# Assignment 0x02 – OSINT, Recon & Network Scanning

Menno Brandt, a1849852

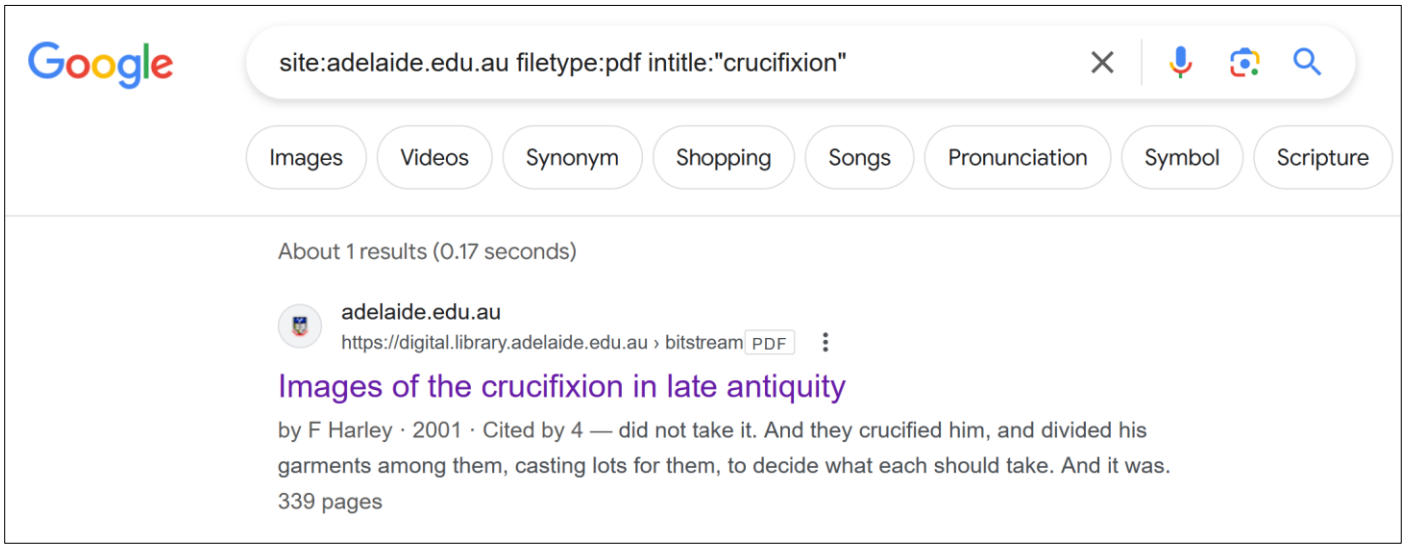
---

Question 1)

The Google search that I used, was site:adelaide.edu.au filetype:pdf intitle:”crucifixion”

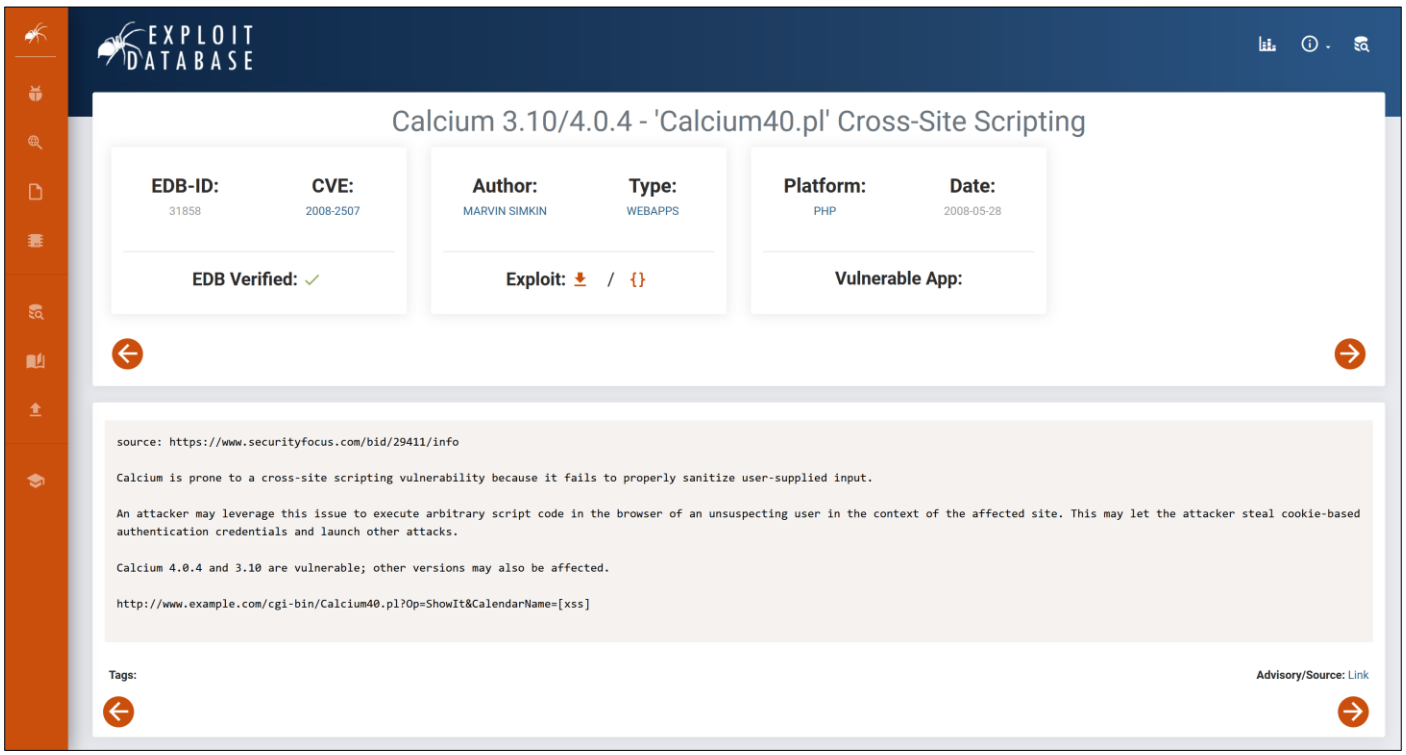
The first part narrows down the search results, to only the University’s domain. Then, the second and third part limit the results to pdf files, that have the word “cruxifixion” in their title.

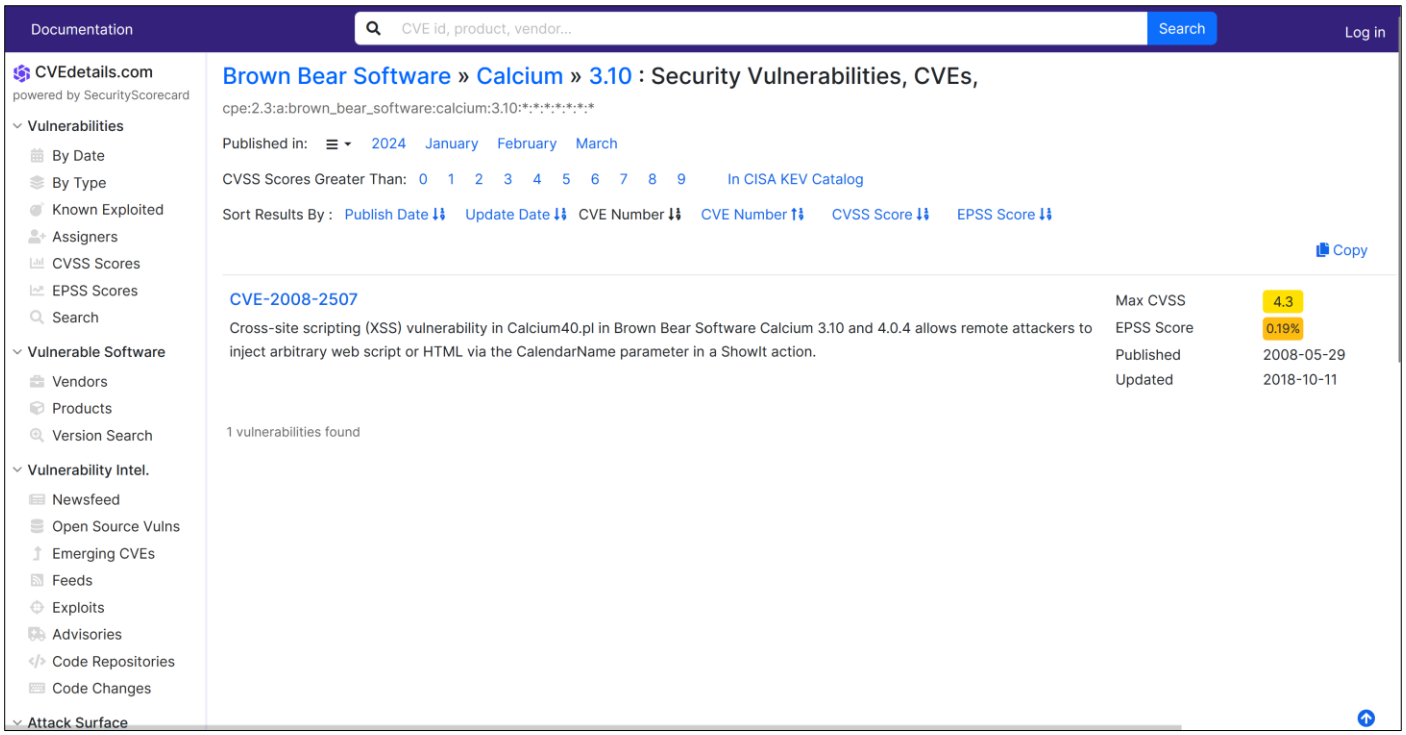
The author of his document is Felicity Harley (F Harley).



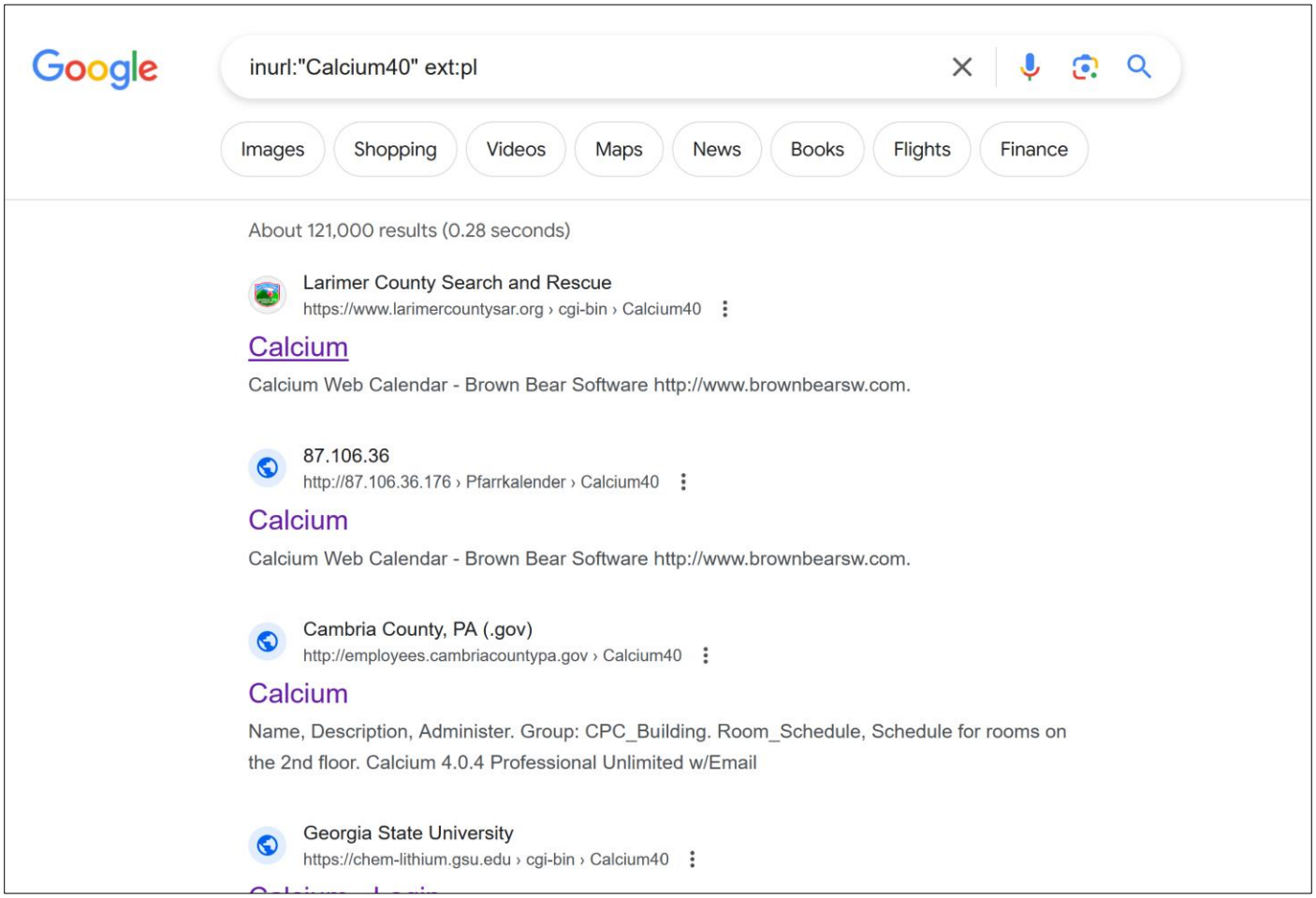

Question 2)

Below are two screenshot of details I found about Calcium’s XSS vulnerability.






To find websites that run Calcium, specifically the version that's exploitable, I entered the following: inurl:"Calcium40" ext:pl. This looks for Calcium40 in the title, and .pl (Pearl) as the extension.




Another way to search for these websites, is to use inurl:"Calcium40.pl". This produces slightly different results. Likely due to search engine optimisation.

Question 3):

The process I used to find the answer, was to create a new workspace (prac2), install the who_is_pocs module, set the source to x.com, and then to let it execute. I then used the command "> show contacts" to put it in a form that's quickly readable.

```
[recon-ng][prac2] > modules load recon/domains-contacts/whois_pocs
[recon-ng][prac2][whois_pocs] > options set SOURCE x.com
SOURCE ⇒ x.com
[recon-ng][prac2][whois_pocs] > run
```

```
[*] 5 total (4 new) contacts found.
[recon-ng][prac2][whois_pocs] > show contacts

+-------+------------+-------------+-----------+-----------+---------------+-----------------+
| rowid | first_name | middle_name | last_name |   email   |     title     |      region     |
+-------+------------+-------------+-----------+-----------+---------------+-----------------+
| 1     | William    |             | Fenech    | wfenech@x.com | Whois contact | San Francisco, CA |
| 2     | David      |             | Weinstein | dw@x.com      | Whois contact | Scotts Valley, CA |
| 3     | Robert     |             | Nordland  | x@x.com       | Whois contact | Carson, CA        |
| 4     | Evgeny     |             | Smirnov   | x@x.com       | Whois contact | London            |
+-------+------------+-------------+-----------+---------------+---------------+-----------------+
```

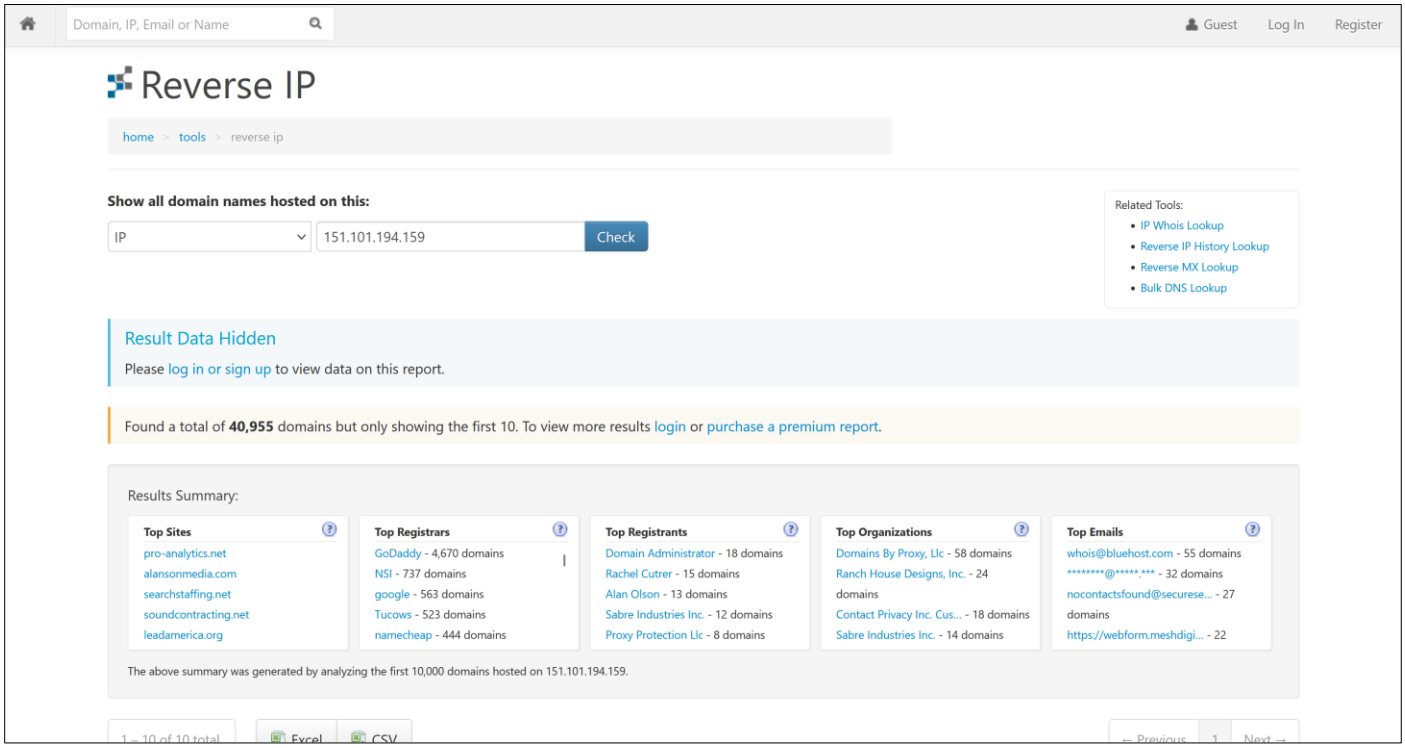
As shown by the screenshot, the person who lives in Carson CA, is Robert Nordland.

Question 4):

| **Question:** | **Answer:** |
|---|---|
| dunstan.org.au resolves to: | 151.101.194.159 |
| Other domain names that resolve to the same address | (small sample of 10).<br>1 Domain<br>2 jeffkaiser.com<br>3 rileypainting.com<br>4 ncfsudbury.com<br>5 ocvacationbibleschool.com<br>6 dementiafriendsiowa.org<br>7 buffalopal.com<br>8 livesportinggoods.com<br>9 6660kennedy.com<br>10 claritywellnesscoaching.com |
| Owner of the IP address | Fastly, Inc |
| The IP address range which the IP address belongs | 151.101.0.0 - 151.101.255.255 |
| The Autonomous System Number (ASN) that contain the IP address | AS54113 |
| Other netblocks registered under the same ASN | Here's a sample, of 10:<br>NETBLOCK / COMPANY<br>103.245.222.0/24 Fastly, Inc<br>103.245.224.0/24 Fastly, Inc<br>104.156.80.0/24 Fastly, Inc.<br>104.156.81.0/24 Fastly, Inc.<br>104.156.82.0/24 Fastly, Inc.<br>104.156.83.0/24 Fastly, Inc.<br>104.156.84.0/24 Fastly, Inc.<br>104.156.85.0/24 Fastly, Inc.<br>104.156.87.0/24 Fastly, Inc.<br>104.156.89.0/24 Fastly, Inc. |

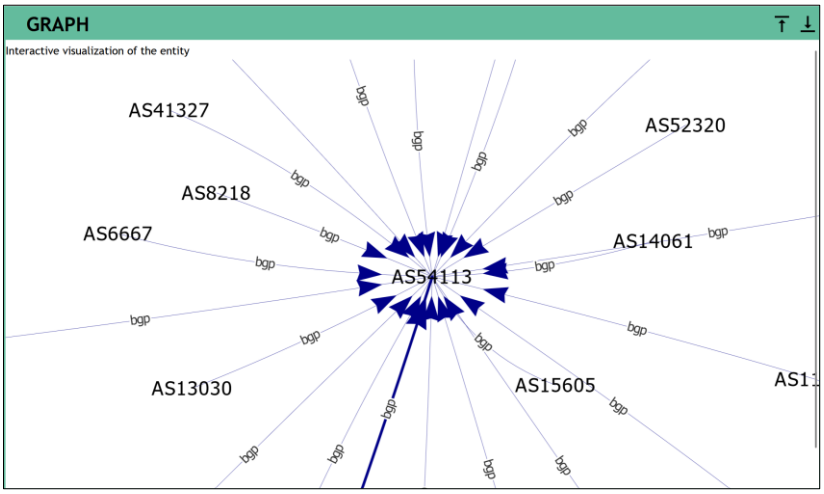
I'll describe the process I used, to fill in this table. To find the IP address of dunstan.org.au, I simply used this command in Kali Linux. "dig" was introduced in the workshop, and I learned about adding +short to just show only the IP in the output.


To find other domains that resolve to this same IP address, I used the IP to Domain lookup tool: https://www.domainiq.com/reverse_ip . This was a tool that I also found in the resources link posted in the workshop (https://www.osinttechniques.com/osint-tools.html). I found that there were thousands of answers, so I only gave a sample of 10 domains. Here is a screenshot of the tool's output, and some of the ~41,000 domains it found.

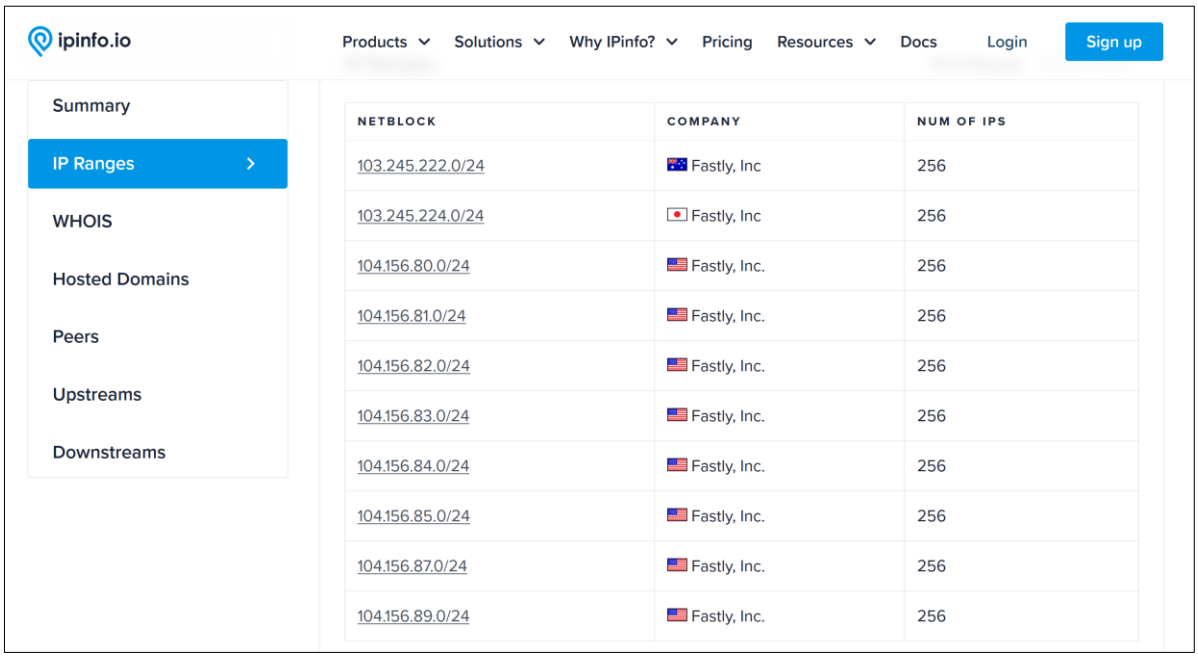



To find the owner of the IP address, I used Kali's WhoIs CLI tool. A screenshot is shown below, and the red arrow points to the organisation's name (Fastly, Inc). I also used the results of this command, to answer the next question in the table. The green arrow points to the IP address range which the IP belongs to.

To find the ASN (Autonomous System Number), I used https://www.robtex.com/ . This was a tool introduced on the workshop page. I just pasted the IP address, searched, and it showed me the following graph, with the ASN in the centre. As can be seen, the ASN is AS54113.

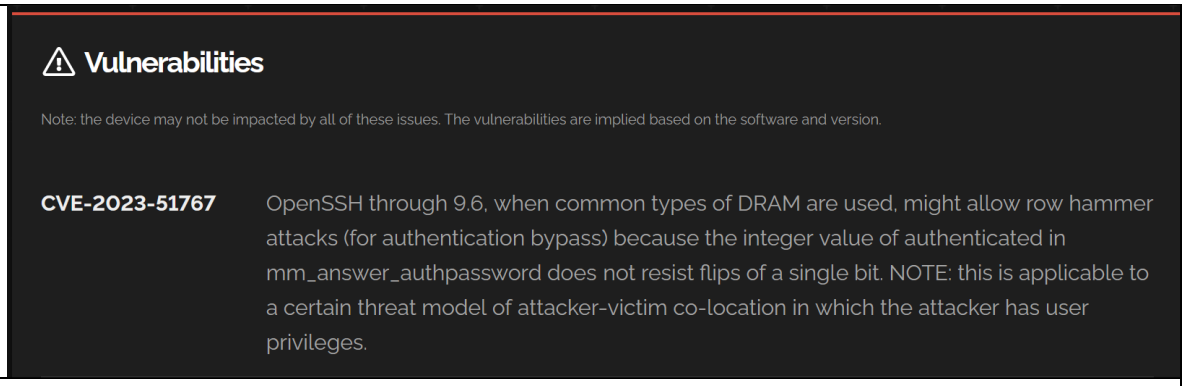



For the final part of the table, I used this ASN in the online tool “ipinfo” and did the following search: https://ipinfo.io/AS54113 . A screenshot of the list of (some) of the ASN’s associated netblocks/IP are shown below.

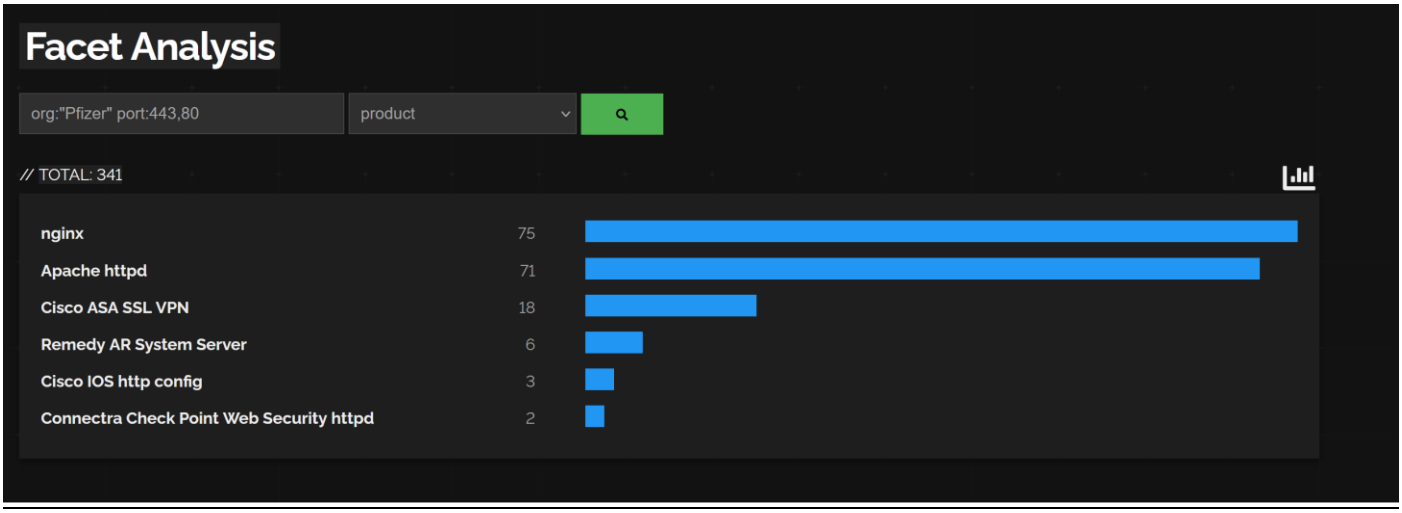



Question 5):

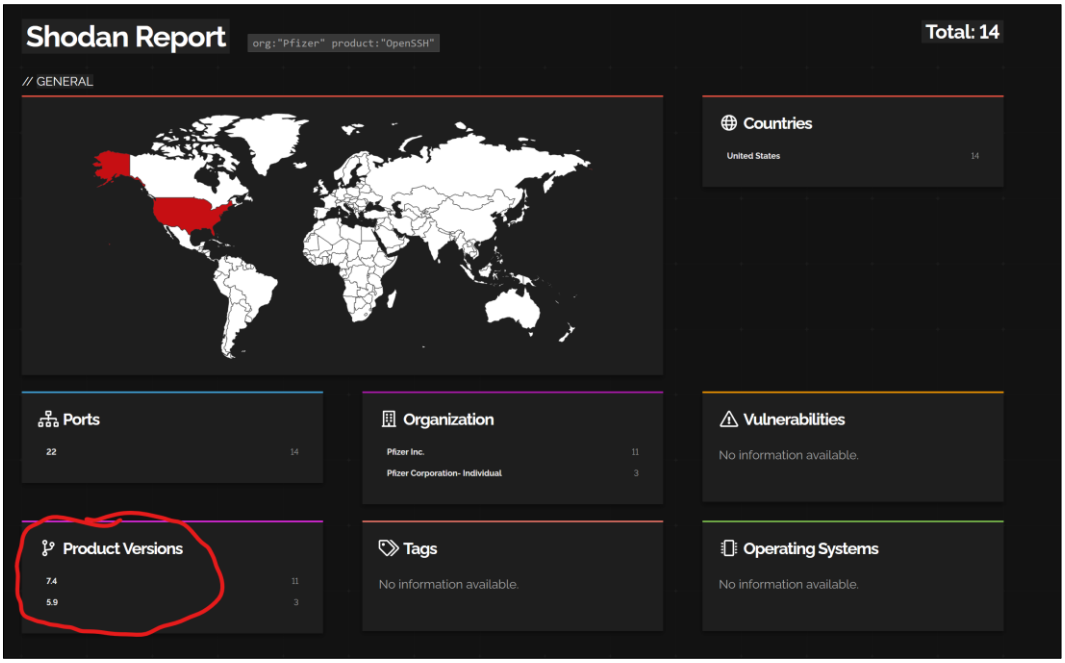
| **Question:** | **Answer:** |
|---|---|
| What web server(s) are used by this company? | nginx, Apache httpd, Cisco ASA SSL VPN, Remedy AR System Server, Cisco IOS https config, and Connectra Check Point Web Security httpd. |
| What versions of OpenSSH are used by this company? | 7.4 and 5.9 |

| | |
|---|---|
| According to Shodan, what are of the vulnerabilities in one of the versions of the OpenSSH servers? |  |
| Choose the most recent vulnerability from above and find the CVSS2.0 string for it by looking it up on nvd.nist.gov | I couldn't find the CVSS2.0 string, here is the CVSS3.1 string:<br><br>CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H |

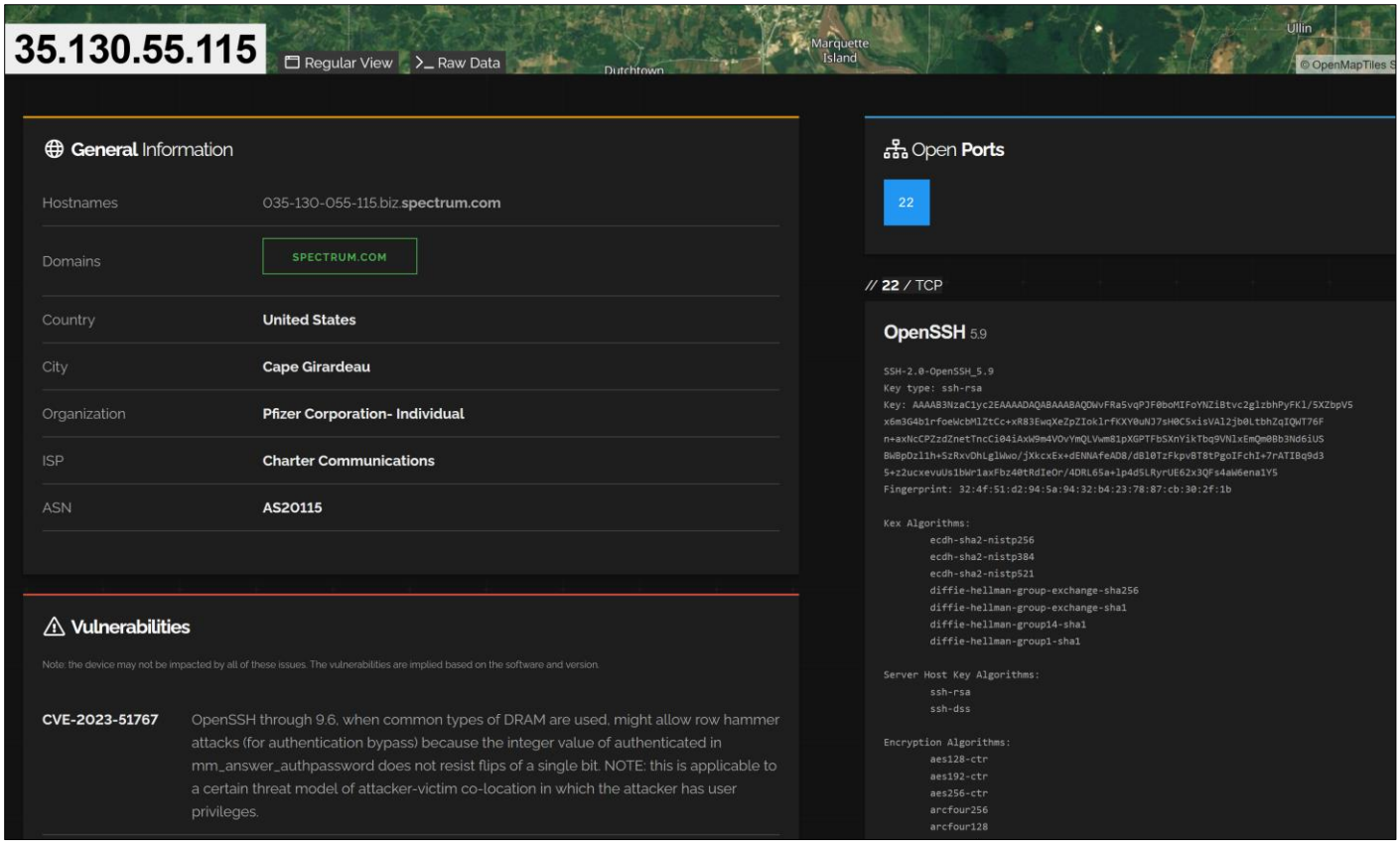
To find the web server(s) used by Pfizer, I used the modifier org:"Pfizer", port:443,80. These two ports (443 and 80) are the default for HTTPS and HTTP. I then looked at the report, to see the products that run on these ports. Clearly, Pfizer uses nginx, Apache httpd, Cisco ASA SSL VPN, Remedy AR System Server, Cisco IOS https config, and Connectra Check Point Web Security httpd.



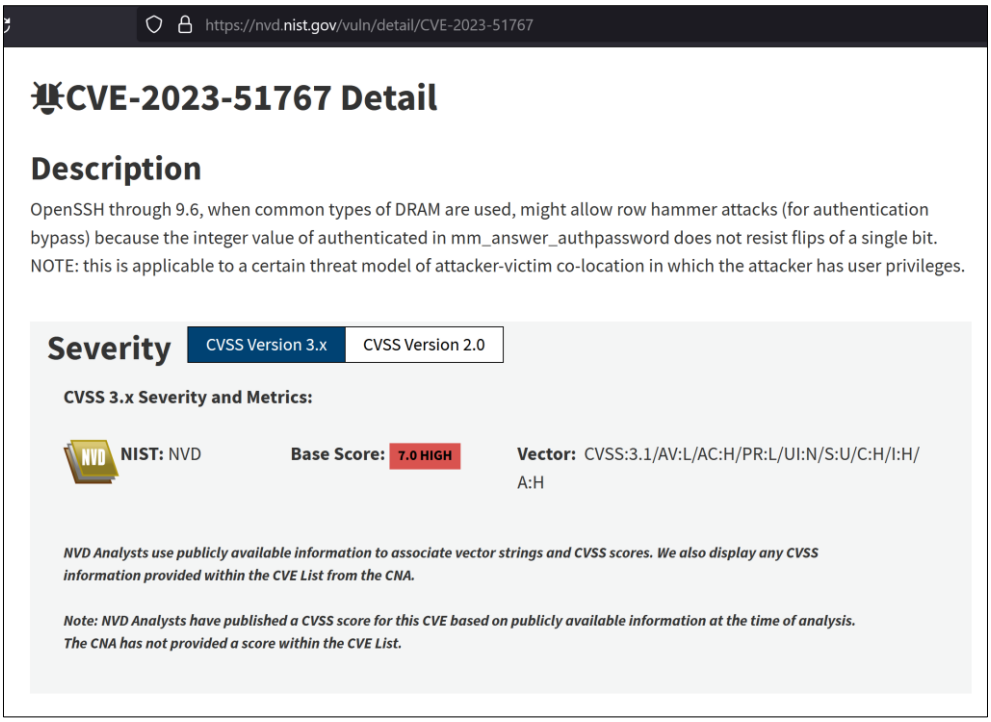

I used the search modifier: org:"Pfizer" product:"OpenSSH", and then clicked "View Report". This showed me some statistics, about Pfizer's use of OpenSSH. As shown by the screenshot, I found out that they use versions 7.4, and 5.9.




To find one of the vulnerabilities, I randomly selected the server 35.130.55.115 that runs OpenSSH version 5.9. As shown in the screenshot, Shodan automatically identified CVE-2023-51767 as a vulnerability of this version

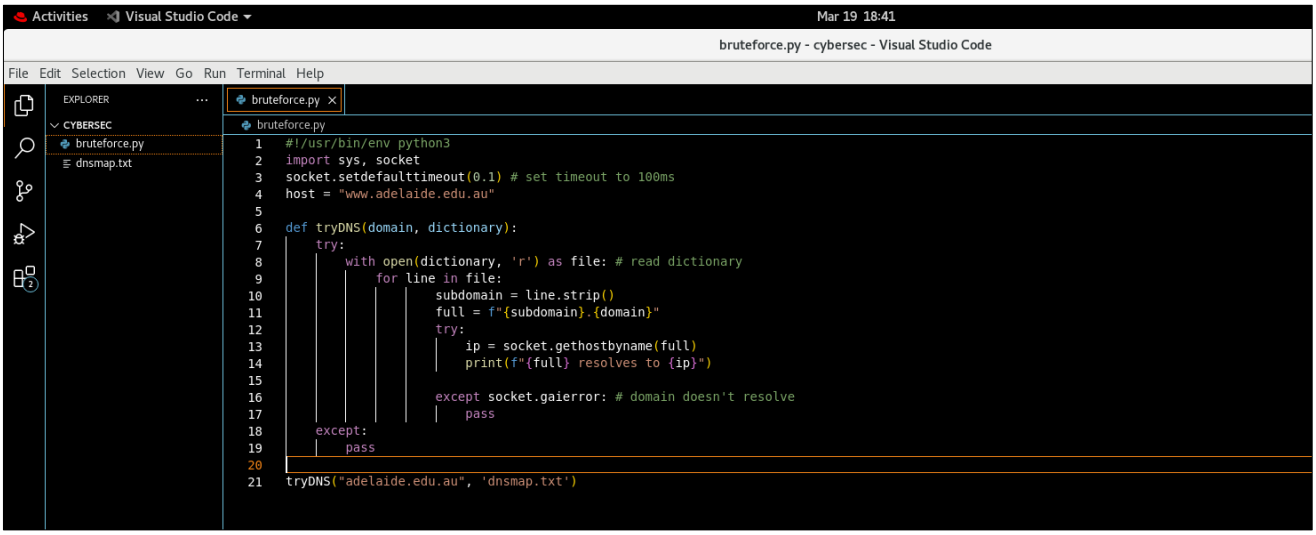



In order to answer the final row in the table, I searched for CVE-2023-51767 on nvd.nist.org.gov. Although it didn't have the CVSS:2.0 string, it had the updated CVSS:3.1 one. I made the assumption that "string" was referring to the "vector" in the details. Below is a screenshot of nvd.nist.gov's website.

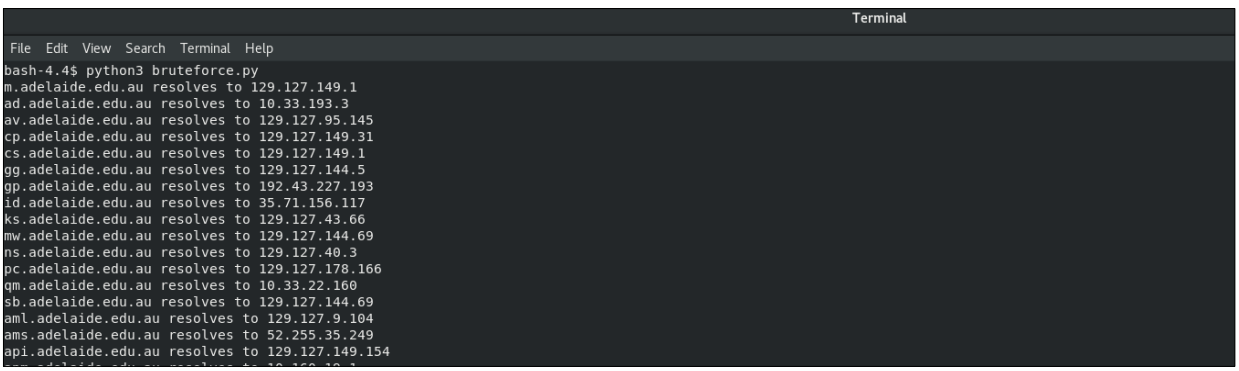



Question 6):

My process for this question, was to first put the dictionary file (dnsmap.txt) and a new python file into the same directory. The Python file's code was initially copied from the example code given to us on the assignment page. I then started the process of modifying it, to loop through and use the combinations stored in the dictionary file. I wrote the function "tryDNS" that reads the dictionary file, appends it to the start of the adelaide.edu.au domain, tries connecting, and then prints if it resolves. My code is shown below:




Below, I have the output of the code during its execution. These are only some preliminary results, as the dictionary file is very long.

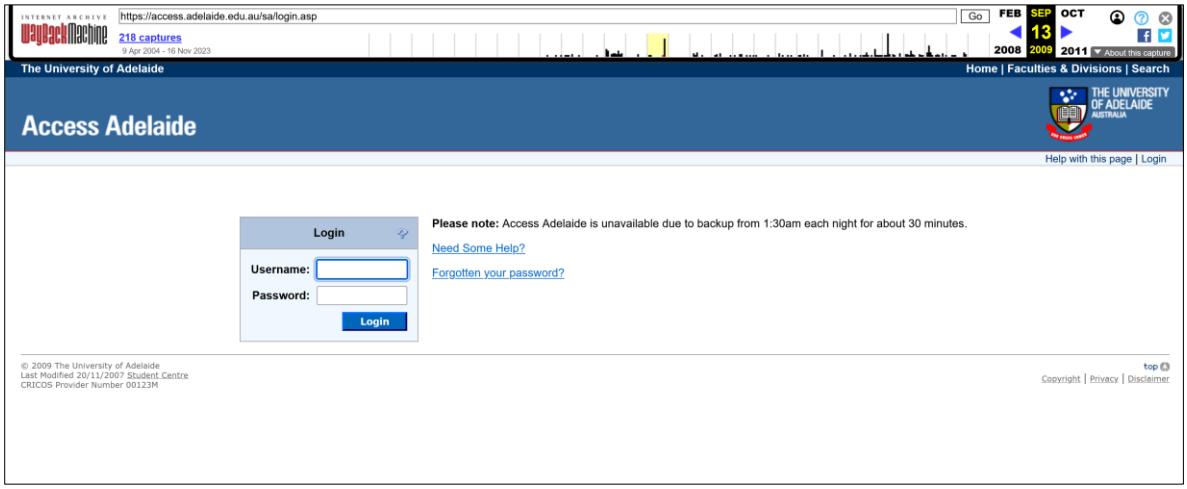


As we can see, domains such as m.adelaide.edu.au, ad.adelaide.edu.au, and av.adelaide.edu.au resolve to their respective IP addresses. Please note that the terminal and code editor look a bit different, because I used Linux on the University's lab computers instead of on my own Kali VM.

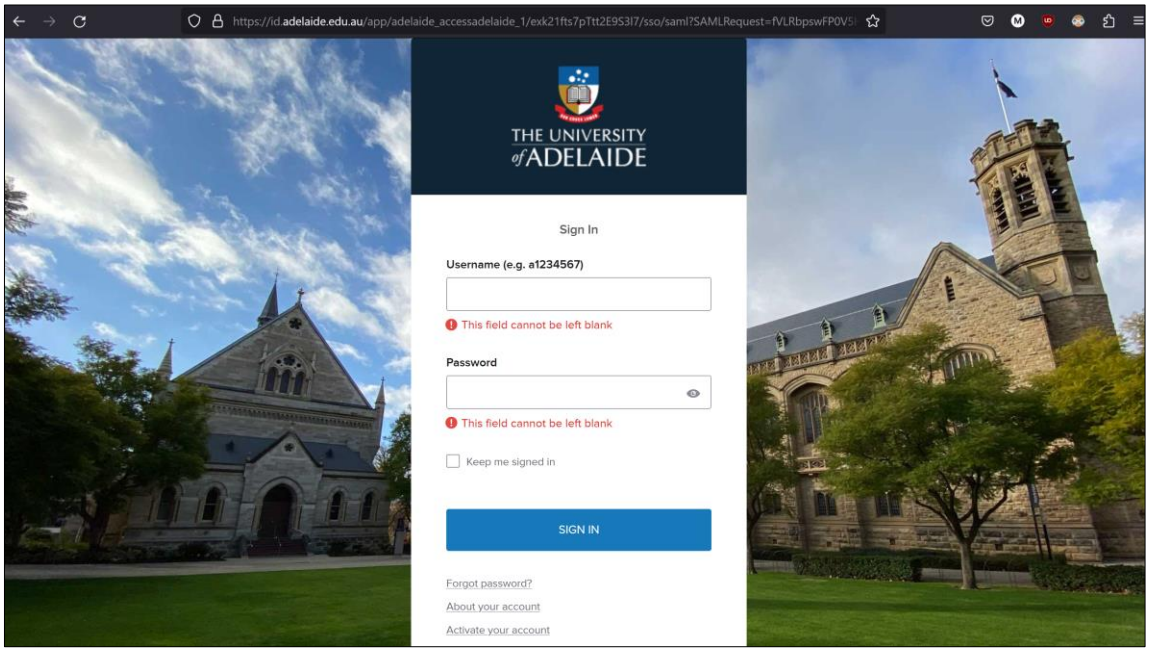
Question 7):

To find out what Access Adelaide used to look like, I went to https://archive.org/web/ , entered the URL https://access.adelaide.edu.au and selected "Browse History". I selected the snapshot that was taken on the 3rd of August 2009 at 14:34:38.

After getting an HTTP 304 error, it redirected and showed me the following login page:



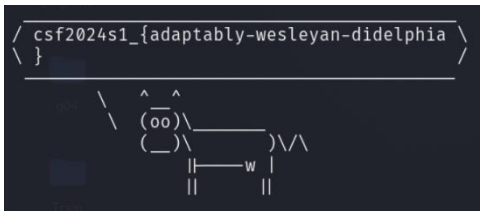

If we compare it the modern 2024 portal, it functions quite differently. Nowadays, it redirects to the generic id.adelaide.edu.au portal, instead of requiring users to sign directly into Access Adelaide. This login page had a big UI update, that can be seen with its background image and the new input fields' designs. It also now features some extra options, such as the "Keep me signed in" checkbox, and the "About your account" and "Activate your account" options. Its font appears to have remained very similar. The University icon has also been updated and modernised and is now featured in the centre.




Question 8):

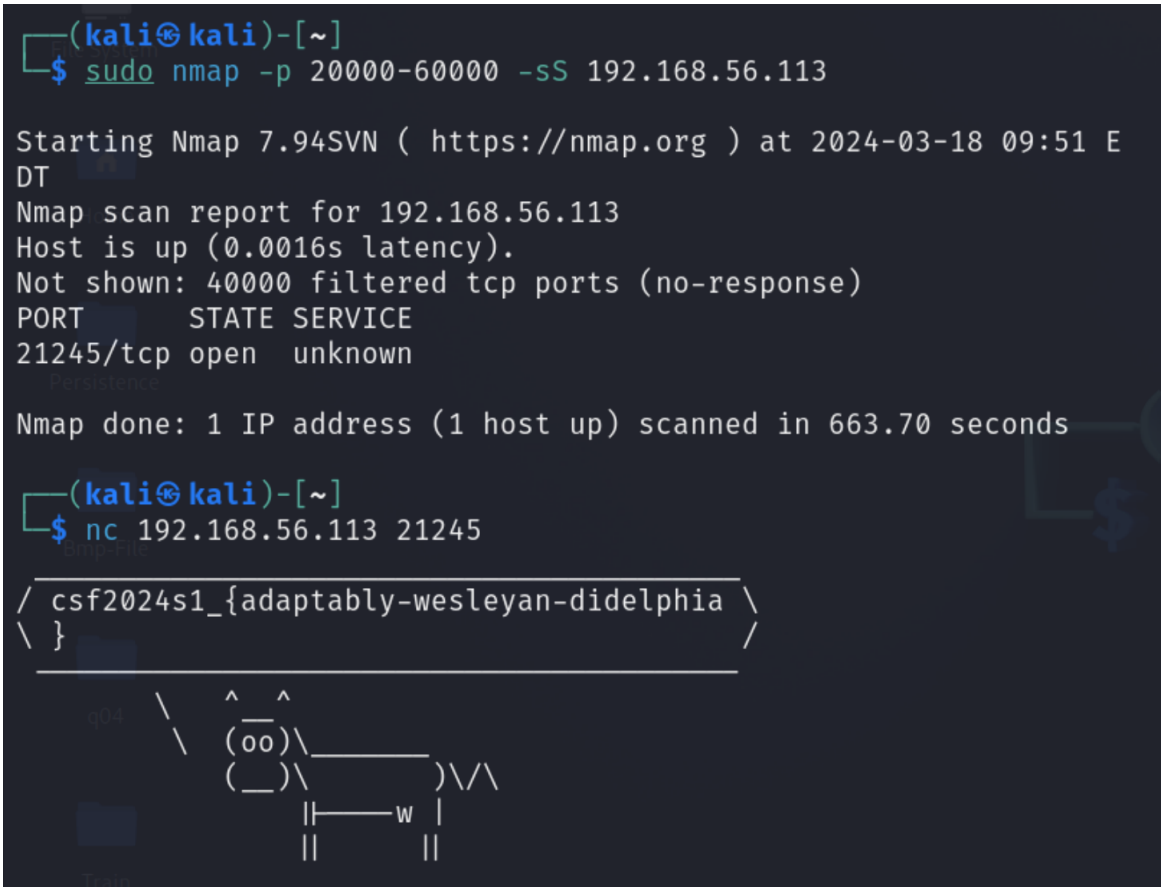
a) The port number is 21245.

b)




c) Explanation of how I identified and retrieved the answer:

To find the port, I used the nmap tool. As displayed in the screenshot below, I used the -p flag to set the tool to search in between ports 20000 and 60000. I also used -sS so that it does a SYN scan and identifies the open TCP port. After the program ran for about 11 minutes, and searched through those ports, nmap found that port 21245 was open. It was the only port to give a SYN/ACK reply. The others gave no response.

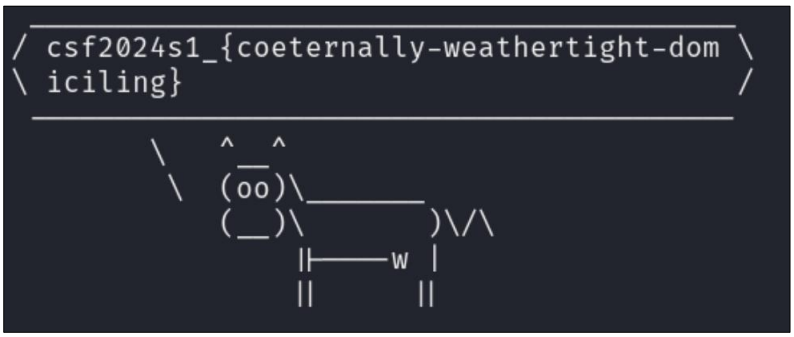



Now that I knew the specific open port, all I had to do was connect to it. I used netcat with the IP and the port as arguments, ran it, and it then displayed the secret. The secret is csf2024s1_{adaptably-wesleyan-didelphia}

Question 9):

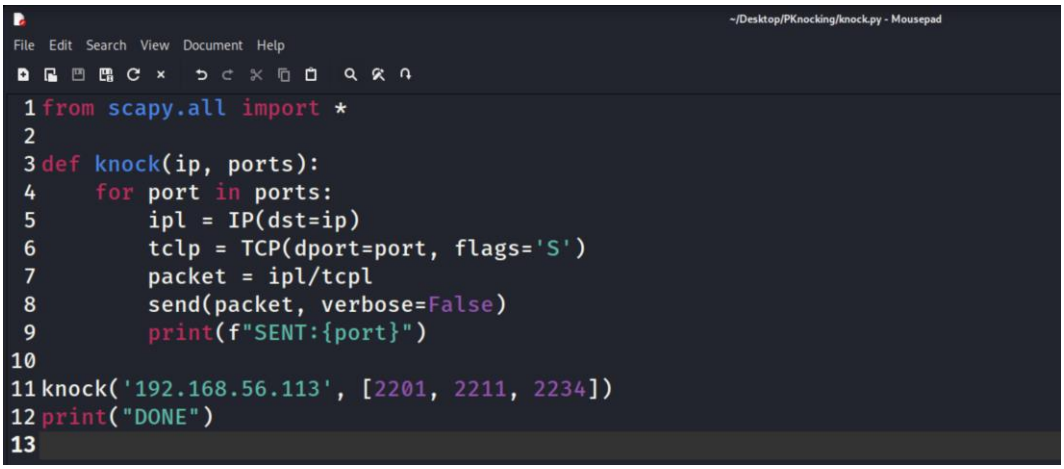
a) nc 192.168.56.113 12345

b)

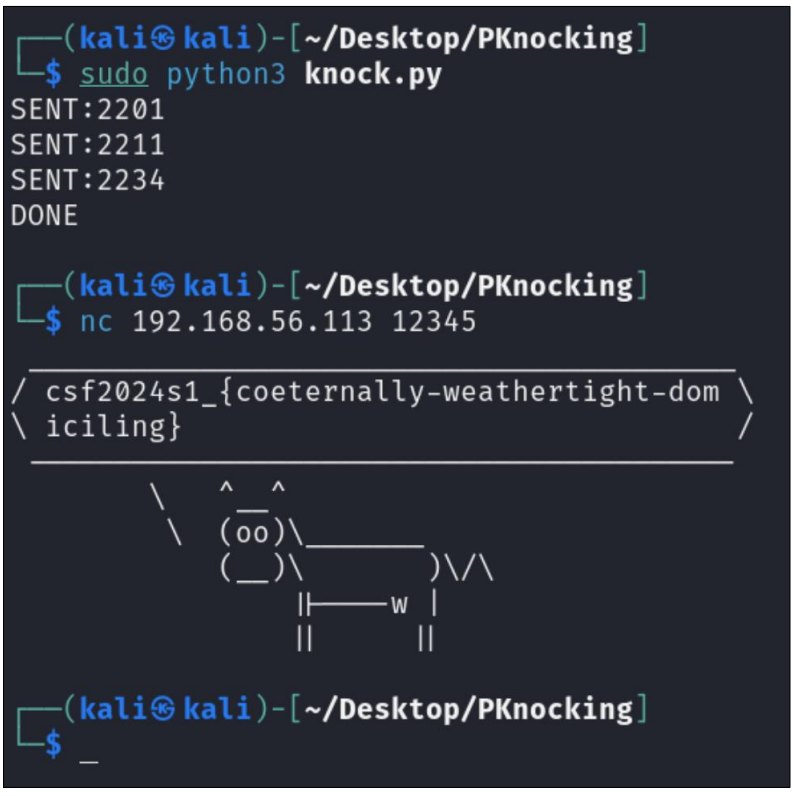


c)

I began this problem by finding a way to send SYN packets to specific ports, using Python. During this search, I found the “Scapy” library, which is used for packet manipulation. Using this library, I wrote the following program. The “knock” function takes in an array of ports (2201, 2211, 2234), and the IP address (192.168.56.116 on my HackLab VM). It then loops through every element in that array of ports and sends packets to them. Hence, “knocking” on these ports.




Although there was a 15 second timeout delay, I figured that if I simply ran the script and pasted the command “nc 192.168.56.113 12345” into my terminal as quickly as I could, it would still connect in time. As demonstrated by the following screenshot, it executed successfully and retrieved the secret. The secret is: csf2024s1_{coeternally-weathertight-domciling}.




These two steps (running the python script, and then running the netcat command), could also have been put into a Bash script. If the port had a smaller lockout time, I probably would have automated this to a greater extent. But, doing it manually still worked.