

# 对CIMC执行手动XML API调用

## 目录

---

[简介](#)

[使用的组件](#)

[背景信息](#)

[对CIMC执行XML API调用](#)

[第 1 步](#)

[第 2 步](#)

[第 3 步](#)

[第 4 步](#)

[故障排除](#)

---

## 简介

本文档介绍如何对思科集成管理控制器(CIMC)执行手动XML API调用。

## 使用的组件

- Linux计算机 ( 任何分发版 ) 。
- Linux计算机与Cisco CIMC之间的网络连接 ( 成功的ping测试足以满足需要 ) 。



注意：本指南中使用的文件名(main.sh、login.xml、get\_summary.xml)是任意的。您可以使用自己的命名约定，只要在整个过程中正确引用它们。

---

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解所有命令的潜在影响。

## 背景信息

在本演示中，使用名为“main.sh”的Bash脚本对服务器执行API调用。

请注意，替代方法(例如CURL（直接从CLI）或Python)也可以实现相同的结果。

## 对CIMC执行XML API调用

CIMC XML API需要使用“aaaLogin”API调用的初始身份验证步骤。在此过程中，将检索会话cookie，然后使用该会话对所有后续API调用进行身份验证。

## 第 1 步

首先在远程Linux计算机上创建一个名为main.sh的文件。此文件包含用于发送API调用的bash脚本。

在本练习中，将多次执行main.sh脚本：首先用于身份验证，然后在后续呼叫中从CIMC检索信息。

以下是main.sh文件的内容：

---

```
#!/bin/bash
# CONFIGURATION - Edit these variables
CIMC_IP="X.X.X.X" # <<<<<<<< Customer CIMC IP address
USERNAME="admin"
PASSWORD="xxxxxxx" # <<<<<<< CIMC Password in plaintext
XML_PAYLOAD_FILE="login.xml" # <<<<<<< Referencing the login.xml file for authentication and cookie retrieval
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_URL}"
```

bash脚本定义CIMC登录参数，还执行XML API调用，该调用在另一个名为login.xml的文件中指定，该文件由名为XML\_PAYLOAD\_FILE的变量标识。

脚本还检查文件login.xml(由XML\_PAYLOAD\_FILE变量定义)是否存在并且是常规文件。

如果由XML\_PAYLOAD\_FILE定义的文件不存在，脚本将打印错误并退出。

在CLI中运行以下命令，保存文件并使其可执行：

---

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

## 第 2 步

然后，在与main.sh文件相同的linux目录中创建calledlogin.xml文件。

此登录文件包含发送至CIMC以检索会话Cookie的actualaaaLoginXML API调用。检索到的Cookie用于后续API调用：

---

请记住用适当的凭证替换CIMC用户名和密码。

在CLI中执行main.sh脚本以检索cookie:

---

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

如果API调用成功，则返回的XML响应包含名为doutCookieve的键，其值为检索到的cookie，如下所示：

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

从输出中，找到outCookie值。

保存此cookie值。

### 第 3 步

在main.sh文件所在的同一目录中创建一个新文件。将该文件命名为get\_summary.xml。

有关可与思科IMC配合使用的API请求的完整列表，请参阅[思科UCS机架式服务器CIMC XML API程序员指南](#)。

新的get\_summary.xml文件用于检索Server Summary Information and Host Power State。使用参考文档中的XML代码块，但用较早的检索Cookie替换cookie密钥值。

---

将<cookie\_value>替换为从较早的login.xmlresponse获取的outCookievalue。更新的请求如下所示：

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

请确保用身份验证过程中检索到的实际Cookie值替换<cookie\_value>。

## 第 4 步

将cookie添加到新的get\_summary.xml文件后，更新main.sh脚本，以将get\_summary.xml引用为此请求的XML\_PAYLOAD\_FILE变量的值。

---

```
#!/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
```

再次运行main.sh脚本以执行更新的API请求。

---

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

然后，您会收到一个XML格式的API响应，从CIMC返回请求的对象。

```
<?xml version="1.0"?>
<configResolveClass cookie="xxxxxxxx/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="XXXXXXXX" 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>
```

## 故障排除

请注意，aaaLogin XML API调用返回一个会话cookie，outRefreshPeriod约为600秒。这意味着，如果未刷新，Cookie将在十(10)分钟后过期，并且需要新的Cookie才能继续执行API请求。

如果您尝试使用过期Cookie（600秒后），将返回552“Authorization required” XML错误响应块：

<?xml version="1.0"?>

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

## 关于此翻译

思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。