對CIMC執行手動XML API呼叫

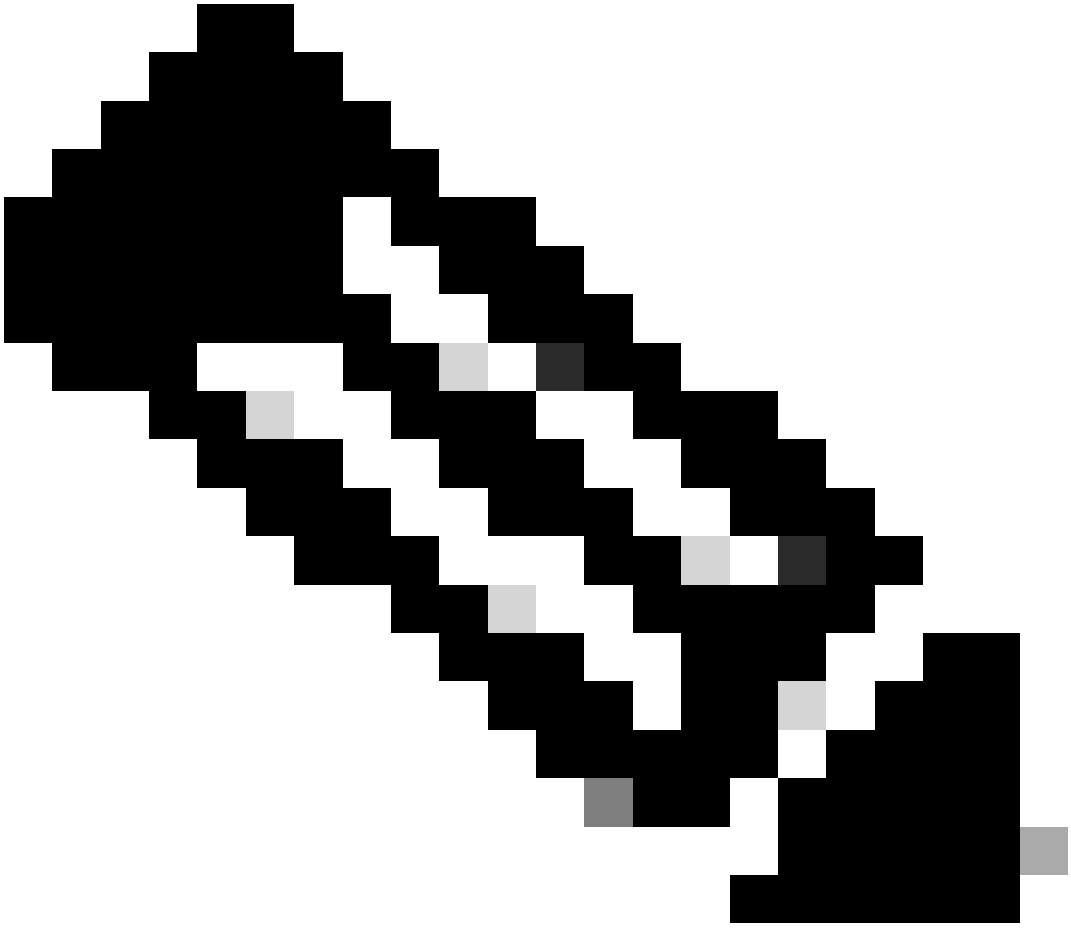
目錄

簡介

本檔案介紹如何對思科整合式管理控制器(CIMC)執行手動XML API呼叫。

採用元件

- Linux電腦（任何發行版）。
- Linux電腦與Cisco CIMC之間的網路連線（成功的ping測試足以滿足需要）。



附註：本指南中使用的檔名(main.sh、login.xml、get_summary.xml)是任意的。您可以自由使用自己的命名約定，只要在整個過程中正確引用它們。

本文中的資訊是根據特定實驗室環境內的裝置所建立。文中使用到的所有裝置皆從已清除（預設）的組態來啟動。如果您的網路運作中，請確保您瞭解任何指令可能造成的影響。

背景資訊

在本演示中，使用名為「main.sh」的Bash指令碼對伺服器執行API呼叫。

請注意，替代方法(例如CURL（直接從CLI）或Python)也可以實現相同的結果。

對CIMC執行XML API呼叫

CIMC XML API要求使用「aaaLogin」API呼叫的初始驗證步驟。在此過程中，將檢索會話cookie，然後使用它來驗證所有後續API呼叫。

步驟 1

首先在遠端Linux電腦中建立名為main.sh的檔案。此檔案包含用於傳送API呼叫的bash指令碼。

main.sh指令碼在本練習中多次執行：在隨後的呼叫中，先用於身份驗證，然後用於從CIMC檢索資訊。

以下是main.sh檔案的內容：

```
#!/bin/bash
# CONFIGURATION - Edit these variables
CIMC_IP="X.X.X.X" # <<<<<<<< Customer CIMC IP address
USERNAME="admin"
PASSWORD="xxxxxxxx" # <<<<<<< CIMC Password in plaintext
XML_PAYLOAD_FILE="login.xml" # <<<<<<< Referencing the login.xml file for authentication and cookie ret
CIMC_URL="https://${CIMC_IP}/nuova" # <<<<<<< Call is made to this URL

# Check if payload file exists
if [ ! -f "$XML_PAYLOAD_FILE" ]; then
echo "Error: XML payload file '$XML_PAYLOAD_FILE' not found."
exit 1
fi
# Send XML request using curl
curl -k -s -u "${USERNAME}:${PASSWORD}" -H "Content-Type: text/xml" -d @"${XML_PAYLOAD_FILE}" "${CIMC_U
```

bash指令碼定義CIMC登入引數，還執行XML API呼叫，該呼叫在另一個名為login.xml的檔案中指定，該檔案由名為XML_PAYLOAD_FILE的變數標識。

指令碼還檢查由變數XML_PAYLOAD_FILE定義的login.xml檔案是否存在且是常規檔案。

如果由XML_PAYLOAD_FILE定義的檔案不存在，指令碼將列印錯誤並退出。

在CLI中運行以下命令，儲存該檔案並使其可執行：

```
$ sudo chmod +x main.sh
```

步驟 2

接下來，在與main.sh檔案相同的linux目錄中建立calledlogin.xml檔案。

此登入檔案包含發送到CIMC以檢索會話cookie的actualaaaLoginXML API呼叫。檢索到的cookie用於後續的API呼叫：

請記得用適當的憑證替換CIMC使用者名稱和密碼。

在CLI中執行main.sh指令碼以檢索cookie:

```
$ sudo ./main.sh
```

如果API呼叫成功，則返回的XML響應包含名為LedoutCookiei的鍵，其值為檢索的cookie，如下所示：

```
<?xml version="1.0"?>
<aaaLogin cookie="" response="yes" outCookie="xxxxxxxxx/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" outRefreshPeriod="600" outPriv="admin" outSessionId="18"
outVersion="4.3(5.250001)"> </aaaLogin>
```

從輸出中，找到outCookie值。

儲存此cookie值。

步驟 3

在main.sh檔案所在的同一目錄中建立一個新檔案。將該檔案命名為get_summary.xml。

有關可用於思科IMC的API請求的綜合清單，請參閱思科UCS機架式伺服器CIMC XML API程式設計師指南。

新的get_summary.xml檔案用於檢索Server Summary Information and Host Power State。使用參考文檔中的XML代碼塊，但將cookie key值替換為較早的檢索cookie。

將<cookie_value>替換為從較早的login.xmlresponse獲取的theoutCookievalue。更新的請求如下所示：

```
<configResolveClass cookie="xxxxxxxxx/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
inHierarchical="false" classId="computeRackUnit"/>
```

請務必將<cookie_value>替換為驗證過程中檢索到的實際cookie值。

步驟 4

將cookie新增到新的get_summary.xml檔案後，更新main.sh指令碼，以便將"get_summary.xml"作為此請求的XML_PAYLOAD_FILE變數的值。

```
#!/bin/bash
# CONFIGURATION - Edit these variables
CIMC_IP="X.X.X.X"
USERNAME="admin"
PASSWORD="xxxxxxxx"
XML_PAYLOAD_FILE="get_summary.xml" # <<<<<<< Referencing the get_summary.xml file for subsequent API ca
CIMC_URL="https://${CIMC_IP}/nuova"
# Check if payload file exists
if [ ! -f "$XML_PAYLOAD_FILE" ]; then
echo "Error: XML payload file '$XML_PAYLOAD_FILE' not found."
exit 1
fi
# Send XML request using curl
curl -k -s -u "${USERNAME}:${PASSWORD}" -H "Content-Type: text/xml" -d @"${XML_PAYLOAD_FILE}" "${CIMC_U
```

再次運行main.sh指令碼以執行更新的API請求。

```
$ sudo ./main.sh
```

然後，您會收到一個XML格式的API響應，該響應從CIMC返回請求的對象。

```
<?xml version="1.0"?>
<configResolveClass cookie="xxxxxxxxx/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" response="yes" classId="computeRackUnit">
<outConfigs>
<computeRackUnit dn="sys/rack-unit-1" adminPower="policy" availableMemory="524288" model="UCSC-C220-M6S" memorySpeed="3200" name="UCS C220
M6S" numOfAdaptors="2" numOfCores="56" numOfCoresEnabled="56" numOfCpus="2" numOfEthHostIfs="0" numOfFcHostIfs="0" numOfThreads="112" operPower
="off" originalUuid="XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" presence="equipped" serverId="1"
serial="XXXXXXXXXX" totalMemory="524288" usrLbl="" uuid="XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" vendor="Cisco Systems
Inc" cimcResetReason="graceful-reboot
" assetTag="Unknown" adaptorSecureUpdate="Enabled" resetComponents="components" storageResetStatus="NA" vicResetStatus="NA" bmcResetStatus="NA" sma
rtUsbAccess="disabled" smartUsbStatus="Disabled" biosPostState="pending"/>
</outConfigs>
</configResolveClass>
```

疑難排解

必須注意的是，aaaLogin XML API呼叫返回會話cookie，outRefreshPeriod約為600秒。這意味著，如果未刷新，cookie將在十(10)分鐘後過期，並且需要新cookie才能繼續執行API請求。

如果您嘗試使用過期cookie（600秒後），將返回552「Authorization required」 XML錯誤響應塊：

<?xml version="1.0"?>
<configResolveDn cookie="xxxxxxxxx/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" response="yes" errorCode="552" invocationResult="service-unavailable" errorDescr="Authorization required"> </configResolveDn>

關於此翻譯

思科已使用電腦和人工技術翻譯本文件，讓全世界的使用者能夠以自己的語言理解支援內容。請注意，即使是最佳機器翻譯，也不如專業譯者翻譯的內容準確。Cisco Systems, Inc. 對這些翻譯的準確度概不負責，並建議一律查看原始英文文件（提供連結）。