REST API存取與驗Active IQ Unified Manager證功能

Active IQ Unified Manager 9.8

NetApp
April 16, 2024

目錄

REST API存取與驗Active IQ Unified Manager 證功能

您可以使用任何可發出HTTP要求的網頁瀏覽器或程式設計平台、來存取此靜態API。Active IQ Unified ManagerUnified Manager支援基本HTTP驗證機制。在呼叫Unified Manager REST API之前、您必須先驗證使用者。

REST存取

您可以使用任何可發出HTTP要求的網頁瀏覽器或程式設計平台來存取Unified Manager REST API。例如、登入Unified Manager之後、您可以在任何瀏覽器中輸入URL、以擷取所有管理工作站的屬性、例如管理站台名稱、金鑰和IP位址。

- 申請

  取得 https://<IP 位址/主機名稱>：<port_number> esei/API/v2/datacenter/叢集/叢集

- 回應

```
{
  "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` 是API伺服器的IP位址或完整網域名稱（FQDN）。
- 連接埠443

  443是預設的HTTPS連接埠。您可以視需要自訂HTTPS連接埠。

若要從網頁瀏覽器發出POST、修補及刪除HTTP要求、您必須使用瀏覽器外掛程式。您也可以使用諸如Curl和Perl等指令碼平台來存取REST API。

驗證

Unified Manager支援API的基本HTTP驗證配置。為了確保資訊流程安全（要求和回應）、REST API只能透過HTTPS存取。API伺服器會為所有用戶端提供自我簽署的SSL憑證、以便進行伺服器驗證。此憑證可由自訂憑證（或CA憑證）取代。

您必須設定使用者對API伺服器的存取權限、才能叫用REST API。使用者可以是本機使用者（儲存在本機資料庫中的使用者設定檔）或LDAP使用者（如果您已將API伺服器設定為透過LDAP驗證）。您可以登入Unified Manager管理主控台使用者介面來管理使用者存取。

HTTP狀態代碼用於Active IQ Unified Manager

在執行API或疑難排解問題時、您應該注意Active IQ Unified Manager 到由資訊技術API使用的各種HTTP狀態代碼和錯誤代碼。

下表列出與驗證相關的錯誤代碼：

HTTP狀態代碼	狀態代碼標題	說明
200	好的	成功執行同步 API 呼叫時傳回。
201.	已建立	透過同步呼叫（例如 Active Directory 組態）來建立新資源。
202.02	已接受	成功執行非同步呼叫以進行資源配置功能（例如建立 LUN 和檔案共用）時傳回。
400	無效要求	表示輸入驗證失敗。使用者必須修正輸入、例如要求內容中的有效金鑰。

HTTP狀態代碼	狀態代碼標題	說明
401.	未獲授權的要求	您無權檢視資源/未獲授權。
403.	禁止的要求	禁止存取您嘗試存取的資源。
404..	找不到資源	找不到您要聯絡的資源。
405	不允許使用方法	不允許使用方法。
429	要求太多	當使用者在特定時間內傳送過多要求時傳回。
500	內部伺服器錯誤	內部伺服器錯誤。無法從伺服器取得回應。此內部伺服器錯誤可能是永久性的、也可能不是永久性的。例如、如果您執行 `GET` 或 `GET ALL` 操作並接收此錯誤、建議您重複此作業至少五次重試。如果是永久性錯誤、則傳回的狀態代碼仍為 500 。如果作業成功、則傳回的狀態代碼為 200 。

使用API進行Active IQ Unified Manager 效能不穩定的建議

在Active IQ Unified Manager 使用API時、您應該遵循某些建議的實務做法。

- 所有回應內容類型必須採用下列格式、才能有效執行：

```
application/json
```

- API版本編號與產品版本編號無關。您應該使用Unified Manager執行個體可用的最新API版本。如需Unified Manager API版本的詳細資訊、請參閱Active IQ Unified Manager 「REST API版本管理功能」一節。
- 使用Unified Manager API更新陣列值時、您必須更新整個值字串。您無法將值附加至陣列。您只能取代現有的陣列。
- 您可以使用篩選運算子、例如（|）和萬用字元來查詢參數。使用篩選運算子萬用字元（*）和管道（|）的組合、避免查詢物件。它可能會擷取不正確的物件數。
- 請注意 `GET` （全部）任何API的要求最多可傳回1000筆記錄。即使您是透過設定來執行查詢 `max_records` 值大於1000的參數、只會傳回1000筆記錄。
- 若要執行管理功能、建議您使用Unified Manager UI。

疑難排解記錄

系統記錄可讓您分析失敗的原因、並疑難排解執行API時可能發生的問題。

從下列位置擷取記錄、以疑難排解與API呼叫相關的問題。

記錄位置	使用
`/var/log/ocie/access_log.log`	包含所有API呼叫詳細資料、例如叫用API的使用者名稱、開始時間、執行時間、狀態和URL。 您可以使用此記錄檔來檢查常用的API、或疑難排解任何GUI工作流程。您也可以根據執行時間、使用它來擴充分析。
`/var/log/ocum/ocumserver.log`	包含所有API執行記錄。 您可以使用此記錄檔來疑難排解及偵錯API呼叫。
`/var/log/ocie/server.log`	包含所有Wildfly伺服器部署及啟動/停止服務相關記錄。 您可以使用此記錄檔來找出開始、停止或部署Wildfly伺服器期間發生任何問題的根本原因。
`/var/log/ocie/au.log`	包含擷取單位相關記錄。 您可以在建立、修改或刪除ONTAP 任何物件時、使用此記錄檔、但Active IQ Unified Manager 這些物件不會反映在整個過程中。

工作物件非同步處理

提供Active IQ Unified Manager `jobs` 擷取執行其他API時所執行工作相關資訊的API。您必須瞭解使用工作物件進行非同步處理的方式。

有些 API 呼叫（尤其是用於新增或修改資源的呼叫）可能需要比其他呼叫更長的時間才能完成。Unified Manager會以非同步方式處理這些長時間執行的要求。

使用工作物件說明的非同步要求

在非同步執行 API 呼叫之後、 HTTP 回應代碼 202 表示該要求已成功驗證並接受、但尚未完成。此要求會以背景工作的形式處理、並在對用戶端的初始 HTTP 回應之後繼續執行。回應包括繫留要求的工作物件、包括其唯一識別碼。

查詢與**API**要求相關聯的工作物件

HTTP回應中傳回的工作物件包含數個內容。您可以查詢狀態內容、以判斷要求是否成功完成。工作物件可以處於下列其中一種狀態：

- `NORMAL`
- `WARNING`

- `PARTIAL_FAILURES`
- `ERROR`

輪詢工作物件以偵測工作的終端機狀態時、您可以使用兩種技巧：成功或失敗：

- 標準輪詢要求：立即傳回目前的工作狀態。
- 長時間輪詢要求：當工作狀態移至時 `NORMAL`、 `ERROR`或 `PARTIAL_FAILURES`。

非同步要求的步驟

您可以使用下列高階程序來完成非同步 API 呼叫：

1. 發出非同步 API 呼叫。
2. 接收 HTTP 回應 202 、表示已成功接受要求。
3. 從回應本文擷取工作物件的識別碼。
4. 在迴圈內、等待工作物件到達終端機狀態 `NORMAL`、 `ERROR`或 `PARTIAL_FAILURES`。
5. 確認工作的終端狀態、並擷取工作結果。

您好API伺服器

_Hello API server_是示範如何Active IQ Unified Manager 使用簡單的REST用戶端、在靜態中叫用REST API的範例程式。範例程式會以Json格式提供API伺服器的基本詳細資料（伺服器僅支援 `application/json` 格式）。

使用的URI為： https://<hostname>/api/datacenter/svm/svms。此範例程式碼採用下列輸入參數：

- API伺服器IP位址或FQDN
- 選用：連接埠號碼（預設：443）
- 使用者名稱
- 密碼
- 回應格式 (`application/json`)

若要叫用REST API、您也可以使用其他指令碼、例如JERSEY和REST-Easy來撰寫Java REST用戶端Active IQ Unified Manager 以供使用。您應該瞭解下列有關範例程式碼的考量事項：

- 使用HTTPS連線Active IQ Unified Manager 來叫用指定的REST URI
- 忽略Active IQ Unified Manager 由供應的憑證
- 在交握期間跳過主機名稱驗證
- 用途`javax.net.ssl.HttpsURLConnection` 用於URI連線
- 使用協力廠商程式庫 (`org.apache.commons.codec.binary.Base64`）用於建構HTTP基本驗證中使用的Base64編碼字串

若要編譯及執行範例程式碼、您必須使用Java編譯器1.8或更新版本。

```
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;
    }

}
```