Active IQ Unified Manager での REST API へのアクセスおよび認証

Active IQ Unified Manager

NetApp
January 15, 2026

目次

Active IQ Unified Manager での REST API へのアクセスおよび認証 . . . 1
- 認証 . . . 3
- Active IQ Unified Manager で使用される HTTP ステータスコード . . . 3
- Active IQ Unified Manager で API を使用する際の推奨事項です . . . 4
- トラブルシューティング用のログ . . . 5
- ジョブオブジェクトの非同期プロセス . . . 6
  - ジョブオブジェクトを使用して記述された非同期要求 . . . 6
  - API要求に関連付けられているジョブオブジェクトを照会する . . . 6
  - 非同期要求の手順 . . . 7
- Hello API server 」と入力します . . . 7

Active IQ Unified Manager での REST API へのアクセスおよび認証

Active IQ Unified Manager REST APIには、基本のHTTP認証メカニズムを使用して問題HTTP要求を実行できるRESTクライアントまたはプログラミングプラットフォームを使用してアクセスできます。

要求と応答の例：

- * リクエスト *

```
GET
https://<IP
address/hostname>:<port_number>/api/v2/datacenter/cluster/clusters
```

- * 応答 *

```
{
  "records": [
    {
      "key": "4c6bf721-2e3f-11e9-a3e2-
00a0985badbb:type=cluster,uuid=4c6bf721-2e3f-11e9-a3e2-00a0985badbb",
      "name": "fas8040-206-21",
      "uuid": "4c6bf721-2e3f-11e9-a3e2-00a0985badbb",
      "contact": null,
      "location": null,
      "version": {
        "full": "NetApp Release Dayblazer__9.5.0: Thu Jan 17 10:28:33
UTC 2019",
        "generation": 9,
        "major": 5,
        "minor": 0
      },
      "isSanOptimized": false,
      "management_ip": "10.226.207.25",
      "nodes": [
        {
          "key": "4c6bf721-2e3f-11e9-a3e2-
00a0985badbb:type=cluster_node,uuid=12cf06cc-2e3a-11e9-b9b4-
00a0985badbb",
          "uuid": "12cf06cc-2e3a-11e9-b9b4-00a0985badbb",
          "name": "fas8040-206-21-01",
          "_links": {
            "self": {
```

```
            "href": "/api/datacenter/cluster/nodes/4c6bf721-2e3f-11e9-
a3e2-00a0985badbb:type=cluster_node,uuid=12cf06cc-2e3a-11e9-b9b4-
00a0985badbb"
          }
        },
        "location": null,
        "version": {
          "full": "NetApp Release Dayblazer__9.5.0: Thu Jan 17
10:28:33 UTC 2019",
          "generation": 9,
          "major": 5,
          "minor": 0
        },
        "model": "FAS8040",
        "uptime": 13924095,
        "serial_number": "701424000157"
      },
      {
        "key": "4c6bf721-2e3f-11e9-a3e2-
00a0985badbb:type=cluster_node,uuid=1ed606ed-2e3a-11e9-a270-
00a0985bb9b7",
        "uuid": "1ed606ed-2e3a-11e9-a270-00a0985bb9b7",
        "name": "fas8040-206-21-02",
        "_links": {
          "self": {
            "href": "/api/datacenter/cluster/nodes/4c6bf721-2e3f-11e9-
a3e2-00a0985badbb:type=cluster_node,uuid=1ed606ed-2e3a-11e9-a270-
00a0985bb9b7"
          }
        },
        "location": null,
        "version": {
          "full": "NetApp Release Dayblazer__9.5.0: Thu Jan 17
10:28:33 UTC 2019",
          "generation": 9,
          "major": 5,
          "minor": 0
        },
        "model": "FAS8040",
        "uptime": 14012386,
        "serial_number": "701424000564"
      }
    ],
    "_links": {
      "self": {
        "href": "/api/datacenter/cluster/clusters/4c6bf721-2e3f-11e9-
```

```
a3e2-00a0985badbb:type=cluster,uuid=4c6bf721-2e3f-11e9-a3e2-
00a0985badbb"
        }
      }
    },
```

- *ip address/hostname* は API サーバの IP アドレスまたは完全修飾ドメイン名 (FQDN) です。
- ポート 443

  443 は、デフォルトの HTTPS ポートです。必要に応じて、 HTTPS ポートをカスタマイズできます。

Webブラウザから問題 HTTP要求を行うには、REST APIブラウザプラグインを使用する必要があります。cURL や Perl などのスクリプトプラットフォームを使用して、 REST API にアクセスすることもできます。

認証

Unified Manager では、 API の基本的な HTTP 認証方式がサポートされます。情報の流れ（要求と応答）をセキュリティで保護するために、 REST API には HTTPS 経由でのみアクセスできます。API サーバは、サーバ検証のためにすべてのクライアントに自己署名 SSL 証明書を提供します。この証明書は、カスタム証明書（または CA 証明書）で置き換えることができます。

REST API を呼び出すには、 API サーバへのユーザアクセスを設定する必要があります。ユーザには、ローカルユーザ（ローカルデータベースに格納されているユーザプロファイル）または LDAP ユーザ（ LDAP 経由で認証するように API サーバを設定している場合）を指定できます。Unified Manager Administration Console のユーザインターフェイスにログインして、ユーザアクセスを管理できます。

Active IQ Unified Manager で使用される HTTP ステータスコード

API の実行時や問題のトラブルシューティング時には、 Active IQ Unified Manager API で使用されるさまざまな HTTP ステータスコードとエラーコードについて理解しておく必要があります。

次の表に、認証に関連するエラーコードを示します。

HTTP ステータスコード	ステータスコードタイトル	説明
200	わかりました	同期 API 呼び出しの実行に成功した場合に返されます。
201	作成済み	Active Directory の設定など、同期呼び出しによって新しいリソースが作成されたことを示します。

HTTP ステータスコード	ステータスコードタイトル	説明
202.	承認済み	LUN やファイル共有の作成など、プロビジョニング機能の非同期呼び出しの実行が成功したときに返されます。
400	無効な要求です	入力検証に失敗したことを示します。ユーザは、要求の本文に有効なキーを指定するなど、入力を修正する必要があります。
401	不正な要求です	リソースを表示する権限がありません / 権限がありません。
403	禁止された要求です	アクセスしようとしていたリソースへのアクセスは禁止されています。
404	リソースが見つかりません	アクセスしようとしているリソースが見つかりません。
405	メソッドを使用できません	メソッドを使用できません
429	要求が多すぎます	指定した時間内にユーザが送信した要求が多すぎる場合に返されます。
500	内部サーバエラー。	内部サーバエラー。サーバから応答を取得できませんでした。この内部サーバエラーは、永続的な場合とそうでない場合があります。たとえば 'get' または 'get all' 操作を実行してこのエラーを受信した場合 ' この操作を最低 5 回繰り返すことをお勧めします永続的なエラーの場合、引き続きステータスコード 500 が返されます。処理が成功すると、ステータスコード 200 が返されます。

Active IQ Unified Manager で API を使用する際の推奨事項です

Active IQ Unified Manager で API を使用するときは、特定の推奨される方法に従ってください。

- 有効に実行するには、すべての応答コンテンツタイプが次の形式である必要があります。

```
application/json
```

- API のバージョン番号は、製品のバージョン番号とは関係ありません。Unified Manager インスタンスで使用可能な最新バージョンの API を使用する必要があります。Unified Manager API のバージョンの詳細については、「Active IQ Unified Manager での ST API のバージョン管理」を参照してください。
- Unified Manager API を使用して配列値を更新する場合は、値の文字列全体を更新する必要があります。配列に値を付加することはできません。既存のアレイのみを交換できます。
- パイプ（|）やワイルドカード（+ * +）などのフィルタ演算子をすべてのクエリパラメータに使用できます。ただし、指標 API の IOPS やパフォーマンスなど、二重値は除きます。
- フィルタ演算子のワイルドカード（*）とパイプ（|）を組み合わせてオブジェクトを照会しないでください。間違った数のオブジェクトが取得される可能性があります。
- フィルタに値を使用する場合は、値に「？」が含まれていないことを確認してください を押します。これは、 SQL インジェクションのリスクを軽減するためです。
- すべての API に対する 'get' (すべて) 要求は ' 最大 1000 件のレコードを返すことに注意してください「 mAX_records 」パラメータを 1000 より大きい値に設定してクエリを実行した場合でも、返されるレコードは 1000 件だけです。
- 管理機能を実行する場合は、 Unified Manager UI を使用することを推奨します。

トラブルシューティング用のログ

システムログを使用して、 API の実行中に発生する可能性のある障害の原因を分析し、問題のトラブルシューティングを行うことができます。

API 呼び出しに関連する問題のトラブルシューティングを行うには、次の場所からログを取得します。

ログの場所	使用
/var/log/ocie/access_log.log	API を呼び出しているユーザ名、開始時刻、実行時間、ステータス、 URL など、 API 呼び出しに関するすべての詳細情報が含まれます。 このログファイルを使用して、頻繁に使用される API を確認したり、 GUI ワークフローのトラブルシューティングを行ったりできます。また、実行時間に基づいて分析の規模を調整することもできます。
/var/log/ocum/ocumentserver.log' のように表示されます	すべての API 実行ログが含まれます。 このログファイルを使用して、 API 呼び出しのトラブルシューティングとデバッグを行うことができます。

ログの場所	使用
/var/log/ocie/sserver.log' のようになります	すべての Wildfly サーバ構成と、サービスの開始 / 停止に関連するログが含まれています。 このログファイルを使用して、 Wildfly サーバの開始、停止、または導入時に発生する問題のルート原因を見つけることができます。
/var/log/mocie /au.log	Acquisition Unit 関連のログが含まれます。 このログファイルは、 ONTAP でオブジェクトを作成、変更、または削除したときに使用できますが、 Active IQ Unified Manager REST API には反映されません。

ジョブオブジェクトの非同期プロセス

Active IQ Unified Manager には、他の API の実行中に実行されたジョブに関する情報を取得する「 jobs'API が用意されています。非同期処理がジョブオブジェクトを使用してどのように動作するかを理解しておく必要があります。

一部の API 呼び出し、特にリソースの追加や変更に使用される API 呼び出しは、他の呼び出しよりも完了に時間がかかることがあります。Unified Manager は、これらの長時間実行されている要求を非同期的に処理します。

ジョブオブジェクトを使用して記述された非同期要求

非同期的に実行される API 呼び出しを行うと、 HTTP 応答コード 202 が返されます。この応答コードは、要求が正常に検証され受け入れられたものの、まだ完了していないことを示します。要求はバックグラウンドタスクとして処理され、クライアントへの最初の HTTP 応答後も引き続き実行されます。応答には、要求に対応するジョブオブジェクトと、その一意の識別子が含まれます。

API要求に関連付けられているジョブオブジェクトを照会する

HTTP 応答で返されるジョブオブジェクトには、いくつかのプロパティが含まれています。状態プロパティを照会して、要求が正常に完了したかどうかを確認できます。ジョブオブジェクトは次のいずれかの状態になります。

- 「 normal 」
- 「警告」
- 「 PARTIAL FAILURES 」
- 「エラー」

ジョブオブジェクトをポーリングするときに、タスクの終了状態（成功または失敗）を検出するために使用できる 2 つの方法があります。

- 標準のポーリング要求：現在のジョブの状態がすぐに返されます。

- 長いポーリング要求 : ジョブの状態が 'normal’error’’partial_failures に移行したとき

非同期要求の手順

非同期 API 呼び出しを完了する大まかな手順を次に示します。

1. 問題：非同期 API 呼び出し。
2. 要求が正常に受け取られたことを示す HTTP 応答 202 を受信します。
3. 応答の本文からジョブオブジェクトの識別子を抽出します。
4. ループ内で ' ジョブ・オブジェクトが NORMAL ' ERROR ' または PARTIAL FAILURES の最終状態になるまで待ちます
5. ジョブの終了状態を確認し、ジョブの結果を取得します。

Hello API server 」と入力します

_Hello API server_は 、シンプルな REST クライアントを使用して Active IQ Unified Manager で REST API を呼び出す方法を示すサンプルプログラムです。サンプルプログラムでは、 API サーバーに関する基本的な詳細情報が JSON 形式で提供されています (サーバーは「 application/json' 形式」のみをサポートしています) 。

使用される URI は「 https://<hostname>/api/datacenter/svm/svms.` 」です このサンプルコードでは、次の入力パラメータを使用します。

- API サーバの IP アドレスまたは FQDN
- オプション：ポート番号（デフォルト： 443 ）
- ユーザ名
- パスワード
- 応答形式 (application/json`)

REST API を呼び出すために、 Jersey や RESTEasy など他のスクリプトを使用して Active IQ Unified Manager 用の Java REST クライアントを作成することもできます。サンプルコードに関して、次の点に注意する必要があります。

- Active IQ Unified Manager への HTTPS 接続を使用して、指定した REST URI を呼び出します
- Active IQ Unified Manager から提供される証明書を無視します
- ハンドシェイク中にホスト名の検証をスキップします
- URI 接続には javax.net.ssl.HttpsURLConnection` を使用します
- サードパーティのライブラリ（ org.apache.commons.codec.binary.Base64` ）を使用して、 HTTP ベーシック認証で使用される Base64 エンコード文字列を作成します

サンプルコードをコンパイルして実行するには、 Java コンパイラ 1.8 以降を使用する必要があります。

```
import java.io.BufferedReader;
```

```
import java.io.InputStreamReader;
import java.net.URL;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;
import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.SSLSession;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import org.apache.commons.codec.binary.Base64;

public class HelloApiServer {

    private static String server;
    private static String user;
    private static String password;
    private static String response_format = "json";
    private static String server_url;
    private static String port = null;

    /*
     * * The main method which takes user inputs and performs the *
necessary steps
     * to invoke the REST URI and show the response
     */ public static void main(String[] args) {
        if (args.length < 2 || args.length > 3) {
            printUsage();
            System.exit(1);
        }
        setUserArguments(args);
        String serverBaseUrl = "https://" + server;
        if (null != port) {
            serverBaseUrl = serverBaseUrl + ":" + port;
        }
        server_url = serverBaseUrl + "/api/datacenter/svm/svms";
        try {
            HttpsURLConnection connection =
getAllTrustingHttpsUrlConnection();
            if (connection == null) {
                System.err.println("FATAL: Failed to create HTTPS
connection to URL: " + server_url);
                System.exit(1);
            }
            System.out.println("Invoking API: " + server_url);
            connection.setRequestMethod("GET");
```

```
            connection.setRequestProperty("Accept", "application/" +
response_format);
            String authString = getAuthorizationString();
            connection.setRequestProperty("Authorization", "Basic " +
authString);
            if (connection.getResponseCode() != 200) {
                System.err.println("API Invocation Failed : HTTP error
code : " + connection.getResponseCode() + " : "
                        + connection.getResponseMessage());
                System.exit(1);
            }
            BufferedReader br = new BufferedReader(new
InputStreamReader((connection.getInputStream())));
            String response;
            System.out.println("Response:");
            while ((response = br.readLine()) != null) {
                System.out.println(response);
            }
            connection.disconnect();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /* Print the usage of this sample code */ private static void
printUsage() {
        System.out.println("\nUsage:\n\tHelloApiServer <hostname> <user>
<password>\n");
        System.out.println("\nExamples:\n\tHelloApiServer localhost admin
mypassword");
        System.out.println("\tHelloApiServer 10.22.12.34:8320 admin
password");
        System.out.println("\tHelloApiServer 10.22.12.34 admin password
");
        System.out.println("\tHelloApiServer 10.22.12.34:8212 admin
password \n");
        System.out.println("\nNote:\n\t(1) When port number is not
provided, 443 is chosen by default.");
    }

    /* * Set the server, port, username and password * based on user
inputs. */ private static void setUserArguments(
            String[] args) {
        server = args[0];
        user = args[1];
        password = args[2];
```

```
        if (server.contains(":")) {
            String[] parts = server.split(":");
            server = parts[0];
            port = parts[1];
        }
    }

    /*
     * * Create a trust manager which accepts all certificates and * use
this trust
     * manager to initialize the SSL Context. * Create a
HttpsURLConnection for this
     * SSL Context and skip * server hostname verification during SSL
handshake. * *
     * Note: Trusting all certificates or skipping hostname verification *
is not
     * required for API Services to work. These are done here to * keep
this sample
     * REST Client code as simple as possible.
     */ private static HttpsURLConnection
getAllTrustingHttpsUrlConnection()    {        HttpsURLConnection conn =
null;        try {            /* Creating a trust manager that does not
validate certificate chains */            TrustManager[]
trustAllCertificatesManager = new                    TrustManager[]{new
X509TrustManager(){
     public X509Certificate[] getAcceptedIssuers(){return  null;}
     public void checkClientTrusted(X509Certificate[]
certs, String authType){}
     public void checkServerTrusted(X509Certificate[]
certs, String authType){}            }};            /* Initialize the
SSLContext with the all-trusting trust manager */
     SSLContext sslContext = SSLContext.getInstance("TLS");
sslContext.init(null, trustAllCertificatesManager, new
SecureRandom());
HttpsURLConnection.setDefaultSSLSocketFactory(sslContext.getSocketFactory(
));            URL url = new URL(server_url);            conn =
(HttpsURLConnection) url.openConnection();            /* Do not perform an
actual hostname verification during SSL Handshake.            Let all
hostname pass through as verified.*/
conn.setHostnameVerifier(new HostnameVerifier() {                public
boolean verify(String host, SSLSession                        session) {
return true;                }            });        } catch (Exception e)
{            e.printStackTrace();        }        return conn;    }

    /*
     * * This forms the Base64 encoded string using the username and
```

```
password *
     * provided by the user. This is required for HTTP Basic
Authentication.
     */ private static String getAuthorizationString() {
        String userPassword = user + ":" + password;
        byte[] authEncodedBytes =
Base64.encodeBase64(userPassword.getBytes());
        String authString = new String(authEncodedBytes);
        return authString;
    }

}
```

著作権に関する情報

商標に関する情報