

Assignment 0x02 – OSINT, Recon & Network Scanning

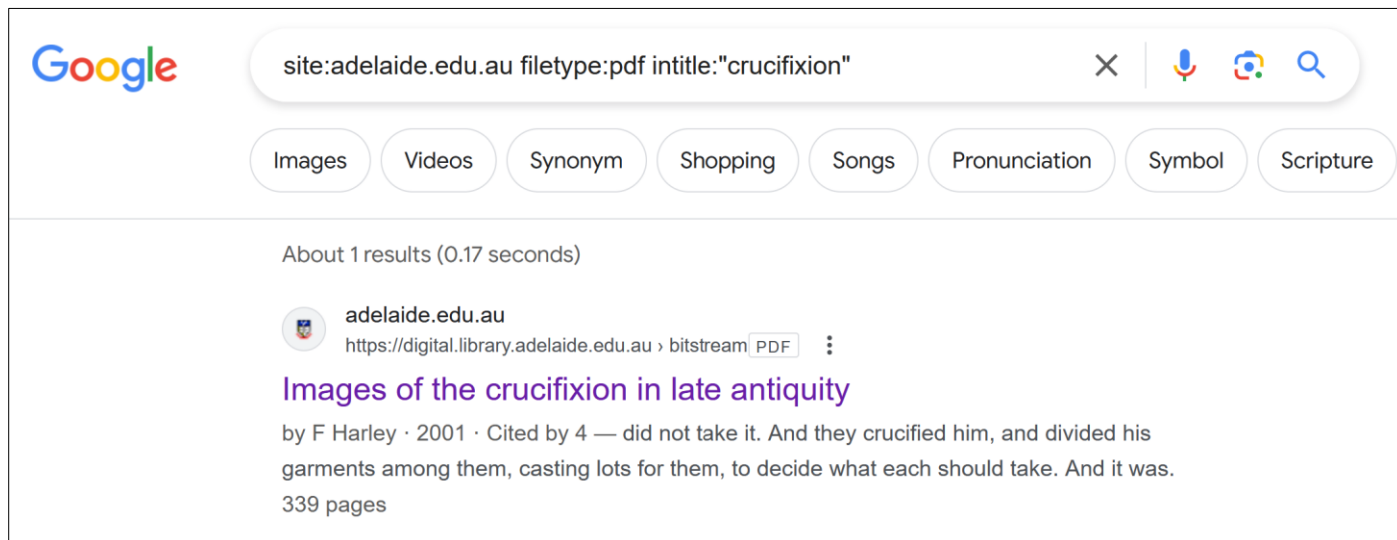
Menno Brandt, a1849852

Question 1)

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

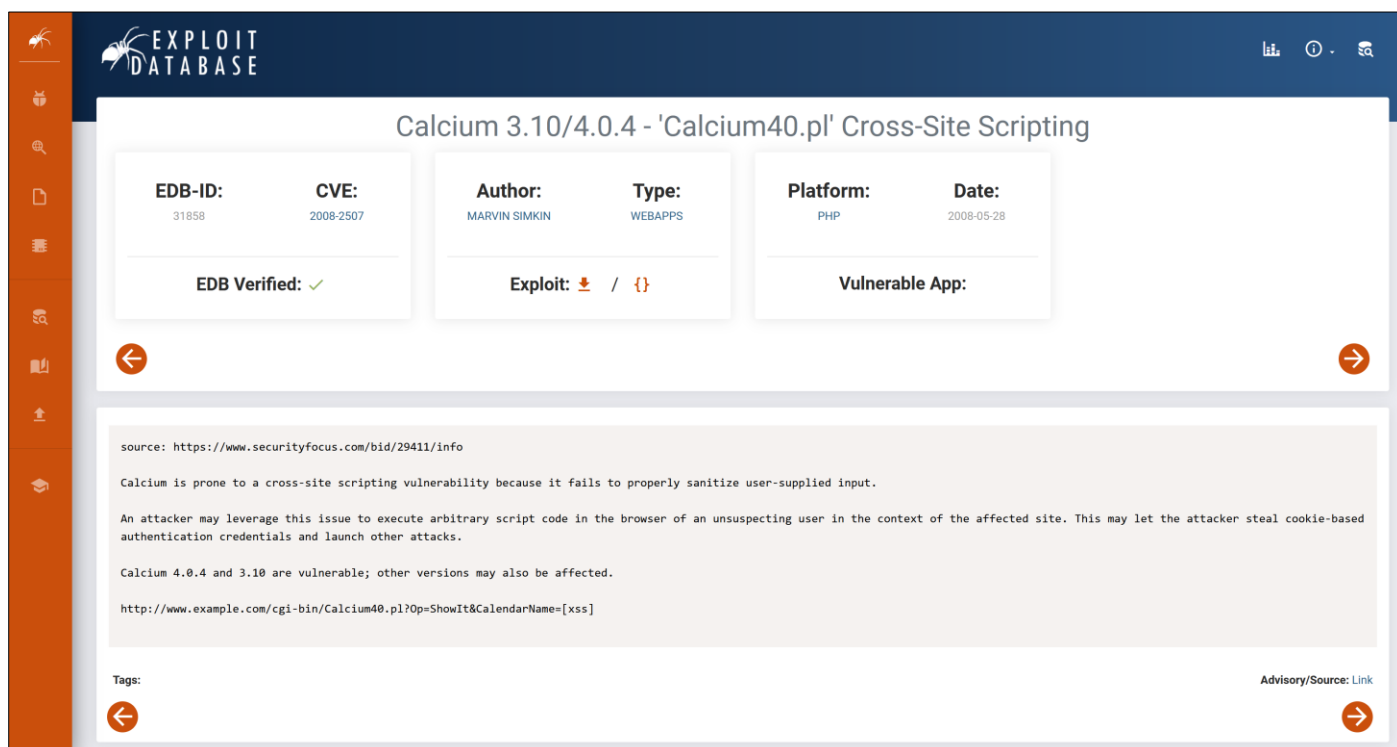
The first part narrows down the search results, to only the University's domain. Then, the second and third part limit the results to pdf files, that have the word "cruxifixion" in their title.

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.



Documentation [Log in](#)

CVEdetails.com
powered by SecurityScorecard

Brown Bear Software » Calcium » 3.10 : Security Vulnerabilities, CVEs,
cpe:2.3:a:brown_bear_software:calcium:3.10:*:*:*:*:*

Published in: [January](#) [February](#) [March](#)

CVSS Scores Greater Than: [0](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [In CISA KEV Catalog](#)

Sort Results By: [Publish Date](#) [Update Date](#) [CVE Number](#) [CVE Number](#) [CVSS Score](#) [EPSS Score](#)

[Copy](#)

CVE-2008-2507	Max CVSS	EPSS Score	Published	Updated
Cross-site scripting (XSS) vulnerability in Calcium40.pl in Brown Bear Software Calcium 3.10 and 4.0.4 allows remote attackers to inject arbitrary web script or HTML via the CalendarName parameter in a ShowIt action.	4.3	0.19%	2008-05-29	2018-10-11

1 vulnerabilities found

Vulnerabilities

- By Date
- By Type
- Known Exploited
- Assigners
- CVSS Scores
- EPSS Scores
- Search

Vulnerable Software

- Vendors
- Products
- Version Search

Vulnerability Intel.

- Newsfeed
- Open Source Vulns
- Emerging CVEs
- Feeds
- Exploits
- Advisories
- Code Repositories
- Code Changes

Attack Surface

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 (Perl) as the extension.

Google

[Images](#) [Shopping](#) [Videos](#) [Maps](#) [News](#) [Books](#) [Flights](#) [Finance](#)

About 121,000 results (0.28 seconds)

Larimer County Search and Rescue
<https://www.larimercountysar.org/cgi-bin/Calcium40>

Calcium
Calcium Web Calendar - Brown Bear Software <http://www.brownbearsw.com>.

87.106.36
<http://87.106.36.176/Pfarrkalender/Calcium40>

Calcium
Calcium Web Calendar - Brown Bear Software <http://www.brownbearsw.com>.

Cambria County, PA (.gov)
<http://employees.cambriacountypa.gov/Calcium40>

Calcium
Name, Description, Administer. Group: CPC_Building. Room_Schedule, Schedule for rooms on the 2nd floor. Calcium 4.0.4 Professional Unlimited w/Email

Georgia State University
<https://chem-lithium.gsu.edu/cgi-bin/Calcium40>

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). <table><tbody><tr><td>1</td><td>Domain</td></tr><tr><td>2</td><td>jeffkaiser.com</td></tr><tr><td>3</td><td>rileypainting.com</td></tr><tr><td>4</td><td>ncfsudbury.com</td></tr><tr><td>5</td><td>ocvacationbibleschool.com</td></tr><tr><td>6</td><td>dementiafriendsiowa.org</td></tr><tr><td>7</td><td>buffalopal.com</td></tr><tr><td>8</td><td>livesportinggoods.com</td></tr><tr><td>9</td><td>6660kennedy.com</td></tr><tr><td>10</td><td>claritywellnesscoaching.com</td></tr></tbody></table>	1	Domain	2	jeffkaiser.com	3	rileypainting.com	4	ncfsudbury.com	5	ocvacationbibleschool.com	6	dementiafriendsiowa.org	7	buffalopal.com	8	livesportinggoods.com	9	6660kennedy.com	10	claritywellnesscoaching.com		
1	Domain																						
2	jeffkaiser.com																						
3	rileypainting.com																						
4	ncfsudbury.com																						
5	ocvacationbibleschool.com																						
6	dementiafriendsiowa.org																						
7	buffalopal.com																						
8	livesportinggoods.com																						
9	6660kennedy.com																						
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: <table><thead><tr><th>NETBLOCK</th><th>COMPANY</th></tr></thead><tbody><tr><td>103.245.222.0/24</td><td> Fastly, Inc</td></tr><tr><td>103.245.224.0/24</td><td> Fastly, Inc</td></tr><tr><td>104.156.80.0/24</td><td> Fastly, Inc.</td></tr><tr><td>104.156.81.0/24</td><td> Fastly, Inc.</td></tr><tr><td>104.156.82.0/24</td><td> Fastly, Inc.</td></tr><tr><td>104.156.83.0/24</td><td> Fastly, Inc.</td></tr><tr><td>104.156.84.0/24</td><td> Fastly, Inc.</td></tr><tr><td>104.156.85.0/24</td><td> Fastly, Inc.</td></tr><tr><td>104.156.87.0/24</td><td> Fastly, Inc.</td></tr><tr><td>104.156.89.0/24</td><td> Fastly, Inc.</td></tr></tbody></table>	NETBLOCK	COMPANY	103.245.222.0/24	 Fastly, Inc	103.245.224.0/24	 Fastly, Inc	104.156.80.0/24	 Fastly, Inc.	104.156.81.0/24	 Fastly, Inc.	104.156.82.0/24	 Fastly, Inc.	104.156.83.0/24	 Fastly, Inc.	104.156.84.0/24	 Fastly, Inc.	104.156.85.0/24	 Fastly, Inc.	104.156.87.0/24	 Fastly, Inc.	104.156.89.0/24	 Fastly, Inc.
NETBLOCK	COMPANY																						
103.245.222.0/24	 Fastly, Inc																						
103.245.224.0/24	 Fastly, Inc																						
104.156.80.0/24	 Fastly, Inc.																						
104.156.81.0/24	 Fastly, Inc.																						
104.156.82.0/24	 Fastly, Inc.																						
104.156.83.0/24	 Fastly, Inc.																						
104.156.84.0/24	 Fastly, Inc.																						
104.156.85.0/24	 Fastly, Inc.																						
104.156.87.0/24	 Fastly, Inc.																						
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.

```
(kali@kali)-[~]
$ dig +short dunstan.org.au
151.101.194.159
```

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.

The screenshot shows the 'Reverse IP' tool interface. At the top, there's a search bar with 'Domain, IP, Email or Name' and a search icon. Below it, the breadcrumb 'home > tools > reverse ip' is visible. The main section is titled 'Show all domain names hosted on this:' and contains a dropdown menu set to 'IP' with the value '151.101.194.159' and a 'Check' button. To the right, a 'Related Tools' box lists: IP Whois Lookup, Reverse IP History Lookup, Reverse MX Lookup, and Bulk DNS Lookup. Below the search bar, a message states 'Result Data Hidden' and 'Please log in or sign up to view data on this report.' A yellow banner below that says 'Found a total of 40,955 domains but only showing the first 10. To view more results login or purchase a premium report.' The 'Results Summary' section contains five boxes: 'Top Sites' (listing pro-analytics.net, alansonmedia.com, searchstaffing.net, soundcontracting.net, leadamerica.org), 'Top Registrars' (listing GoDaddy - 4,670 domains, NSI - 737 domains, google - 563 domains, Tucows - 523 domains, namecheap - 444 domains), 'Top Registrants' (listing Domain Administrator - 18 domains, Rachel Cutrer - 15 domains, Alan Olson - 13 domains, Sabre Industries Inc. - 12 domains, Proxy Protection Llc - 8 domains), 'Top Organizations' (listing Domains By Proxy, Llc - 58 domains, Ranch House Designs, Inc. - 24 domains, Contact Privacy Inc. Cus... - 18 domains, Sabre Industries Inc. - 14 domains), and 'Top Emails' (listing whois@bluehost.com - 55 domains, *****@***** - 32 domains, nocontactsfound@secure... - 27 domains, https://webform.meshdigi... - 22 domains). At the bottom, a note states 'The above summary was generated by analyzing the first 10,000 domains hosted on 151.101.194.159.' Navigation controls at the bottom show '1 - 10 of 10 total', 'Excel', 'CSV', and 'Previous', '1', 'Next' buttons.

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.

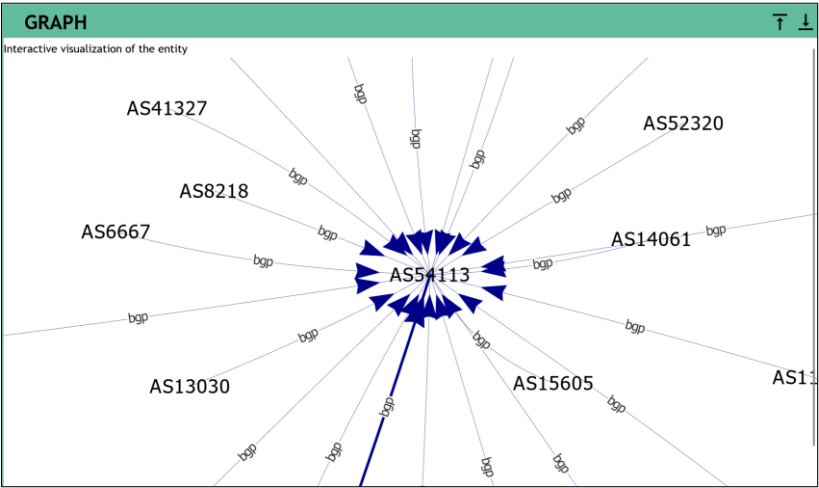
```
(kali@kali)-[~]
$ whois 151.101.194.159

#
# ARIN WHOIS data and services are subject to the Terms of Use
# available at: https://www.arin.net/resources/registry/whois/tou/
#
# If you see inaccuracies in the results, please report at
# https://www.arin.net/resources/registry/whois/inaccuracy_reporting/
#
# Copyright 1997-2024, American Registry for Internet Numbers, Ltd.
#

NetRange: 151.101.0.0 - 151.101.255.255
CIDR: 151.101.0.0/16
NetName: SKYCA-3
NetHandle: NET-151-101-0-0-1
Parent: RIPE-ERX-151 (NET-151-0-0-0-0)
NetType: Direct Allocation
OriginAS:
Organization: Fastly, Inc. (SKYCA-3)
RegDate: 2016-02-01
Updated: 2021-12-14
Ref: https://rdap.arin.net/registry/ip/151.101.0.0

OrgName: Fastly, Inc.
OrgId: SKYCA-3
```

To find the ASN (Autonomous System Number), I used <https://www.robtext.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.

ipinfo.io

Products Solutions Why IPinfo? Pricing Resources Docs Login

Sign up

Summary

IP Ranges >

WHOIS

Hosted Domains

Peers


Upstreams

Downstreams

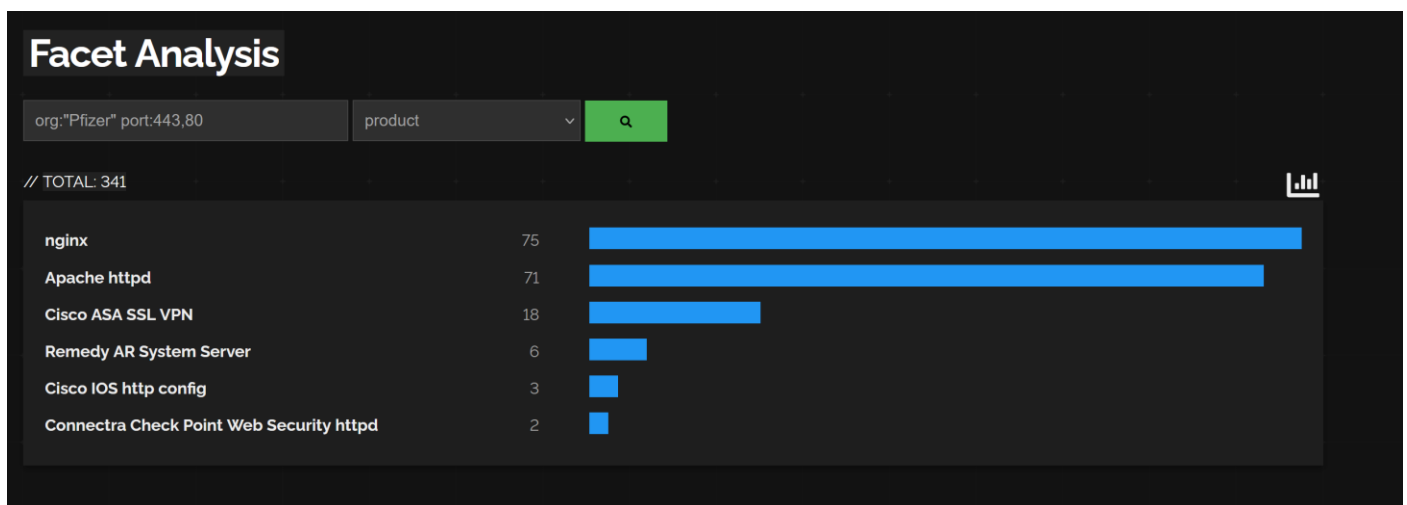
NETBLOCK	COMPANY	NUM OF IPS
103.245.222.0/24	Fastly, Inc	256
103.245.224.0/24	Fastly, Inc	256
104.156.80.0/24	Fastly, Inc.	256
104.156.81.0/24	Fastly, Inc.	256
104.156.82.0/24	Fastly, Inc.	256
104.156.83.0/24	Fastly, Inc.	256
104.156.84.0/24	Fastly, Inc.	256
104.156.85.0/24	Fastly, Inc.	256
104.156.87.0/24	Fastly, Inc.	256
104.156.89.0/24	Fastly, Inc.	256

Question 5):

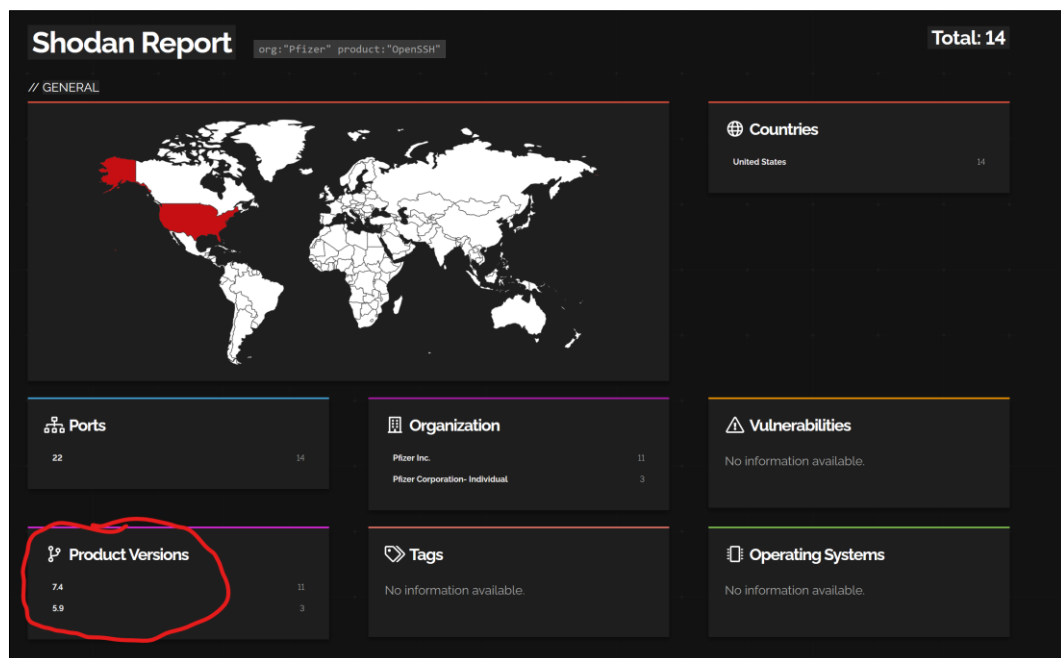
Question:	Answer:
What web server(s) are used by this company?	nginx, Apache httpd, Cisco ASA SSL VPN, Remedy AR System Server, Cisco IOS https config, and Connectra Check Point Web Security httpd.
What versions of OpenSSH are used by this company?	7.4 and 5.9

According to Shodan, what are of the vulnerabilities in one of the versions of the OpenSSH servers?	<div> <div>  Vulnerabilities </div> <div> <p>Note: the device may not be impacted by all of these issues. The vulnerabilities are implied based on the software and version.</p> </div> </div> <div> <div> CVE-2023-51767 </div> <div> <p>OpenSSH through 9.6, when common types of DRAM are used, might allow row hammer attacks (for authentication bypass) because the integer value of authenticated in mm_answer_authpassword does not resist flips of a single bit. NOTE: this is applicable to a certain threat model of attacker-victim co-location in which the attacker has user privileges.</p> </div> </div>
Choose the most recent vulnerability from above and find the CVSS2.0 string for it by looking it up on nvd.nist.gov	<p>I couldn't find the CVSS2.0 string, here is the CVSS3.1 string:</p> <p>CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H</p>

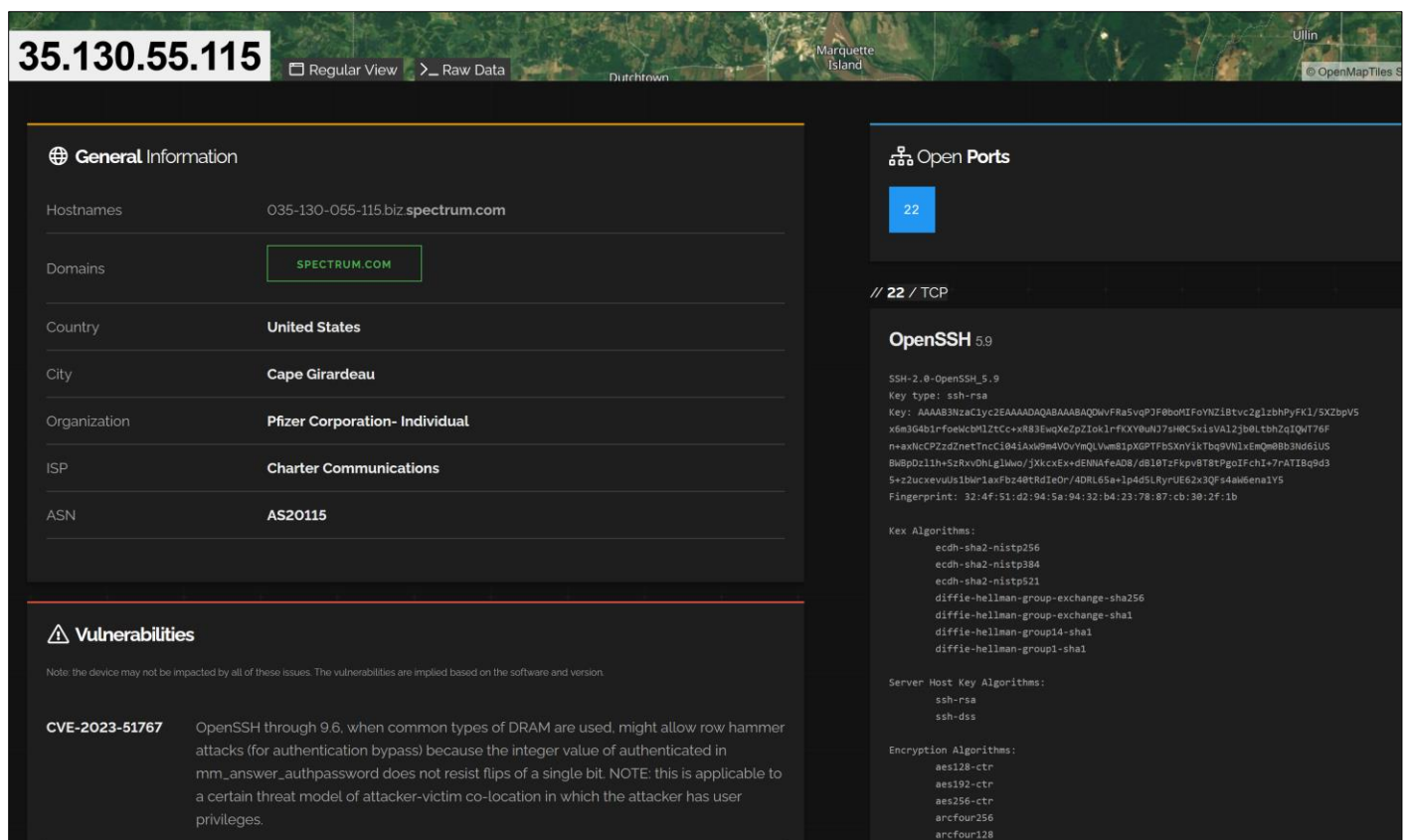
To find the web server(s) used by Pfizer, I used the modifier **org:"Pfizer"**, **port:443,80**. These two ports (443 and 80) are the default for HTTPS and HTTP. I then looked at the report, to see the products that run on these ports. Clearly, Pfizer uses nginx, Apache httpd, Cisco ASA SSL VPN, Remedy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

https://nvd.nist.gov/vuln/detail/CVE-2023-51767

CVE-2023-51767 Detail

Description

OpenSSH through 9.6, when common types of DRAM are used, might allow row hammer attacks (for authentication bypass) because the integer value of authenticated in mm_answer_authpassword does not resist flips of a single bit. NOTE: this is applicable to a certain threat model of attacker-victim co-location in which the attacker has user privileges.

Severity

CVSS Version 3.x CVSS Version 2.0

CVSS 3.x Severity and Metrics:

NIST: NVD **Base Score: 7.0 HIGH** **Vector:** CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H

NVD Analysts use publicly available information to associate vector strings and CVSS scores. We also display any CVSS information provided within the CVE List from the CNA.

Note: NVD Analysts have published a CVSS score for this CVE based on publicly available information at the time of analysis. The CNA has not provided a score within the CVE List.

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:

```

1 #!/usr/bin/env python3
2 import sys, socket
3 socket.setdefaulttimeout(0.1) # set timeout to 100ms
4 host = "www.adelaide.edu.au"
5
6 def tryDNS(domain, dictionary):
7     try:
8         with open(dictionary, 'r') as file: # read dictionary
9             for line in file:
10                 subdomain = line.strip()
11                 full = f"{subdomain}.{domain}"
12                 try:
13                     ip = socket.gethostbyname(full)
14                     print(f"{full} resolves to {ip}")
15                 except socket.gaierror: # domain doesn't resolve
16                     pass
17     except:
18         pass
19
20
21 tryDNS("adelaide.edu.au", 'dnsmap.txt')

```

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

```

bash-4.4$ python3 bruteforce.py
m.adelaide.edu.au resolves to 129.127.149.1
ad.adelaide.edu.au resolves to 10.33.103.3
av.adelaide.edu.au resolves to 129.127.95.145
cp.adelaide.edu.au resolves to 129.127.149.31
cs.adelaide.edu.au resolves to 129.127.149.1
gg.adelaide.edu.au resolves to 129.127.144.5
gp.adelaide.edu.au resolves to 192.43.227.193
id.adelaide.edu.au resolves to 35.71.156.117
ks.adelaide.edu.au resolves to 129.127.43.66
mw.adelaide.edu.au resolves to 129.127.144.69
ns.adelaide.edu.au resolves to 129.127.40.3
pc.adelaide.edu.au resolves to 129.127.178.166
qm.adelaide.edu.au resolves to 10.33.22.160
sb.adelaide.edu.au resolves to 129.127.144.69
aml.adelaide.edu.au resolves to 129.127.9.104
ams.adelaide.edu.au resolves to 52.255.35.249
api.adelaide.edu.au resolves to 129.127.149.154

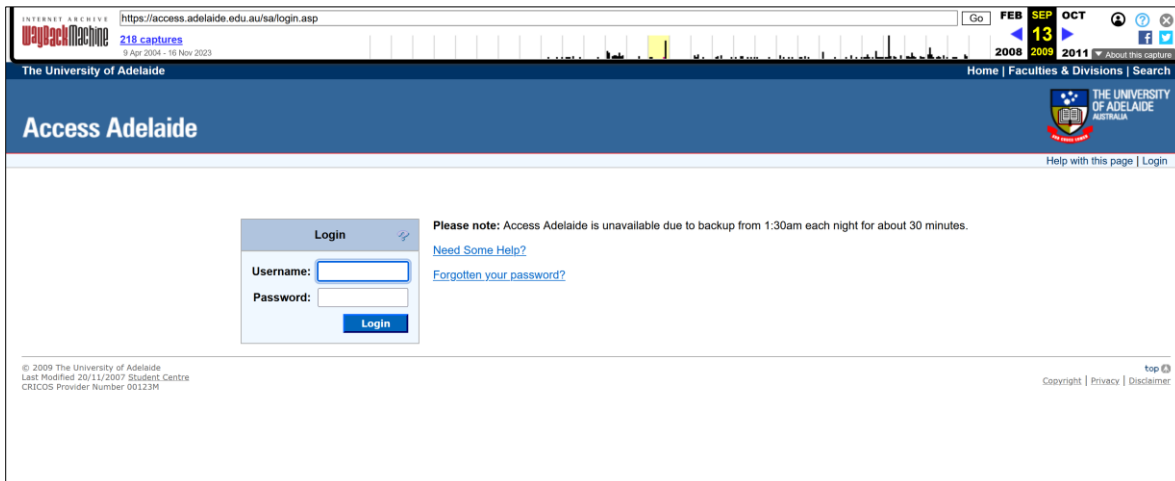
```


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):

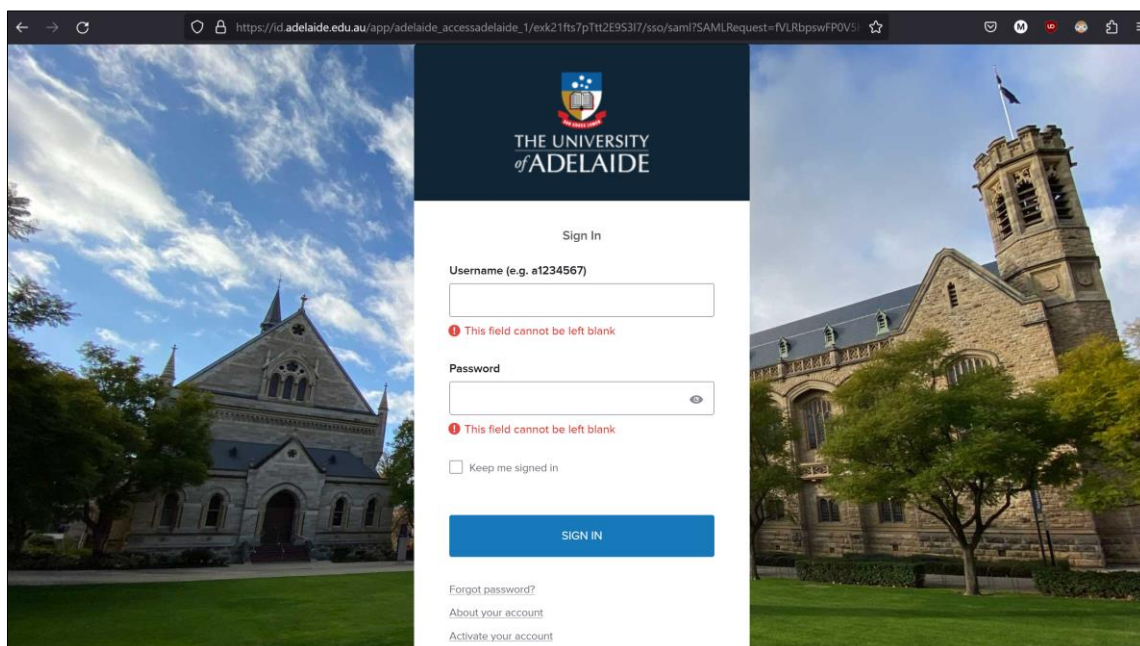
To find out what Access Adelaide used to look like, I went to <https://archive.org/web/>, entered the URL <https://access.adelaide.edu.au> and selected "Browse History". I selected the snapshot that was taken on the 3rd of August 2009 at 14:34:38.

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



The screenshot shows a web browser window with the URL <https://access.adelaide.edu.au/sa/login.asp>. The page has a blue header with "The University of Adelaide" and "Access Adelaide" text. Below the header, there is a login form with fields for "Username:" and "Password:", and a "Login" button. To the right of the form, there is a "Please note" message: "Access Adelaide is unavailable due to backup from 1:30am each night for about 30 minutes." Below this note are links for "Need Some Help?" and "Forgotten your password?". At the bottom of the page, there is a copyright notice: "© 2009 The University of Adelaide. Last Modified 20/11/2007 Student Centre. CRICOS Provider Number 00123M." and a "top" link.

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.



The screenshot shows a modern web browser window with the URL https://id.adelaide.edu.au/app/adelaide_accessadelaide_1/exk21fts7pTt2E9S3I7/sso/saml?SAMLRequest=IVLRbbspwFP0V5I. The page features a large background image of a university building. In the center, there is a "Sign In" section with a "Username (e.g. a1234567)" field and a "Password" field. Below these fields are error messages: "This field cannot be left blank" for both. There is a "Keep me signed in" checkbox and a "SIGN IN" button. At the bottom, there are links for "Forgot password?", "About your account", and "Activate your account".

Question 8):

- The port number is 21245.

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.

```
File Edit Search View Document Help
~/Desktop/PKnocking/knock.py - Mousepad
1 from scapy.all import *
2
3 def knock(ip, ports):
4     for port in ports:
5         ipl = IP(dst=ip)
6         tcpl = TCP(dport=port, flags='S')
7         packet = ipl/tcpl
8         send(packet, verbose=False)
9         print(f"SENT:{port}")
10
11 knock('192.168.56.113', [2201, 2211, 2234])
12 print("DONE")
13
```

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-weatheright-domciling}.

```
(kali㉿kali)-[~/Desktop/PKnocking]
$ sudo python3 knock.py
SENT:2201
SENT:2211
SENT:2234
DONE

(kali㉿kali)-[~/Desktop/PKnocking]
$ nc 192.168.56.113 12345

/ csf2024s1_{coeternally-weatheright-dom \
\ iciling} \

      ^ ^
      (oo)\_____
      (__) \       )\/\
           ||----w |
           ||     ||

(kali㉿kali)-[~/Desktop/PKnocking]
$ _
```

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.