Coursework 1: Set Exercises - PCAP Analysis

Github Link: <https://github.com/Mattrfish/COMP3010-SecOps-and-Incident-Management.git>

Youtube Walkthrough Link:

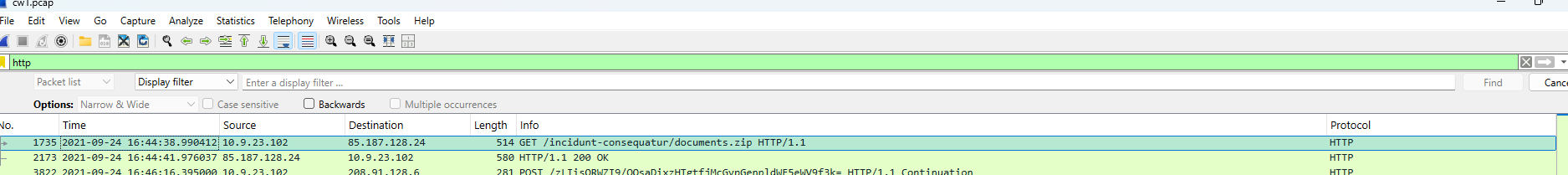
# Introduction

This incident report presents a detailed analysis of a provided PCAP file to investigate a suspected intrusion on the network. The purpose of this analysis is to identify the infected system, determine how the infection occurred, describe what type of attack was involved, and highlight the key indications of compromise (IOCs).

This report is structured as follows: The methodology section outlines what tools and techniques that will be used throughout the analysis and explains how they were applied to investigate the intrusion. The results section presents the key findings that I have gathered from the PCAP analysis, supported by evidence and screenshots. Finally, the conclusion summarises the overall findings and discusses prevention strategies to mitigate similar incidents in the future, and it reflects on any issues or challenges that were encountered during the investigation.

# Methodology

For this intrusion analysis I am going to be using Wireshark to analyse the pcap file as it is the most effective open source tool for packet-level network traffic analysis.



A screenshot of a computer program

AI-generated content may be incorrect. A screenshot of a computer

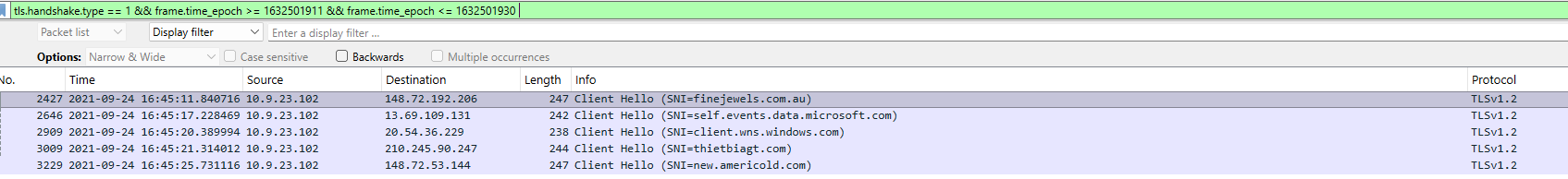
AI-generated content may be incorrect.

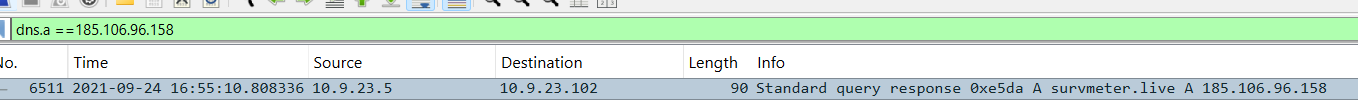
The analysis began by inspecting HTTP traffic using the “http” display filter, to detect the initial infection. I used the “follow > http stream” feature, to inspect communications between the victim and malicious domains and trace the infection process to inspect connections further by examining server information, headers, and communication patters. It also enabled me to inspect transferred file types without exporting them, reducing the risk of executing malicious content on my local machine.

A screenshot of a computer

AI-generated content may be incorrect.

I used the packet details pane to gain insight into the packet information and find out details like domain names, server headers, and SSL certificates.



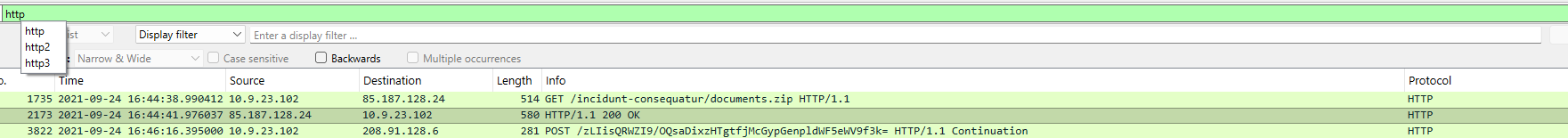


A screenshot of a computer

AI-generated content may be incorrect.

I used filters like frame.time and tls.handhsake to inspect the traffic further and find post-infection details. The dns.a filter identified the domains associated with c2 servers. I used the “statistics > conversations” to analyse packet counts and beaconing patterns to further understand the c2.

# Results



A screenshot of a computer program

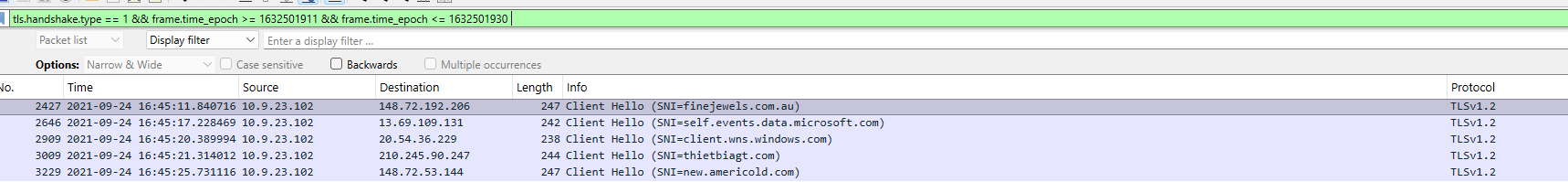
AI-generated content may be incorrect.

I started the analysis by filtering the data to find the initial http connection which occurred on 2021-09-24 16:44:38 between the victim host (10.9.23.102) and malicious destination server (85.187.128.24). Following the http stream, it was discovered that the victim downloaded a compressed file named “documents.zip” from the domain “attirenepal.com”, which contained a file called “chart-1530076591.xls”. The destination server was identified as running “LiteSpeed” web server software with version “PHP /7.2.34”.

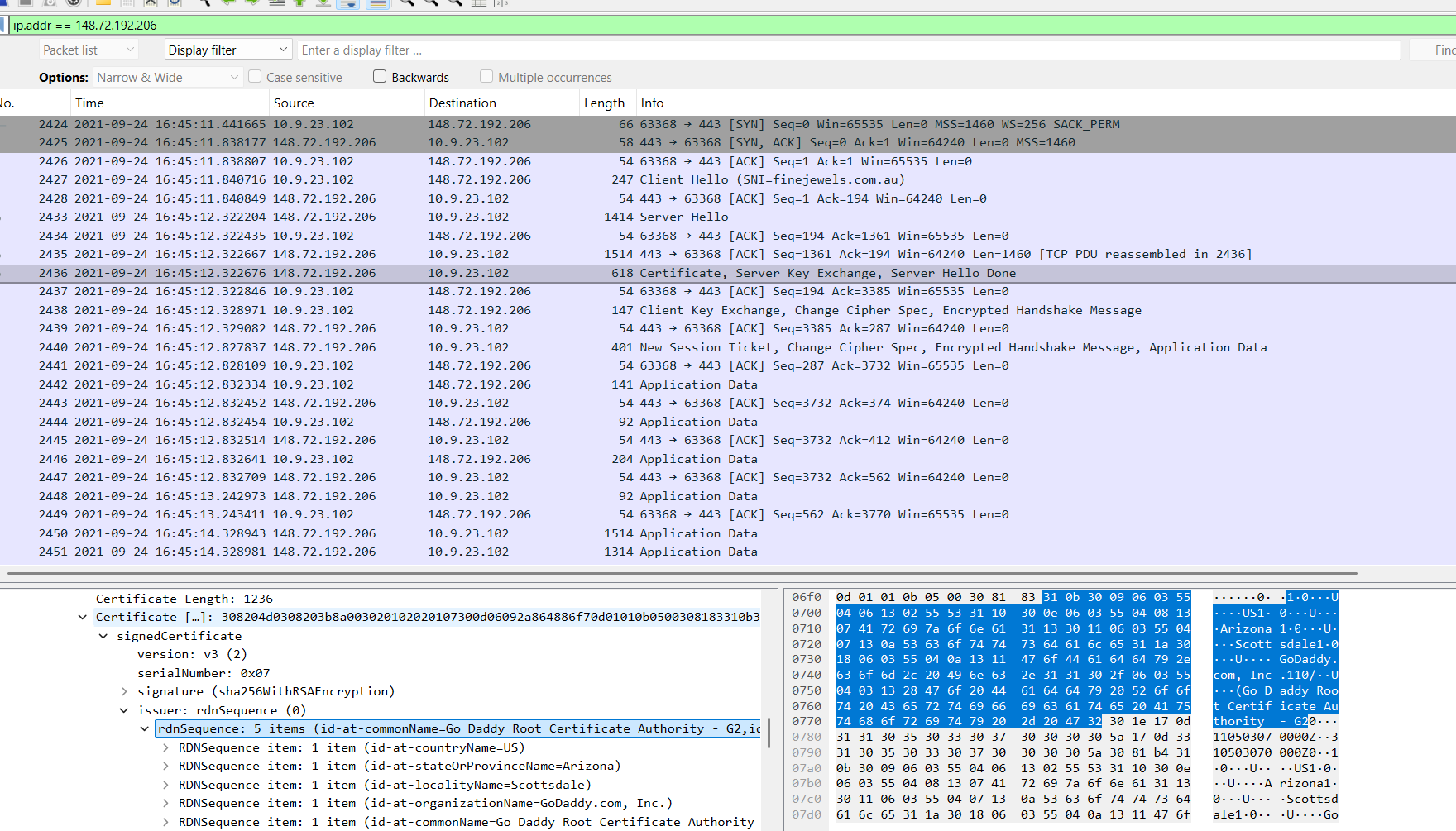
I decided to go back and inspect to http timeline further, which shows a POST request at 16:46:16. I decided to look closer at the traffic during the minute between the initial HTTP OK response and the POST request.

A screenshot of a computer

AI-generated content may be incorrect.



I identified several suspicious TLS requests and applied a filter to isolate only the “ClientHello” packets. This revealed three additional domains that were involved in downloading malicious files to the victim host: finejewels.com.au, thietbiagt.com and new.americold.com.



I looked up the IP of the first domain to find the CA issuer. I found the issue request and inspected the packet to find that it was Go Daddy Root Certificate Authority.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

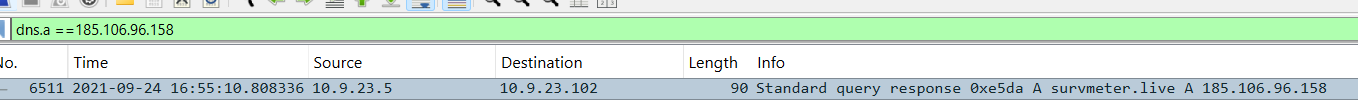
A screenshot of a computer

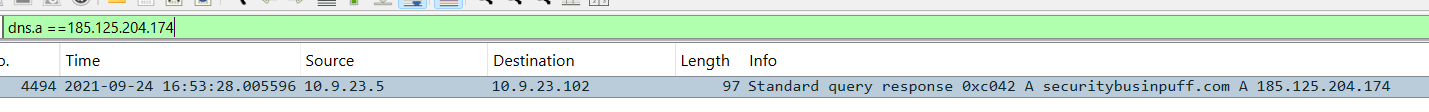
AI-generated content may be incorrect.

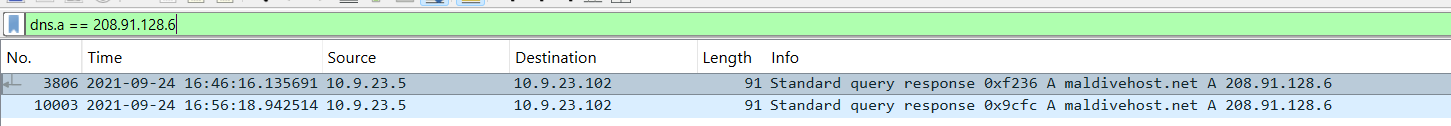
After revisiting the POST request, I noticed multiple requests being continuously sent every 30 seconds to a suspicious domain. To investigate further and determine whether it might be a command-and-control (c2) server, I examined the conversations section in the statistics. There I identified two other ips (185.106.96.158 and 185.125.204.174) that had a similar behaviour.

Using a display filter, I analysed these Ips and confirmed that they also had a high frequency of requests and a large volume of packets sent. Based on this consistent beaconing activity, I could tell these were C2 servers. The POST request had a much lower packet count suggesting that it was using for data exfiltration rather than beaconing.

Additionally I found the host header for the first IP which was ocsp.verisign.com\r\n.







To identify the domains associated with these Ips, I applied a DNS filter to match the IP to the domain. The first IP resolved to survmeter.live and the second to securitybusinpuff.com. The third one, which was associated with the post-incident requests, to maldivehost.net.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Next I examined the first request to the c2 server in more details and found that the payload was 281bytes in length with the message beginning with zLIisQRWZI9. Following thehttp stream further, I identified the sever header as “Apache/2.4.49 (cPanel) OpenSSL/1.1.1l mod\_bwlimited/1.4”.

# Conclusion

# Appendix