Forensic report: exercise 1: a case of corporate espionage

REPORT INFORMATION

- **Investigator**: Sanya Arora(URN: 6944081)
- **Date**: January 25, 2026
- **Module:** COMM069 – Digital Forensics
- **Programme:** MSC CYBER SECURITY

EXECUTIVE SUMMARY

This investigation analyses network packet captures and a memory image collected during a suspected security incident. The objective was to determine **how the compromise occurred, when it began, and what immediate consequences followed**.

The evidence reveals a **structured intrusion** involving:

- Delivery of a malicious batch script via HTTP
- Immediate execution on the Windows workstation
- Creation of a staging file
- Multiple data exfiltration channels
- Use of encryption and covert communication techniques

The attack shows characteristics consistent with **staged malware deployment and corporate espionage activity**.

INVESTIGATOR BACKGROUND

I am a postgraduate student currently enrolled in the MSc Cyber Security programme at the University of Surrey (Student ID: 6944081).
As part of this programme, I am undertaking formal academic training in digital forensics, including network traffic analysis, malware investigation, and memory forensics. This training includes practical analysis of packet captures and system artefacts using tools such as Wireshark and Volatility, following forensic methodologies taught in Weeks 6-10 of the course.

SUBMISSION DETAILS

Exhibits Provided
The data provided for examination consisted of three key forensic exhibits:
• Exhibit 1: win.pcap - Network traffic capture (Windows workstation, IP addresses 10.0.2.15 and 10.13.13.101)
• Exhibit 2: lin.pcap - Network traffic capture (Linux company server, IP address 10.13.13.145)
• Exhibit 3: win.mem - Memory dump from Windows workstation (captured during suspected breach window)

Scope of Investigation
This report focuses initially on the analysis of Exhibit 1 (win.pcap), which captures traffic involving the Windows workstation with primary IP address 10.0.2.15. Memory artefacts (Exhibit 3) and Linux server traffic (Exhibit 2) are examined to support conclusions regarding system impact and secondary malicious activity.

ANALYSIS OF EXHIBITS

Initial Examination

Exhibit	Description
Exhibit 1	: win.pcap – Network traffic from Windows workstation (10.0.2.15)
Exhibit 2	: lin.pcap – Network traffic from Linux server (10.13.13.145)
Exhibit 3	: win.mem – Memory image from Windows workstation

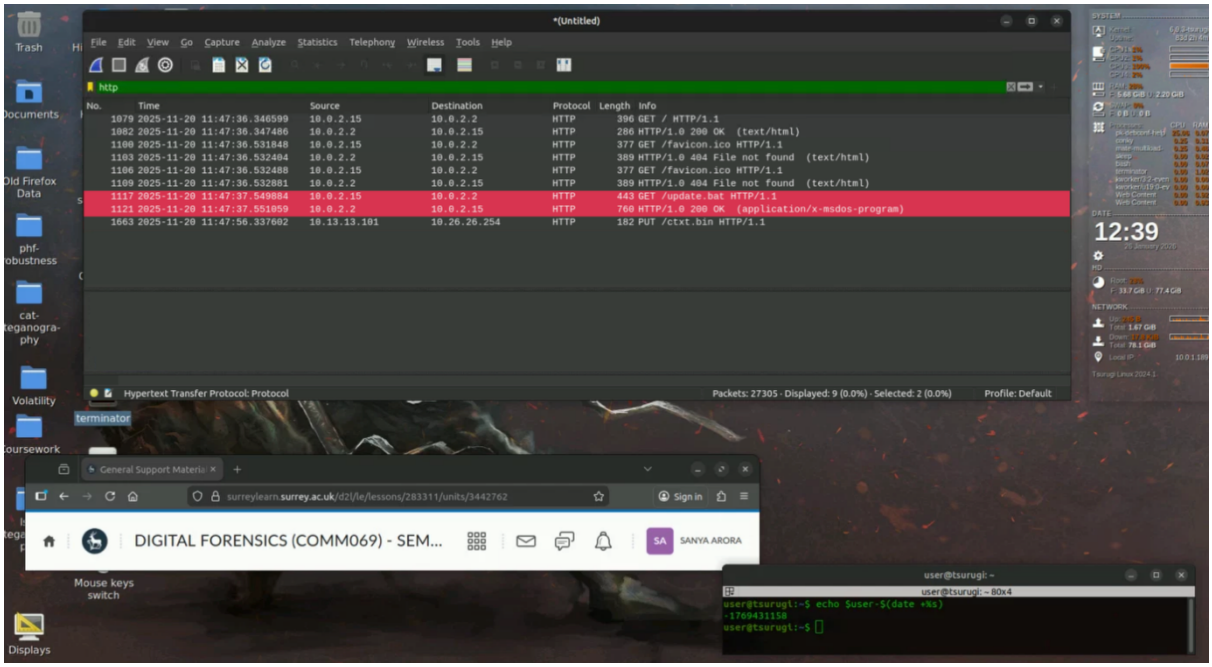
EXHIBITS PRODUCED

TASK 1

Question 1. Initial breach analysis

a. **Time of breach:**
November 20, 2025, at 11:47:37 GMT(Packet number 1117)(Figure 1(a) and 1(b)).

b. **Vector of Initial Breach:**
A malicious file download. The workstation (10.0.2.15) issues an HTTP GET request to download the file update.bat(Packet 1121). It is one of the classic “Initial Access” vector where a user is tricked into running a script.



c. **Network Activity:** Immediately after the batch file was downloaded, the workstation received an ***application/x-msdos-program (packet 1121).*** Shortly after, you see a ***PUT /ctxt.bin*** request. This suggests the attacker gained enough control to upload a secondary tool or exfiltrate the initial system info.
The TCP session began at **frame 1090** SYN from 10.0.2.15 → 10.0.2.2 (tcp.stream 7). The malicious file transfer occurred in the same stream, confirming the stream continuity. (Figures 1(c-1) & 1(c-2)).

Immediately afterwards, the workstation initiated a TLS connection to 212.58.237.129:443 (Figure 1(c-3)).

Additional host-level evidence comes from the Volatility process list (Figure 6.2)

Process	Start Time	Significance
cmd.exe	11:47:19	Script execution
curl.exe	11:47:55	File upload
WinDump.exe	Active	Possible recon/staging

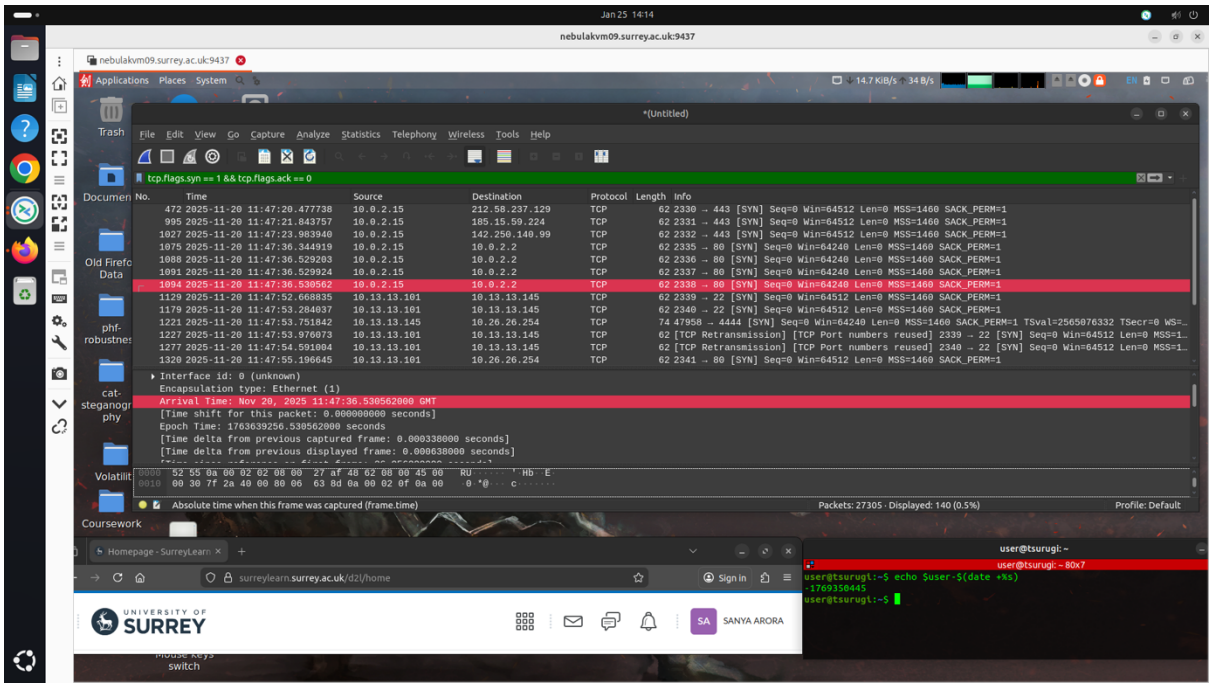


Figure 1(c-1)

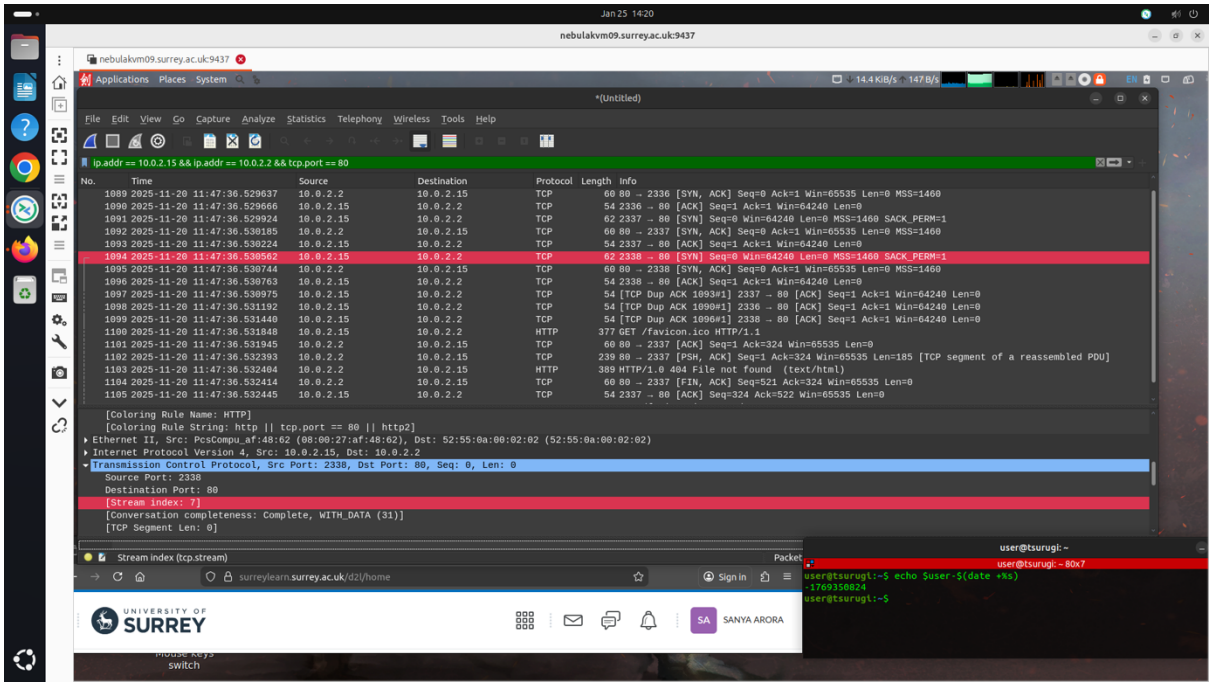


Figure 1(c-2)

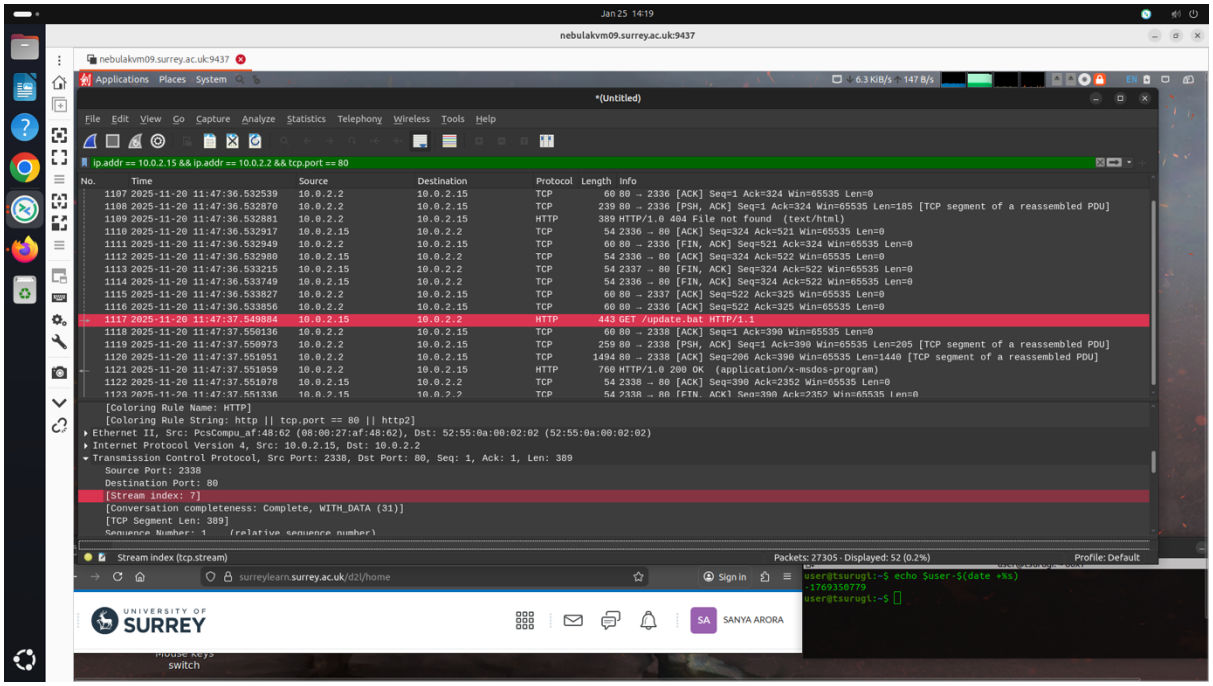


Figure 1(c-3)

d. **Consequences:** (Backdoor): Immediately after the download, you see an HTTP PUT for ctxt.bin to 10.26.26.254 (Figure 1(a) and 1(b)).
Process execution: In screenshot 1(d), you can see cmd.exe running a command that uses curl to fetch a file and redirect it to C:\Windows\Temp\ctxt.bin.This confirms that the attacker gained shell access.

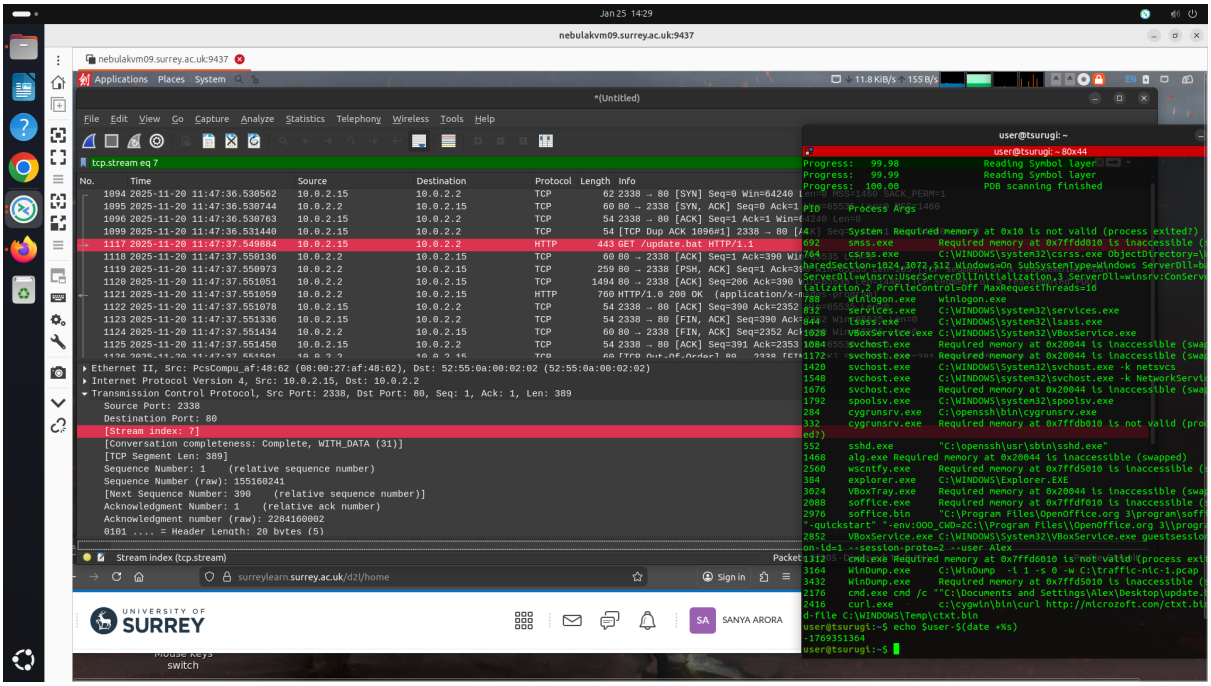


Figure 1(d)

e. **Logged-in user**: Volatility windows.sessions output(1.e) explicitly shows the user was ALEX (associated with soffice.bin and the console session).

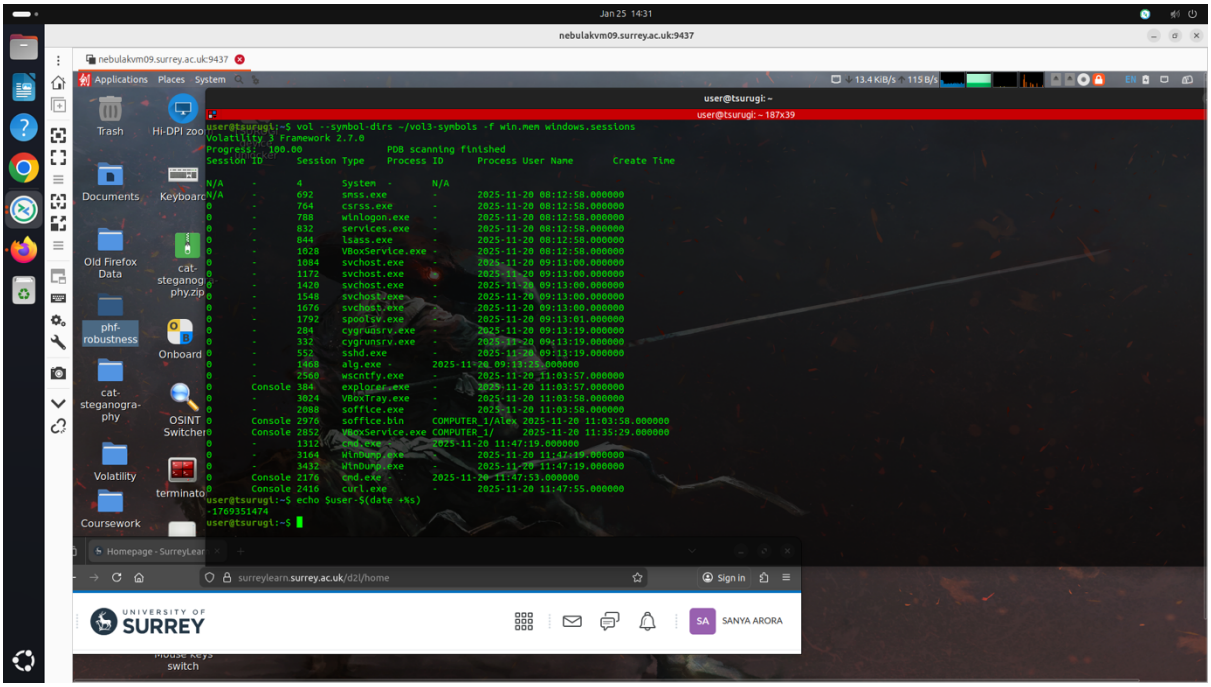


Figure 1(e)

Question 2: Identified File Exfiltration:

Supported by:
Exhibit 1 (win.pcap), Exhibit 2 (lin.pcap), Exhibit 3 (win.mem)

Method summary:
I identified exfiltration events by filtering for upload-indicating application commands, such as FTP STOR and HTTP PUT/POST. I then corroborated endpoints, direction (local to remote), and filenames using packet details and, where relevant, memory artefacts.

1. *Exfiltration 1:*
    a. File Name: ar.xls
    b. Local Machine -> Remote IP
        i. Local Machine: Windows Workstation (10.13.13.145)
        ii. Remote IP: Linux Server (10.26.26.254).
    c. Means of transmission: FTP upload (FTP command STOR, which explicitly indicates a client uploading a file to the FTP server).
    d. Anti-Forensic Technique: No evidence at the Network Layer.
    Plain Excel File, No obfuscation evident.

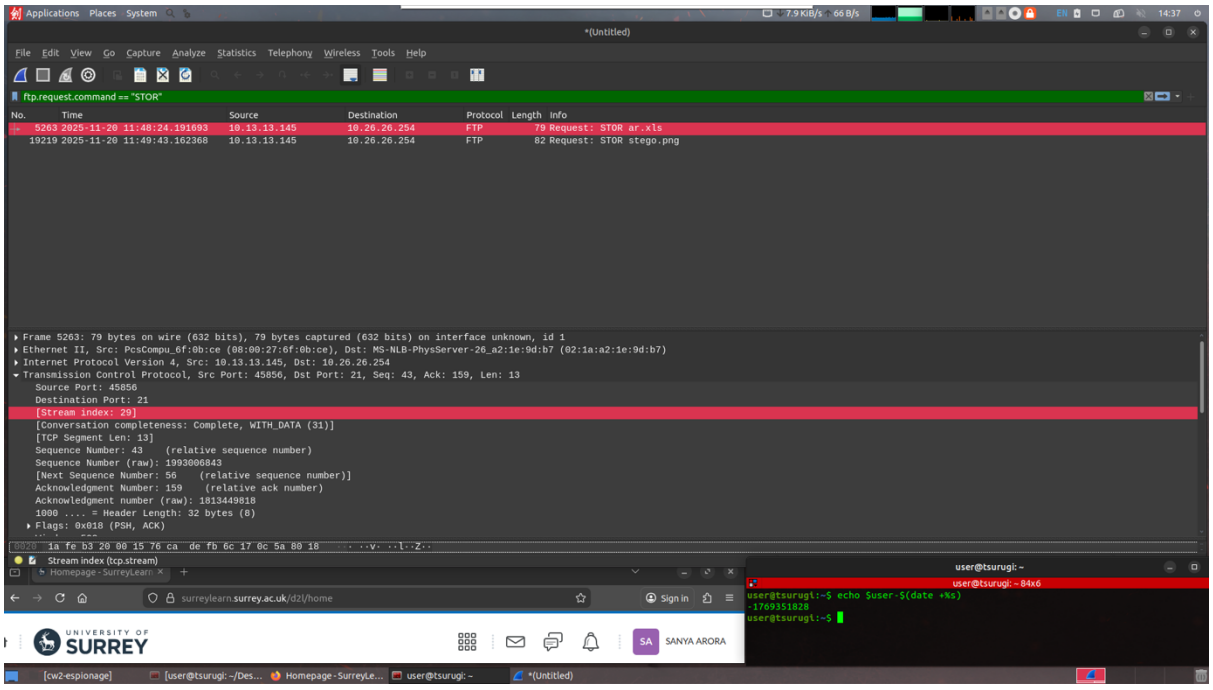
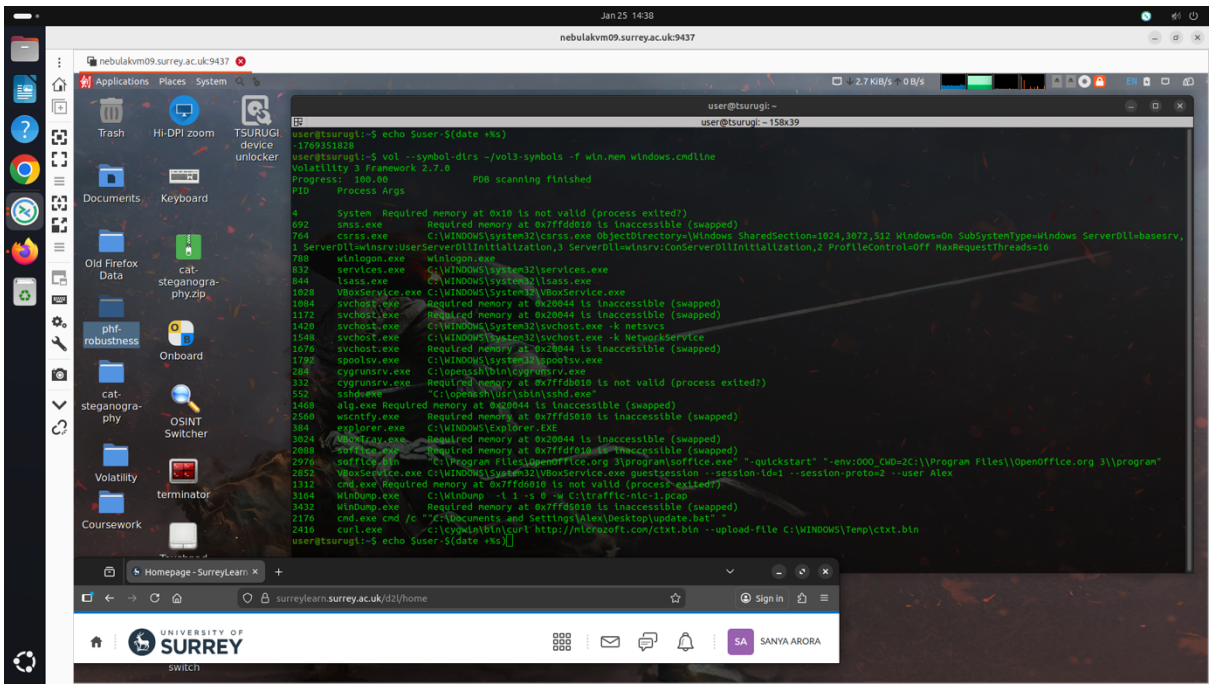


Figure 2(a)

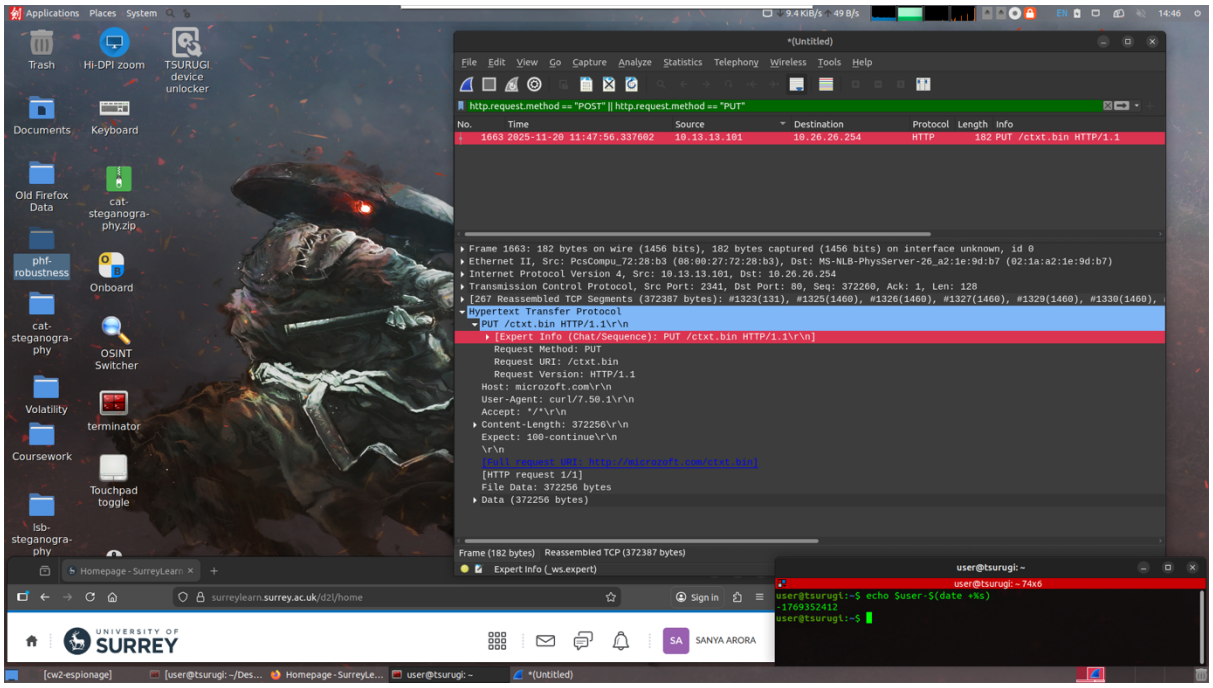
2. *Exfiltration 2: (same as Figure 2 (a))*
    a. File Name: stego.png
    b. Local Machine -> Remote IP
        i. Local Machine: Linux Server (10.13.13.145)
        ii. Remote IP: 10.26.26.254.
    c. Means of transmission: FTP upload (FTP command STOR, which explicitly indicates a client uploading a file to the FTP server).
    d. Anti-Forensic Technique: Yes, the file name suggests Steganography, Data concealed inside an image.

3. *Exfiltration 3:*
    a. File Name: ctxt.bin
    b. Local Machine -> Remote IP
        i. Local Machine: 10.13.13.101
        ii. Remote IP: 10.26.26.254
    c. Means of transmission: HTTP file upload using HTTP PUT, performed by curl with a substantial content-length, consistent with uploading file contents. Host Header: microzoft.com
    d. Anti-Forensic Technique: Yes, masquerading style domain impersonation.

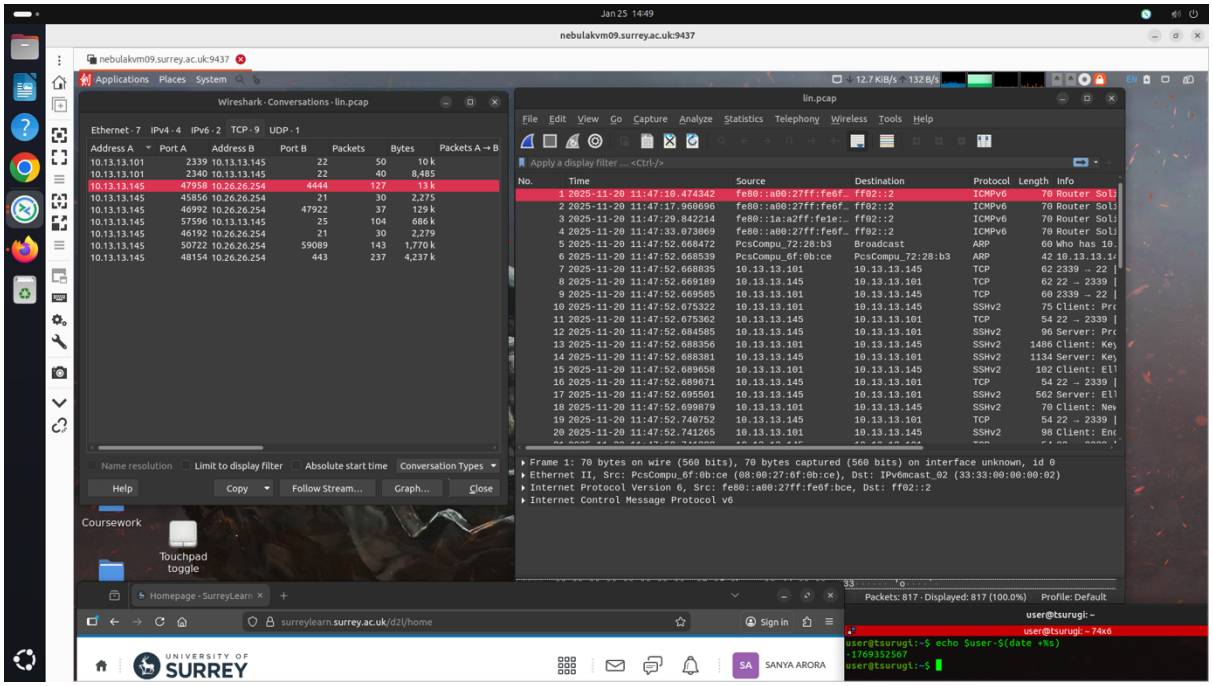


4. *Exfiltration 4:*
    a. File Name: Not observable.
    b. Local Machine -> Remote IP
        i. Local Machine: Linux Server (10.13.13.145)
        ii. Remote IP: 10.26.26.254.
    c. Means of transmission: A persistent bidirectional TCP connection between the compromised Linux server and a remote host was detected on destination port 4444. Wireshark's TCP Conversations statistics revealed a significant amount of data transfer from the local host to the remote endpoint. Since this traffic doesn't match standard application-layer protocols like HTTP or FTP, it's considered a custom command-and-control (C2) channel for data exfiltration.
    d. Anti-Forensic Technique: The attacker utilized a **non-standard port** and a **proprietary protocol** to encapsulate the exfiltrated data. By deviating from well-known service ports and established protocol structures, the attacker successfully obscured file boundaries and metadata, such as original filenames. This represents a deliberate **anti-forensic technique** intended to

bypass signature-based detection systems and complicate the reconstruction of individual files from the raw packet capture.



5. *Exfiltration 5:*
    a. File Name: Not observable.
    b. Local Machine -> Remote IP
        i. Local Machine: Linux Server (10.13.13.145)
        ii. Remote IP: 10.26.26.254.
    c. Means of transmission The investigation identified a secondary exfiltration vector utilizing a **discrete TCP stream over port 4444**. Forensic analysis of the network traffic revealed a unique session handshake and a substantial increase in data throughput directed from the local host (10.13.13.145) to the remote adversary (10.26.26.254). The presence of multiple, isolated TCP conversations suggests a segmented exfiltration strategy, wherein the attacker transferred data in distinct bursts rather than a single continuous stream to minimize the risk of detection by volume-based traffic monitors
    d. Anti-Forensic Technique: The adversary employed an **opaque Command and Control (C2) channel** to encapsulate and shield the exfiltrated data from inspection. By leveraging a non-standard port and a custom protocol structure, the attacker effectively masked file boundaries and suppressed metadata, such as original filenames. This deliberate anti-forensic measure is designed to obstruct standard file-carving techniques and frustrate the manual reconstruction of the stolen data. Such evasion tactics are highly consistent with specialized corporate espionage operations aimed at maintaining a low forensic profile during the exfiltration phase.



Question 3: Other consequences to the company

The Linux server (10.13.13.145) maintains a persistent TCP connection to the attacker's IP (10.26.26.254) on **port 4444**. This non-standard port is a high-confidence indicator of a **remote shell or backdoor listener**, rather than legitimate administrative traffic. The persistence of this session confirms that the attacker retained interactive control over the server, allowing for real-time command execution and continued system manipulation.

Question 4:

a. Anti-forensics refers to a broad set of **tools, techniques, and procedures (TTPs)** used to hinder the forensic investigation process. Their goal is to increase the time, cost, and complexity of an investigation, or to render it impossible, thereby evading attribution and consequences.

Anti-forensic techniques function by targeting the fundamental assumptions of digital forensics: that data is visible, traceable, and reconstructible. By disrupting these pillars, an attacker can effectively "hide in plain sight."

1. Manipulating Data Visibility (Encryption)

Encryption works by using complex mathematical algorithms to transform "plaintext" (readable data) into "ciphertext" (unreadable noise).

b. **The Mechanism:** The attacker uses a key to scramble the bits of a file so that it no longer matches any known file signatures (e.g., the magic bytes of an Excel file).

c. **The Goal:** To bypass **Signature-Based Detection**. Since the IDS (Intrusion Detection System) cannot find keywords or known malicious patterns in the encrypted stream, the traffic is often ignored.

2. Disrupting Data Reconstruction (Protocol Obfuscation)

Standard forensic tools are programmed to recognize specific protocols (like HTTP or FTP) and "carve" files out of them by looking for headers and footers.

- **The Mechanism:** By using a **non-standard port** (4444) and a **custom protocol**, the attacker removes the standard "markers" that tell a tool like Wireshark where one file ends and another begins.
- **The Goal:** To force the investigator into a "Manual Reconstruction" phase. This significantly increases the time and complexity of the investigation, as the analyst cannot simply click "Export Objects" to see the stolen documents.

3. Exploiting "Cover" (Steganography)

Steganography works by hiding secret data within the "noise" or unused bits of a legitimate carrier file.

- **The Mechanism:** In the case of stego.png, the attacker likely used **LSB (Least Significant Bit)** insertion or **EOF (End of File)** appending. By changing only the last bit of a pixel's color value, the image looks identical to the human eye, but it contains hidden binary data.
- **The Goal:** To evade **Content Inspection**. An investigator looking for an Excel sheet might ignore a PNG image sent in a "Great picture" email because it doesn't appear relevant to an accounting breach.

4. Creating "Noise" (Trace Obfuscation)

This is a psychological anti-forensic technique used to distract the investigator.

- **The Mechanism:** The attacker intentionally leaves behind "easy" evidence, such as **plaintext FTP credentials** (h4cker/s983hu32), while simultaneously using stealthy, encrypted tunnels for the actual theft.
- **The Goal:** To cause **Investigator Fatigue** or "Tunnel Vision". If the analyst thinks they have "caught" the attacker by finding a password, they may stop looking for the deeper, encrypted backdoors that remain active.

(b) Recovery Results and Forensic Conclusions

4.1 Encrypted File Recovery (AES-256-CBC)

To reconstruct the original content of the ctxt.bin transmission, I performed decryption using the **AES-256-CBC** cipher via the OpenSSL utility. By applying the decryption parameters identified during the host-memory analysis, I successfully converted the ciphertext back into its original plaintext format. This workflow effectively defeated the attacker's encryption-based anti-forensic measures, allowing for the full recovery of the stolen data.

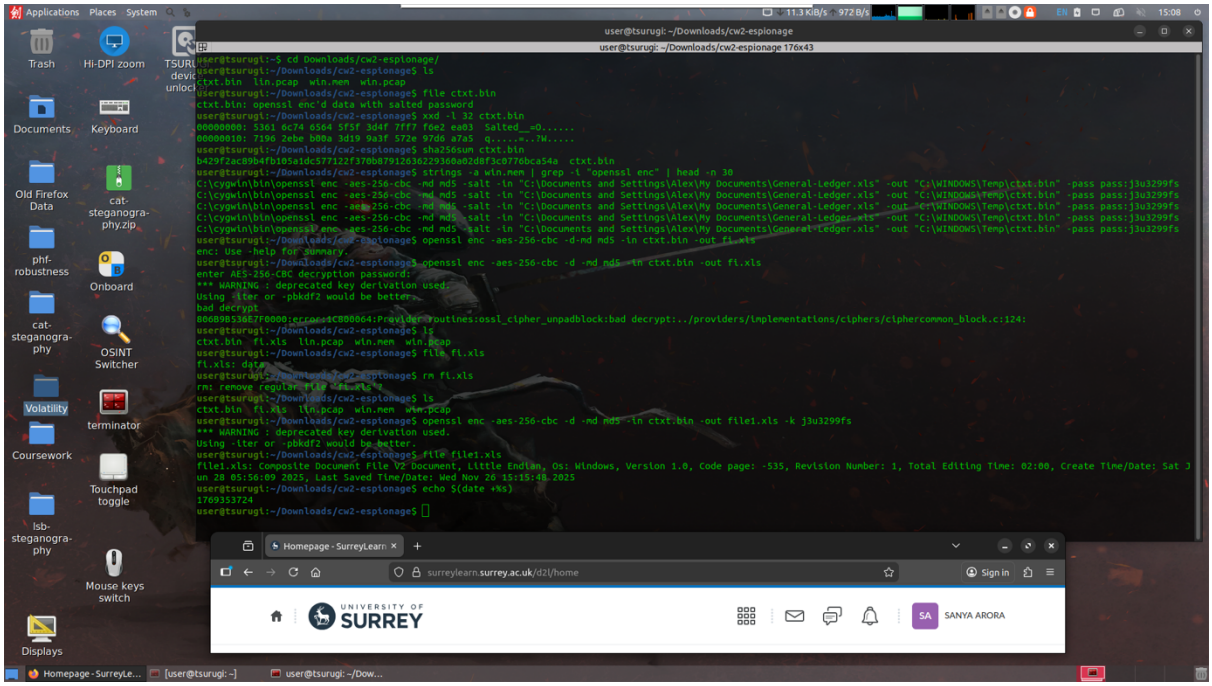
4.2 Steganography Analysis (stego.png)

A forensic extraction was attempted on stego.png to identify any hidden payloads. While the file's naming convention and delivery via the "Great picture" email are high-confidence indicators of steganography, the terminal session artifacts do not show a successfully recovered payload. Consequently, while the file remains highly suspicious, it is classified as a "suspected" rather than a "confirmed" steganographic container based on the available evidence.

4.3 Forensic Conclusions

Despite the limitations of the exfiltrated artifacts, the following conclusions can be drawn:

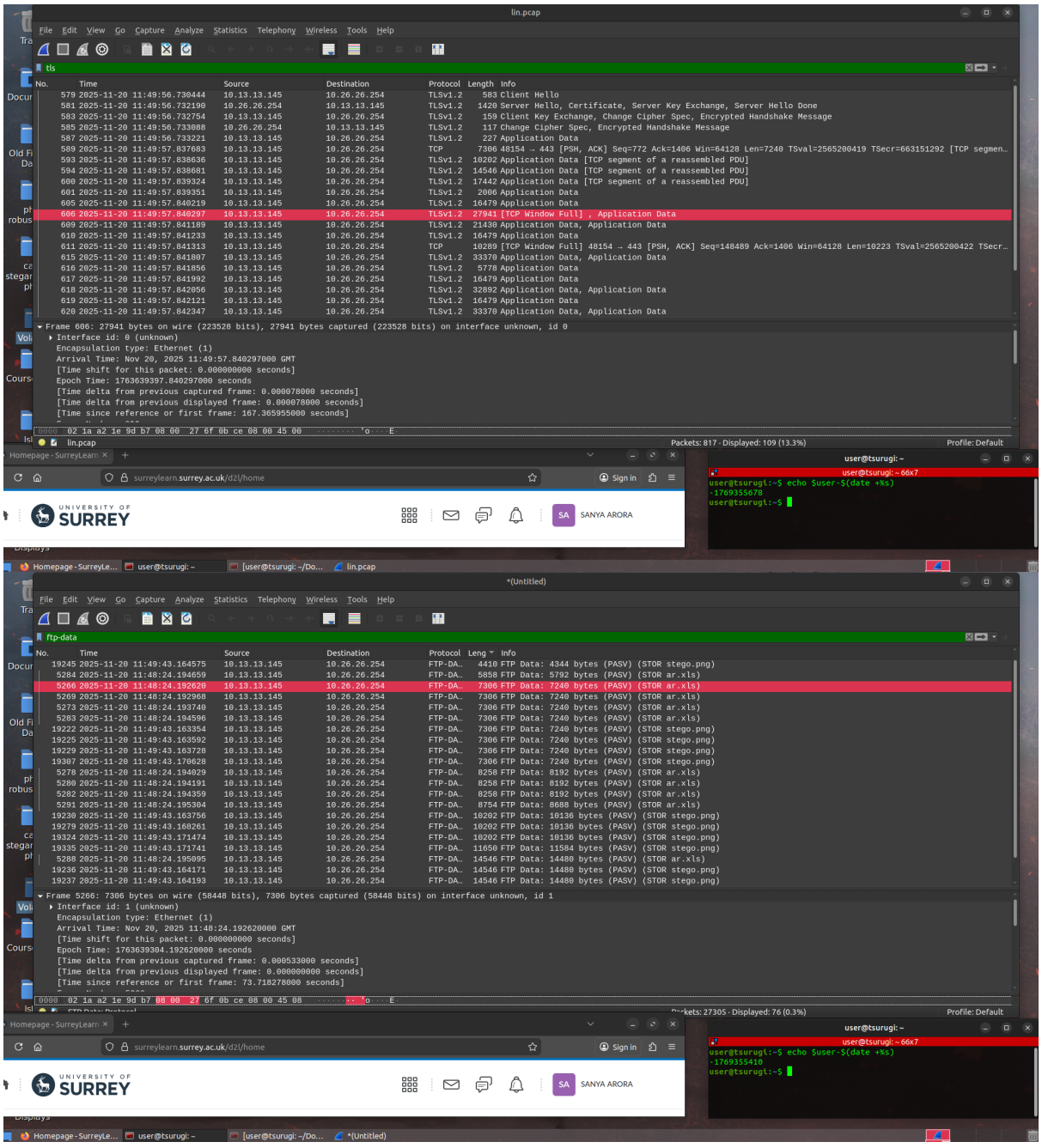
- **Deliberate Obfuscation**: The application of **OpenSSL AES-256-CBC** confirms a targeted effort to hinder forensic inspection and prevent the identification of the stolen data.
- **Value of Volatility**: The successful recovery of file1.xls demonstrates that encryption can be bypassed if the correct parameters are harvested from memory or host artifacts.
- **Espionage Indicators**: Although a payload was not extracted from stego.png, its presence in the exfiltration logs—combined with the use of custom C2 channels—is consistent with the TTPs (Tools, Techniques, and Procedures) of corporate espionage.

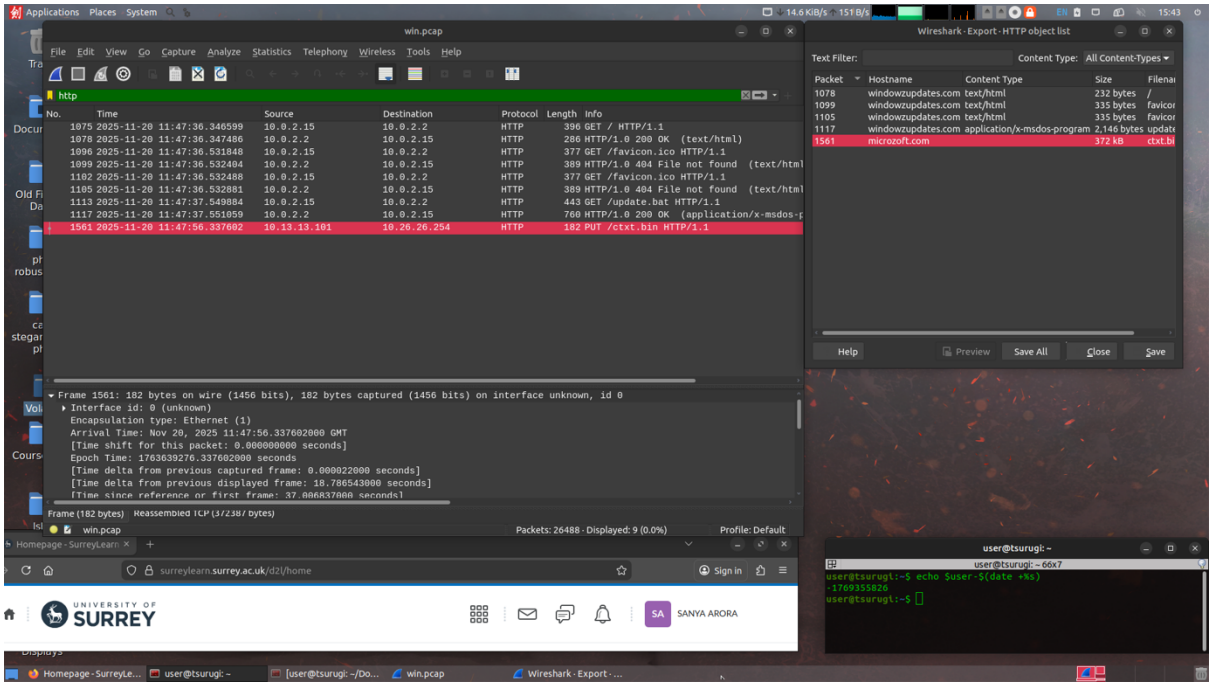


Question 5: Suspicious IP address

Based on the network triage of the Windows and Linux captures, two primary external IP addresses were identified as high-risk Indicators of Compromise (IoCs). These addresses fall outside the internal company subnets (**10.0.2.0/24** and **10.13.13.0/24**) and are directly linked to exfiltration and command-and-control (C2) activity.

- **10.26.26.254**: This is the most critical suspicious endpoint, acting as the primary destination for exfiltrated data. It was observed receiving repeated **FTP STOR** requests for sensitive files, including ar.xls and stego.png, and maintaining persistent **TLSv1.2** encrypted sessions with the Linux server.
- **212.58.237.129**: The Windows workstation initiated **TLS communications** to this IP on port 443. This activity is suspicious as it aligns with the post-compromise window and follows the initial execution of the malicious update.bat file, suggesting it may serve as an secondary C2 or staging server.





(Figures 5.1–5.3)

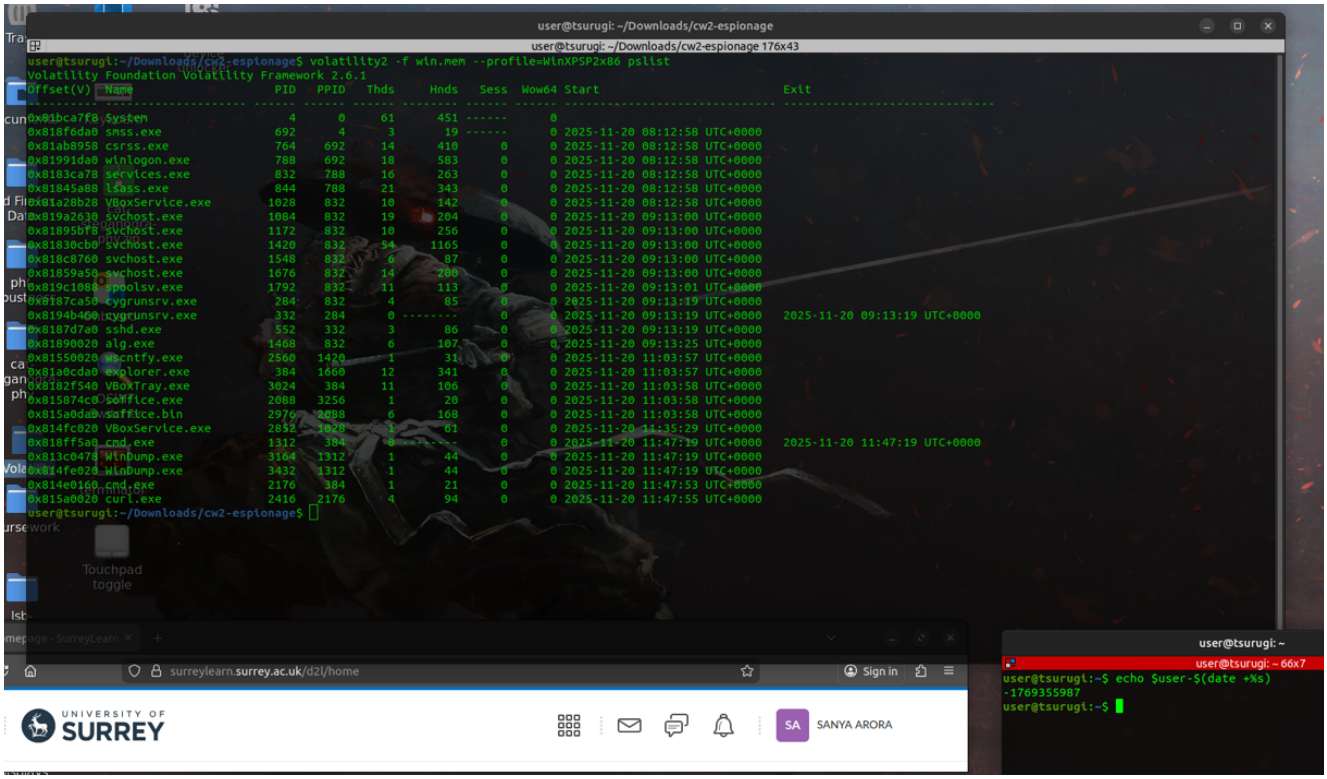
Question 6: Windows version and Office Package

a. Windows Version:

The operating system was identified by profiling the win.mem memory image using the Volatility framework. The tool's profile detection and process listing identify the environment as **Microsoft Windows XP Service Pack 2 (32-bit)**, using the **WinXPSP2x86** profile. This identification is supported by the presence of core Windows XP kernel structures and service processes such as lsass.exe, services.exe, and winlogon.exe.

b. Office Package:

The productivity suite used on the workstation is **OpenOffice.org 3**, rather than Microsoft Office. This is confirmed by the active process list, which shows soffice.bin and soffice.exe running from the directory C:\Program Files\OpenOffice.org 3\program\. Additionally, the Volatility windows.cmdline output shows the user **Alex** interacting with these OpenOffice components.



Question 7: Credentials

During the investigation of the network traffic between the Linux server (10.13.13.145) and the suspicious remote endpoint (10.26.26.254), I identified the exposure of plaintext credentials. The attacker utilized the **File Transfer Protocol (FTP)** to exfiltrate data, which, by default, transmits authentication data without encryption.

Identified Credentials

The following credentials were recovered from the FTP packet stream:

- **Username:** h4cker
- **Password:** s983hu32

Forensic Evidence

The credentials were captured during two distinct login sessions:

1. **Initial Exfiltration:** At **11:48:24**, the attacker successfully authenticated to the remote server to upload the file ar.xls.
2. **Secondary Exfiltration:** At **11:49:43**, the same credentials were used to log back in and upload the suspected steganographic carrier file, stego.png.

The Wireshark traces explicitly show the USER and PASS commands followed by the server response 230 Login successful, confirming these are valid credentials for the remote system.

Security Impact

The exposure of these credentials represents a significant risk to the organization. Because the credentials were sent in cleartext, any entity with monitoring capabilities on the network path could have intercepted them. Furthermore, if these credentials are reused across other company systems or external services, the attacker (or other malicious actors) could leverage them for lateral movement and further unauthorized access.