっている場合に**API**にサインインする

これらの手順は、SSOのIDプロバイダーとしてPingFederateを使用している場合に適用されます

開始する前に

- StorageGRID ユーザーグループに属するフェデレーションユーザーの SSO ユーザー名とパスワードを把握しています。
- テナント管理APIにアクセスするには、テナントアカウントIDを知っている必要があります。

タスク概要

認証トークンを取得するには、以下のいずれかの例を使用できます。

- `storagegrid-ssoauth.py` Python スクリプトは、StorageGRID インストールファイルディレクトリ（(`./rpms` RHEL の場合、 `./debs` Ubuntu または Debian の場合、 `./vsphere` VMware の場合）にあります。
- curlリクエストのワークフロー例。

  curlワークフローは、実行速度が遅すぎるとタイムアウトする可能性があります。次のようなエラーが表示される場合があります： `A valid SubjectConfirmation was not found on this Response`。

  > この例のcurlワークフローでは、パスワードが他のユーザーに見られるのを防ぐことはできません。

  URLエンコードに問題がある場合、次のようなエラーが表示されることがあります： `Unsupported SAML version`。

手順

1. 認証トークンを取得するには、以下のいずれかの方法を選択してください。
   - `storagegrid-ssoauth.py`Python スクリプトを使用します。ステップ 2 に進みます。
   - curl リクエストを使用します。ステップ 3 に進みます。
2. `storagegrid-ssoauth.py`スクリプトを使用する場合は、Pythonインタープリタにスクリプトを渡して実行してください。

   指示が表示されたら、以下の引数に値を入力してください。

   - SSO方式。「pingfederate」の任意のバリエーション（PINGFEDERATE、pingfederateなど）を入力できます。
   - SSOユーザ名
   - StorageGRID がインストールされているドメイン。このフィールドは、PingFederate では使用されません。空欄のままでも、任意の値を入力しても構いません。
   - StorageGRID のアドレス
   - テナント管理APIにアクセスする場合は、テナントアカウントIDが必要です。

```
python3 storagegrid-ssoauth.py
sso_method: pingfederate
saml_user: my-sso-username
saml_domain:
sg_address: storagegrid.example.com
tenant_account_id: 12345
Enter the user's SAML password:
********************************************************************************

********************************************************************************
StorageGRID Auth Token: 56eb07bf-21f6-40b7-afob-5c6cacfb25e7
```

StorageGRID 認証トークンが出力に含まれています。これで、SSO を使用していない場合と同様に、トークンを他のリクエストに使用できます。

3. curlリクエストを使用する場合は、以下の手順に従ってください。

   a. ログインに必要な変数を宣言します。

```
export SAMLUSER='my-sso-username'
export SAMLPASSWORD='my-password'
export TENANTACCOUNTID='12345'
export STORAGEGRID_ADDRESS='storagegrid.example.com'
```

   グリッド管理APIにアクセスするには、0を `TENANTACCOUNTID`として使用します。

   b. 署名付き認証URLを受け取るには、 `/api/v3/authorize-saml`にPOSTリクエストを発行し、レスポンスから余分なJSONエンコーディングを削除します。

   この例は、TENANTACCOUNTID の署名付き認証 URL に対する POST リクエストを示しています。結果は、JSON エンコーディングを削除するために python -m json.tool に渡されます。

```
curl -X POST "https://$STORAGEGRID_ADDRESS/api/v3/authorize-saml" \
  -H "accept: application/json" -H  "Content-Type: application/json"
\
  --data "{\"accountId\": \"$TENANTACCOUNTID\"}" | python -m
json.tool
```

   この例に対する応答には、URLエンコードされた署名付きURLが含まれていますが、追加のJSONエンコード層は含まれていません。

```
{
    "apiVersion": "3.0",
    "data": "https://my-pf-baseurl/idp/SSO.saml2?...",
    "responseTime": "2018-11-06T16:30:23.355Z",
    "status": "success"
}
```

c. 応答から取得した `SAMLRequest`を、後続のコマンドで使用するために保存します。

```
export SAMLREQUEST="https://my-pf-baseurl/idp/SSO.saml2?..."
```

d. レスポンスとクッキーをエクスポートし、レスポンスを出力します。

```
RESPONSE=$(curl -c - "$SAMLREQUEST")
```

```
echo "$RESPONSE" | grep 'input type="hidden" name="pf.adapterId"
id="pf.adapterId"'
```

e. 'pf.adapterId' の値をエクスポートし、レスポンスをエコーします：

```
export ADAPTER='myAdapter'
```

```
echo "$RESPONSE" | grep 'base'
```

f. 「href」の値をエクスポートし（末尾のスラッシュ / を削除）、レスポンスを出力します：

```
export BASEURL='https://my-pf-baseurl'
```

```
echo "$RESPONSE" | grep 'form method="POST"'
```

g. 「action」の値をエクスポートします。

```
export SSOPING='/idp/.../resumeSAML20/idp/SSO.ping'
```

h. 認証情報とともにクッキーを送信する：

```
curl -b <(echo "$RESPONSE") -X POST "$BASEURL$SSOPING" \
--data "pf.username=$SAMLUSER&pf.pass=
$SAMLPASSWORD&pf.ok=clicked&pf.cancel=&pf.adapterId=$ADAPTER"
--include
```

i. 非表示フィールドから `SAMLResponse`を保存します：

```
export SAMLResponse='PHNhbWxwOlJlc3BvbnN...1scDpSZXNwb25zZT4='
```

j. 保存した `SAMLResponse`を使用して、StorageGRID`/api/saml-response`リクエストを作成し、StorageGRID認証トークンを生成します。

`RelayState`には、テナントアカウントIDを使用するか、Grid Management APIにサインインする場合は0を使用してください。

```
curl -X POST "https://$STORAGEGRID_ADDRESS:443/api/saml-response" \
  -H "accept: application/json" \
  --data-urlencode "SAMLResponse=$SAMLResponse" \
  --data-urlencode "RelayState=$TENANTACCOUNTID" \
  | python -m json.tool
```

レスポンスには認証トークンが含まれています。

```
{
    "apiVersion": "3.0",
    "data": "56eb07bf-21f6-40b7-af0b-5c6cacfb25e7",
    "responseTime": "2018-11-07T21:32:53.486Z",
    "status": "success"
}
```

a. レスポンス内の認証トークンを `MYTOKEN`として保存します。

```
export MYTOKEN="56eb07bf-21f6-40b7-af0b-5c6cacfb25e7"
```

これで、SSO を使用しない場合と同様に、 `MYTOKEN`を他のリクエストに使用できます。

シングル サインオンが有効になっている場合にAPIからサインアウトする

シングル サインオン（SSO）が有効になっている場合は、Grid Management APIまたはTenant Management APIからサインアウトするために、一連のAPIリクエストを発行する必要があります。これらの手順は、SSO

のIDプロバイダーとしてPingFederateを使用している場合に適用されます。

タスク概要

必要に応じて、組織のシングル ログアウト ページからログアウトすることで、StorageGRID API からサインアウトできます。または、StorageGRID から シングル ログアウト（SLO）をトリガーすることもできます。その場合、有効な StorageGRID ベアラートークンが必要です。

手順

1. 署名付きログアウトリクエストを生成するには、`cookie "sso=true"` を SLO API に渡します。

```
curl -k -X DELETE "https://$STORAGEGRID_ADDRESS/api/v3/authorize" \
-H "accept: application/json" \
-H "Authorization: Bearer $MYTOKEN" \
--cookie "sso=true" \
| python -m json.tool
```

ログアウト用のURLが返されます：

```
{
    "apiVersion": "3.0",
    "data": "https://my-ping-
url/idp/SLO.saml2?SAMLRequest=fZDNboMwEIRfhZ...HcQ%3D%3D",
    "responseTime": "2021-10-12T22:20:30.839Z",
    "status": "success"
}
```

2. ログアウトURLを保存してください。

```
export LOGOUT_REQUEST='https://my-ping-
url/idp/SLO.saml2?SAMLRequest=fZDNboMwEIRfhZ...HcQ%3D%3D'
```

3. ログアウト URL にリクエストを送信して SLO をトリガーし、StorageGRID にリダイレクトします。

```
curl --include "$LOGOUT_REQUEST"
```

302レスポンスが返されます。リダイレクト先は、APIのみのログアウトには適用されません。

```
HTTP/1.1 302 Found
Location: https://$STORAGEGRID_ADDRESS:443/api/saml-
logout?SAMLResponse=fVLLasMwEPwVo7ss%...%23rsa-sha256
Set-Cookie: PF=QoKs...SgCC; Path=/; Secure; HttpOnly; SameSite=None
```

4. StorageGRID ベアラートークンを削除します。

   StorageGRID ベアラートークンの削除は、SSO を使用しない場合と同じように機能します。`cookie "sso=true" が指定されていない場合、ユーザーは SSO の状態に影響を与えることなく StorageGRID からログアウトされます。

```
curl -X DELETE "https://$STORAGEGRID_ADDRESS/api/v3/authorize" \
-H "accept: application/json" \
-H "Authorization: Bearer $MYTOKEN" \
--include
```

   A `204 No Content`応答は、ユーザーがサインアウトしたことを示します。

```
HTTP/1.1 204 No Content
```

API を使用して StorageGRID 機能を無効化する

Grid Management API を使用すると、StorageGRID システムの特定の機能を完全に無効にできます。機能が無効化されると、その機能に関連するタスクを実行するための権限を誰にも割り当てることができなくなります。

タスク概要

無効化された機能システムを使用すると、StorageGRID システム内の特定の機能へのアクセスを防止できます。機能を無効化することは、ルートユーザー、または **Root access** 権限を持つ管理者グループに属するユーザーがその機能を使用できないようにする唯一の方法です。

この機能がどのように役立つかを理解するために、次のシナリオを考えてみましょう。

A社は、テナントアカウントを作成することで自社の StorageGRID システムのストレージ容量をリースするサービスプロバイダーです。A社は、賃借人のオブジェクトのセキュリティを保護するため、アカウントが展開された後、自社の従業員がテナントアカウントに一切アクセスできないようにしたいと考えています。

A社は、グリッド管理APIの「機能無効化」システムを使用することで、この目標を達成できます。Grid Manager（UIとAPIの両方）で「テナントルートパスワードの変更」機能を完全に無効化することで、A社は、管理者ユーザー（ルートユーザーおよび「ルートアクセス」権限を持つグループに属するユーザーを含む）が、どのテナントアカウントのルートユーザーのパスワードも変更できないようにしています。

手順

1. グリッド管理APIのSwaggerドキュメントにアクセスしてください。"グリッド管理APIを使用する"を参照してください。
2. 機能無効化エンドポイントを探してください。
3. テナントのルートパスワードの変更などの機能を無効にするには、次のような本文を API に送信します。

```
{ "grid": {"changeTenantRootPassword": true} }
```

リクエストが完了すると、「テナントのルートパスワードを変更する」機能は無効になります。「テナントのルートパスワードの変更」管理権限はユーザーインターフェースに表示されなくなり、テナントのルートパスワードを変更しようとする API リクエストはすべて「403 Forbidden」で失敗します。

無効化された機能を再有効化する

デフォルトでは、グリッド管理APIを使用して、無効化された機能を再度有効化できます。ただし、無効化した機能が二度と有効化されないようにするには、**activateFeatures** 機能そのものを無効化できます。

activateFeatures 機能は再有効化できません。この機能を無効化すると、無効化された他の機能を再有効化する機能が永久に失われることにご注意ください。失われた機能を復元するには、テクニカルサポートにお問い合わせください。

手順

1. グリッド管理APIのSwaggerドキュメントにアクセスします。
2. 機能無効化エンドポイントを探してください。
3. すべての機能を再度有効にするには、次のような本文をAPIに送信してください：

```
{ "grid": null }
```

このリクエストが完了すると、テナントのルートパスワード変更機能を含むすべての機能が再有効化されます。テナントのルートパスワードの変更 管理権限がユーザーインターフェイスに表示されるようになり、ユーザーが ルートアクセス または テナントのルートパスワードの変更 管理権限を持っている場合、テナントのルートパスワードを変更しようとするすべての API リクエストは成功します。

上記の例では、無効化されていたすべての機能が再有効化されます。無効化された機能のうち、無効化されたままにしておくべき機能がある場合は、PUT リクエストでそれらを明示的に指定する必要があります。例えば、テナントルートパスワードの変更機能を再有効化し、storageAdmin 管理権限を引き続き無効化するには、次の PUT リクエストを送信してください：+ `{ "grid": {"storageAdmin": true} }`

著作権に関する情報

商標に関する情報