Introduction

This report will see me act as a security analyst to investigate a security incident. The objective is to identify the infected host, dictate how it occurred and classify the type of infection. This investigation was primarily conducted using Wireshark to filter, analyse and trace the infected traffic. The report is structured like so: Methodology will describe the tools and techniques used; the results section will showcase the key evidence of my findings; and the conclusion will highlight preventative measures and challenges faced.

Methodology

The analysis of the PCAP file was conducted using Wireshark as the primary tool while also using the Linux terminal when necessary. I performed intrusion analysis to identify the infected system by using a variety of Wireshark filters such as:

- http.request – Filter to only show http requests.
- Show HTTPS traffic (443); tls.handshake==1 reveals client hello and frame.time sets the time range (Figure 1).

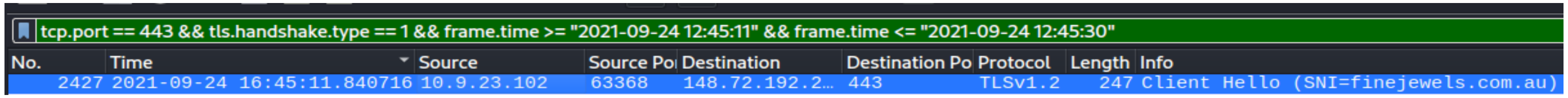

Figure 1. Finding Malicous domains

- Tls.handshake.certificate – Filter to locate the certificate.
- Ip.addr == "185.106.96.158" – Filter for specific IP address.
- Filter specific IP address and show only the server's name (Figure 2).

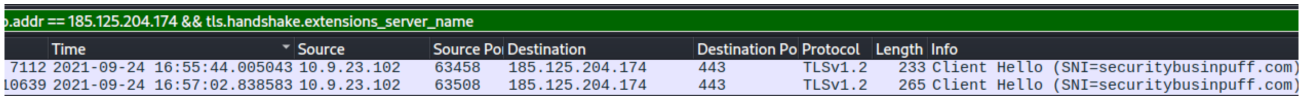

Figure 2. IP+Server Name Filter

- http.request.method=="POST" && ip.src=="IP" – Filter to show only post requests from a given IP.

- DNS filter for IP check request. Uses common IP address checkers (Figure 3)

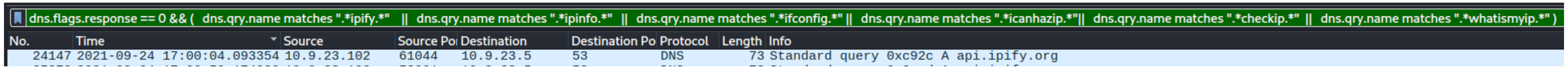

Figure 3. IP check requests

- SMTP – A filter to show only Simple-Mail-Transfer-Protocol.

Using these filters, I was able to identify the first indication of infection, identify how much the infection has spread and what caused the system to be infected in the first place.

To identify the certification authority, I found the specific TLS handshake packet in Wireshark. The Transport Layer was expanded, and the certificate was now visible. Next, I had to export the packet bytes and save them to my system. Once opened the entire certificate was visible. This inspection of the CA proved that the attacker was using a trusted certificate to evade detection and blend into normal traffic (Figure 5).

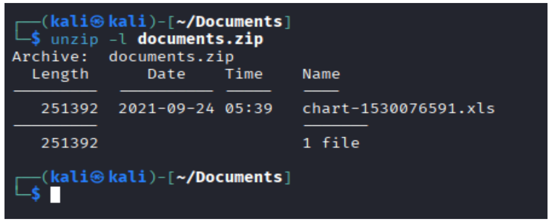

Figure 4. Unzip -l Command

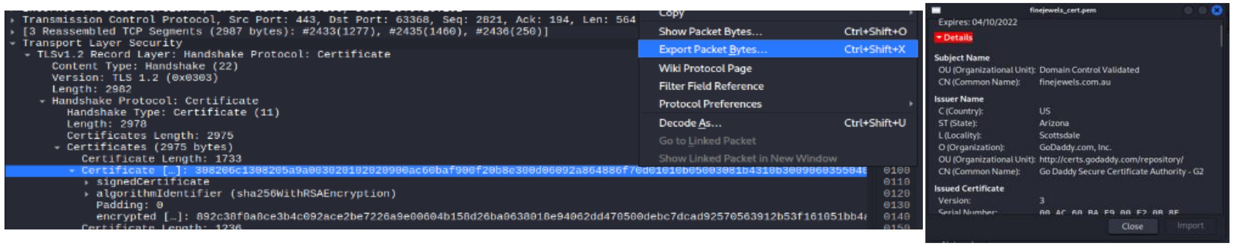

Figure 5. TLS Certificate

To locate the downloaded file for inspection I went to the object list menu and found the corresponding file (Figure 6). Next, I needed to export it to my system without extracting as this is a malicious file. To see the contents inside the file without extracting I used the “uzip -l” command to view the contents (Figure 4).

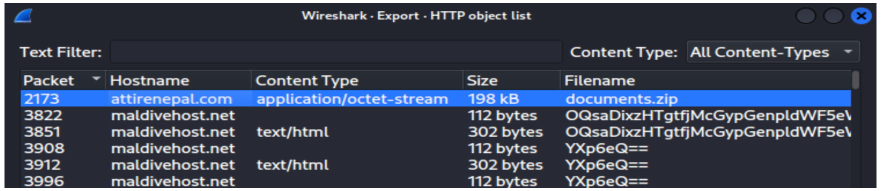

Figure 6. Object list Menu

Throughout my inspection following the relevant streams was crucial in exposing the infection throughout the PCAP file (Figure 8). On the initial infection I followed the TCP stream which revealed more information about the infection like Host, File name, and Server name (Figure 7).

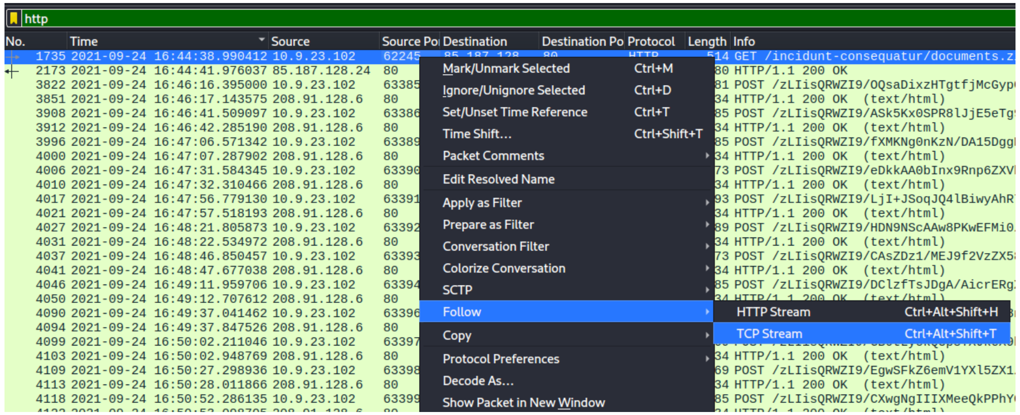

Figure 8. TCP Stream

Figure 7. TCP Stream details

Results

The initial HTTP connection occurred at **2021-09-24 at 16:44:38 UTC** when the host made a HTTP GET request to an external domain **attirenepal.com** (Figure 10). The file that the host downloaded was **documents.zip** and the contents of the file was a malware file named **chart-1530076591.xls**. The IP that delivered the malware is at **(85.187.128.24)** and is running the **LiteSpeed** web server with version **PHP/7.2.34** (Figure 9). It's clear at this point that the host fell victim to a type of Phishing attack.

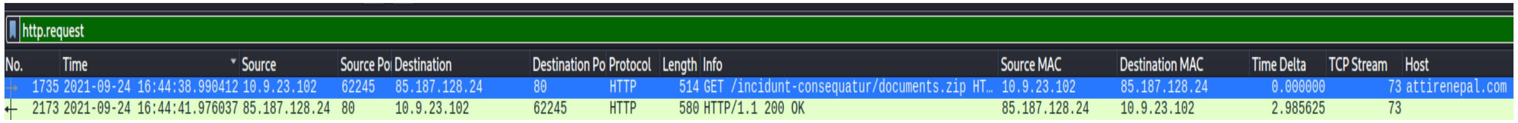

Figure 10. Initial HTTP connection

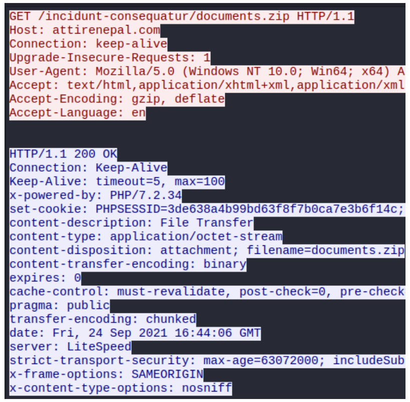

Figure 9. HTTP Stream

They were additional domains that played their part in infecting this system quickly after infecting the system. The three additional domains were **finejewels.com.au, thietbiagt.com,** and **new.americold.com** all between **16:45:11–16:45:30 UTC** (Figure 11). The domain **finejewels.com.au** has an SSL certificate from **GoDaddy** which was extracted from the TLS handshake. The attacker utilised a legitimate certificate to evade detection as an unsigned or invalid certificate would typically be flagged.

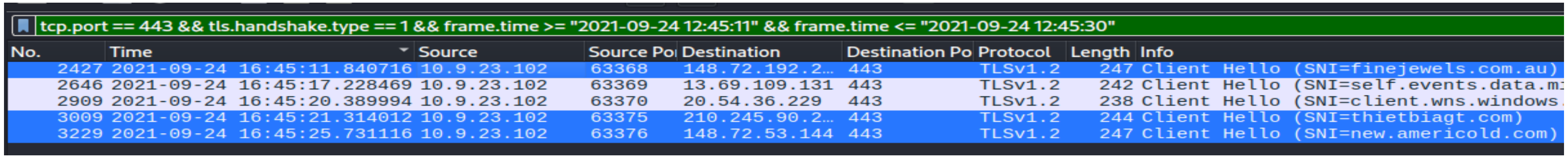

Figure 11. Additional domains

Once established the malware connected to two Cobalt Strike servers at the following addresses **185.106.96.158, 185.125.204.174**. The first IP **185.106.96.158** has the domain name **survmeter.live** that was captured (Figure 13). The malware used a host header **ocsp.verisign.com** to try and masquerade as legitimate traffic to try and deceive anyone who looked at it (Figure 12). This is a technique called domain fronting which uses different domains on the same HTTPS traffic. (Arntz, 2023) p.addr == 185.106.96.158

	Time	Source	Source Po	Destination	Destination Po	Protocol	Length	Info	Source MAC	Destination MAC	Time Delta	TCP Stream	Host
6323	2021-09-24 16:55:08.330879	10.9.23.102	63447	185.106.96.158	80	TCP	66	63447 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS…	10.9.23.102	185.106.96.158	0.000000	163	
6324	2021-09-24 16:55:08.592910	185.106.96.158	80	10.9.23.102	63447	TCP	58	80 → 63447 [SYN, ACK] Seq=0 Ack=1 Win=6424…	185.106.96.158	10.9.23.102	0.262031	163	
6325	2021-09-24 16:55:08.593239	10.9.23.102	63447	185.106.96.158	80	TCP	54	63447 → 80 [ACK] Seq=1 Ack=1 Win=65535 Len…	10.9.23.102	185.106.96.158	0.000329	163	
6326	2021-09-24 16:55:08.593562	10.9.23.102	63447	185.106.96.158	80	HTTP	306	GET /spfooh/cacerts.crl HTTP/1.1	10.9.23.102	185.106.96.158	0.000323	163	ocsp.verisign.

Figure 12. First Cobalt Strike Server

dns.a==185.106.96.158

No.	Time	Source	Source Po	Destination	Destination Po	Protocol	Length	Info
6511	2021-09-24 16:55:10.808336	10.9.23.5	53	10.9.23.102	58930	DNS	90	Standard query response 0xe5da A survmeter.live A 185.106.96.158

Figure 13. DNS Response Packet

The second IP **185.125.204.174** had a domain **securitybusinpuff.com** and was captured by filtering to any TLS handshakes connected to that IP (Figure 14).

ip.addr == 185.125.204.174 && tls.handshake.extensions_server_name

No.	Time	Source	Source Po	Destination	Destination Po	Protocol	Length	Info
7112	2021-09-24 16:55:44.005043	10.9.23.102	63458	185.125.204.174	443	TLSv1.2	233	Client Hello (SNI=securitybusinpuff.com)
10639	2021-09-24 16:57:02.838583	10.9.23.102	63508	185.125.204.174	443	TLSv1.2	265	Client Hello (SNI=securitybusinpuff.com)

Figure 14. Second Strike Server

The victim host now acts as a beacon by sending frequent POST requests back to the command server. The domain that is used for the post infection traffic is **maldivehost.net** and is captured sending regular POST requests (Figure 16). The POST request is about 281 bytes and contains data such as **"zLIisQRWZI9"**. The malicious domain was hosted on a server running **Apache/2.4.49 (cPanel) OpenSSL/1.1.1l mod_bwlimited/1.4 (Figure 15)**. The malware performed a DNS query to determine the victims IP address at **2021-09-24 17:00:04 UTC**. The malware used a domain named **api.ipify.org** (Figure 17).

```
POST /zLIisQRWZI9/OQsaDixzHTgtfjMcGypGenpldWF5eWV9f3k= HTTP/1.1
Host: maldivehost.net
Content-Length: 112

Dw8YBxsEGmYFAAEJfR4NQkMmLTYqZDk5KyQmOyRGQglxEBo4Lzk/EyYrMi1hOT8vI


HTTP/1.1 200 OK
Date: Fri, 24 Sep 2021 16:46:15 GMT
Server: Apache/2.4.49 (cPanel) OpenSSL/1.1.1l mod_bwlimited/1.4
```

Figure 15. POST Request HTTP Stream

http.request.method == "POST" && ip.src == 10.9.23.102

.	Time	Source	Source Port	Destination	Destination Po	Protocol	Length	Info	Source MAC	Destination MAC	Time Delta	TCP Stream	Host
3822	2021-09-24 12:46:16.…	10.9.23.102	63385	208.91.128.6	80	HTTP	281	POST /zLIisQRWZI9/OQsaDixzHTgtfjMcGypGenpldWF5eWV9f…	10.9.23.102	208.91.128.6	0.000000	104	maldivehost.net
3908	2021-09-24 12:46:41.…	10.9.23.102	63386	208.91.128.6	80	HTTP	285	POST /zLIisQRWZI9/ASk5Kx0SPR8lJjE5eTg9GkN6fGFyZHl/Y…	10.9.23.102	208.91.128.6	25.114097	105	maldivehost.net
3996	2021-09-24 12:47:06.…	10.9.23.102	63389	208.91.128.6	80	HTTP	285	POST /zLIisQRWZI9/fXMKNg0nKzN/DA15DggBI0N6fGFyZHl/Y…	10.9.23.102	208.91.128.6	25.062245	109	maldivehost.net
4006	2021-09-24 12:47:31.…	10.9.23.102	63390	208.91.128.6	80	HTTP	273	POST /zLIisQRWZI9/eDkkAA0bInx9Rnp6ZXVheXllfX95 HTTP…	10.9.23.102	208.91.128.6	25.013003	110	maldivehost.net

Figure 16. Beaconing connection

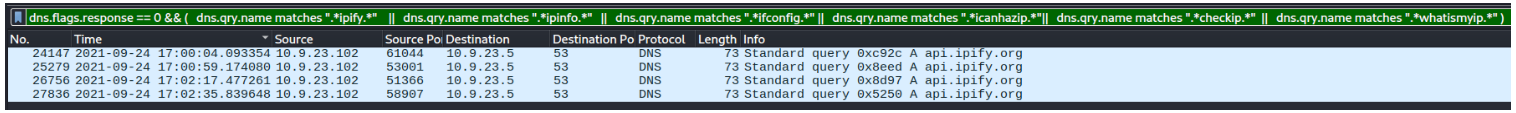

Figure 17. DNS Query

Within the PCAP file a significant vulnerability was present that saw email traffic in clear text. The first email address observed in the Simple Mail Transfer Protocol was farshin@mailfa.com (Figure 19)**.** Following the SMTP stream, credentials were discovered encoded in base64. These credentials belonged to the user ***ho3ein.sharifi*** and the password used was **"13691369" (Figure 18)**.

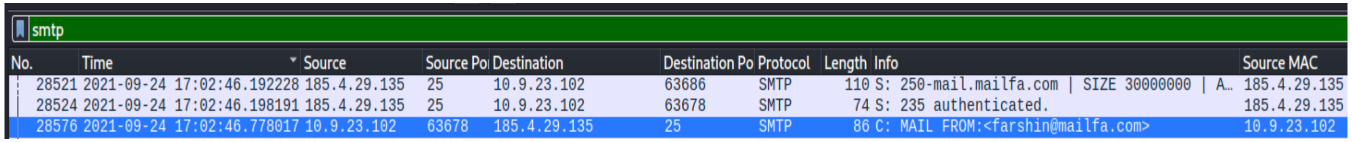

Figure 19. SMTP Traffic

Figure 18. SMTP stream

Conclusion

To prevent such an attack from happening again implementing additional security measures such as:

- Enforce encryption on sensitive data like email traffic this would have prevented the email leak.
- Implement logging and monitoring so that suspicious domains are reviewed. If logging was present the Cobalt Server may have been detected sooner.
- Training users to recognise patterns for phishing and other attacks such as the malicious file downloaded in the PCAP file.

By implementing these measures will reduce the vulnerabilities that led to this system being infected and will make the entire system more robust to future attacks.

GITHUB Link: https://github.com/shane-thompson211/COMP3010---Set-Exercises.git

YouTube Video Walkthrough: https://youtu.be/nZYttKH08Il

Student Declaration of AI Tool use in this Assessment

Please indicate your level of usage of generative AI for this assessment - please tick the appropriate category(s).

If the “Assisted Work” or “Partnered Work” category is selected, please expand on the usage and in which elements of the assignment the usage refers to.

Solo Work	**S1 - Generative AI tools have not been used for this assessment.**	☐
Assisted Work	**A1 – Idea Generation and Problem Exploration** Used to generate project ideas, explore different approaches to solving a problem, or suggest features for software or systems. Students must critically assess AI-generated suggestions and ensure their own intellectual contributions are central.	☐
	A2 - Planning & Structuring Projects AI may help outline the structure of reports, documentation and projects. The final structure and implementation must be the student’s own work.	☐
	A3 – Code Architecture AI tools maybe used to help outline code architecture (e.g. suggesting class hierarchies or module breakdowns). The final code structure must be the student’s own work.	☐
	A4 – Research Assistance Used to locate and summarise relevant articles, academic papers, technical documentation, or online resources (e.g. Stack Overflow, GitHub discussions. The interpretation and integration of research into the assignment remain the student’s responsibility.	☐
	A5 - Language Refinement Used to check grammar, refine language, improve sentence structure in documentation not code. AI should be used only to provide suggestions for	☐

	improvement. Students must ensure that the documentation accurately reflects the code and is technically correct.	
	A6 – Code Review AI tools can be used to check comments within the code and to suggest improvements to code readability, structure or syntax. AI should be used only to provide suggestions for improvement. Students must ensure that the code accurately reflects their knowledge and is technically correct.	☐
	A7 - Code Generation for Learning Purposes Used to generate example code snippets to understand syntax, explore alternative implementations, or learn new programming paradigms. Students must not submit AI-generated code as their own and must be able to explain how it works.	☐
	A8 - Technical Guidance & Debugging Support AI tools can be used to explain algorithms, programming concepts, or debugging strategies. Students may also help interpret error messages or suggest possible fixes. However, students must write, test, and debug their own code independently and understand all solutions submitted.	☐
	A9 - Testing and Validation Support AI may assist in generating test cases, validating outputs, or suggesting edge cases for software testing. Students are responsible for designing comprehensive test plans and interpreting test results.	☐
	A10 - Data Analysis and Visualization Guidance AI tools can help suggest ways to analyse datasets or visualize results (e.g. recommending chart types or statistical methods). Students must perform the analysis themselves and understand the implications of the results.	☐

	A11 - Other uses not listed above Please specify:	☐
Partnered Work	**P1 - Generative AI tool usage has been used integrally for this assessment** Students can adopt approaches that are compliant with instructions in the assessment brief. Please Specify: • Summarising and shortening sentences to meet word count. • Report guidance to ensure criteria met. • To help with understanding how a certain Wireshark filter works. • Clarifying errors such as why a certain filter shows no packets. • Researching new concepts like Cobalt Strike Servers. • Generate Readme file for Git repo	☒

Please provide details of AI usage and which elements of the coursework this relates to:
Generative AI was used in a supportive role in the report section of this coursework. I leveraged AI to help summarise large sentences to lower word count. Provide guidance so that the report aligned with the marking criteria. Also, AI was used to help understanding certain Wireshark filters and troubleshoot any issues. **All the PCAP analysis, filtering, data extraction and interpretation was carried out independently by me.**

I understand that the ownership and responsibility for the academic integrity of this submitted assessment falls with me, the student.	☒
I confirm that all details provide above are an accurate description of how AI was used for this assessment.	☒

References

Arntz, P. (2023, 12 01). *Explained: Domain fronting*. Retrieved from ThreatDown by Malwarebytes: https://www.threatdown.com/blog/explained-domain-fronting/