Case-Study Project

8th August, 2025

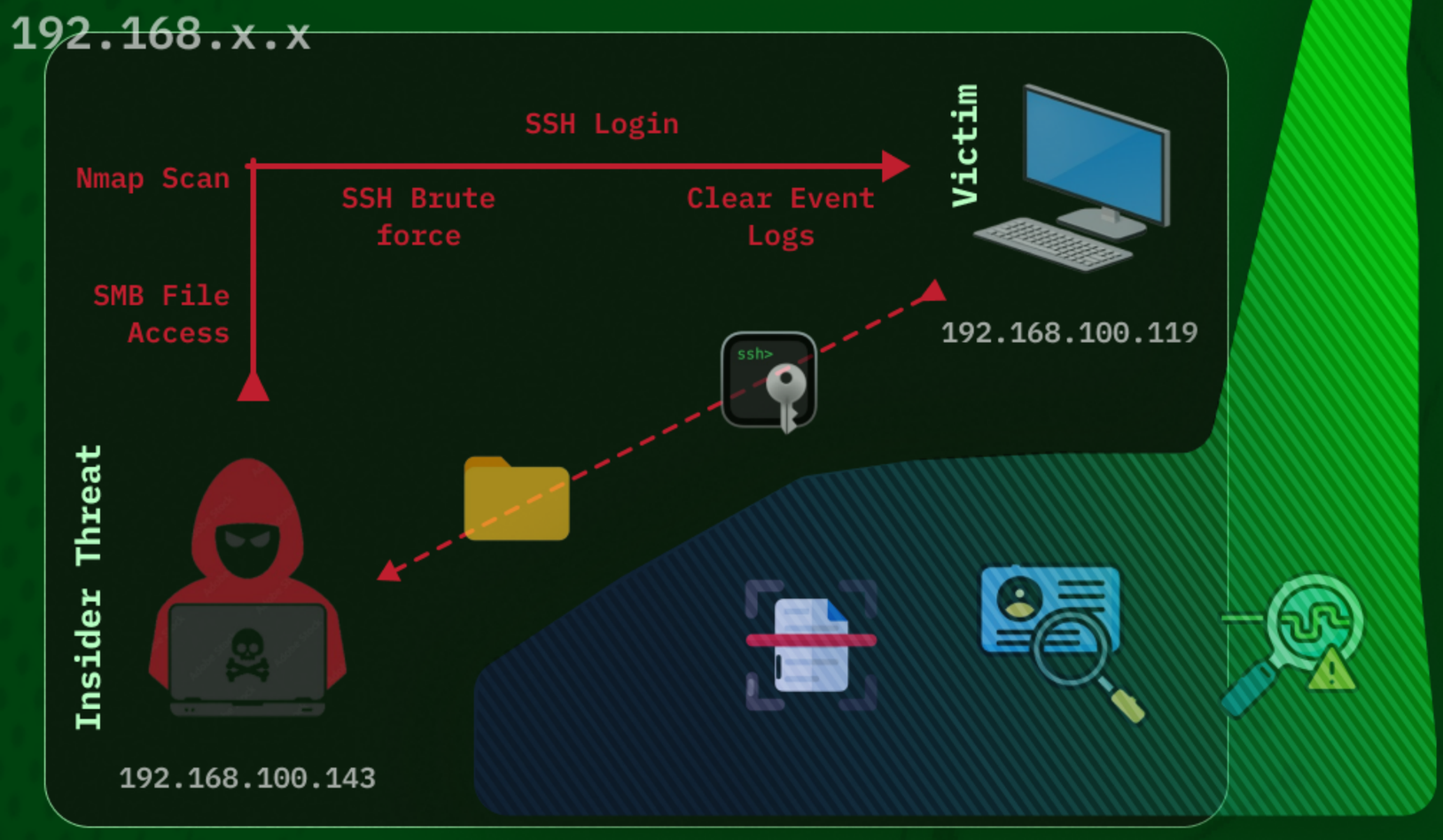
Insider Threat's Lateral Movement

NetworkMiner3.0 & Zeek

GOAL Simulate a stealthy lateral movement attack by a disgruntled insider. Capture the network traffic of the entire operation, and then, acting as a security analyst, use NetworkMiner & Zeek to analyze the capture file.

Simulation | Detection | Identification

OBJECTIVE To simulate the intrusion as a threat actor, identify the actions, extract forensic evidence, and document the complete attack chain



Security Analyst

Athar Imran

Insider Threat's Lateral Movement Simulation, Detection & Identification using NetworkMiner3.0 & Zeek

---

The Goal

Simulate a stealthy lateral movement attack by a disgruntled insider. Capture the network traffic of the entire operation, and then, acting as a security analyst, use NetworkMiner & Zeek to analyze the capture file.

The Objective

To simulate the intrusion as a threat actor, identify the actions, extract forensic evidence, and document the complete attack chain.

Case-Study Project

Author: Athar Imran
Role: Security Analyst
Date: August 08, 2025
Organization: Independent Project

The silent file access is the real threat; the noisy brute-force is just the analyst's lucky break.

Key Tools: Zeek, NetworkMiner, Nmap, Hydra, tcpdump, smbclient

Contents

Introduction

"68% of data breaches involved a 'non-malicious human element', such as human error or falling for social engineering." - Verizon Data Breach Investigations Report

The most dangerous threats often come from within. A malicious insider, armed with legitimate access, can operate undetected for extended periods, their actions blending in with normal network activity.

This project simulates a realistic insider threat scenario, demonstrating how a seemingly benign act of data access can escalate into a full-scale lateral movement attack, designed to compromise additional systems and exfiltrate sensitive information.

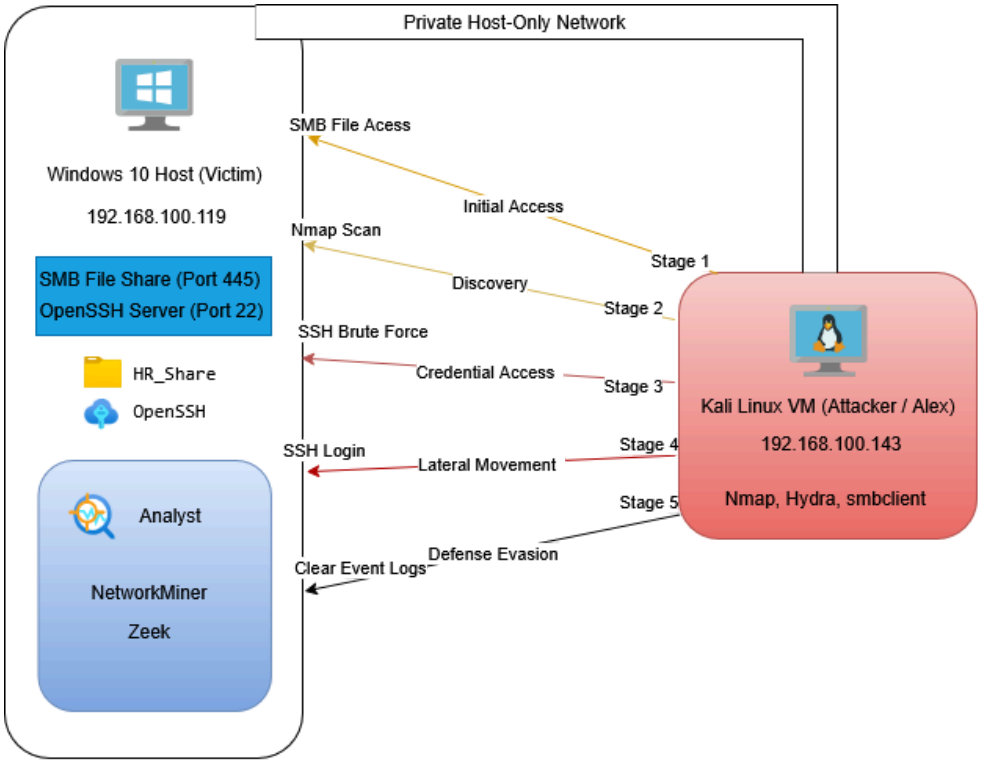
Using a virtualized environment, this guide systematically walks through the stages of a simulated attack, from subtle reconnaissance and active scanning to a successful brute-force and post-exploitation cleanup.

The core of this exercise lies in the detection and analysis phase, where we use open-source security tools like **Zeek** and **NetworkMiner** to piece together the evidence, create a forensic timeline, and highlight the critical role of network and host-based monitoring in catching sophisticated internal threats.

Overview

This project simulates a realistic scenario where an insider, an employee named **Alex**, exploits their legitimate network access to launch a multi-stage attack. Initially, Alex's actions fly under the radar, as their activity appears to be normal file access. However, their motive to find sensitive financial data quickly escalates, leading to aggressive lateral movement attempts.

The primary objective is to demonstrate how a series of low-risk, unflagged events can precede a high-priority security alert, which then serves as the trigger for a full-scale forensic investigation.



Key Elements:

- **The Insider:** Alex, a disgruntled HR employee with standard privileges to a shared HR folder.
- **The Motive:** Financial gain from leaking sensitive corporate documents.
- **The Trigger:** The moment Alex shifts from normal file access to aggressive port scanning and brute-force attacks from their workstation, a clear sign of malicious intent and an attempt to pivot to other systems.

The Technical Environment:

- **Victim Machine (Target): Windows 10 Host**, representing a corporate server or a high-value workstation containing sensitive data.
- **Attacker Machine:** A **Kali Linux VM**, which simulates the insider's compromised workstation and is used to launch the attack.

Phase 0 - Environment Setup & Tools:

First, let's prepare our lab. The key is to ensure both components (Windows 10 Host and Kali VM) can communicate with each other on the same network.

- Windows 10 will serve as the main host.
- Kali Linux VM will be deployed on the host.

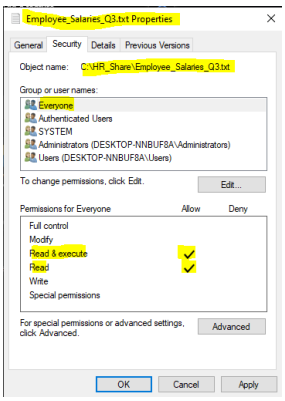
0.1 - Configure Networking:

Ensure your Kali VM can communicate with the Windows 10 Host.

- In the VM manager, set the VM's network adapter to **NAT Network** or **Host-Only Adapter**. This creates a private network between the host and the VM.
- **Find IP Addresses:** We'll need both IPs.
    - On **Windows 10 Host**, open Command Prompt and type: `ipconfig`
    - On **Kali Linux VM**, open a terminal and type: `ip a`
    - Note these IP addresses. For this guide, let's assume:
        - Windows 10 (Victim) IP: `192.168.100.119`
        - Kali Linux (Attacker) IP: `192.168.100.143`

0.2 - Preparing Windows 10 as the Victim Machine:

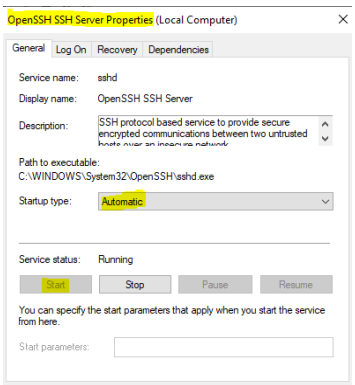
1. **Create a "Sensitive" File & Share:**
    - Create a folder on C: drive named `C:\HR_Share`.
    - Inside this folder, create a text file named `Employee_Salaries_Q3.txt`. Write some dummy text inside like "Confidential Salary Data".
    - Right-click the `HR_Share` folder -> Properties -> Sharing -> Advanced Sharing.
    - Check "Share this folder" and click "Permissions." Give the "Everyone" group "Read" access.



This simulates a basic, perhaps poorly configured, file share.

2. **Enable a Remote Service for Lateral Movement:**

    We'll enable SSH, as it's a common vector.



- Go to Windows Settings -> Apps -> Optional features -> Add a feature.
- Find and install **OpenSSH Server**.
- Once installed, open the "Services" app. Find "OpenSSH SSH Server," start it, and set its startup type to "Automatic."

0.3 - Install Tools on Kali Linux:

Update package list first: `sudo apt update`

- **Zeek:** `sudo apt install zeek`
- **NetworkMiner:** NetworkMiner isn't in the default repositories.
  - Go to the official NetworkMiner website.
  - Download the latest free version.
  - Extract the downloaded `.zip` file. Run it from the extracted folder.
- **Attack Tools:** These are usually pre-installed on Kali.
  - `nmap` (for scanning)
  - `hydra` (for brute-forcing)
  - `smbclient` (for accessing Windows shares)

Phase 1 - The Attack Simulation:

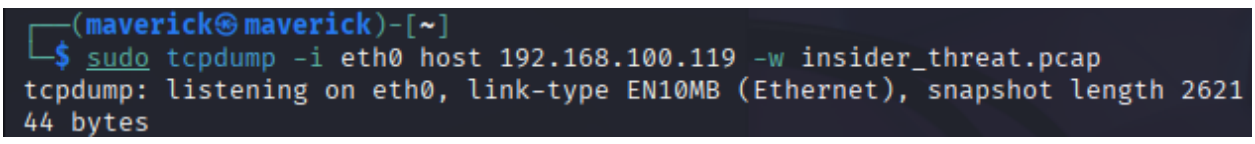
Now, we'll execute the attack chain from the Kali VM while capturing all the network traffic.

1.0 - Packet Capture Start:

On **Kali VM**, open a terminal. We need to capture all traffic between Kali and the Windows host.

```
sudo tcpdump -i eth0 host 192.168.56.101 -w insider_threat.pcap
```

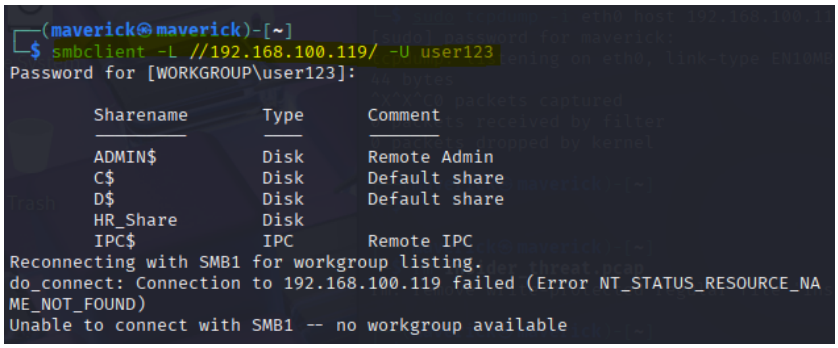
Leave this terminal running. All subsequent actions will be captured in the `insider_threat.pcap` file.



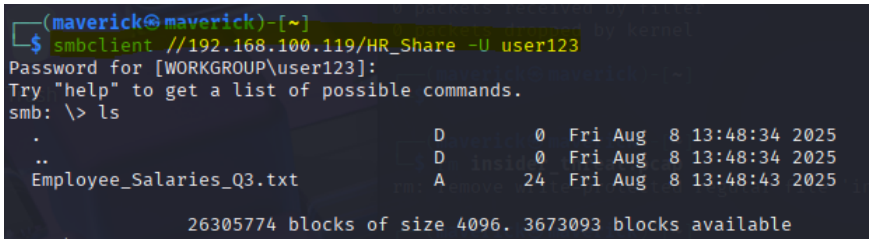
Stage 1: Initial "Normal" Activity (Reconnaissance):

From a **new terminal** on the Kali VM, Alex accesses the HR share at an "odd hour" to look for information. This simulates initial, low-profile snooping.

`smbclient -L //192.168.100.119/`
This command lists the shares on the Windows machine. It will prompt for the Windows account password. Just press Enter.

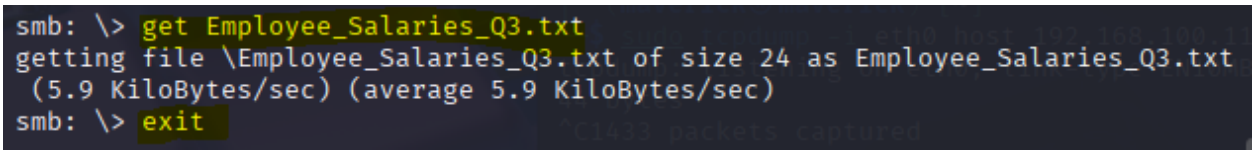


Now, connect to the share and download the file.
`smbclient`



`//192.168.100.119/HR_Share`

Once connected, we'll get an smb: `\> prompt`. Type:
`get Employee_Salaries_Q3.txt xit`

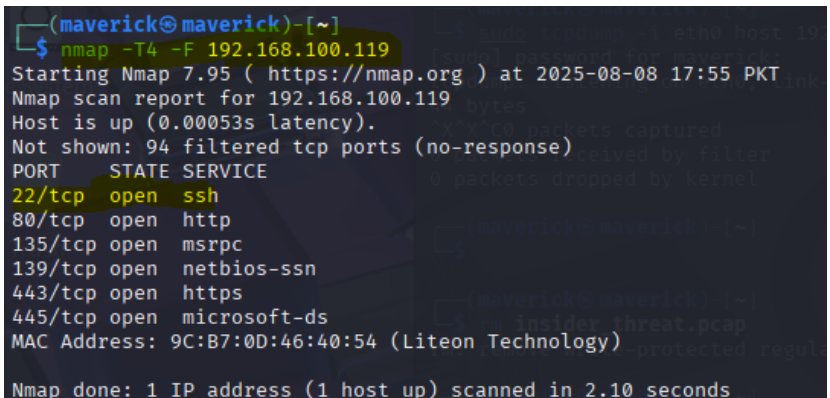


This action is suspicious but might not trigger a high-priority alert on its own.

Stage 2: Active Scanning (Lateral Movement Prep):

Alex didn't find what he wanted. He now actively scans the Windows machine for other open services to exploit. **This is the noisy activity that should trigger an alert.**

Run a quick scan on the Windows host to find open ports.



```
nmap -T4 -F 192.168.100.119
```

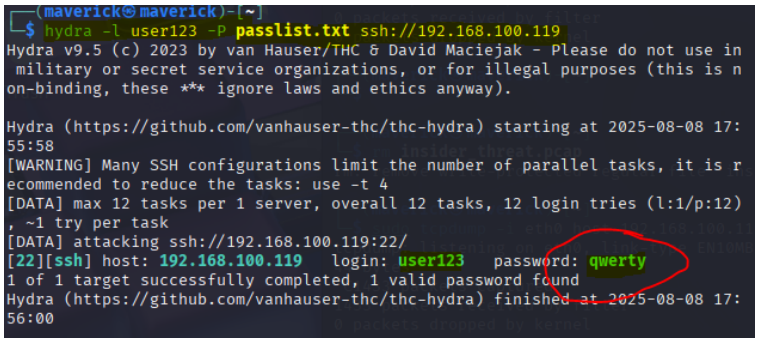
The scan reveals that port 22 (SSH) is open.

Stage 3: Brute-Force Attack (Lateral Movement):

Alex will now try to guess the password for the Windows user account via SSH.

- First, create a small password list. `echo "Password123" > pass.list echo "admin" >> pass.list echo "YourActualWindowsPassword" >> pass.list` *(Add real password here so the attack succeeds)*
- Launch the brute-force attack with Hydra.

```
hydra -l user123 -P pass.list ssh://192.168.100.119
```

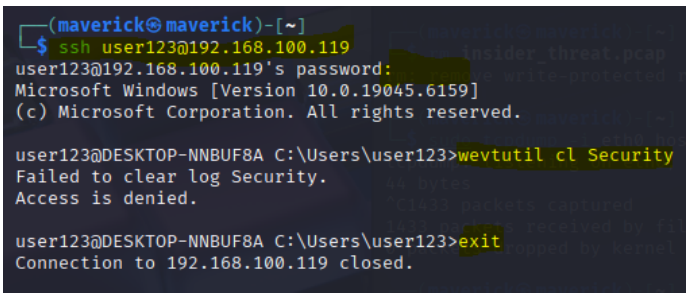


Hydra will rapidly try the passwords, and one will succeed. **This is a major indicator of an attack.**

Stage 4: Post-Exploitation & Covering Tracks:

Alex now has access.

- **Log in via SSH:** `ssh user123@192.168.100.119` (Enter the correct password when prompted)
- **Cover Up:** Once logged in to the Windows machine via the SSH session, Alex clears the security event logs to hide the brute-force attempts. `wevtutil cl Security`
- **Exit the SSH session:** `exit`



1.5 - Packet Capture Stop:

Go back to the first Kali terminal (the one running `tcpdump`) and press **`Ctrl + C`** to stop the capture. We now have the `insider_threat.pcap` file containing all the evidence.

```
┌──(maverick㉿maverick)-[~]
└─$ sudo tcpdump -i eth0 host 192.168.100.119 -w insider_threat.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 2621
44 bytes
^C1433 packets captured
1433 packets received by filter
0 packets dropped by kernel
```

Phase 2 - Detection & Analysis:

2.0 - The scenario:

Our Security Operations Center (SOC) gets an alert. Let's assume Zeek generated a notice for `Security::Failed log dump attempt`. Our job as analysts is to investigate.

2.1 - Analyze with Zeek:

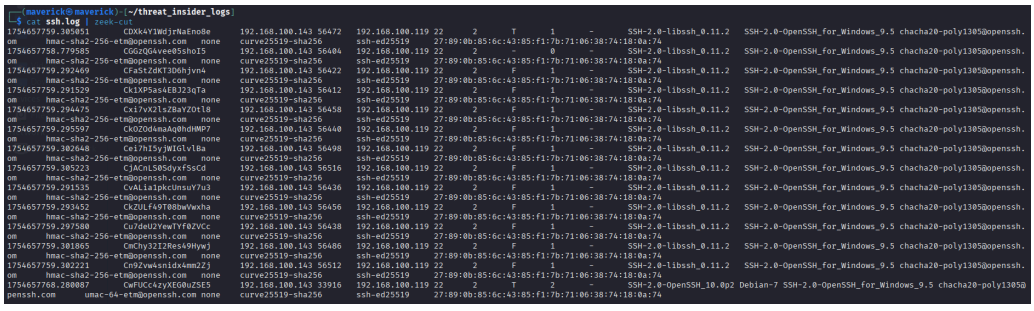
On the Kali VM, process the PCAP file with Zeek.

```
zeek -C -r insider_threat.pcap
```

This command will generate several `.log` files in the current directory. These are tab-separated text files, perfect for analysis.

```
┌──(maverick㉿maverick)-[~/threat_insider_logs]
└─$ ls
conn.log      insider_threat.pcap  smb_files.log
dce_rpc.log   ntlm.log             smb_mapping.log
files.log     packet_filter.log    ssh.log
```

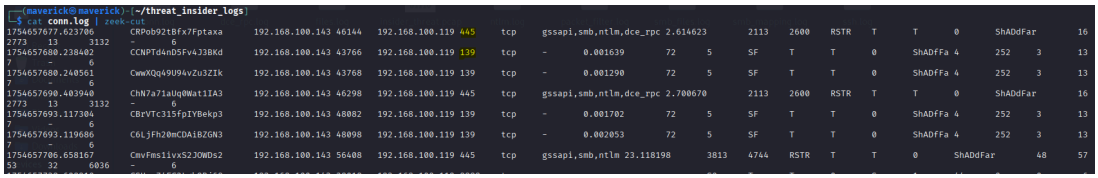
- **`ssh.log`**: `cat ssh.log | zeek-cut`
  - This log will show dozens of failed login attempts from the attacker's IP, followed by one successful authentication. This is damning evidence of the brute-force attack.



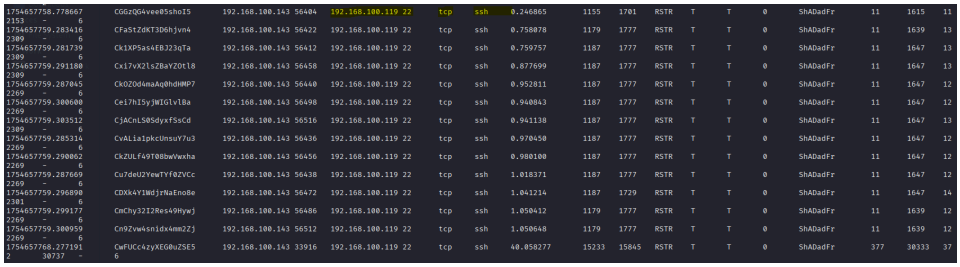
- **smb_files.log**: `cat smb_files.log | zeek-cut`
  - Here we will see the evidence of the initial activity: Alex's IP accessing `Employee_Salaries_Q3.txt` from the `HR_Share`. This helps establish the timeline *before* the aggressive attack.

```
┌──(maverick㉿maverick)-[~/threat_insider_logs]
└─$ cat smb_files.log | zeek-cut
1754657714.360214       CmvFms1ivxS2JOWDs2      192.168.100.143 56408   192.168.100.119 445     -       S
MB::FILE_OPEN   \\\\192.168.100.119\\HR_Share   <share_root>    0       -       1754642914.395482       1
754657384.486122        1754642889.784823       1754655243.443528
1754657721.737217       CmvFms1ivxS2JOWDs2      192.168.100.143 56408   192.168.100.119 445     -       S
MB::FILE_OPEN   \\\\192.168.100.119\\HR_Share   Employee_Salaries_Q3.txt        24      -       1754
642923.805453   1754655243.637542       1754642913.035179       1754655243.443528
```

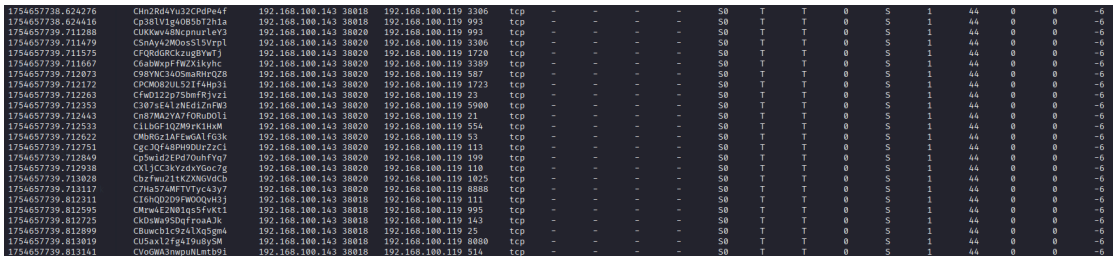
- **conn.log**: `cat conn.log | zeek-cut`
  - This log provides a complete overview of all connections:
    - Port 445 (SMB) traffic.



    - A flurry of port 22 (SSH) connections.



    - Port scanning activity from Nmap might show up as connections to many different ports in a short time.



- **files.log**: `cat files.log | zeek-cut`

- This is the same as smb_files.log, but it also shows all other files accessed (which is not in our case):

```
┌──(maverick㉿maverick)-[~/threat_insider_logs]
└─$ cat files.log | zeek-cut
1754657721.740444       FDgj8T2B2h7cPldi3a      CmvFms1ivxS2JOWDs2      192.168.100.143 56408   192.
168.100.119     445     SMB     0       (empty) text/plain      Employee_Salaries_Q3.txt        0.00
0000    T       F       24      24      0       0       F       -       -       -       -       -   -
-
```

2.2 - Investigate with NetworkMiner:

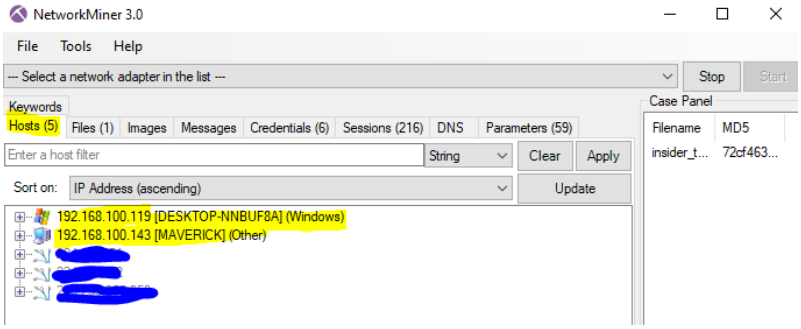
Zeek gave us the logs; NetworkMiner will give us the visual evidence and extracted files.

Open the networkminer and import the `insider_threat.pcap` file.

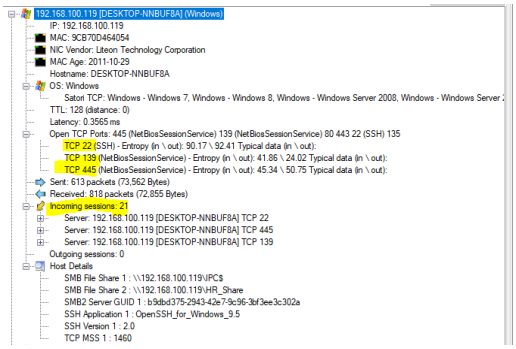
Below is the thorough investigation:

Hosts Tab:

- We'll see two primary hosts: your Kali VM and your Windows Host.



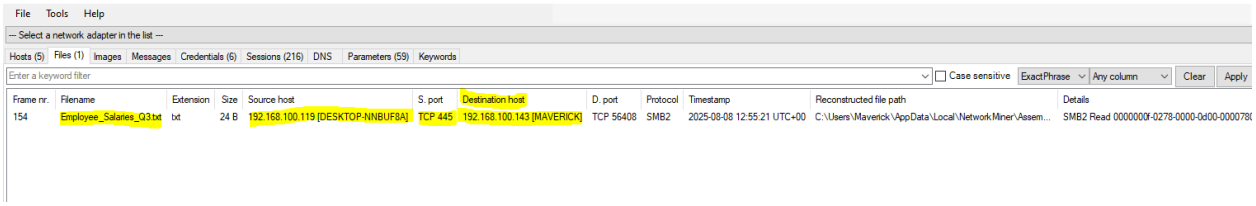
- Clicking on the Windows Host (`192.168.56.101`) will show all open ports detected, incoming sessions, and a fingerprint of its OS.



Files Tab: (Key Evidence):

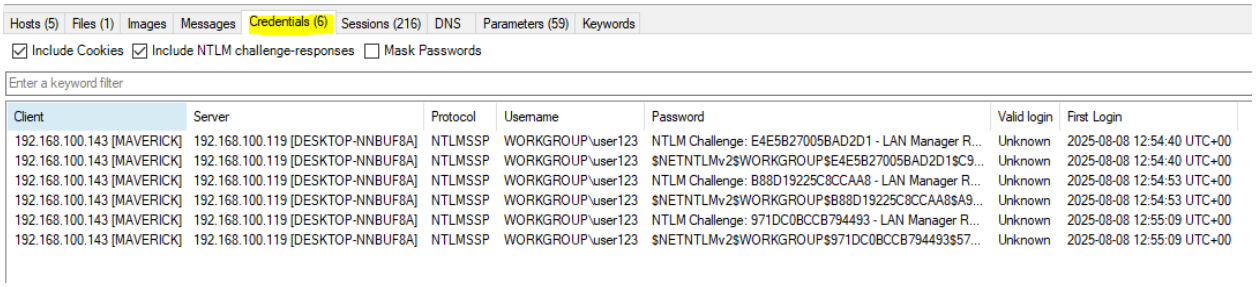
- NetworkMiner automatically reassembles and extracts files transferred over the network.

- We can see the **`Employee_Salaries_Q3.txt`** file listed! Right-click and open it to see the contents. This is irrefutable proof that the attacker downloaded this specific file.



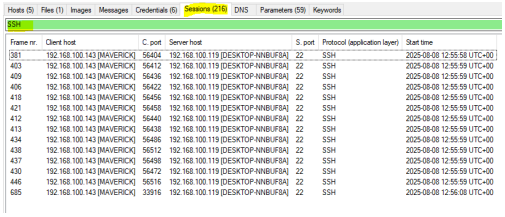
Credentials Tab:

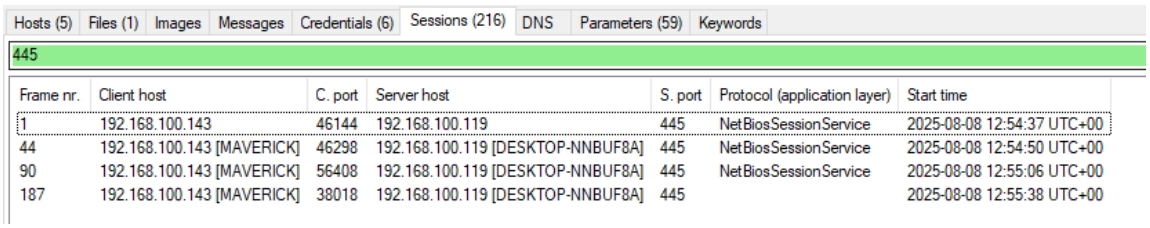
- NetworkMiner will extract any credentials sent in cleartext.
- We can likely see the SMB username used to access the share. You might not see the SSH password due to encryption, but the attempt itself is logged.



Sessions Tab:

- We can see every TCP session. It can be filtered by port (445 for SMB, 22 for SSH) to inspect the raw communication flow between the attacker and victim.





Phase 3 - Incident Report & Findings:

Here is the final report on this incident:

3.1 - Executive Summary:

On August 8, 2025, an automated alert was triggered for a suspected SSH brute-force attack originating from IP `192.168.100.143` and targeting server `192.168.100.119`. Investigation revealed this was part of a multi-stage attack by an insider. The actor first accessed a sensitive file on an HR share, then scanned the target for open services, and successfully gained access via SSH. The actor attempted to cover their tracks by clearing security logs post-compromise. All malicious activity originated from the host assigned to employee "Alex".

3.2 - Timeline of Attack & MITRE ATT&CK Mapping:

Time (Simulated)	Action	Evidence	MITRE ATT&CK Tactic	MITRE ATT&CK Technique
22:05	File Share Access	smb_files.log, Extracted file in NetworkMiner	Reconnaissance	T1595.001: Active Scanning: Scanning IP Blocks
22:10	Port Scan	conn.log showing probes, Nmap fingerprint in NetworkMiner	Reconnaissance	T1046: Network Service Scanning
22:12	SSH Brute-Force	ssh.log, notice.log (Password_Guessing)	Credential Access	T1110.001: Brute Force: Password Guessing
22:14	Successful SSH Login	ssh.log showing successful auth	Lateral Movement	T1021.004: Remote Services: SSH
22:15	Clear Security Logs Atempt	Command in SSH session (inferred)	Defense Evasion	T1070.001: Indicator Removal: Clear Windows Event Logs

Phase 4 - Analyst Notes & Detection Gaps:

4.1 - What NetworkMiner Excelled At:

- **File Extraction:** Automatically carving the `Employee_Salaries_Q3.txt` file from SMB traffic provided concrete proof of data access.
- **Visualization:** The `Hosts` tab provided an immediate, clear picture of the actors and assets involved.
- **OS Fingerprinting:** Identifying the victim OS via passive analysis is an extremely useful context.

4.2 - Detection Gaps & Limitations:

- **Encrypted Traffic:** NetworkMiner could not see *what commands were typed* inside the successful SSH session because the payload was encrypted. We only know a session was established.
- **Log Clearing:** The act of clearing the logs (`wevtutil cl Security`) happened *inside* the encrypted SSH tunnel. We can't see the command in the PCAP. We infer this action happened because a successful login was followed by a loss of host-level logs. **This highlights the need to correlate network evidence (PCAP) with host evidence (or lack thereof).**

4.3 - Future Improvements & Automation:

- **Real-time Detection:** This manual analysis is slow. A **SIEM** (Security Information and Event Management) system like **Splunk**, **ELK Stack**, or **Wazuh** would provide real-time alerting.
- **Integration:** Zeek can feed its logs directly into a SIEM in real-time. An analyst could then get an alert (`SSH Brute Force`) and immediately pivot to a dashboard showing the preceding SMB access and Nmap scan from the same IP, cutting investigation time from hours to minutes.
- **Host-based Monitoring:** Tools like Wazuh or Sysmon on the Windows host would have generated a high-priority alert the moment `wevtutil cl Security` was executed, catching the cover-up attempt even though it was in an encrypted channel.

About me: Find me on: Linkedin | Github