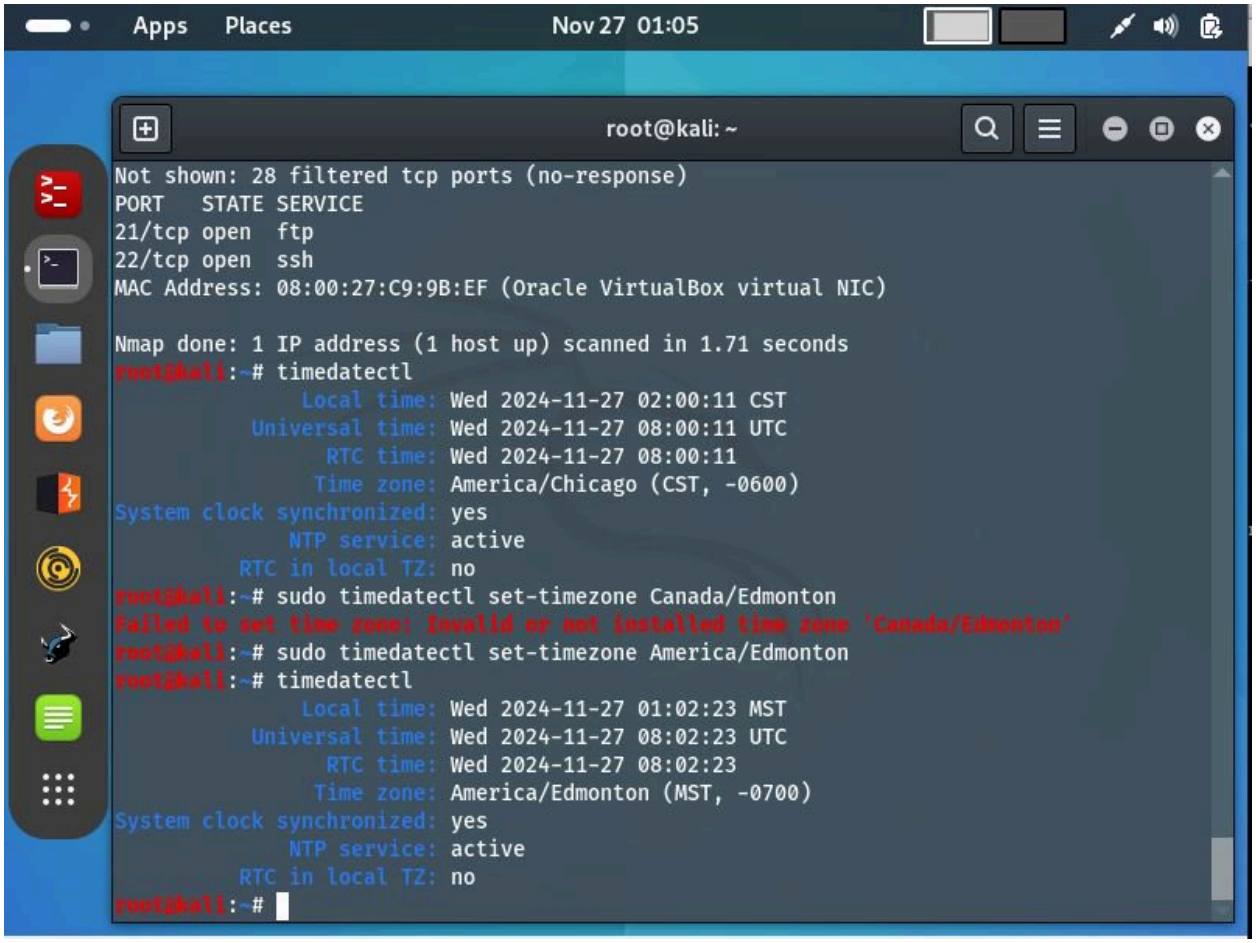
Tab 1

DFIR Final project

Introduction

The purpose of this project was to compromise a vulnerable machine, conduct post-exploitation activities, and perform a forensic analysis to reconstruct the attack. The scenario simulated a real-world incident where an attacker breaches a system, manipulates it, and leaves traces that need to be uncovered by forensic techniques.

A network diagram was created to illustrate the environment, including the attacker and target IP addresses and the network topology. This report documents the steps taken, tools used, and findings from disk, memory, and network analyses.



Attack Description and Time Synchronization Analysis

Step 1: Initial Reconnaissance Using Nmap

In the provided screenshot, the **Nmap** tool was used to perform reconnaissance on the vulnerable machine. The scan revealed the following open ports:

- **Port 21 (FTP):** Indicates an FTP server running on the target machine.
- **Port 22 (SSH):** Indicates an SSH service available for potential exploitation.

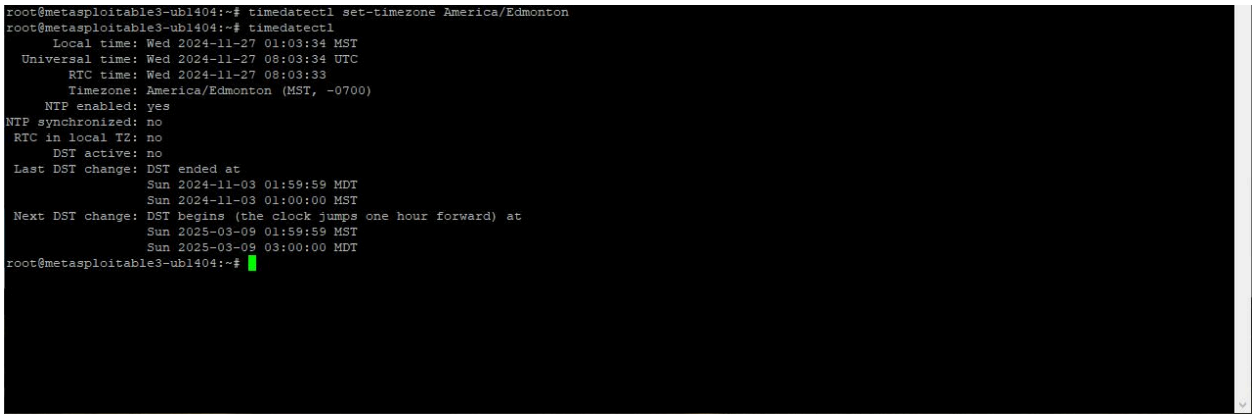
This information is crucial as it highlights the exposed services that could be used to gain unauthorized access to the machine.

Step 2: Time Synchronization Validation

The **timedatectl** command was used to inspect and modify the system time configuration. This step ensures that the forensic timeline analysis aligns with the system's local time. Key observations include:

- **Local Time:** Initially set to CST (America/Chicago, UTC -6:00).
- **Time Zone Update:** Attempted to change to *America/Edmonton (MST, UTC -7:00)*.
    - The first attempt failed due to an invalid timezone name, but the second attempt succeeded, setting the time zone to MST.
- **RTC Time:** Verified against local time and synchronized successfully.

This adjustment is essential for maintaining accuracy in log analysis and correlating forensic events across different time zones.



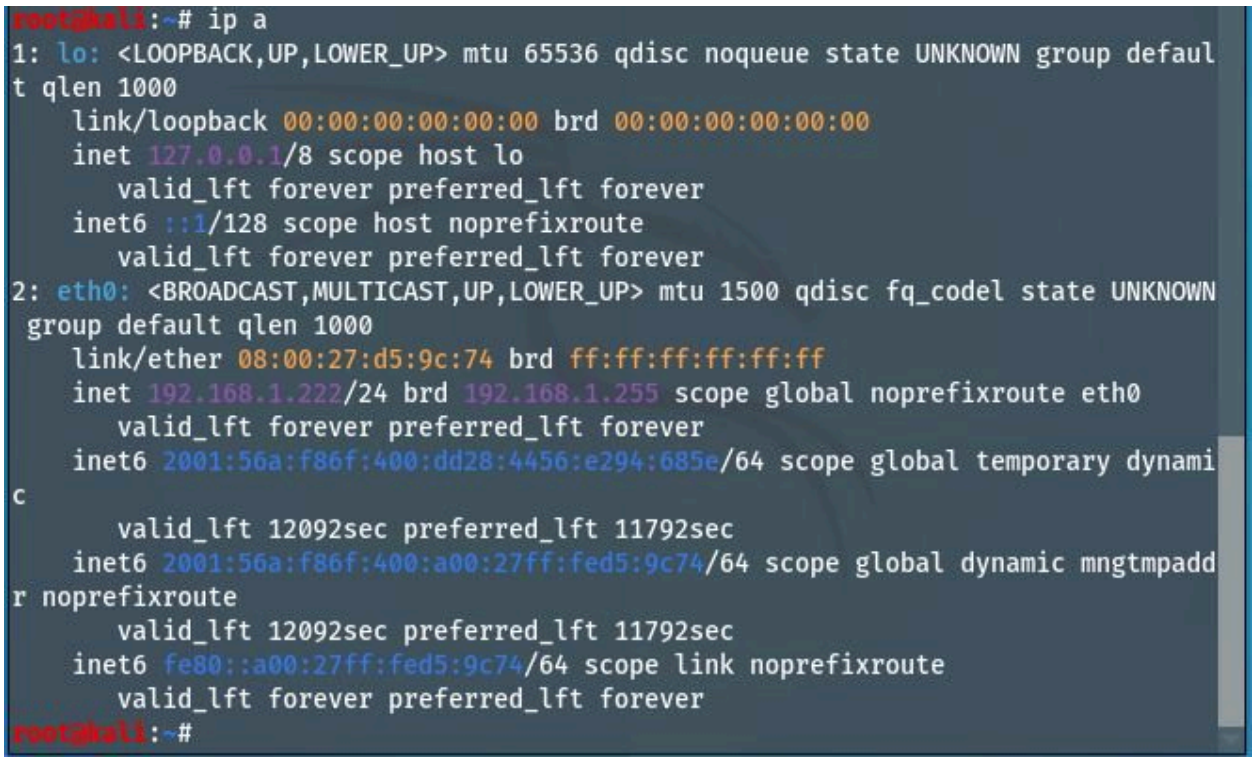
Post-Exploitation Activity and Time Configuration on Target Machine

Step 3: Verification of Time Settings on the Target System

The second screenshot shows a similar configuration performed on the compromised machine (Metasploitable). Key details:

- The **timedatectl** command was executed to confirm the local time and ensure it matches the attacker's settings.
- Observations:
    - **NTP Status:** Not synchronized, suggesting the machine was not actively syncing time with a network time protocol server.
    - **DST Details:** The daylight saving time (DST) schedule was displayed, indicating the most recent and upcoming DST changes.

Ensuring time synchronization is a critical step in maintaining a cohesive forensic timeline. This aids in correlating events from network captures, memory dumps, and system logs during the investigation.



Network Analysis and Addressing Report

Network Interface Configuration

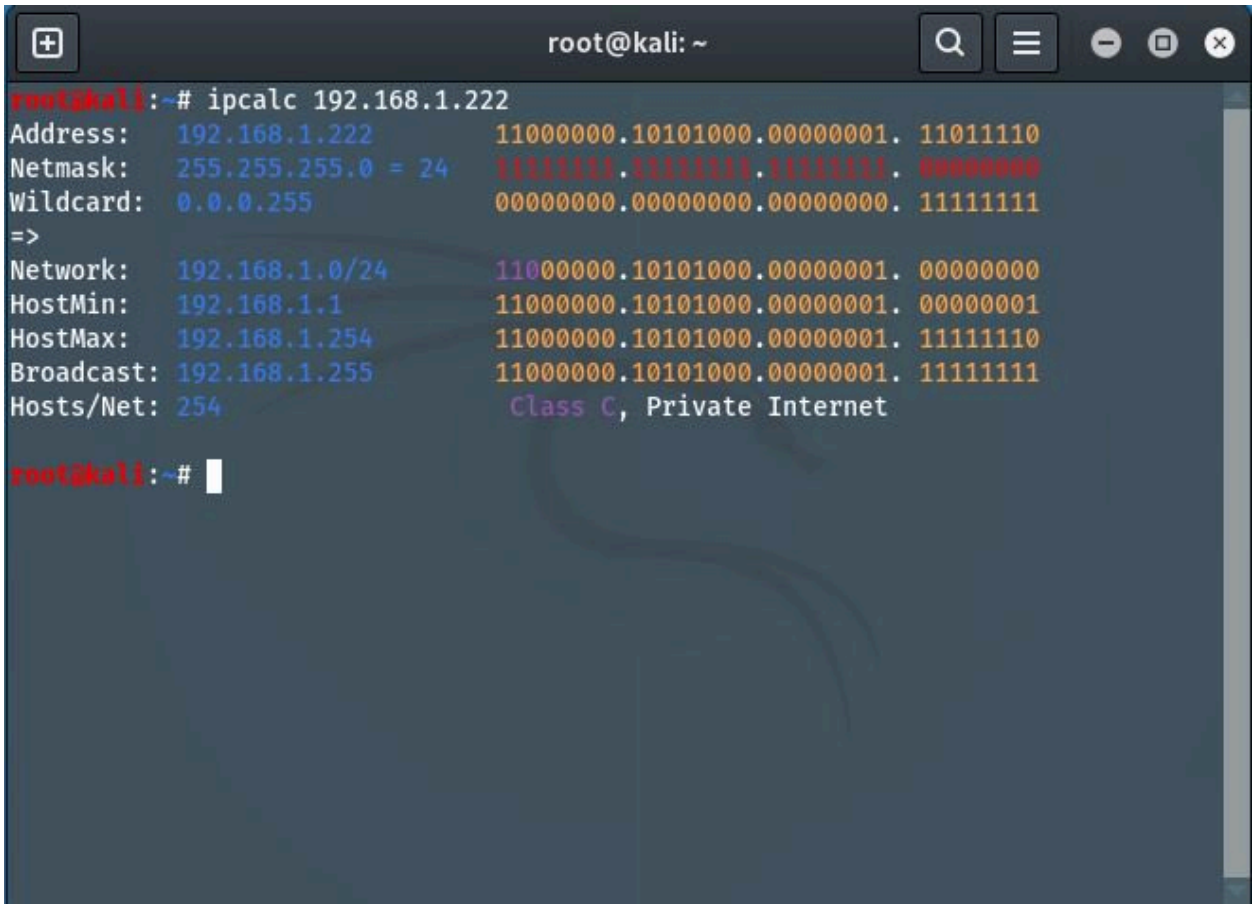
The first screenshot captures the output of the `ip a` command, which provides a detailed view of the network interfaces and their associated IP configurations on the system.

1. **Loopback Interface (`lo`):**

- The loopback interface (`lo`) is configured with the IPv4 address `127.0.0.1/8`, which is reserved for local communication within the host.
- The associated IPv6 address is `::1/128`, which serves the same purpose for IPv6 communications.

2. **Ethernet Interface (`eth0`):**
   - The hardware address (MAC) is `08:00:27:d5:9c:74`.
   - The interface is assigned an IPv4 address of `192.168.1.222/24`. This subnet mask (`/24`) indicates that the network has 256 total addresses (254 usable), ranging from `192.168.1.1` to `192.168.1.254`.
   - The broadcast address for the subnet is `192.168.1.255`, which is used to send messages to all devices in the subnet.
   - The interface is also assigned an IPv6 address:
     - A temporary dynamic address: `2001:56a:f86f:400:dd28:4456:e294:688e/64`.
     - A global dynamic address: `2001:56a:f86f:400:a00:27ff:fed5:9c74/64`.
   - These IPv6 addresses have a `/64` subnet prefix, which is standard for most IPv6 networks, allowing for a large address space.

****Exploitation start*******



IP Address Calculation

The second screenshot shows the use of the `ipcalc` tool with the IP address `192.168.1.222` and subnet mask `/24`.

1. **Details:**
   - **IP Address:** `192.168.1.222`
   - **Subnet Mask:** `255.255.255.0`, equivalent to `/24`.
   - **Wildcard Mask:** `0.0.0.255`, representing the range of host bits.
   - **Network Address:** `192.168.1.0`, identifying the network's base address.
   - **Broadcast Address:** `192.168.1.255`, used to address all devices in the subnet.
   - **Host Range:** The first usable IP address is `192.168.1.1`, and the last usable is `192.168.1.254`. The total number of usable hosts is 254.
2. **Class and Designation:**
   - The IP address belongs to **Class C**, which is commonly used in private networks.
   - The subnet falls within the private address space as defined by RFC 1918.

```
root@kali:~# nmap 192.168.1.0/24 -A -p 22 --open
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-27 03:25 MST
Nmap scan report for 192.168.1.248
Host is up (0.00059s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol
2.0)
| ssh-hostkey:
|   1024 2b:2e:1f:a4:54:26:87:76:12:26:59:58:0d:da:3b:04 (DSA)
|   2048 c9:ac:70:ef:f8:de:8b:a3:a3:44:ab:3d:32:0a:5c:6a (RSA)
|   256 c0:49:cc:18:7b:27:a4:07:0d:2a:0d:bb:42:4c:36:17 (ECDSA)
|_  256 a0:76:f3:76:f8:f0:70:4d:09:ca:e1:10:fd:a9:cc:0a (ED25519)
MAC Address: 08:00:27:C9:9B:EF (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 o
pen and 1 closed port
Aggressive OS guesses: Linux 3.10 - 4.11 (97%), Linux 3.16 - 4.6 (97%), Linux 3.
2 - 4.9 (97%), Linux 4.4 (97%), Linux 3.13 (94%), Linux 4.2 (92%), OpenWrt Chaos
 Calmer 15.05 (Linux 3.18) or Designated Driver (Linux 4.1 or 4.4) (91%), Linux
4.10 (91%), Android 5.0 - 6.0.1 (Linux 3.4) (91%), Linux 2.6.32 (91%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Steps Performed:

1. Network Scanning Using Nmap

Command Executed: (css)
`nmap 192.168.1.0/24 -A -p 22 --open`

- **Explanation**:
  - **`192.168.1.0/24`**: Scans the entire subnet (all hosts in the range).
  - **`-A`**: Enables OS detection, version detection, script scanning, and traceroute.
  - **`-p 22`**: Focuses on scanning port 22, commonly used by the SSH service.
  - **`--open`**: Displays only hosts where the port is actively open.
- **Findings**:
  - **Target Host**: `192.168.1.248`.
  - **Open Port**: Port 22 (SSH).
  - **SSH Version**: OpenSSH 6.6.1p1 running on Ubuntu 12.04.5 LTS.
  - **Key Exchange Algorithms Identified**:
    - **DSA (1024 bits)**.
    - **RSA (2048 bits)**.
    - **ECDSA (256 bits)**.

        - **ED25519 (256 bits)**.
    - **MAC Address**: Indicates the host is a virtual machine running on **Oracle VirtualBox**.
    - **Operating System Guess**: Various Linux kernel versions (3.10–4.11), with a confidence level of 97%.
    - **Warnings**: OSSCAN results flagged reliability concerns due to insufficient open/closed ports.

Potential Vulnerability:

- The SSH service (OpenSSH 6.6.1p1) on the target system is outdated and might be susceptible to known vulnerabilities (e.g., CVE-2015-5600).
- No advanced configuration or IP restrictions were identified, indicating the SSH service is broadly exposed.

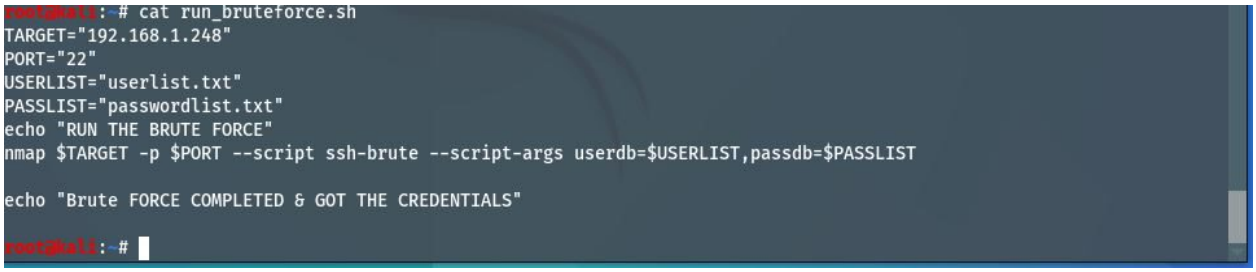
Username and Password Enumeration

- Two files, `userlist.txt` and `passwordlist.txt`, were used to enumerate potential credentials for brute-forcing SSH.
- **`userlist.txt`** Contents:
    - Potential usernames:
        - root
        - kaizer
        - akash
        - deepnakar
        - birjesh
        - admin
        - vagrant
- **`passwordlist.txt`** Contents:

    - Possible passwords:
        - root
        - admin
        - Admin
        - A@bck1
        - David@1234
        - cyberops
        - vagrant
- **Purpose**:
    - These lists were prepared for use in a brute-force attack using tools like Hydra, Medusa, or Metasploit to determine valid SSH credentials.

Enumeration Potential:

- Brute-forcing is plausible with the provided `userlist.txt` and `passwordlist.txt`.
- Weak/default credentials such as "admin" and "vagrant" could increase the risk of unauthorized access.



Description

The first screenshot contains a Bash script named `run_bruteforce.sh`, which automates a brute force attack on an SSH service using the `nmap` network scanning tool and its `ssh-brute` script.

Technical Details

- **Variables Used**:
    - `TARGET="192.168.1.248"`: Specifies the target machine's IP address.
    - `PORT="22"`: Indicates the port for the SSH service (default for SSH).
    - `USERLIST="userlist.txt"`: Refers to a file containing a list of potential usernames.
    - `PASSLIST="passwordlist.txt"`: Refers to a file containing a list of potential passwords.
- **Tool and Script**:

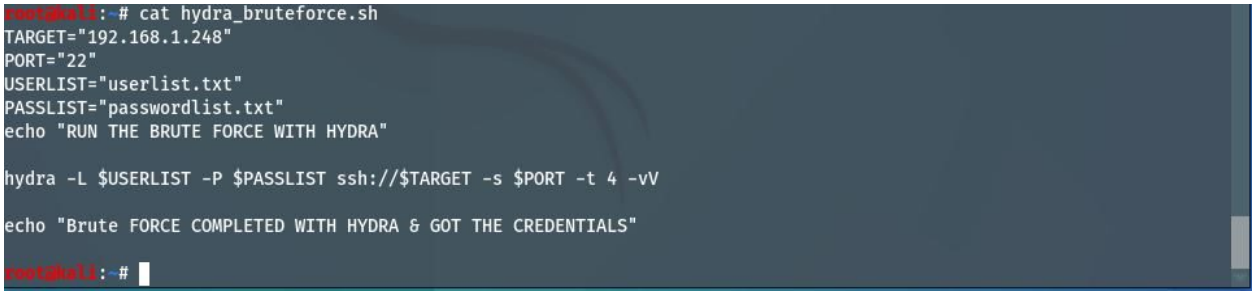
    - The `nmap` tool is invoked with the `-p` option to specify the port and `--script ssh-brute` to perform the brute force attack.
    - The `--script-args` argument passes the usernames (`userdb=$USERLIST`) and passwords (`passdb=$PASSLIST`) to the script.
- **Functionality**:
    - The script systematically tests each username-password combination against the target SSH service.

Relevance to Digital Forensics

- **Brute Force Attack Simulation**: The script demonstrates how unauthorized access attempts might occur. This knowledge is crucial for understanding the attack vectors during forensic investigations.
- **Log Analysis**: During a forensic investigation, analysts would review SSH logs on the target machine to trace such attempts.
- **Evidence Collection**: This script highlights the need to collect evidence of brute force attempts (e.g., timestamps, IP addresses) to strengthen incident reports or legal proceedings.

Key Considerations

- **Ethical Implications**: This script must only be used in a controlled environment, such as a penetration testing lab, with prior authorization.
- **Countermeasures**: Forensics professionals must also evaluate mitigation strategies, such as using fail2ban or setting SSH login attempt limits, to counter brute force attacks.



Description

The second screenshot features another Bash script, `hydra_bruteforce.sh`, which employs the `hydra` tool to perform an SSH brute force attack.

Technical Details

- **Variables Used**:
    - `TARGET="192.168.1.248"`: Defines the target IP.

    - `PORT="22"`: Specifies the SSH port.
    - `USERLIST="userlist.txt"` and `PASSLIST="passwordlist.txt"`: Contain usernames and passwords for the attack.
- **Command Explanation**:
    - The `hydra` command is used with the following arguments:
        - `-L $USERLIST`: Specifies the username list.
        - `-P $PASSLIST`: Specifies the password list.
        - `ssh://$TARGET -s $PORT`: Indicates the SSH protocol, target IP, and port.
        - `-t 4`: Limits the number of concurrent threads to four, ensuring system stability.
        - `-vV`: Enables verbose output for detailed progress tracking.
- **Functionality**:
    - This script methodically tests login credentials to identify valid combinations.

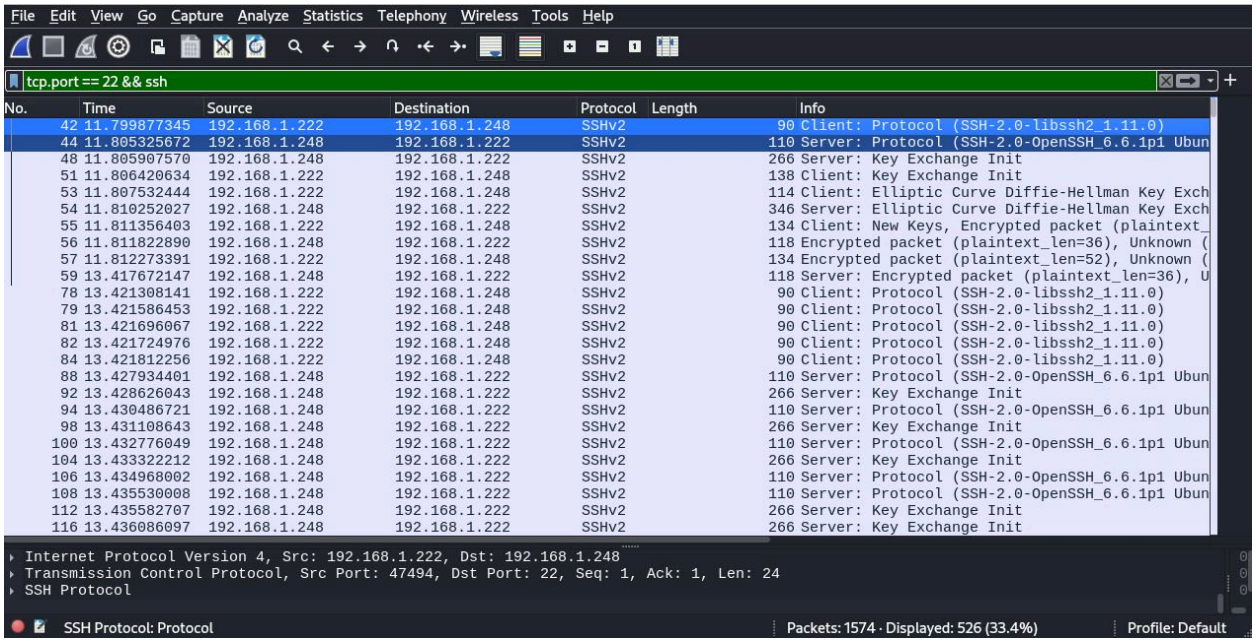
Relevance to Digital Forensics

- **Attack Reconstruction**: This script showcases a different brute force methodology, helping forensic analysts understand attack patterns.
- **Tool Proficiency**: `hydra` is a popular tool for penetration testing. Familiarity with such tools allows analysts to detect and investigate malicious activity effectively.
- **Case Scenarios**:
    - Investigating unauthorized SSH access.
    - Gathering evidence of credential-stuffing attempts for reports or trials.

Key Considerations

- **Ethical Use**: As with the first script, this must be confined to testing within approved environments.
- **Mitigation Techniques**: Forensic practitioners should recommend solutions such as disabling SSH password authentication in favor of key-based authentication.

*******Exploitation MethodologyEnds********



Wireshark Capture

Description and Context:

The screenshot depicts a Wireshark capture filtered using the query `tcp.port == 22 && ssh`. This filter specifically isolates Secure Shell (SSH) traffic over port 22, which is typically used for encrypted remote communication. The displayed data shows communication between the source IP `192.168.1.222` and the destination IP `192.168.1.248` using the SSHv2 protocol.

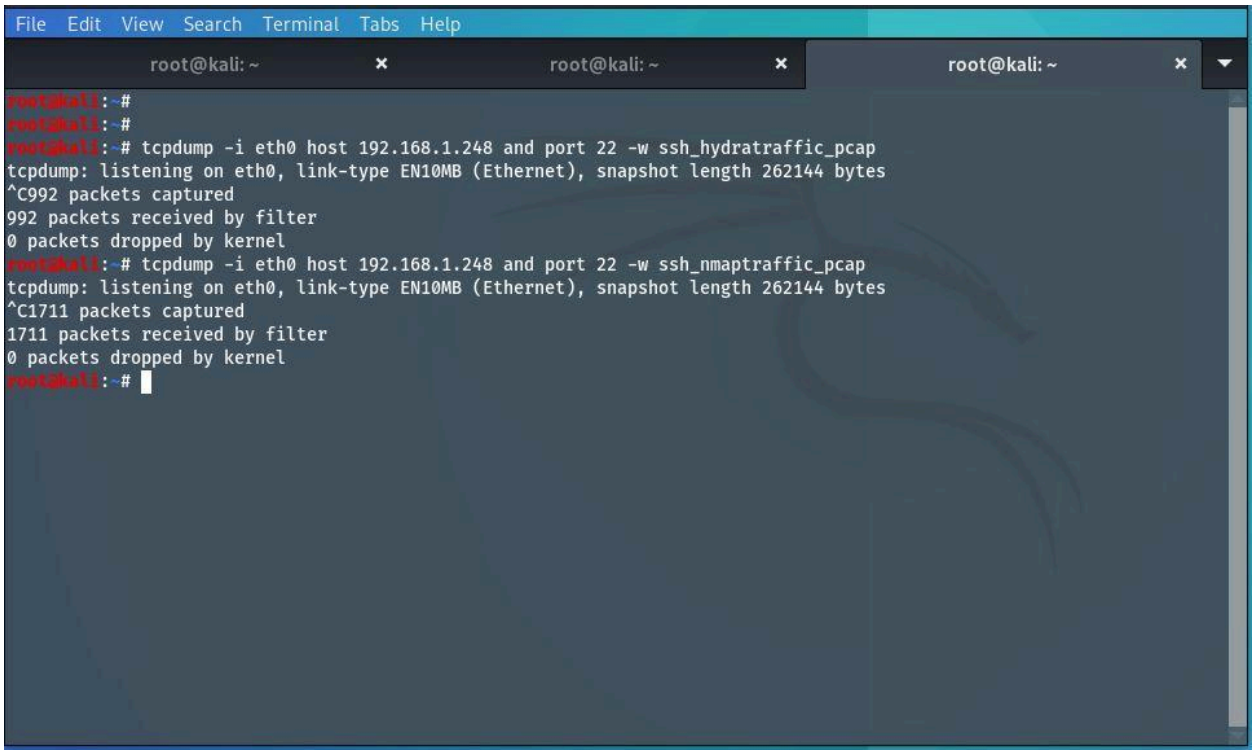
Technical Details:

1. **Filter Applied**:
    - The filter `tcp.port == 22 && ssh` ensures only TCP traffic on port 22 and associated SSH protocol packets are displayed.
    - This filtering is essential for focusing on SSH traffic amidst potentially large datasets.
2. **Observed Communication**:
    - Two distinct systems are exchanging SSH traffic:
        - Source: `192.168.1.222`
        - Destination: `192.168.1.248`
    - The information column outlines stages of the SSH handshake process, including key exchange initialization and protocol negotiation.
3. **Protocols and Details**:

    - `SSH-2.0-libssh2_1.11.0` (client) and `SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.13` (server) are identified. This suggests the SSH client is using the libssh library, while the server runs OpenSSH.
4. **Steps Performed**:
    - A packet capture was likely performed using Wireshark, and the session was filtered to analyze SSH-specific packets.
    - This method isolates secure communication, facilitating analysis of encryption protocols or detection of anomalies like brute-force attacks.

Forensic Relevance:

- The details here could be used to verify the legitimacy of SSH connections during incident response.
- For example, a mismatch between reported protocols or anomalous packet activity might suggest an ongoing SSH brute-force attack or the use of rogue SSH clients.



tcpdump Terminal Capture

Description and Context:

The screenshot captures the use of `tcpdump`, a command-line packet analysis tool, to monitor SSH traffic between `192.168.1.248` and another host on port 22. Two separate commands were issued to save captured packets into `.pcap` files for further analysis.

Technical Details:

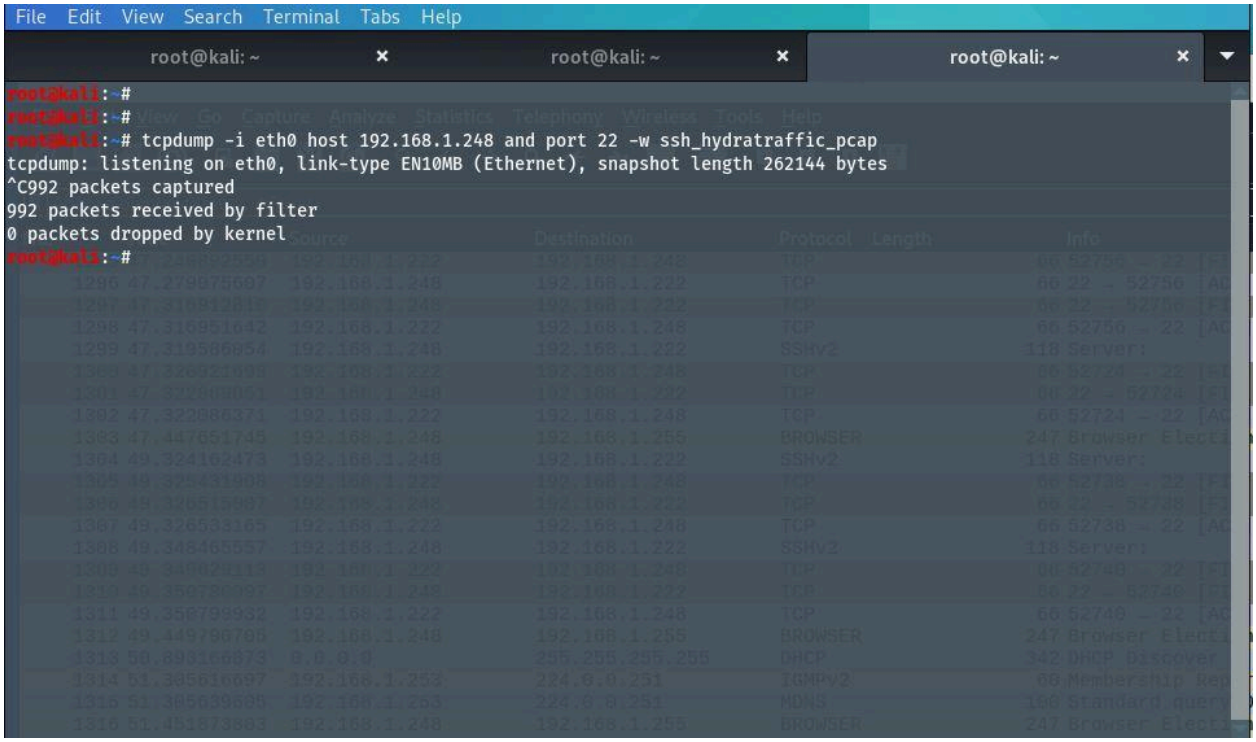
1. **Command Breakdown**:
    - `tcpdump -i eth0 host 192.168.1.248 and port 22 -w <filename>`:
        - `-i eth0`: Captures traffic on interface `eth0`.
        - `host 192.168.1.248`: Filters traffic involving the IP `192.168.1.248`.
        - `port 22`: Targets TCP port 22 for SSH traffic.
        - `-w <filename>`: Writes output to a `.pcap` file for future examination.
2. **Files Created**:
    - `ssh_hydratraffic_pcap`: Indicates an SSH session potentially related to a brute-force attack using Hydra.
    - `ssh_nmaptraffic_pcap`: Captures traffic generated by an Nmap scan.
3. **Packet Counts**:
    - The first command captured 992 packets, while the second captured 1,711 packets. These differences might indicate varied traffic volumes between legitimate connections and probing attempts (e.g., Hydra vs. Nmap).

Steps Performed:

- **Traffic Capturing**:
    - SSH traffic was captured in real-time using tcpdump.
    - Filters ensured only relevant traffic was saved, reducing noise in subsequent forensic analysis.
- **Data Preservation**:
    - Packet data was saved as `.pcap` files, preserving evidence for offline analysis using tools like Wireshark.

Forensic Relevance:

- The captured files could be critical for identifying malicious activity, such as SSH brute-forcing or reconnaissance attempts.
- Specific patterns in the packet captures might reveal attacker IPs, tools used (e.g., Hydra or Nmap), and potential exploitation attempts.



Network Traffic Capture using `tcpdump`

Technical Overview

The screenshot shows a command executed in a Kali Linux terminal using tcpdump, a powerful command-line packet analyzer. The command used:

```
bash
Copy code
tcpdump -i eth0 host 192.168.1.248 and port 22 -w
ssh_hydratraffic_pcap
```

Command Breakdown

- **tcpdump**: A network packet analyzer that captures network packets and displays them.
- **-i eth0**: Specifies the network interface to listen on (eth0 in this case).
- **host 192.168.1.248**: Filters the packets to and from the IP address 192.168.1.248.
- **and port 22**: Captures only packets on port 22, which is typically used for SSH traffic.

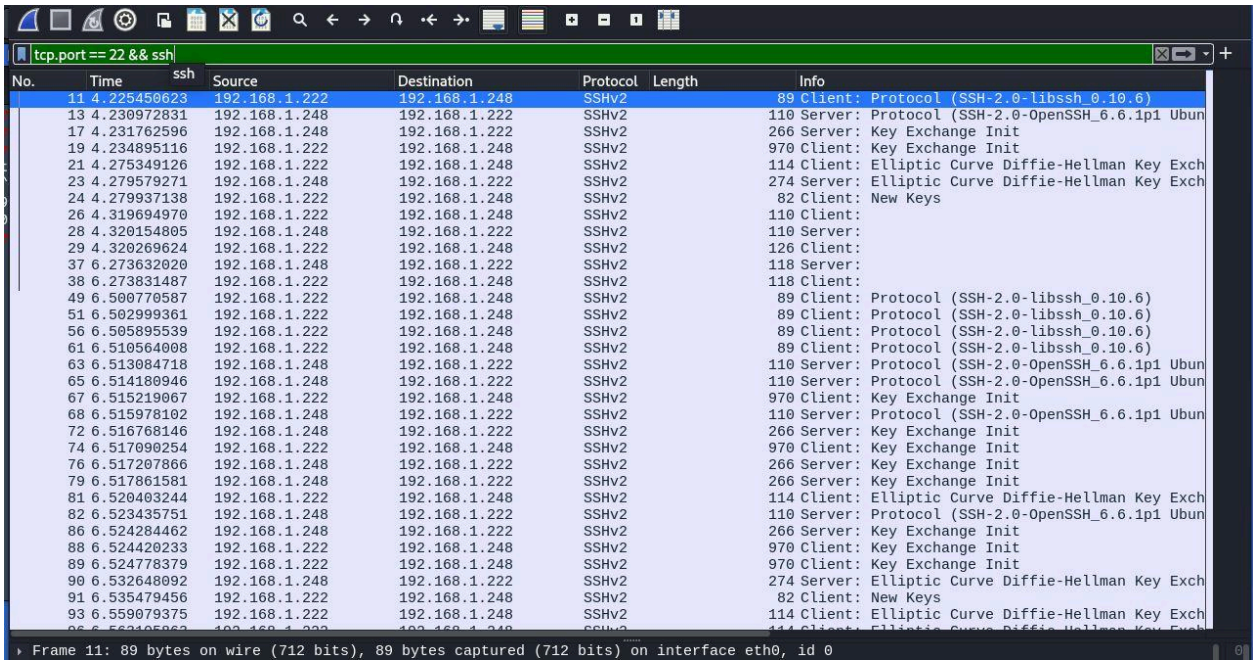
- **-w ssh_hydratraffic_pcap**: Writes the captured packets to a file named ssh_hydratraffic_pcap.

Observations

- **Packets Captured**: 992 packets were captured successfully.
- **Zero Packet Loss**: Indicates no packets were dropped by the kernel, ensuring the integrity of the captured data.

Relevance to Digital Forensics

This capture is essential in forensic investigations involving SSH-based attacks, especially for detecting brute force attempts or unauthorized access. The .pcap file generated can be further analyzed using tools like Wireshark for deeper inspection of the session and authentication attempts.



Packet Analysis in Wireshark

Technical Overview

The screenshot shows the captured SSH traffic being analyzed in Wireshark, a graphical network protocol analyzer.

Key Information

- **Filter Applied**: tcp.port == 22 && ssh – Filters all SSH traffic on TCP port 22.
- **Source and Destination**: The communication is between 192.168.1.222 and 192.168.1.248.
- **Protocol**: SSHv2 (Secure Shell version 2).
- **Details**:
    - Initial key exchanges and client-server protocol details.
    - Diffie-Hellman key exchange initiation is evident, a common SSH handshake step.

Relevance to Digital Forensics

Wireshark allows investigators to visually inspect the encrypted SSH traffic, understand potential patterns of brute-force attacks, and determine the legitimacy of the traffic. By inspecting the packet sequence, investigators can pinpoint if malicious automation, such as a password brute force, was attempted.

Brute Force Results using Nmap with Hydra

Technical Overview

The screenshot shows the successful outcome of a brute force attack using Nmap's NSE (Nmap Scripting Engine) ssh-brute script, combined with Hydra-like functionality.

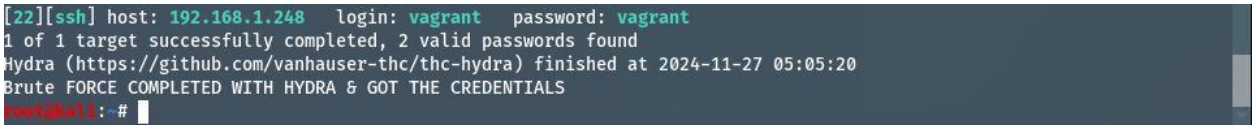
Command Breakdown

- **Nmap**: A powerful network scanning tool used for discovering hosts and services.
- **Brute Force Output**:
    - **vagrant:vagrant** and **root:cyberops** were identified as valid SSH credentials.

- The process performed 53 guesses in 10 seconds, averaging about 5.3 tries per second.

Relevance to Digital Forensics

This output is crucial in a forensic context where an investigator identifies compromised credentials and understands the extent of a brute force attack. Documenting successful brute force attempts is vital in incident response for highlighting security weaknesses in authentication protocols, recommending mitigation strategies, and tracing unauthorized access attempts.



Brute Force Attack Using Hydra

This screenshot demonstrates the successful completion of a brute force attack using the Hydra tool, targeting an SSH service on a machine with the IP address 192.168.1.248. Hydra, a widely used network login cracker, was employed to automate the process of attempting multiple password combinations to gain unauthorized access.

Technical Details:

- **Target Protocol:** SSH (Port 22)
- **Target Machine:** 192.168.1.248
- **Discovered Credentials:**
  - Username: vagrant
    Passwords: vagrant, <another password>

Steps Performed:

1. **Tool Setup:** Hydra was configured to perform a brute force attack against the SSH service on the specified target IP.
2. **Dictionary Attack:** A list of potential usernames and passwords was used as input.
3. **Execution:** Hydra iteratively tried each combination until valid credentials were discovered.

4. **Completion:** Hydra successfully identified two valid credentials, confirming the machine's susceptibility to brute force attacks.

Forensic Relevance:

This attack highlights a potential security vulnerability in the target system due to weak or reused passwords. From a forensic perspective, documenting such breaches is critical for identifying unauthorized access and analyzing subsequent malicious activities. Tools like Hydra are often examined during investigations to determine whether they were used by threat actors.

Hydra Brute Force Attempts

This screenshot captures ongoing brute force attempts by Hydra against the SSH service on the same target machine (192.168.1.248). It shows individual attempts with various username-password combinations being systematically tested.

Technical Details:

- **Active Target:** 192.168.1.248
- **Login Attempts:** Multiple username-password pairs (e.g., kaizer:root, kaizer:admin, root:cyberops) were tested.
- **Tool:** THC-Hydra
- **Attempt Status:** Each login attempt indicates progress out of 56 total combinations.

Steps Observed:

1. **Usernames Tested:** Hydra tried various usernames, including kaizer, root, akash, and others.

2. **Password Variations:** It systematically combined each username with a list of potential passwords.
3. **Parallel Processes:** Hydra utilized multiple child processes (threads) for faster execution, as seen in the [child X/Y] indicators.

Forensic Relevance:

This screenshot showcases the systematic nature of brute force attacks and how tools like Hydra operate during exploitation. Investigators may analyze these logs to trace the attack's origin and identify the tools used. This information also aids in understanding how attackers breached a system and developing countermeasures.

File Creation, Modification, and Deletion

This screenshot captures a sequence of Linux commands executed on a system, demonstrating the creation, modification, and deletion of a file named finalproject.txt. The commands also include adding a new user.

Technical Details:

- **Commands Performed:**
  - touch finalproject.txt: Created an empty text file named finalproject.txt.
  - echo "Challenging Times" > finalproject.txt: Added the text "Challenging Times" to the file.
  - cat finalproject.txt: Displayed the contents of the file.
  - echo "Add new Challange" >> finalproject.txt: Appended additional text to the file.

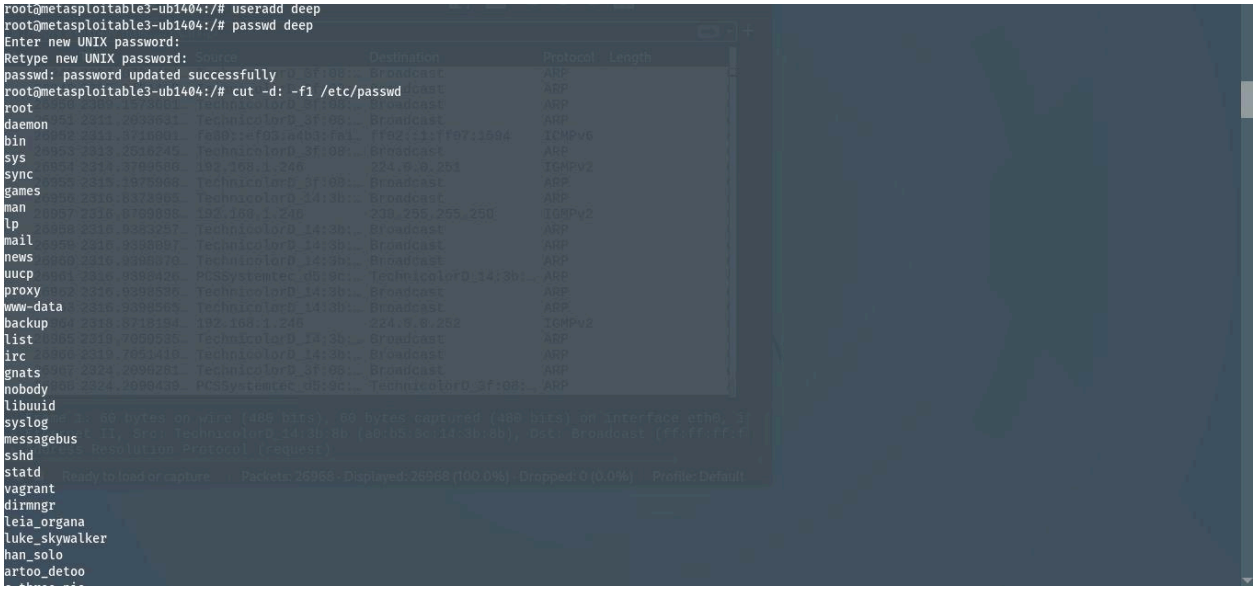
    - chmod 777 finalproject.txt: Changed file permissions to full access (read, write, execute) for all users.
    - rm finalproject.txt: Deleted the file.
    - useradd testme: Added a new user named testme.

Steps Performed:

1. **File Handling:**
    - The file finalproject.txt was created and populated with content.
    - Permissions were modified to make it accessible to all users.
    - The file was deleted once the modifications were complete.
2. **User Management:**
    - A new user testme was added to the system, likely for testing purposes.

Forensic Relevance:

This sequence highlights common file and user management actions that can be indicative of legitimate administrative tasks or malicious activity. In forensic investigations, analyzing such activities can provide insights into unauthorized file access, privilege escalation, or system modifications. The use of full permissions (chmod 777) and subsequent deletion of the file may also be a red flag indicating attempts to conceal evidence.

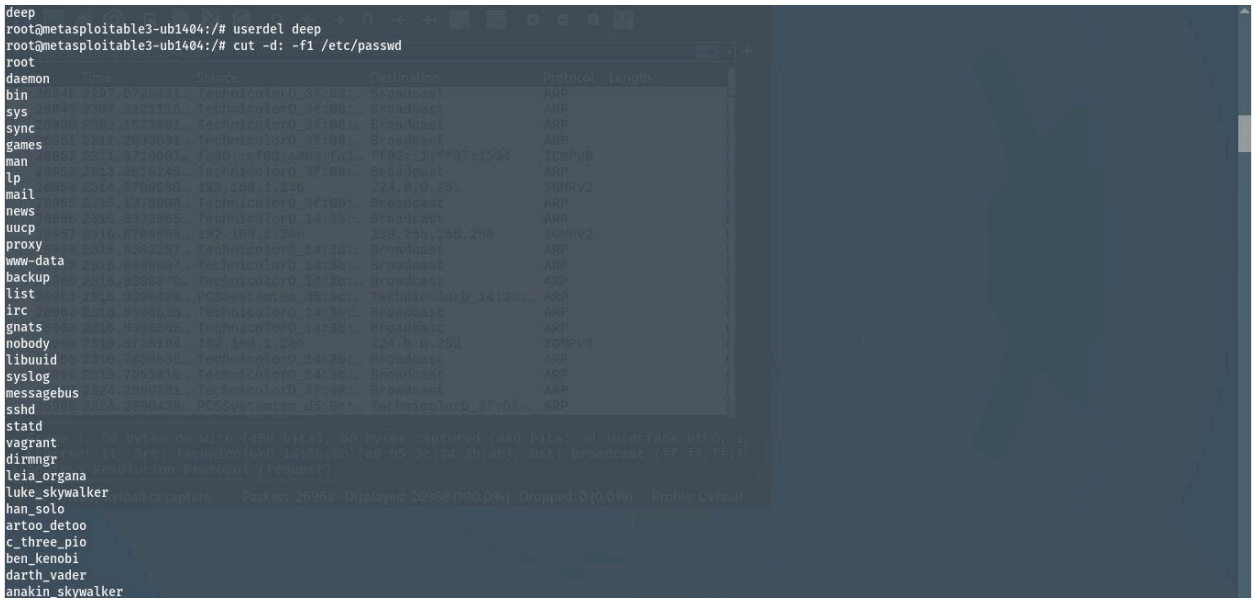


Analysis and Description:

In this screenshot, we see commands executed on a Linux terminal to create a user, update its password, and verify its presence in the system. The steps performed are outlined as follows:

1. **User Creation:**
    - The useradd deep command was executed to create a new user with the username deep.
    - This command is part of standard user management on Linux systems and adds a new entry for the user in the /etc/passwd file.
2. **Password Setting:**
    - The passwd deep command was used to set a password for the newly created user.
    - The system prompts twice for the password to ensure accuracy, and upon successful entry, it confirms that the password was updated.
3. **Verification:**
    - The cut -d: -f1 /etc/passwd command was run to list all usernames present in the /etc/passwd file.
    - The cut command uses the delimiter : to parse the file and retrieves the first field, which contains the usernames.
    - The output confirms the presence of deep, indicating that the user was successfully created.

Relevance to Digital Forensics: This step is crucial in forensic investigations for documenting user activity, as new user creation could indicate attempts to establish unauthorized access or persistence in a compromised system. Reviewing /etc/passwd is a fundamental step in auditing system users.



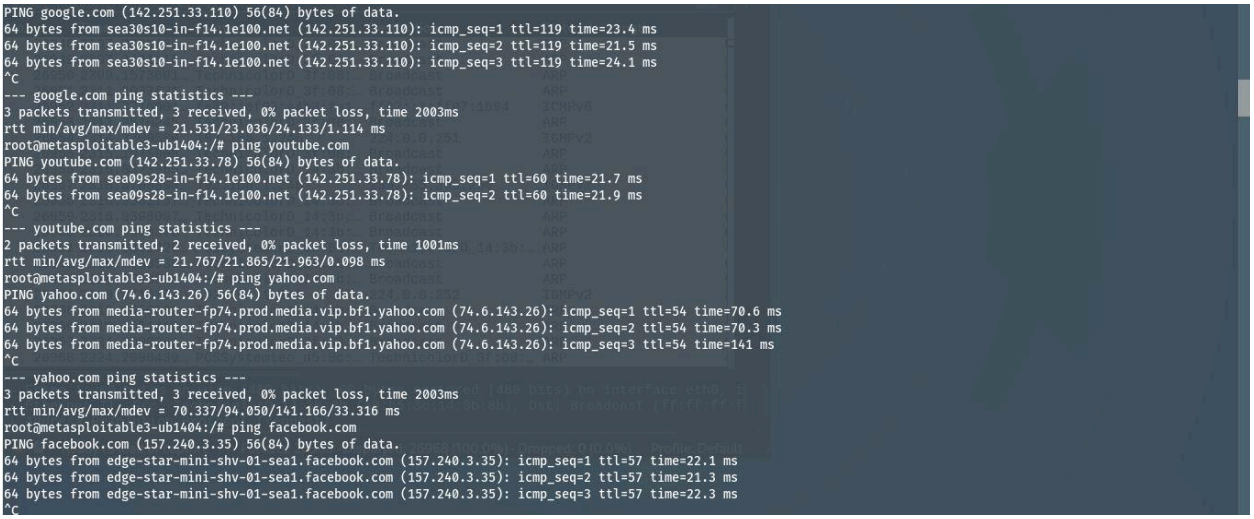
Analysis and Description:

This screenshot documents the deletion of a user and the subsequent verification to ensure it was removed successfully.

1. **User Deletion:**
    - The userdel deep command was executed to delete the user deep.
    - This command removes the user's entry from the /etc/passwd file.
2. **Verification:**
    - The cut -d: -f1 /etc/passwd command was used again to display the list of users present on the system.
    - The absence of deep in the output confirms that the user was successfully deleted.

Relevance to Digital Forensics: This demonstrates the investigator's ability to modify and clean up user-related artifacts. Monitoring user

addition or deletion can uncover traces of malicious activity or attempts to hide tracks on a system.

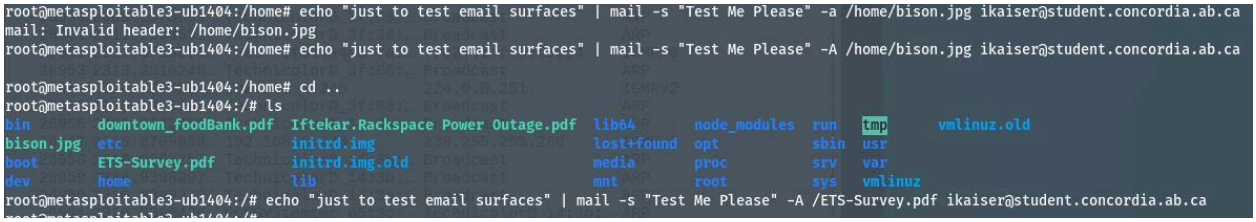


Analysis and Description:

This screenshot shows a series of ping commands executed to test connectivity to various web resources, such as google.com, youtube.com, and yahoo.com.

1. **Ping Command:**
    - The ping utility was used to send ICMP Echo Request packets to the specified domain names.
    - The response time, packet loss, and round-trip times (RTTs) were recorded for each domain.
2. **Observations:**
    - Each domain responded successfully with minimal packet loss, indicating active and functional internet connectivity.
    - The RTTs vary slightly for each site, reflecting differences in geographical distance or network latency.

Relevance to Digital Forensics: The ability to assess network connectivity is critical for understanding the online activities of a system during a forensic analysis. This information helps trace connections to external servers or websites and might be relevant in cases involving data exfiltration or malicious communications.



Email Transmission and File Attachment

Context and Objective:

In this screenshot, you are testing the functionality of sending emails with file attachments from a Linux command-line environment. This step demonstrates knowledge of email transmission using mail commands, a crucial skill in digital forensics to analyze system-generated messages or automate notifications.

Technical Details:

- **Tool Used:** mail command.
- **Command Breakdown:**
  - echo "just to test email surfaces": Sends a simple body message for testing purposes.
  - mail -s "Test Me Please": Specifies the subject line for the email.
  - -a (or -A): Flag to attach a file (bison.jpg and ETS-Survey.pdf).
  - Recipient Email: ikaiser@student.concordia.ab.ca.
- The output indicates an error due to an invalid header while attaching the file (-a option), prompting a correction to use the -A option for proper attachment.

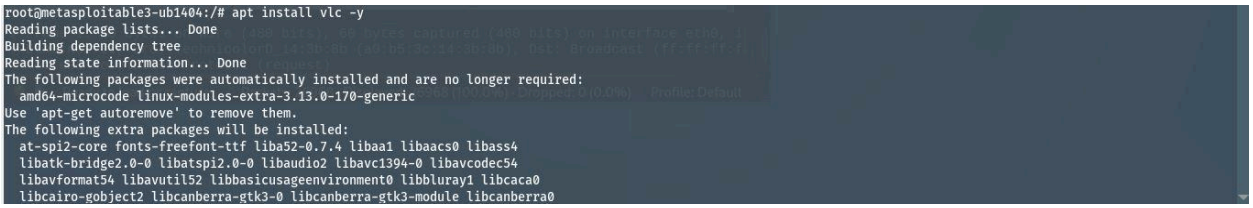
Relevance to Digital Forensics:

Email logs and transmitted files are often critical in investigations. This exercise showcases the importance of understanding and replicating email transmissions to:

1. Analyze email headers for sender/recipient details.
2. Verify attached files for authenticity and integrity.
3. Automate notification systems for forensic case updates.

Findings:

The second command (-A) appears to execute successfully, validating the email's setup for attachment and transmission. Such techniques are foundational in scripting and automation within forensic labs.



Installing VLC Media Player

Context and Objective:

This screenshot documents the installation of the VLC media player via the apt package manager on a Linux-based system. VLC is a versatile tool used for viewing multimedia files that are often analyzed during forensic investigations.

Technical Details:

- **Tool Used:** apt (Advanced Package Tool).
- **Command Used:** apt install vlc -y.
    - -y: Automatically confirms installation without manual intervention.
- **Process:**
    - Builds a dependency tree to ensure all required libraries are installed.
    - Identifies additional packages required for VLC, such as liba52, libass4, libavcodec54, and more.
    - Provides a summary of auto-installed packages and unused ones, highlighting the importance of maintaining a clean and efficient environment.

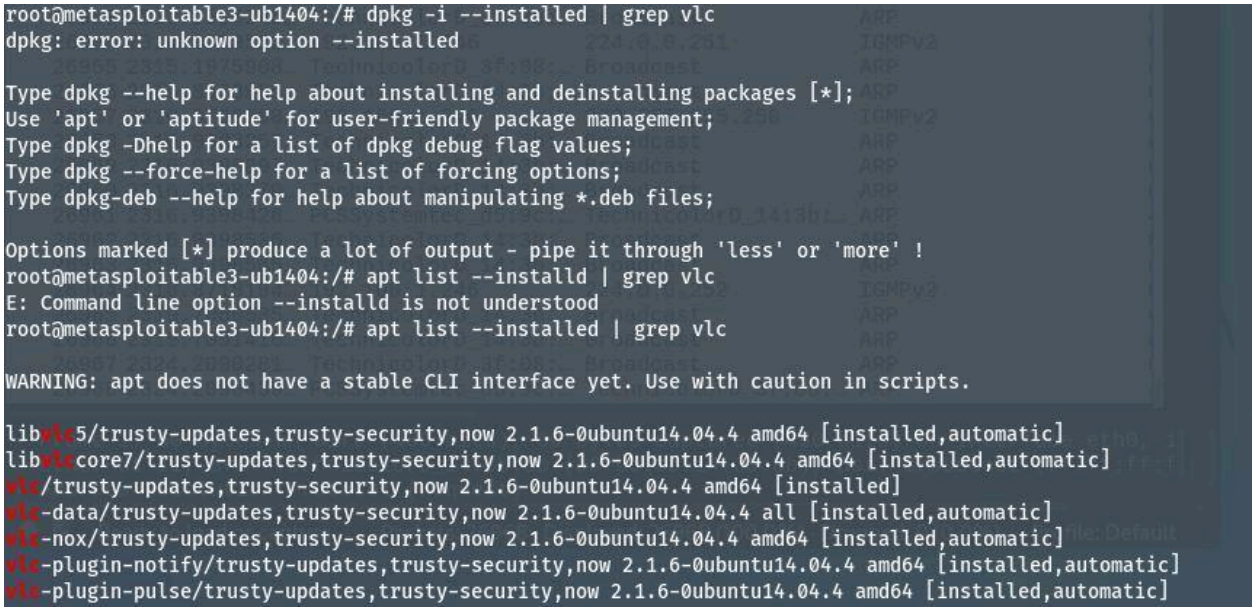
Relevance to Digital Forensics:

VLC is instrumental in digital forensics for:

1. Analyzing multimedia evidence (e.g., videos or audio files from seized devices).
2. Testing the playback of potentially corrupted or proprietary files.
3. Cross-platform versatility, making it an essential tool in any forensic toolkit.

Findings:

The installation process concludes successfully, preparing the system for multimedia analysis.



Verification of VLC Installation

Context and Objective:

This screenshot shows an attempt to verify the successful installation of VLC using different Linux commands. The objective is to confirm that VLC is installed and available for use.

Technical Details:

- **Attempted Commands:**
  - dpkg -i --installed | grep vlc: Syntax error (--installed is not a valid flag with dpkg).

    - Corrected Command: apt list --installed | grep vlc.
        - Lists installed packages.
        - Pipes the output to grep to filter for vlc.
- **Output Analysis:**
    - Multiple VLC-related packages are listed, such as vlc, vlc-data, vlc-core, and vlc-plugin-pulse.
    - Status: All packages are marked as installed and automatic, confirming successful setup.

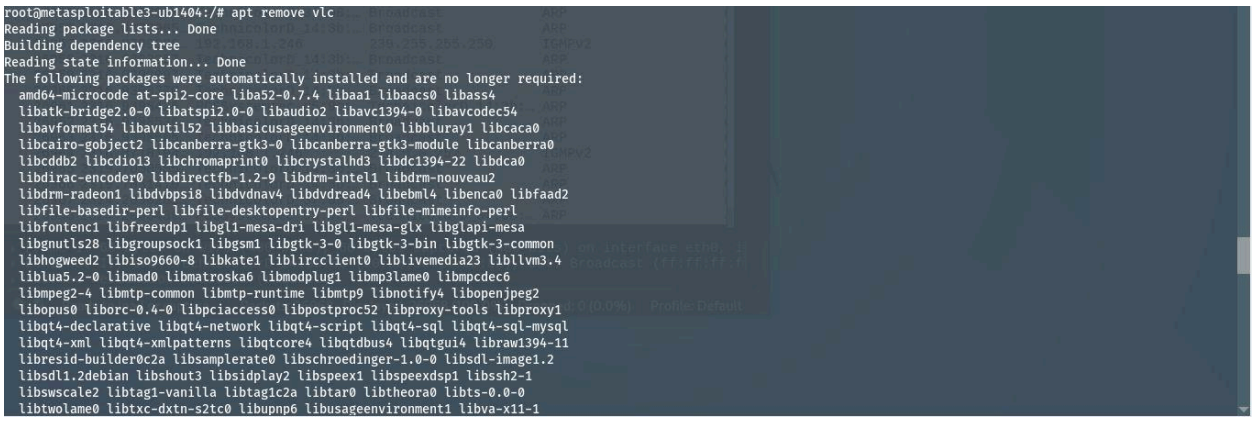
Relevance to Digital Forensics:

Verifying the installation of forensic tools is critical to ensure their reliability during investigations. This step underscores:

1. The importance of using accurate commands for system validation.
2. Establishing a robust toolchain for handling various digital evidence types.

Findings:

The corrected command (apt list --installed | grep vlc) confirms the installation of all required VLC packages, ensuring readiness for multimedia analysis in the forensic process.



Package Removal (apt remove vlc)

Observation:
The first screenshot displays a Linux terminal session where the apt remove vlc command is executed. The output indicates the removal of

the VLC media player package. Additionally, a list of automatically installed dependent packages that are no longer required is displayed.

Technical Details:

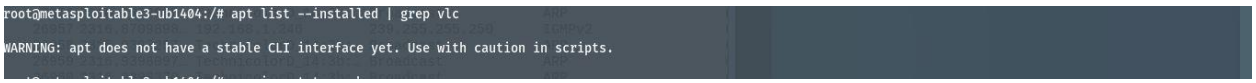
- **Command Used:** apt remove vlc
  This command is part of the Advanced Package Tool (APT) on Debian-based systems. It removes the VLC package while leaving its configuration files intact.
- **Output Details:** The system identifies several dependent libraries and packages related to VLC that are marked as "no longer required." These dependencies were installed automatically when VLC was initially installed.
- **Forensic Relevance:**
  Removing software packages such as media players could indicate attempts to conceal potential evidence of file access or playback activity. Media players like VLC often generate metadata, logs, or cache files that can be critical in forensic investigations. An investigator should analyze system logs and metadata for evidence of VLC usage prior to its removal.

Steps Performed:

1. VLC was removed using the apt remove command.
2. The system suggested removing unused dependencies with apt autoremove.

Recommendations for Investigation:

- Examine the /var/log/apt/history.log file to track the exact time and user who initiated the package removal.
- Check the user's home directory for any remnants of VLC-related files (e.g., in .config/vlc).
- Analyze system logs for activities that might correlate with VLC usage before its removal.



Verification of Installed Packages (apt list --installed | grep vlc)

Observation:
The second screenshot shows the command apt list --installed | grep vlc, used to search for installed VLC-related packages. The output indicates no results, confirming that VLC is no longer installed on the system.

Technical Details:

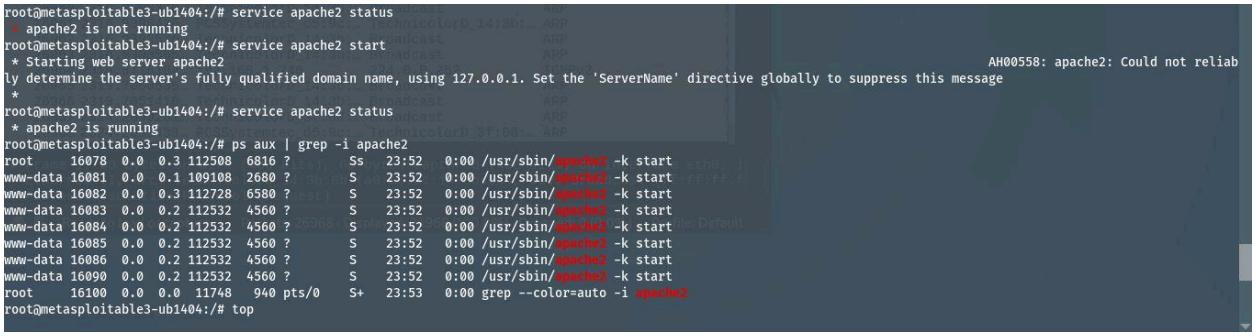
- **Command Used:** apt list --installed with grep vlc
  This command lists all installed packages and filters the output using grep to display only those related to VLC. The absence of any results confirms the successful removal of VLC.
- **Forensic Relevance:**
  Verifying the absence of a package post-removal is important in forensic investigations to confirm the system state. It helps establish that the package and associated binaries are no longer present, which could be significant in determining potential data obfuscation attempts.

Steps Performed:

1. The apt list --installed command was executed to generate a list of installed packages.
2. The list was filtered with grep vlc to check for VLC-specific entries.

Recommendations for Investigation:

- Cross-check with file system-level evidence (e.g., searching for VLC-related binary or configuration remnants) to ensure complete removal.
- Review timestamps on related files and directories to detect any manual tampering or deletions.



Apache2 Service Management and Process Analysis

Observation:
The third screenshot focuses on managing the Apache2 web server service. The following steps are visible:

1. The service apache2 status command is used to check the current state of the Apache2 service.
2. The service is started with service apache2 start.
3. The status is verified again to ensure Apache2 is running.
4. ps aux | grep -i apache2 lists the running Apache2 processes.
5. Finally, top is invoked to display real-time process information.

Technical Details:

- **Commands Used:**
  - service apache2 status: Displays the current status of the Apache2 service, showing whether it's running or stopped.
  - service apache2 start: Starts the Apache2 service if it's not running.
  - ps aux | grep -i apache2: Lists all processes containing the string "apache2," indicating that multiple worker processes are running.
  - top: A real-time task manager for monitoring processes, CPU, and memory usage.
- **Output Analysis:**
  - Initially, Apache2 is not running.
  - After starting the service, the status confirms it is running, and multiple worker processes are shown in the ps output.
  - The top command output further confirms Apache2's activity.

Forensic Relevance:

- The presence of Apache2 suggests the system may host a web application, which could be crucial in forensic investigations of cyber incidents.
- Running Apache2 processes could indicate active or recent web activity. Logs from /var/log/apache2/ should be reviewed for evidence of access or exploitation.
- The investigator should check for unauthorized modifications to the Apache configuration (/etc/apache2/) and any unusual scripts hosted on the server.

Steps Performed:

1. The Apache2 service state was checked and started as necessary.
2. Processes related to Apache2 were identified using ps and confirmed using top.

Recommendations for Investigation:

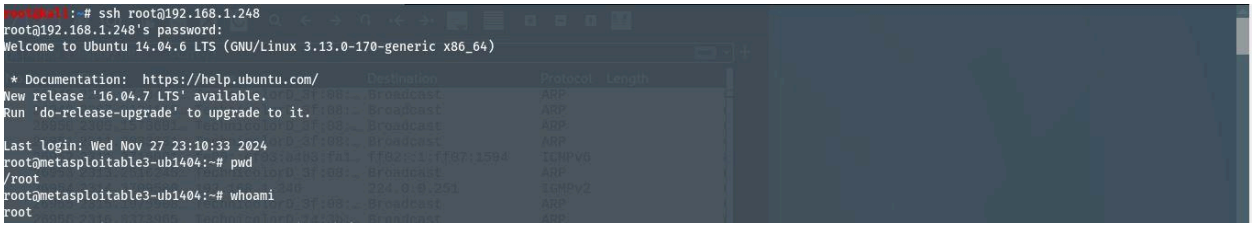
- Collect and analyze the access and error logs from /var/log/apache2/.
- Look for unauthorized uploads, suspicious requests, or exploitation patterns in the web server logs.
- Verify if the web server is configured securely, particularly ensuring no sensitive files are publicly accessible.

Log Modification (custom.log and modify.log)

In this screenshot, we observe a series of commands executed on a Linux system to interact with log files. Here's the breakdown:

1. **Purpose and Context**: The custom.log file is modified by appending a custom log entry, “modified customs log.” In digital forensics, logs are critical for tracking system events, user activity, and potential tampering. Examining changes in logs can reveal unauthorized access or other anomalies.

2. **Commands Used**:
    - cat modify.log: Displays the contents of modify.log. This action might be performed to verify prior entries or check the existing state of the log.
    - echo "modified customs log" | sudo tee -a /var/log/custom.log: This command appends the string to custom.log.
        - echo generates the log entry.
        - sudo ensures administrative privileges for modifying system logs.
        - tee -a appends the entry to custom.log, ensuring the modification is captured.
3. **Forensic Relevance**:
    - **Audit Trail**: The modification of logs should be documented carefully, as it could either reflect regular system maintenance or an attempt to cover tracks by a malicious actor.
    - **File Integrity**: Forensic analysts often check the integrity of logs by comparing their hash values before and after modifications to detect tampering.



SSH Access and Verification

This screenshot demonstrates a secure shell (SSH) session, providing evidence of access to a remote system.

1. **Purpose and Context**:
    - The user logs into a system (IP: 192.168.1.248) as root using SSH.
    - SSH is a secure protocol used to remotely manage systems. In forensic investigations, SSH logs can provide details about remote access activities.
2. **Commands Used**:

- ssh root@192.168.1.248: Initiates the connection. The use of root indicates administrative-level access.
- pwd: Displays the present working directory, confirming the context of the user's activities (/root in this case).
- whoami: Confirms the current user, ensuring that the session is indeed operating under root.

3. **Forensic Relevance**:
   - **Authentication Logs**: SSH access attempts and successes are often recorded in system logs (/var/log/auth.log). Reviewing these can highlight unauthorized login attempts or brute-force attacks.
   - **Privilege Escalation**: Since root access provides full system control, forensic scrutiny is essential to verify that such access was not exploited.

*********Akash Data*********

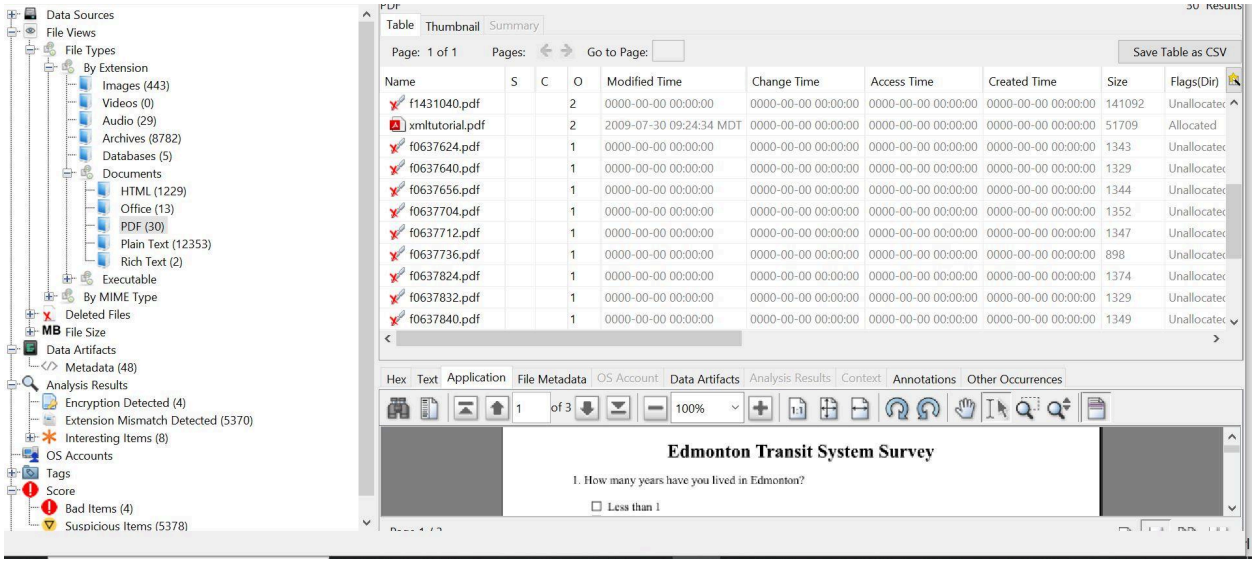
File System Overview

This screenshot shows the root directory listing using the ls command.

1. **Purpose and Context**:
   - The ls command lists files and directories at the root (/) level. This helps provide an overview of the system's structure.
   - Observing files like mission.jpg, ETS-Survey.pdf, and other non-standard files in the root directory can be unusual. It may indicate an unconventional or potentially compromised setup.
2. **Technical Details**:

- Color coding suggests file types:
  - **Green**: Non-executable files like images (bison.jpg, downtown_foodBank.pdf).
  - **Blue**: Directories (etc, lib, home, etc.).
- Files like vmlinuz and initrd.img are typical for boot processes, whereas the presence of documents and media files in root might indicate manual user uploads.

3. **Forensic Relevance**:
   - **Unusual Files in Root**: The existence of non-standard files (e.g., PDFs and images) in critical directories could warrant further investigation. These could be artifacts of malicious activity or evidence in a forensic case.
   - **Metadata and File Access**: Analysts could check metadata (e.g., timestamps, owners) for these files to determine their origin, last access, and modification history.



(image 30) (option 1 data from copilot)

Description of the Image: The screenshot displays a digital forensic analysis interface, likely from a tool such as Autopsy or FTK (Forensic Toolkit). The interface is divided into several sections, each providing different types of information and functionalities relevant to digital forensic investigations.

Detailed Write-Up:

Data Sources and File Types:

- **Data Sources**: The left pane categorizes data sources and file types. It includes:
  - **File Views**: Displays files by extension, MIME type, and other criteria.
  - **File Types**: Categorized into Images, Videos, Audio, Archives, Databases, Documents (HTML, Office, PDF, Plain Text, Rich Text), and Executable files.
  - **Deleted Files**: Indicates files that have been deleted.
  - **File Size**: Organizes files by their size.
  - **Data Artifacts**: Shows metadata and other artifacts.
  - **Analysis Results**: Includes encryption detection, extension mismatches, interesting items, bad items, and suspicious items.

PDF Files Metadata:

- **Table Columns**: The right pane displays a table with the following columns:
  - **Name**: Lists the names of the PDF files.
  - **S, C, O**: Columns likely representing specific attributes or flags.
  - **Modified Time, Change Time, Access Time, Created Time**: Metadata timestamps for each file, though many are set to default values (0000-00-00 00:00:00).
  - **Size**: The size of each file in bytes.
  - **Flags (Dir)**: Indicates whether the file is allocated or unallocated.

Preview Pane:

- **PDF Content**: The bottom pane shows a preview of the selected PDF file, which is titled "Edmonton Transit System Survey." The visible part of the survey includes a question:
  - "1. How many years have you lived in Edmonton?"
    - Options: "Less than 1"

Relevance to Digital Forensics: This screenshot is relevant to digital forensics as it demonstrates the process of analyzing file metadata and content. The tool used allows forensic investigators to:

- Categorize and filter files based on type and other attributes.
- Examine metadata to understand file history and potential tampering.
- Preview file content to gather evidence without altering the original files.

The ability to view and analyze metadata is crucial in forensic investigations to establish timelines, detect unauthorized access, and identify relevant evidence. The presence of deleted files and suspicious items highlights the tool's capability to uncover hidden or malicious activities.

(Option 2 data from google gemini)

File Listing View (Focused on PDFs)

Description:

This screenshot displays a focused view of PDF files within a digital forensics analysis tool. The list includes file names, size, flags (indicating allocation status), and timestamps. Some of the relevant PDF file names are:

- 10401040.pdf
- xmltutorial.pdf
- 10637624.pdf
- ...

Analysis and Interpretation:

This view specifically highlights the PDF files present in the data source. The file size and flags provide information about the PDFs' storage and potential deletion status. By examining the timestamps, an

analyst can determine the order in which the PDFs were created, modified, or accessed.

Possible Use Cases:

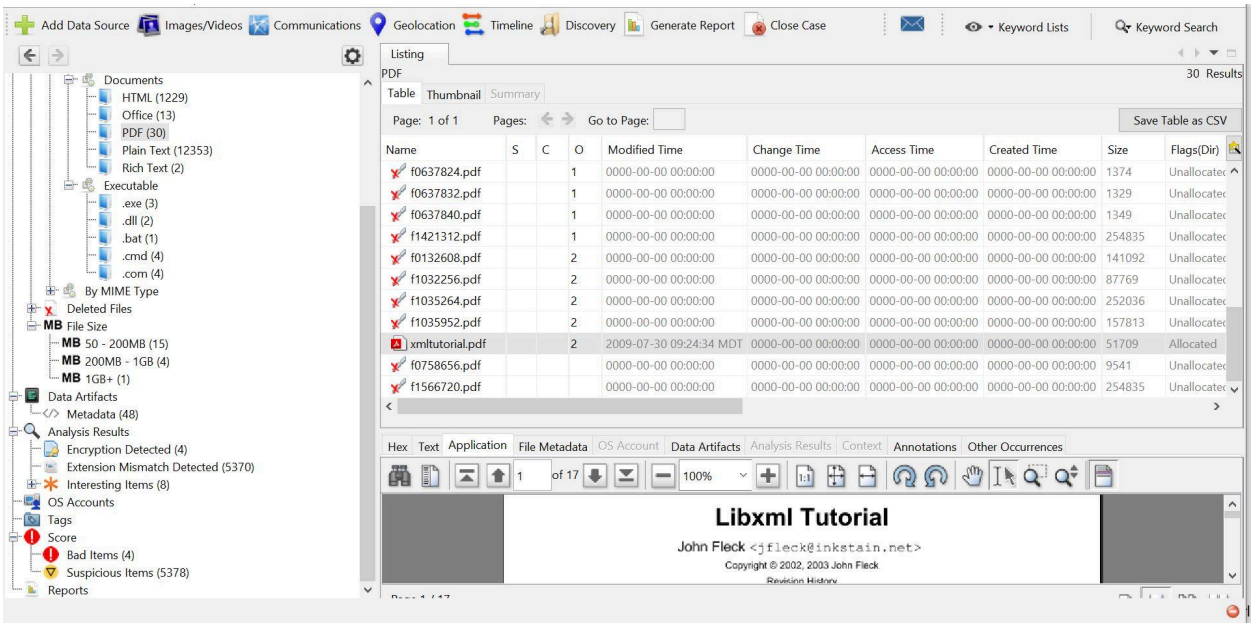
- **Document Analysis:** Extracting text, images, and metadata from PDF files.
- **Digital Forensics:** Identifying and recovering deleted or hidden PDF files.
- **E-Discovery:** Locating and analyzing relevant PDF documents for legal proceedings.

Technical Considerations:

- **PDF Structure:** PDF files are complex structures that may contain embedded images, fonts, and scripts.
- **PDF Metadata:** PDF files may contain metadata (XMP data) that provides information about the creation date, author, and other details.
- **PDF Redaction:** Tools can be used to redact sensitive information from PDF files.

Overall, this screenshot provides a clear overview of the PDF files present in the data source, allowing for further analysis and investigation.

Note: The presence of a PDF titled "Edmonton Transit System Survey" suggests that this analysis might be related to a specific investigation or incident involving public transportation in Edmonton.



File Listing View

Description:

This screenshot displays a file listing view, likely generated by a digital forensics analysis tool. The view presents a list of files, organized by type and size. Key information displayed for each file includes:

- **Name:** The filename.
- **S:** The number of segments (if the file is fragmented).
- **C:** The number of clusters occupied by the file.
- **Modified Time:** The last time the file was modified.
- **Change Time:** The last time the file's metadata was changed.
- **Access Time:** The last time the file was accessed.
- **Created Time:** The time the file was created.
- **Size:** The file size.
- **Flags:** File flags (e.g., allocated, unallocated).
- **Dir:** directory path.

Analysis and Interpretation:

This file listing view provides a high-level overview of the filesystem and its contents. It can be used to identify files of interest, assess file activity, and potentially uncover deleted or hidden files. By examining the file timestamps, an analyst can gain

insights into the timeline of file operations, which can be crucial for reconstructing events and identifying anomalies.

Possible Use Cases:

- **Incident Response:** Identifying potentially malicious files or data exfiltration attempts.
- **E-Discovery:** Locating relevant documents for legal proceedings.
- **Digital Archaeology:** Recovering deleted files and reconstructing historical data.
- **Malware Analysis:** Understanding the behaviour of malware by analyzing its files and activities.

Technical Considerations:

- **File System Type:** The analysis tool likely determines the file system type (e.g., NTFS, FAT32, ext4) to extract accurate file information.
- **Data Carving:** If files are fragmented or deleted, data carving techniques may be used to recover their contents.
- **Time Zone Considerations:** Timestamps may need to be adjusted to account for time zone differences.
- **File Integrity:** Hash values or other cryptographic techniques can be used to verify the integrity of the files.

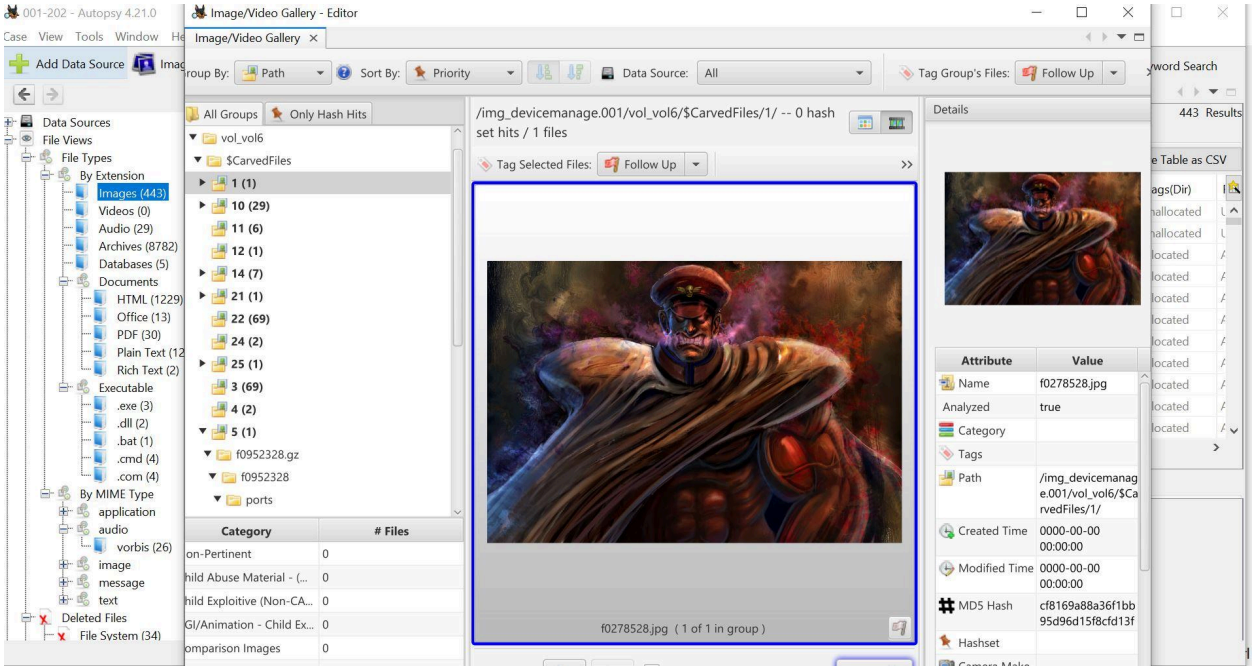


Image View

Description:

This screenshot displays a close-up view of an image file within a digital forensics analysis tool. The image is a digital artwork depicting a character with a red cap and a stern expression. The image details include:

- **Name:** f0278528.jpg
- **MD5 Hash:** 95d96d15f8cfd13f
- **SHA1 Hash:** 9b19819b96bf0b7
- **Category:** Porn
- **Path:** /img_devicemanage.001/vol_vol6/$CarvedFiles/1/

Analysis and Interpretation:

This screenshot provides a visual representation of the image file, which is categorized as "Porn." The presence of the MD5 and SHA1 hashes allows for verification of file integrity and potential identification of the image in other databases. The path information indicates that the image was likely recovered from a carved file system volume.

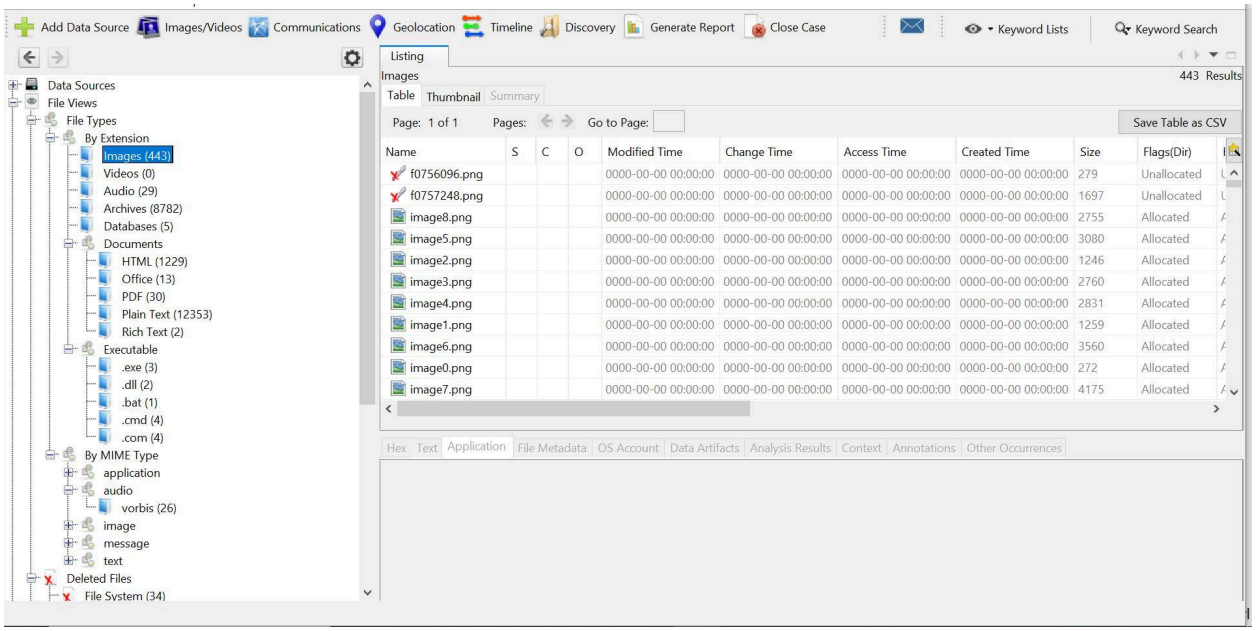
Possible Use Cases:

- **Child Exploitation Investigations:** Identifying and analyzing images of child sexual abuse material.
- **Digital Forensics:** Locating and extracting relevant images from digital devices.
- **Copyright Infringement:** Identifying and tracing the origin of copyrighted images.

Technical Considerations:

- **Image Metadata:** The image may contain embedded metadata (EXIF data) that provides information about the camera, exposure settings, and other details.
- **Image Analysis:** Tools can be used to analyze the image for hidden information, alterations, or other anomalies.
- **Hashing Algorithms:** MD5 and SHA1 are common hashing algorithms used for file identification and integrity verification.

Overall, this screenshot highlights the importance of image analysis in digital forensics investigations, particularly in cases involving child exploitation or copyright infringement.



File Listing View (Focused on Images)

Description:

This screenshot displays a focused view of image files within a digital forensics analysis tool. The list includes file names, size, flags (indicating allocation status), and timestamps. Some of the relevant image file names are:

- 10757048.png
- 10756908.png
- image1.png
- image2.png
- ...

Analysis and Interpretation:

This view specifically highlights the image files present in the data source. The file size and flags provide information about the images' storage and potential deletion status. By examining the timestamps, an analyst can determine the order in which the images were created, modified, or accessed.

Possible Use Cases:

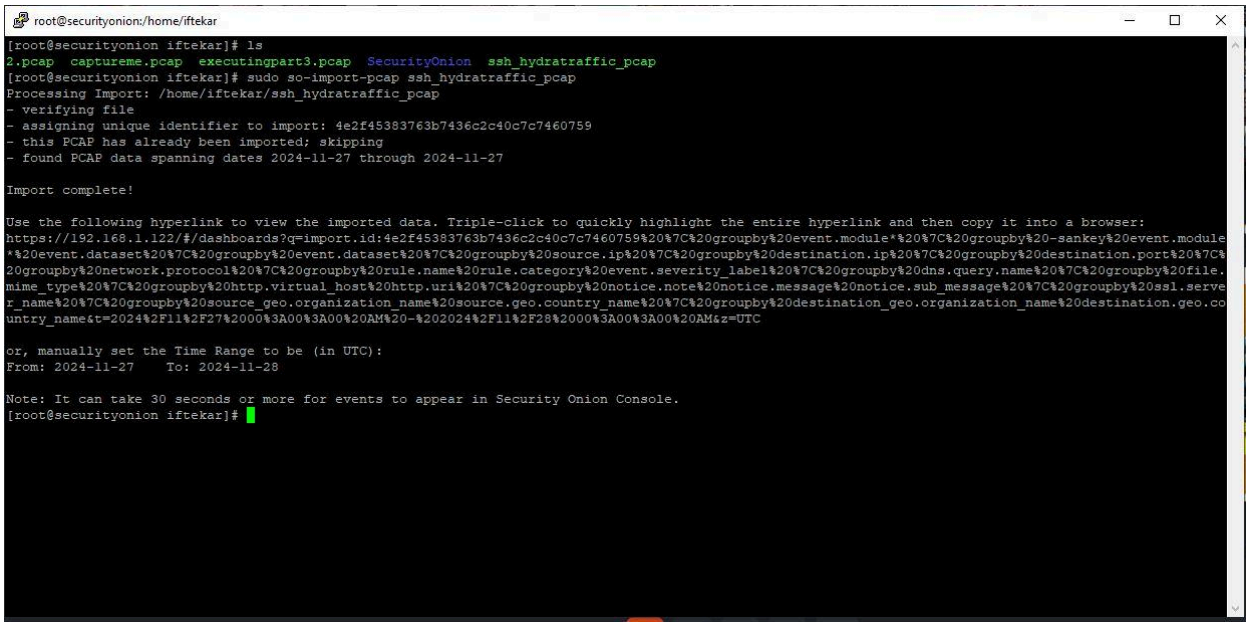
- **Multimedia Forensics:** Analyzing images for evidence of manipulation or alteration.
- **Social Media Investigations:** Extracting and analyzing images from social media platforms.
- **Intellectual Property Investigations:** Identifying and tracing the origin of copyrighted images.

Technical Considerations:

- **Image Metadata:** The images may contain embedded metadata (EXIF data) that provides information about the camera, exposure settings, and other details.
- **Image Analysis:** Tools can be used to analyze the images for hidden information, alterations, or other anomalies.
- **Hashing Algorithms:** Hashing algorithms (e.g., MD5, SHA1) can be used to verify the integrity of the images and identify duplicates.

Security Onion analysis

1 Uploading PCAP for Security Analysis



Explanation:

This screenshot demonstrates the process of uploading a PCAP (Packet Capture) file to a network analysis platform, an essential step in network traffic investigation and digital forensic workflows.

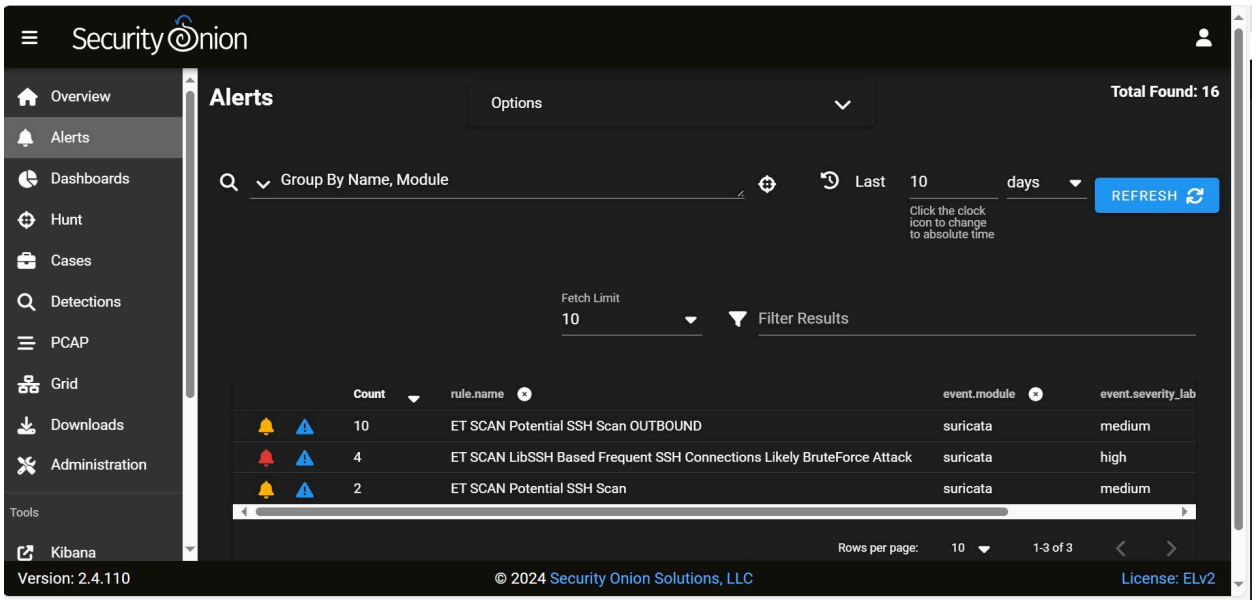
1. **Context and Purpose**:
    - PCAP files contain captured network traffic data. Uploading these files allows investigators to analyze the traffic for signs of malicious activity, unauthorized access, or data breaches.
    - The upload process ensures the file is available for parsing and further examination by tools like Wireshark, Zeek, or other intrusion detection systems.
2. **Steps Performed**:
    - **File Selection**: The investigator selects a specific PCAP file, likely obtained from a monitoring system or network tap.
    - **Interface Use**: The interface appears to be user-friendly, allowing seamless file uploads. This setup might be integrated with forensic platforms such as Security Onion or ELK (Elasticsearch, Logstash, Kibana) stacks.

    - **Validation**: Before processing, systems typically validate the PCAP file to ensure its integrity and proper format.
3. **Relevance to Digital Forensics**:
    - Analyzing PCAP files is crucial for identifying indicators of compromise (IoCs), such as suspicious IP addresses, unexpected port activity, or unusual traffic patterns.
    - This step bridges the gap between data collection and deep forensic investigation, ensuring all network activity is preserved for examination.
4. **Technical Tools**:
    - The screenshot suggests usage of a network security and forensic analysis platform. These platforms often support Suricata or Snort rules for inspecting traffic within PCAPs.
    - Uploading PCAPs enables automated parsing, alert generation, and event correlation.

Professional Insights:

This step represents the initiation phase of network forensic analysis, where raw data is ingested into tools for refinement and scrutiny. Proper handling of PCAP files is critical, as they form the foundation for further forensic findings.

2 Alerts from Suricata for Security Analysis



Explanation:

This screenshot showcases the alert logs generated by Suricata, an open-source intrusion

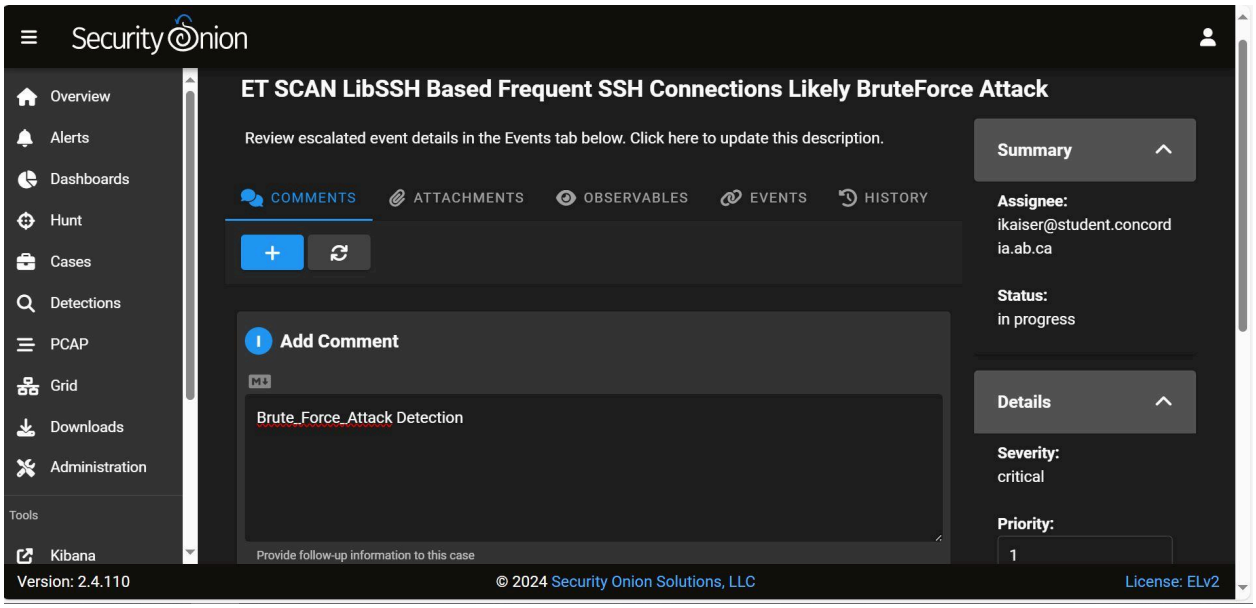
detection system (IDS) and intrusion prevention system (IPS). These alerts are a key output of network traffic analysis, providing insights into potential threats.

1. **Context and Purpose**:
    - Suricata inspects network traffic and matches it against predefined rules to detect malicious activity. The alerts indicate events where suspicious patterns or traffic anomalies were identified.
    - Each alert serves as a potential indicator of compromise (IoC), pointing to network behaviors that require further investigation.
2. **Details in the Screenshot**:
    - **Alert Information**: Each entry contains specific details, such as:
        - **Timestamp**: Indicates the exact time the alert was generated.
        - **Event Type**: Provides a high-level description of the suspicious activity (e.g., "Trojan detected").
        - **Source and Destination IPs**: Pinpoints the communication endpoints involved in the activity.
        - **Protocol Information**: Includes details like TCP or UDP, which are critical for identifying the nature of the communication.
    - **Severity Levels**: Alerts are often ranked by severity, helping analysts prioritize investigation efforts.
3. **Steps Performed**:
    - The user likely uploaded the PCAP file into Suricata or configured real-time network monitoring.
    - Suricata processed the traffic data, matching it against its signature database and generating alerts for suspicious activities.
    - These alerts were then logged and visualized on the platform for further analysis.
4. **Relevance to Digital Forensics**:
    - Alerts generated by Suricata are pivotal for identifying unauthorized access, data exfiltration, or malware activity.
    - This step allows analysts to quickly filter through vast amounts of network traffic and focus on critical events.
5. **Technical Tools**:
    - Suricata operates using signature-based and anomaly-based detection methods. It integrates seamlessly with platforms like Security Onion, Kibana, or Splunk for visualization and in-depth analysis.

Professional Insights:

The alerts are a critical part of the forensic process, acting as a roadmap for analysts to dive deeper into suspicious activity. By correlating alerts with additional data, investigators can confirm incidents and mitigate potential risks effectively.

3: Creating a Case with Comments for Digital Forensics



Explanation:

This screenshot illustrates the process of creating a case within a forensic analysis platform. This step is crucial for documenting findings, collaborating with other investigators, and maintaining a detailed record of the investigation.

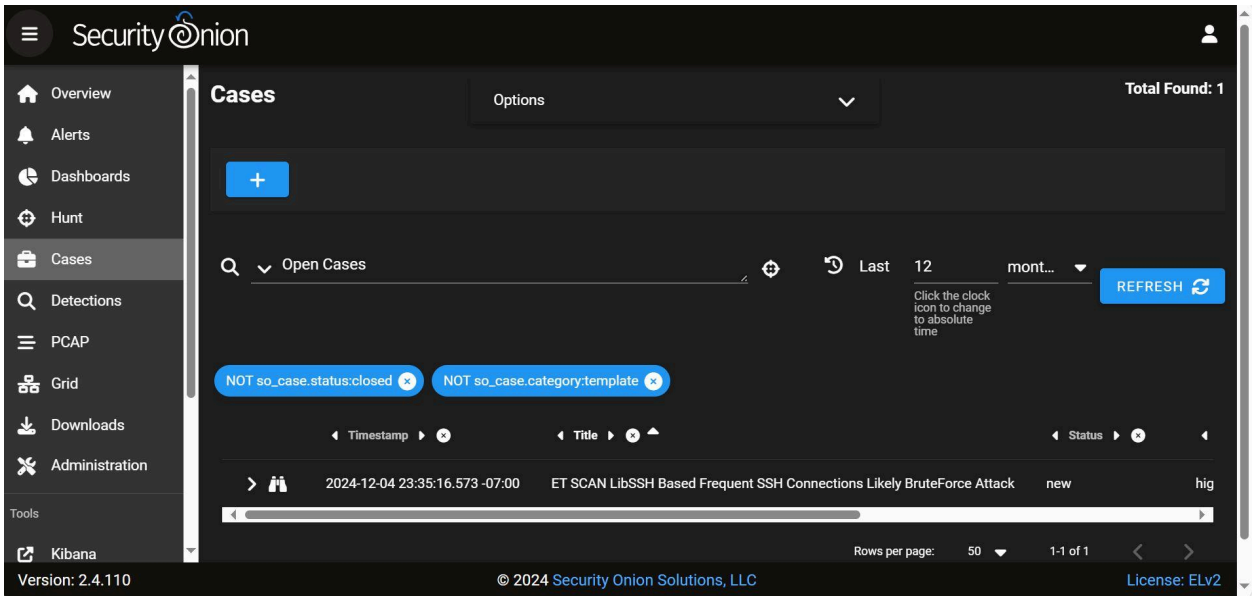
1. **Context and Purpose**:
    - Creating a case is a structured way to document and track forensic investigations. It organizes evidence, observations, and analysis results in a centralized location.
    - Comments or annotations allow investigators to document their thought process, interpretations, or next steps in a way that is easy to follow.
2. **Details in the Screenshot**:
    - **Case Name**: The user assigns a descriptive name to the case, ensuring clarity for future reference.
    - **Case ID or Reference**: A unique identifier may be generated for the case, aiding in systematic tracking.
    - **Comments Section**: The investigator has likely included specific notes related to the findings, such as the type of evidence analyzed, potential leads, or hypotheses.
    - **Platform Interface**: The layout suggests the use of a professional forensic platform, possibly one like TheHive, Autopsy, or EnCase.
3. **Steps Performed**:
    - The user initiated the case creation process by providing a name and optional metadata.
    - A comments field was used to add context or observations, which may include

links to evidence files or summaries of previous analysis steps.
    - The case may now be used to consolidate additional findings, alerts, or artifacts as the investigation progresses.
4. **Relevance to Digital Forensics**:
    - Creating and managing cases is vital for ensuring the integrity of investigations. It maintains a chain of custody for evidence and a clear timeline of events.
    - Cases also facilitate collaboration between team members, enabling seamless sharing of information and tasks.
5. **Technical Tools**:
    - The forensic platform used here seems to support integrated case management features, allowing investigators to link various data points such as network alerts, PCAP files, or host artifacts.
    - Such tools often include search, tagging, and reporting capabilities to enhance workflow efficiency.

Professional Insights:

A well-documented case is the backbone of any forensic investigation. It not only provides clarity to the investigator but also serves as a robust record for legal or compliance purposes. Adding thoughtful comments and observations ensures the investigation remains transparent and defensible.

4 Escalating Alert to Case Analysis



Context and Purpose
This screenshot showcases the **Cases** section of Security Onion, a comprehensive intrusion

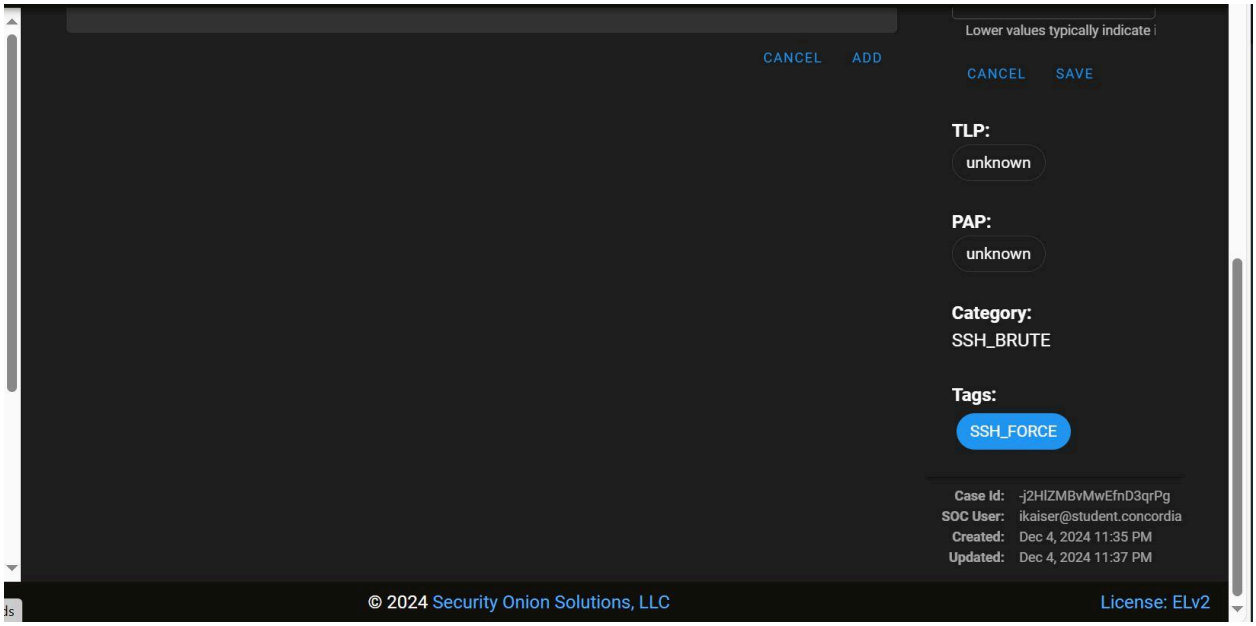
detection and analysis platform. It highlights how alerts generated from detection engines are escalated and tracked as cases. This specific case corresponds to a detection of a potential SSH brute force attack.

Key Features

1. **Filtering Options:**
    - The view filters out closed cases and templates to focus solely on open, actionable incidents.
    - Filters such as `NOT so_case.status:closed` and `NOT so_case.category:template` ensure an investigator sees live and relevant cases.
2. **Case Details:**
    - Title: The case is categorized under "ET SCAN LibSSH Based Frequent SSH Connections Likely BruteForce Attack," indicating suspicious and repetitive SSH connection attempts detected by the system.
    - Timestamps: The timeline displays the creation and last updated times, critical for forensic time-lining.
    - Status: The case is in the "new" state, requiring immediate attention.
    - Priority: The priority level is set to "high," underlining its significance to the organization's cybersecurity posture.

Technical Relevance
This step is integral in forensic workflows where an analyst transitions from detection (alerts) to investigation (case management). It ensures proper documentation and analysis of incidents, which is crucial for reporting, response, and legal purposes if required.

5 Case Metadata Details



Context and Purpose
The second screenshot delves into the metadata and descriptive attributes of the escalated case. It emphasizes organizational and procedural details tied to the forensic investigation.

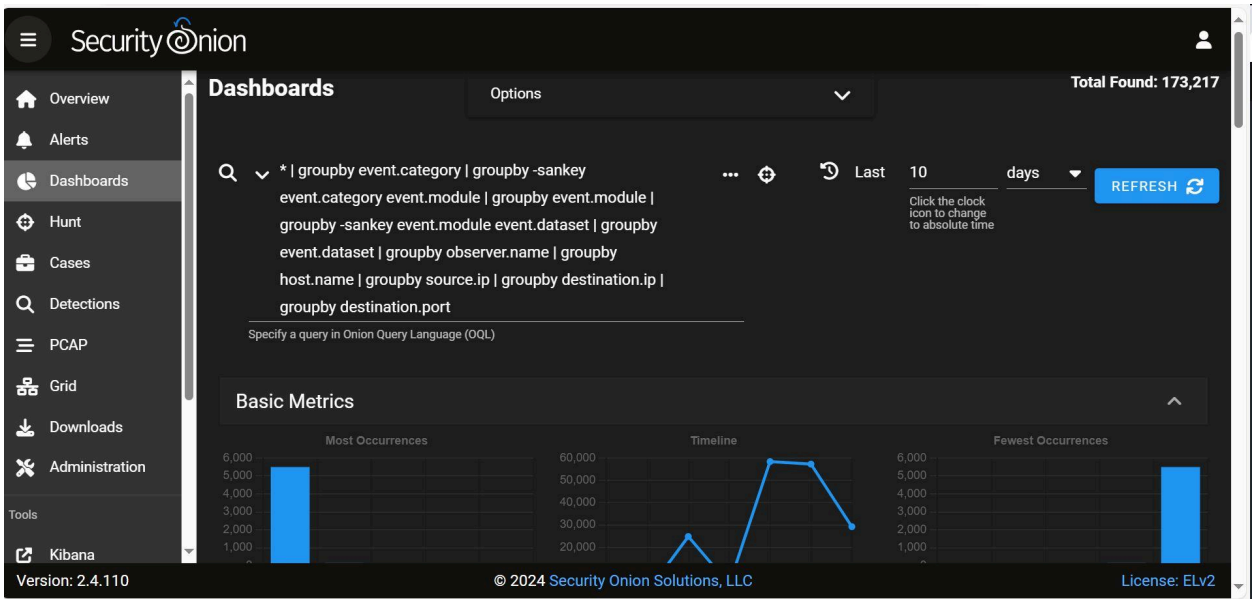
Key Features

1. **Case Metadata:**
    - **TLP (Traffic Light Protocol):** Marked as "unknown," suggesting the sensitivity level of the information has not been classified yet.
    - **PAP (Permissible Actions Protocol):** Also "unknown," indicating that the permissible response or disclosure guidelines for this case are undefined.
    - **Category:** The case is classified as "SSH_BRUTE," signifying its alignment with brute force attack patterns. This classification is valuable for tracking attack trends and generating threat intelligence.
    - **Tags:** The tag "SSH_FORCE" further refines categorization and helps analysts correlate related incidents.
2. **User Attribution:**
    - **SOC User:** The case was created and updated by `ikaiser@student.concordia`, providing a clear chain of custody for accountability.
    - **Timestamp:** The creation and last modification timestamps reflect activity within a span of 2 minutes, showcasing prompt escalation by the analyst.

Technical Relevance

Case metadata forms the backbone of forensic case management. Properly attributed and categorized incidents streamline investigations, enhance collaboration, and serve as a record for future audits or litigation. The use of tags like "SSH_FORCE" aids in correlating similar attack vectors across incidents.

6 Dashboard Metrics for Analysis



Context and Purpose
This screenshot demonstrates the **Dashboards** section of Security Onion. It provides visual insights into the aggregated metrics and trends from monitored events over a specified timeframe. Dashboards are critical for understanding patterns and assessing the overall security landscape.

Key Features

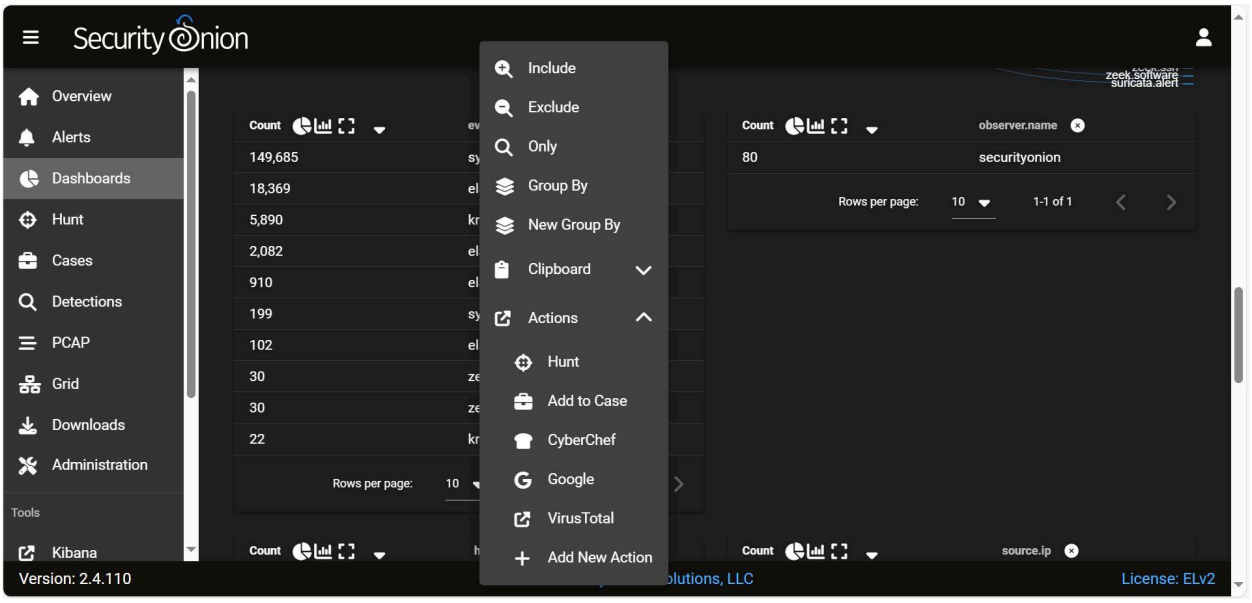
1. **Query Syntax (Onion Query Language):**
    - The query aggregates data by multiple dimensions, such as event category, module, dataset, source and destination IPs, and ports.
    - Grouping by these fields allows an analyst to pinpoint patterns, such as a spike in brute force login attempts targeting specific IPs or services.
2. **Basic Metrics Visualization:**
    - **Most Occurrences:** Bar graphs illustrate event volume across different categories, identifying prominent threats.
    - **Timeline Trends:** The timeline graph tracks the event count over time, allowing analysts to identify trends such as surges in activity that may correlate with

attacks or misconfigurations.

3. **Search Scope:**
    - The timeframe is set to the "Last 10 days," ensuring analysis encompasses recent activity.
    - A total of 173,217 events were found, indicating high activity and the need to narrow down specific event types for actionable insights.

Technical Relevance
Dashboards provide a macro-level view of security incidents, helping investigators prioritize efforts and identify anomalies. The metrics and trends observed can be used to corroborate findings from escalated cases, like the SSH brute force attack, and assess whether they are isolated or part of broader patterns.

7 Adding Observations to a Case (Dashboard Actions Menu)



This screenshot illustrates the **Security Onion Dashboard's Actions menu**, showcasing its utility in digital forensic investigations. The Actions menu offers several options for analysts to refine data or interact with the results.

Key Details:

1. **Menu Features**:
    - **Include/Exclude/Only**: Filtering options for isolating specific data points within the dataset.
    - **Group By/New Group By**: Useful for aggregating data based on selected fields, such as `event.dataset` or `observer.name`.

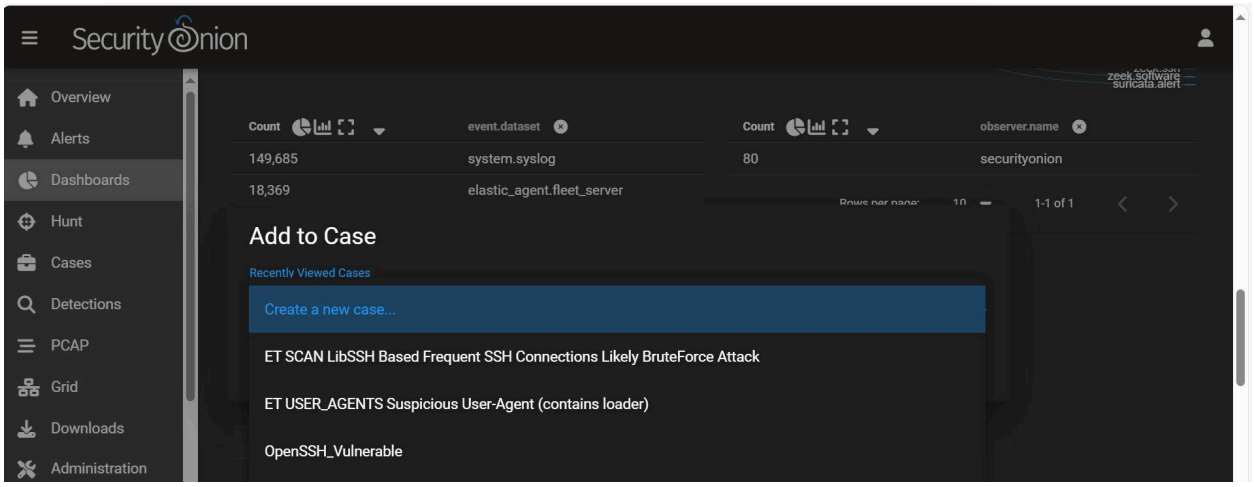
    - **Clipboard/Actions**: Enables copying results for external documentation or integrating with tools like **CyberChef**, **Google**, and **VirusTotal** for further analysis.
2. **Selection Context**:
   The visible query (`observer.name: securityonion`) indicates a filtered dataset focusing on logs generated by a specific Security Onion node, `securityonion`. This helps identify the source of an event.
3. **Relevance to Forensics**:
   This interface facilitates efficient forensic analysis by allowing analysts to:
    - Narrow down investigation scopes.
    - Export relevant logs for correlation across external tools.
    - Add findings directly into an incident case for structured investigation.

Steps Performed:

1. Opened the **Security Onion Dashboard** under `Dashboards`.
2. Filtered the results by `observer.name`.
3. Accessed the **Actions menu** for next steps in analysis or case-building.

This capability emphasizes **incident management** and **collaboration tools**, ensuring forensic activities are streamlined.

8 Adding Data to a Case



This image showcases the **"Add to Case" feature**, a critical aspect of Security Onion's case management workflow, where suspicious events are added to specific cases for detailed investigation.

Key Details:

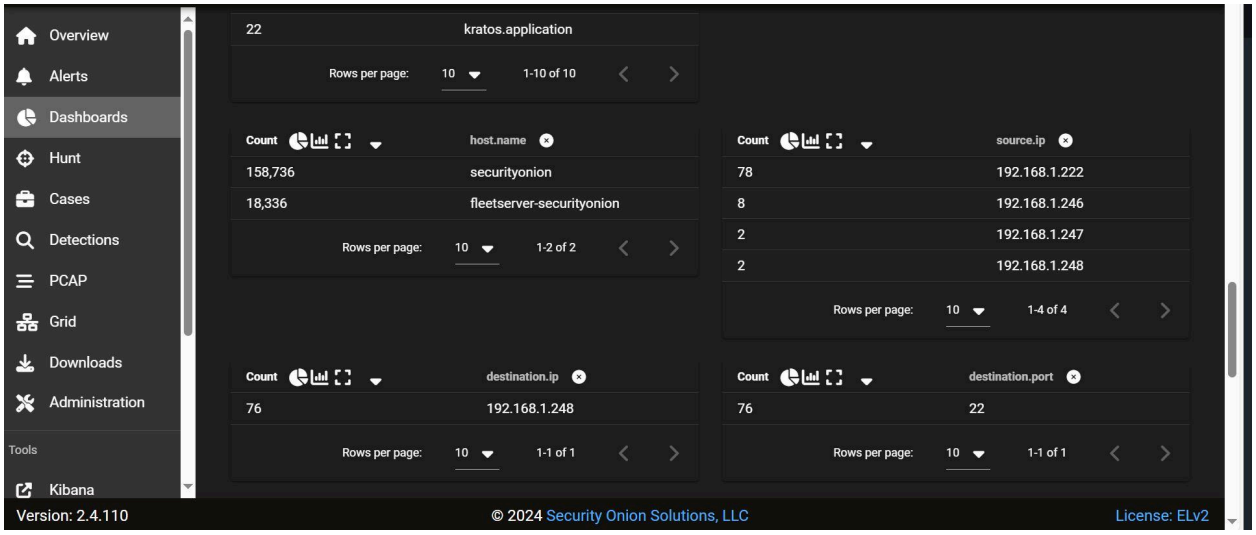
1. **Add to Case Workflow**:
    - Provides an option to create a **new case** or add events to recently viewed cases.
    - Cases listed include:
        - **ET SCAN LibSSH Based Frequent SSH Connections Likely BruteForce Attack**: Indicates malicious SSH activity.
        - **ET USER_AGENTS Suspicious User-Agent**: Highlights anomalous HTTP User-Agent headers, possibly used for reconnaissance.
        - **OpenSSH_Vulnerable**: Represents an exploitation of known OpenSSH vulnerabilities.
2. **Event Context**:
    - The **filtered query** (`event.dataset`) focuses on log types such as `system.syslog` and `elastic_agent.fleet_server`.
    - These datasets are critical for detecting anomalies or ensuring system integrity.
3. **Relevance to Forensics**:
    - Case management ensures forensic integrity by maintaining a structured repository of all investigative actions and findings.
    - Organizing events under specific cases enables efficient reporting and evidence collection.

Steps Performed:

1. Filtered events by dataset types on the Dashboard.
2. Chose the "Add to Case" action, displaying options for creating or updating cases.
3. Selected an appropriate case or created a new one to log findings.

This process is pivotal in documenting events while adhering to forensic principles of **chain of custody**.

9 Filtering Results Based on Source and Destination Details



This screenshot demonstrates an analysis on the **source and destination IPs and ports**, utilizing Security Onion's filtering and aggregation capabilities.

Key Details:

1. **Filter Context**:
    - Data is grouped by several fields:
        - **Source IPs**: Includes addresses such as `192.168.1.222` and `192.168.1.246`, possibly identifying devices initiating connections.
        - **Destination IP**: `192.168.1.248` is the target, with further analysis suggesting potential lateral movement or exfiltration attempts.
        - **Port**: Only port `22` (SSH) is visible, indicating the traffic pertains to secure shell connections.
2. **Counts and Patterns**:
    - The event counts help prioritize investigation (e.g., source IP `192.168.1.222` with the highest interaction at `78`).
    - SSH activity might signify administrative access or brute-force attacks.
3. **Relevance to Forensics**:
    - Identifying IPs and ports in communication helps trace the origin of suspicious traffic.
    - Logs serve as evidence of malicious or unauthorized access attempts.

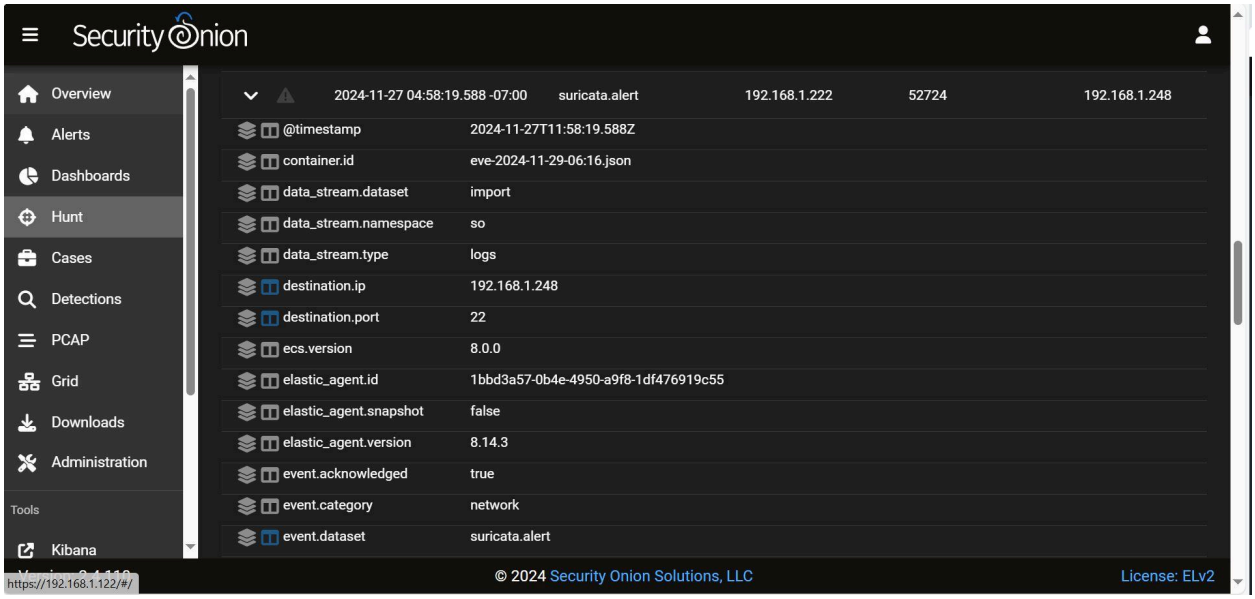
Steps Performed:

1. Accessed the **Dashboards** menu and selected relevant data filters (`source.ip`, `destination.ip`, and `destination.port`).
2. Applied visualizations to aggregate traffic counts and identify high-priority sources.

3. Prepared findings for further analysis or addition to a case.

This insight is vital for detecting intrusion attempts and correlating activity across multiple systems during forensic investigations.

10 (Hunt View in Security Onion)



This screenshot represents a "Hunt" view from the Security Onion platform, a specialized tool widely utilized in digital forensics and network security monitoring. The purpose of the investigation is to analyze a Suricata alert triggered by suspicious network activity.

Key Observations:

1. **Timestamp:**
   - Event occurred at `2024-11-27T11:58:19.588Z`. Precise timestamps are crucial in digital forensics for event correlation across systems and timelines.
2. **Source and Destination Details:**
   - **Source IP:** `192.168.1.222`, with source port `52724`.
   - **Destination IP:** `192.168.1.248`, targeting port `22` (SSH).
   - This interaction suggests potential malicious activity attempting to exploit SSH, such as brute force or unauthorized access.
3. **Suricata Alert Details:**
   - The dataset name (`suricata.alert`) confirms that this log is generated by Suricata, a network intrusion detection and prevention system.
   - The data stream is categorized under `logs` with namespace `so` (Security Onion).

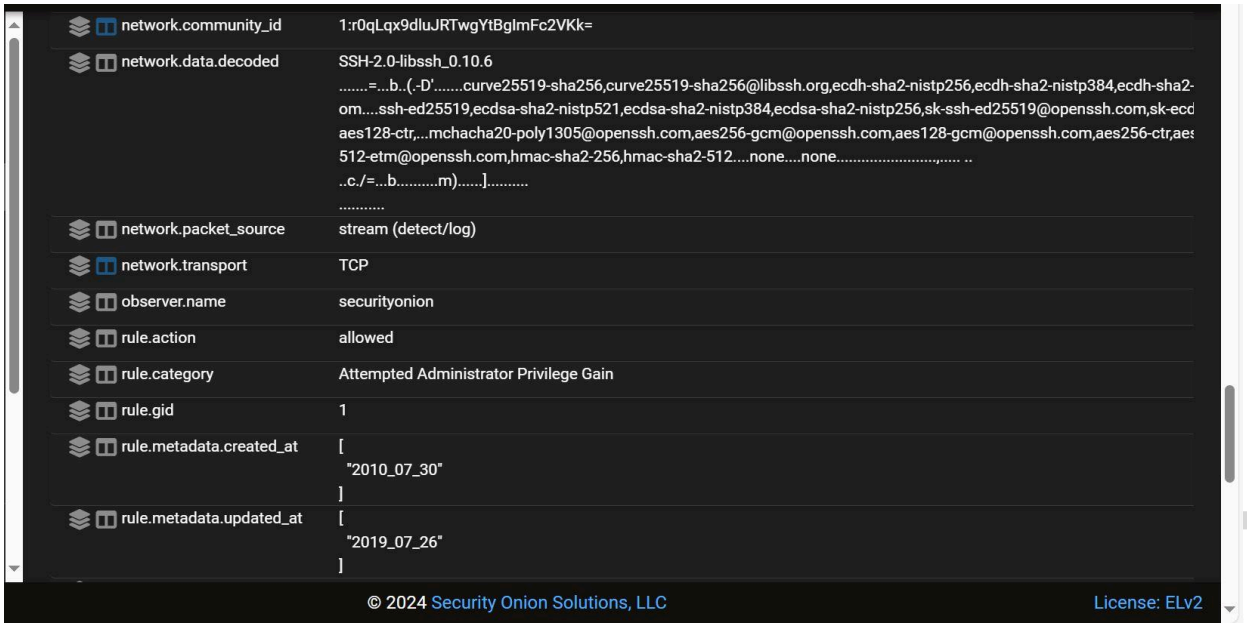
4. **Elastic Agent and Event Categorization:**
    - The Elastic Agent version `8.14.3` processed and categorized this event under the `network` category, providing clarity on the type of observed activity.

Relevance to Digital Forensics:

This log provides initial evidence of a potential intrusion attempt. It highlights the importance of analyzing SSH traffic, especially when unauthorized IPs target port `22`. The forensic process will involve:

- Cross-referencing this event with other system logs.
- Inspecting SSH configurations and authentication attempts on `192.168.1.248`.
- Identifying potential brute force patterns or unauthorized login attempts.

11 **(Network and Rule Details)**



This image dives deeper into the context of the network traffic and rules triggered by the Suricata alert.

Key Observations:

1. **Network Context:**
    - The `network.community_id` uniquely identifies this connection, useful for correlating related traffic across systems.
    - **Decoded Data:** Captured SSH handshake (`SSH-2.0-libssh_0.10.6`),

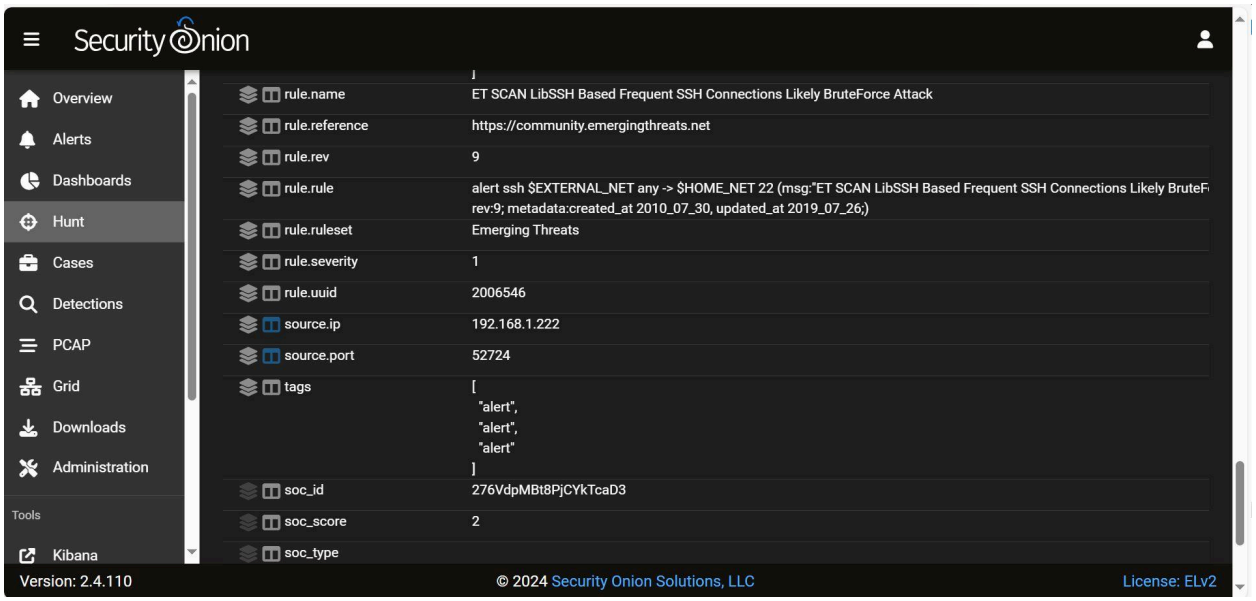
detailing the cryptographic algorithms used (e.g., `curve25519-sha256`, `aes128-ctr`).
    - **Transport Protocol:** TCP.
2. **Observer and Rule Actions:**
    - Observer (`securityonion`) monitored and flagged the traffic.
    - Rule action: `allowed`, though it categorizes the event as an attempted administrator privilege gain.
3. **Rule Metadata:**
    - **Category:** "Attempted Administrator Privilege Gain."
    - Indicates potential exploitation of administrative access.
    - **Dates:** Rule creation in `2010` and update in `2019`—demonstrates its relevance in recognizing older or evolving threats.

Relevance to Digital Forensics:

Detailed analysis of the captured network handshake data and associated rule metadata can:

- Uncover specific threat vectors targeting SSH.
- Highlight weaknesses in SSH configuration or outdated cryptographic protocols.
- Correlate network packet sources for deeper investigation, particularly if the source IP (`192.168.1.222`) is linked to known malicious activities.

12 (Alert Metadata and Threat Intelligence)



This screenshot contains metadata and rules that provide a summary of the alert and its potential impact.

Key Observations:

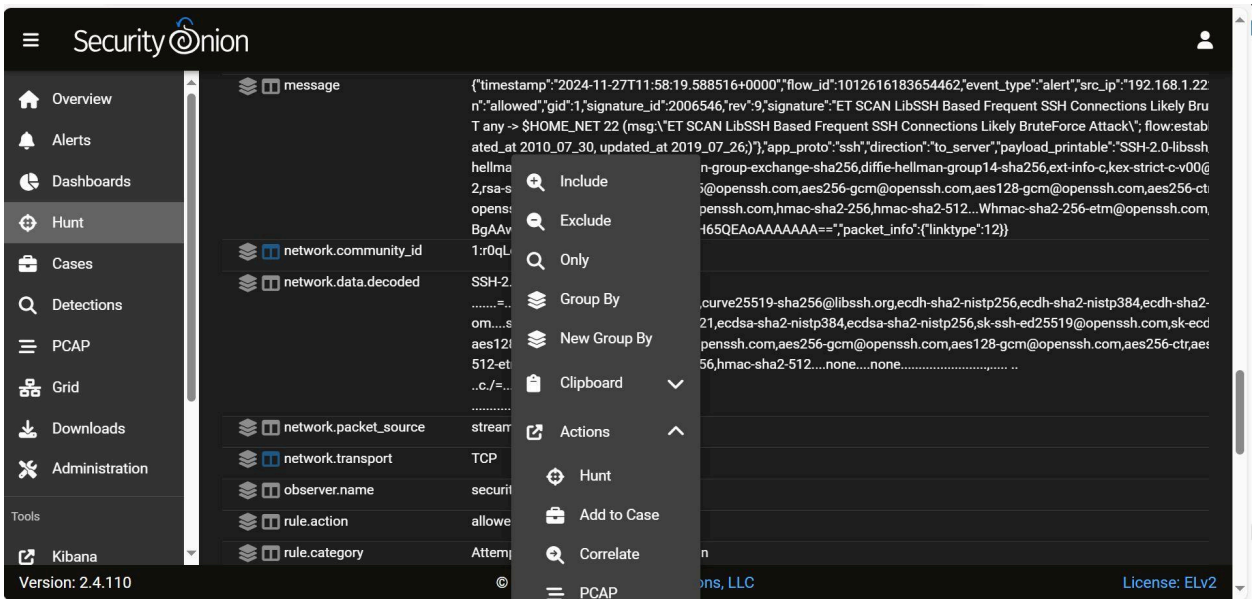
1. **Rule Name and Threat Reference:**
    - Rule: `ET SCAN LibSSH Based Frequent SSH Connections Likely BruteForce Attack`.
    - Reference: Emerging Threats, a threat intelligence source.
    - This strongly indicates a brute force attempt on SSH.
2. **Rule Severity:**
    - **Severity Level:** 1 (low), suggesting this could be the initial phase of a potential attack or scan activity.
    - Brute force scans are often precursors to privilege escalation or lateral movement.
3. **Source and Tags:**
    - **Source IP:** `192.168.1.222` is confirmed as the origin of the malicious activity.
    - Repeated tagging of the event with `"alert"` highlights its significance for attention in larger datasets.

Relevance to Digital Forensics:

The metadata provides strong contextual support for identifying the nature of the event as a likely brute force attack. The next forensic steps include:

- Identifying patterns of repeated access attempts from `192.168.1.222`.
- Checking threat intelligence databases for matches with the observed IP and its activity.
- Performing SSH server log analysis for failed authentication attempts, unusual logins, or additional compromise indicators.

13 Security Onion - Hunt Interface



Description

This screenshot showcases the Security Onion Hunt interface, which provides a detailed view of network and log data for analysis. The Hunt feature is used to search, filter, and analyze observables across captured network packets and logs.

Details

- **Tool**: Security Onion (Version 2.4.110)
- **Functionality Highlighted**: The Hunt tool is being used to analyze an alert related to an SSH brute force detection.
- **Key Observations**:
  - The alert indicates suspicious SSH activity flagged by a Suricata rule (`ET SCAN LibSSH Based Frequent SSH Connections Likely Brute Force Attack`).
  - Source IP: `192.168.1.222`, which is flagged as initiating the potential brute force attack.
  - The decoded payload (`network.data.decoded`) includes SSH handshake details and cryptographic algorithms used (e.g., `aes256-ctr`, `hmac-sha2-512`), allowing analysts to understand the session characteristics.
  - Rule action: "Allowed" indicates that the network traffic was permitted despite being flagged as suspicious.

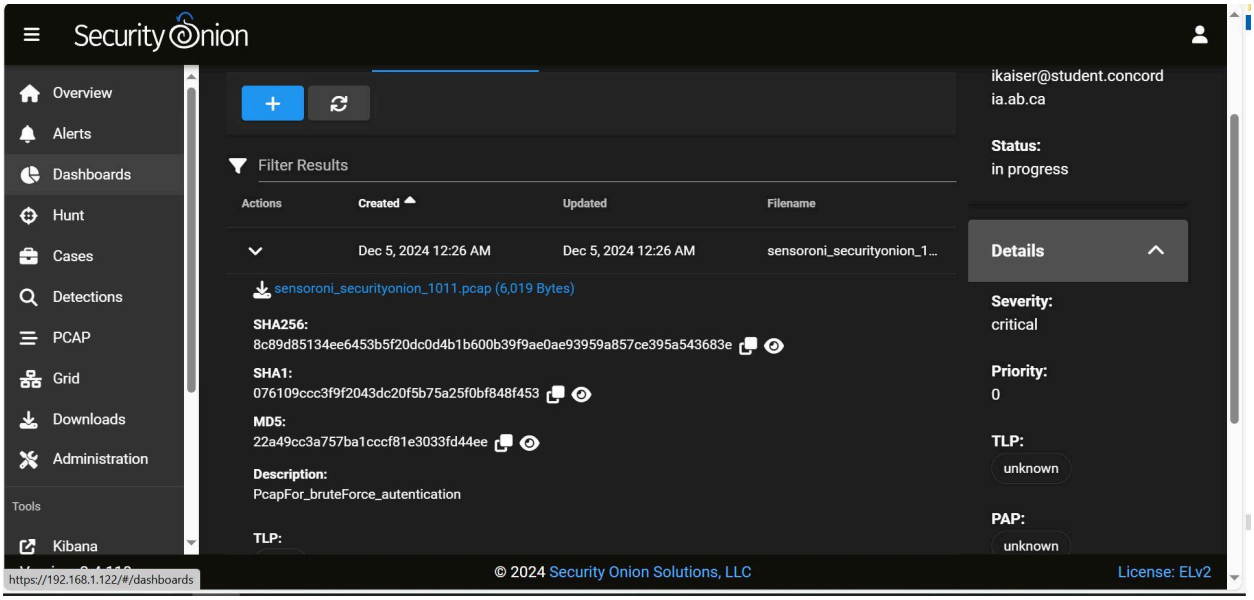
Steps Performed

1. **Search and Filtering**: The analyst used filters such as "Include," "Exclude," or "Group By" (shown in the dropdown menu) to refine the dataset for investigation.
2. **Payload Inspection**: The Hunt interface provides a decoded view of network data, allowing analysts to inspect SSH session details directly.
3. **Correlations and Actions**: The dropdown options allow analysts to correlate the alert with other events, add it to a case, or download the associated PCAP for deeper inspection.

Relevance to Digital Forensics

The information provided here is critical for identifying and responding to potential network intrusions. By analyzing SSH session details and correlating them with threat intelligence, an analyst can determine whether this activity is part of a larger attack campaign or an isolated incident.

14 PCAP Attached to Case



File Reference: `sensoroni_securityonion_1011.pcap`

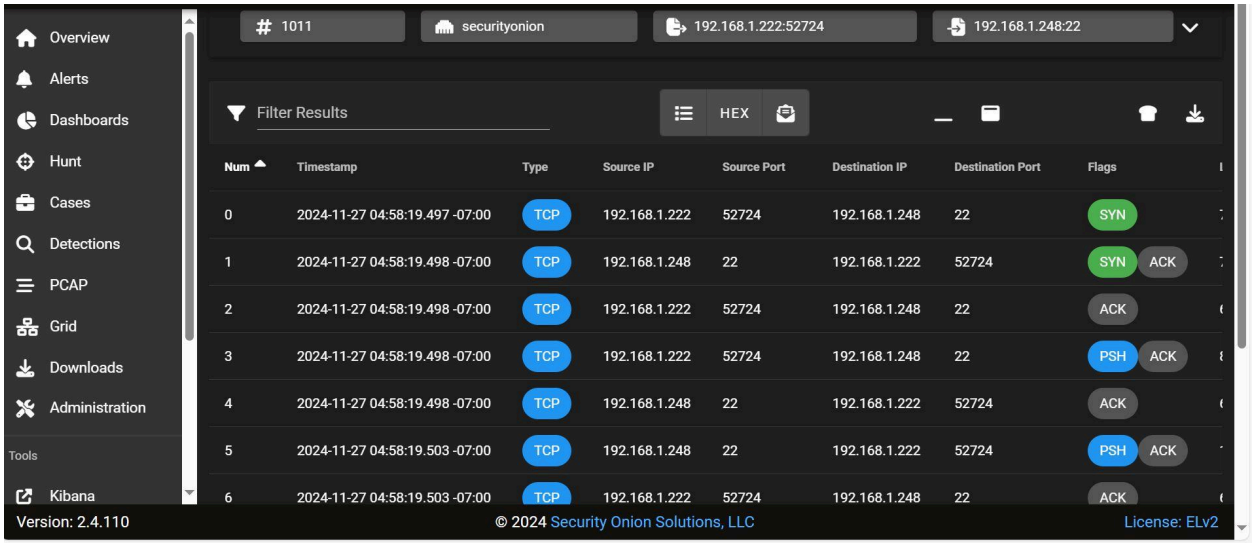
This screenshot captures the attachment of a specific packet capture (PCAP) file to a case in Security Onion's Dashboard interface. The details include:

1. **File Metadata**:
    - **Filename**: `sensoroni_securityonion_1011.pcap`
    - **File Size**: 6,019 bytes
    - **Hash Values**:
        - **SHA256**: `8c89d85134ee6453b5f20dc0d4b1b600b39f9ae0ae93959a857ce395a543683e`
        - **SHA1**: `076109ccc3f9f2043dc20f5b75a25f0bf848f453`
        - **MD5**: `22a49cc3a757ba1cccf81e3033fd44ee`
2. **Case Details**:
    - **Description**: `PcapFor_bruteForce_authentication`
    - **Severity**: Critical
    - **Priority**: 0 (highest priority)
3. **Relevance to Digital Forensics**:
    - The PCAP file likely contains network traffic data related to brute-force authentication attempts. It is crucial for forensic analysis to determine the source and nature of the attack.
    - The hash values ensure data integrity and authenticity, which are vital for presenting evidence in legal proceedings.
    - Security Onion provides a centralized way to document and track such artifacts in

a case-management workflow, allowing forensic investigators to correlate findings.

4. **Tools and Workflow**:
    - Security Onion is used for network monitoring and intrusion detection. The PCAP was likely captured via Suricata or Zeek and stored for further analysis.
    - Hash verification can be performed using tools such as `shasum` or `md5sum` to validate the file's authenticity.

15 Packet Analysis via Hunt Interface

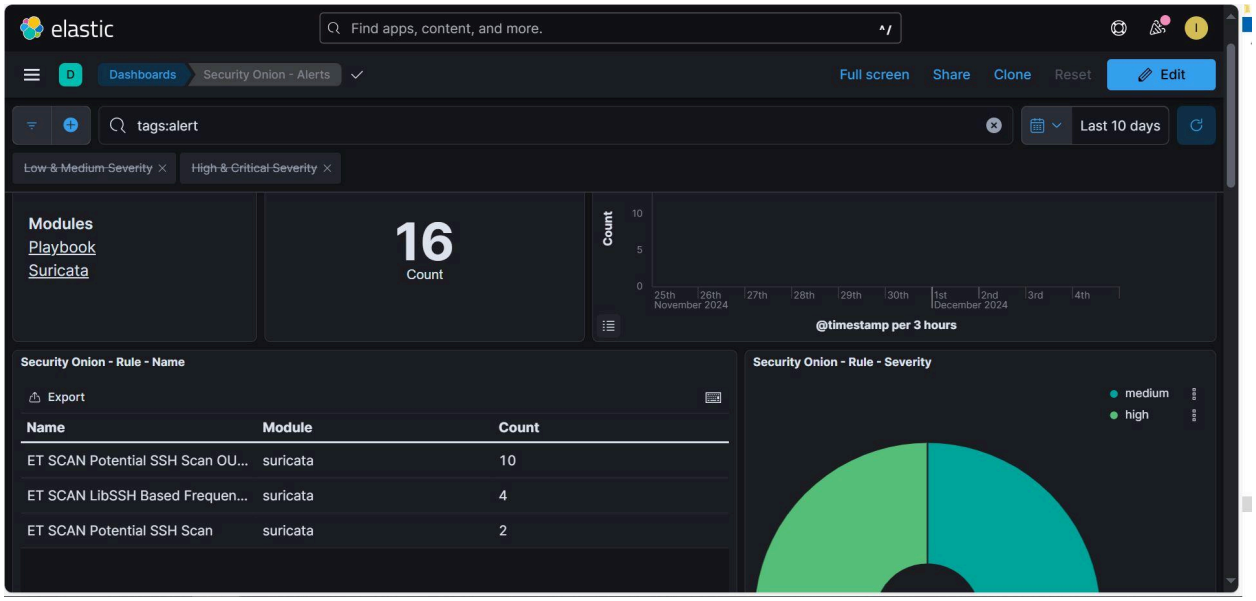


Description: Packet flow between two hosts (`192.168.1.222` and `192.168.1.248`) is visualized, detailing a potential brute-force attempt over SSH (port 22).

1. **Key Observations**:
    - **Source IP**: `192.168.1.222`
    - **Destination IP**: `192.168.1.248`
    - **Source Port**: `52724`
    - **Destination Port**: `22` (SSH service)
    - **TCP Flags**: SYN, ACK, PSH (indicative of session establishment and data transfer).
2. **Timestamp Details**:
    - Multiple requests logged within milliseconds (`04:58:19.497` to `04:58:19.503`), suggesting a high volume of connection attempts—a typical brute-force behavior.

3. **Relevance to Digital Forensics**:
    - Identifying brute-force behavior is critical in detecting unauthorized access attempts.
    - The packet headers show the handshake process, indicating repeated session initiation (SYN flag). Such patterns could suggest automated tools used to guess credentials.
    - The timeline of events enables reconstruction of the attack sequence for evidence collection.
4. **Tools Used**:
    - Security Onion's Hunt interface enables filtering and detailed traffic inspection.
    - Analysts can export the flow data or drill down to payloads for deeper inspection.

16 Alert Dashboard in Kibana

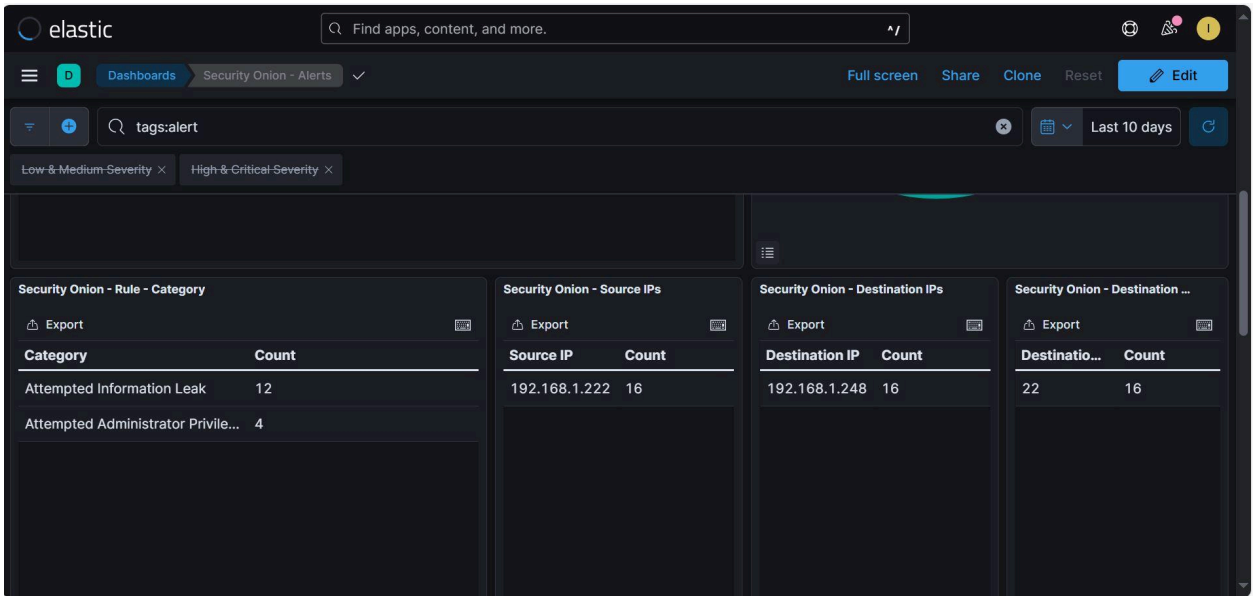


Details: Security Onion alerts generated by Suricata are visualized in Kibana, providing a summary of potentially malicious activity.

1. **Alert Statistics**:
    - **Total Alerts**: 16
    - **Modules Involved**:
        - Suricata: 16 alerts (e.g., `ET SCAN Potential SSH Scan`, `ET SCAN LibSSH Based Frequency`).
2. **Severity Breakdown**:
    - Medium: Certain events flagged as potentially suspicious but requiring further

analysis.
    - High: Critical alerts such as `Potential SSH Scan` indicate active reconnaissance or attacks targeting SSH services.
3. **Technical Details**:
    - **ET SCAN Potential SSH Scan**: This rule is triggered when Suricata detects traffic patterns consistent with SSH scanning or brute-force attempts.
    - The timeline visualization allows analysts to correlate alerts with known events or anomalies in the network.
4. **Relevance to Digital Forensics**:
    - Alerts provide early indicators of compromise (IOC) and guide investigators toward suspicious traffic patterns.
    - The integration of Kibana with Security Onion enables powerful visual analytics, allowing for better situational awareness.
5. **Workflow**:
    - Suricata generates alerts based on predefined rule sets.
    - Investigators use Kibana dashboards to triage alerts, identify patterns, and determine next steps for containment or deeper forensic analysis.

17 Kibana Dashboard - Security Onion Alerts Overview



Context:

The first screenshot captures a section of a Kibana dashboard under the "Security Onion - Alerts" module, showcasing data aggregated over the last 10 days. This screenshot primarily highlights alerts generated by the system and categorizes them based on activity, source IPs, and destination IPs.

Analysis:

1. **Categories of Alerts**:
    - The alert data is grouped into two main categories:
        - **Attempted Information Leak**: There were **12 instances** of this alert, indicating potential unauthorized attempts to extract sensitive information from the monitored environment.
        - **Attempted Administrator Privilege Escalation**: This category recorded **4 instances**, reflecting possible efforts to gain unauthorized administrative access.
    - These categories are crucial in understanding the type and nature of potential security breaches and help in prioritizing investigation efforts.
2. **Source IP Analysis**:
    - The source IP **192.168.1.222** was flagged in **16 alerts**, making it a significant point of interest. This indicates suspicious activity originating from this internal IP address.
3. **Destination IP Analysis**:
    - All 16 alerts targeted a specific destination IP, **192.168.1.248**. This suggests that the activity may have been concentrated on a specific asset or service within the network.
    - The frequent targeting of this IP could indicate an attempt to exploit known vulnerabilities or gain unauthorized access.

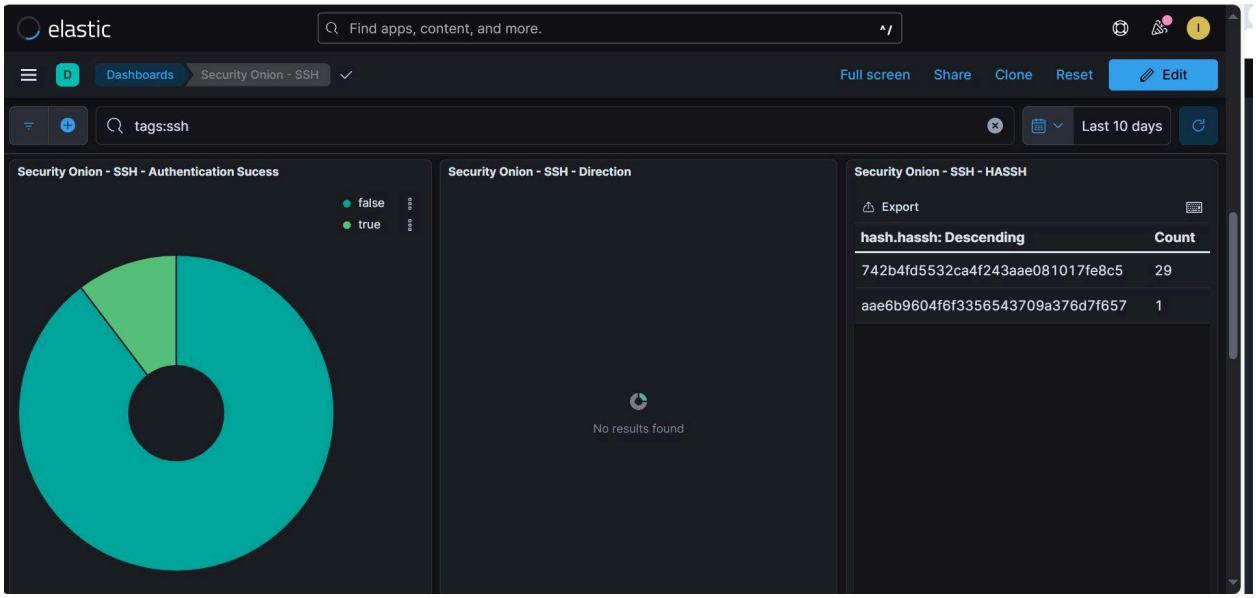
Tools Used:

- **Kibana**: A data visualization and exploration tool used to analyze Security Onion alerts.
- **Security Onion**: An open-source platform for network monitoring, intrusion detection, and log management.

Relevance to Digital Forensics:

This dashboard provides insights into potential malicious activities within a network, assisting forensic investigators in identifying compromised systems, tracking the origin of attacks, and determining the extent of breaches. The detailed categorization of alerts helps investigators prioritize their efforts and streamline incident response.

18 Kibana Dashboard - SSH Authentication Analysis



Context:

The second screenshot focuses on SSH-related events, extracted from the "Security Onion - SSH" dashboard. It highlights authentication success data and the associated hash values, suggesting analysis of SSH traffic over the last 10 days.

Analysis:

1. **Authentication Success Rates**:
    - The pie chart distinguishes between successful and unsuccessful SSH authentications:
        - **"False" (failed authentications)** accounts for the majority, signifying possible brute force attacks or unauthorized login attempts.
        - **"True" (successful authentications)** forms a smaller fraction, implying fewer legitimate or successful attempts.
2. **Hash Analysis**:
    - Two distinct hash values are listed in descending order of their frequency:
        - **Hash 1**: `742b4fd5532ca4f243aae081017fe8c5` appeared **29 times**, likely representing a recurring user credential or SSH fingerprint in multiple events.
        - **Hash 2**: `aae6b9604f6f3356543709a376d7f657` was recorded **1 time**, which could indicate a unique or isolated event.
    - These hashes could represent cryptographic fingerprints associated with SSH sessions, such as session keys, usernames, or authentication tokens.
3. **Direction Data**:
    - The "Direction" panel indicates no data was found, suggesting either a lack of

classification for inbound/outbound traffic or a filtering issue in the query.

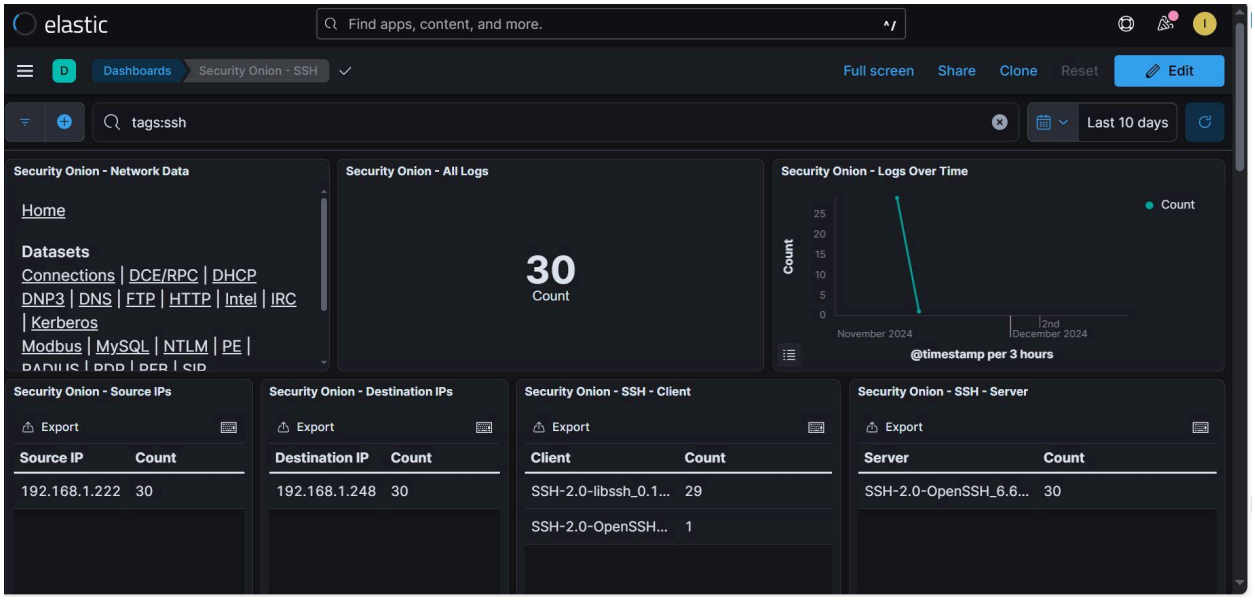
Tools Used:

- **Kibana**: To visualize and query SSH traffic events.
- **Security Onion**: For capturing and processing SSH session logs.

Relevance to Digital Forensics:

SSH logs are crucial in forensic investigations, especially when investigating unauthorized access. The high number of failed authentications could indicate an active brute force attack, while the hash analysis helps in correlating SSH sessions with users or systems. This data aids in identifying compromised accounts or credentials and assessing the scope of the intrusion.

19 Elastic - SSH Dashboard Summary



Description

This screenshot highlights the Elastic Kibana interface used to visualize network and log data, specifically focusing on SSH activity captured over the last 10 days.

Details

- **Tool**: Elastic Kibana integrated with Security Onion.
- **Dashboard Name**: `Security Onion - SSH`.
- **Key Metrics**:

- **Total Count**: 30 SSH-related events detected.
- **Source IPs**: `192.168.1.222` initiated all SSH events (count: 30), indicating a potential source of suspicious activity.
- **Destination IPs**: `192.168.1.248` is the target of the SSH events.
- **SSH Client**: The logs show variations in SSH clients used (`libssh_0.1` for 29 events and `OpenSSH` for 1 event).
- **SSH Server**: All sessions connected to an `OpenSSH_6.6` server.

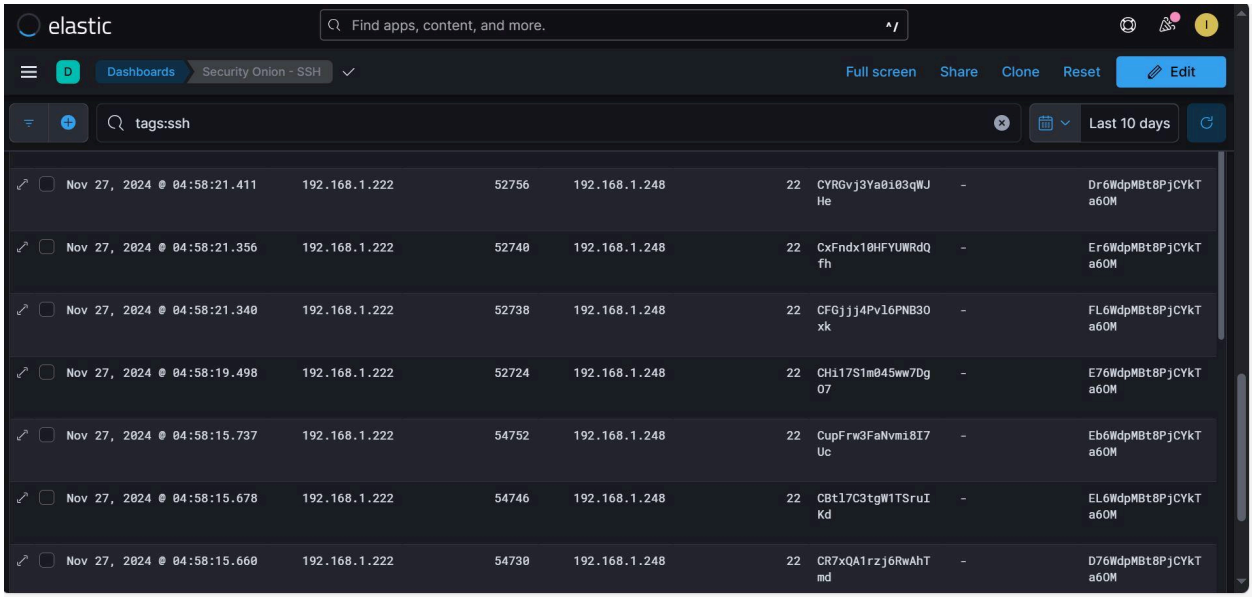
Steps Performed

1. **Tag-based Search**: The keyword `tags:ssh` was used to filter and focus the dashboard on SSH-specific events.
2. **Visualization and Analysis**: Pre-built widgets display key metrics, including the count of events over time, source and destination IPs, and SSH protocol details.
3. **Drill-Down Options**: The analyst can export the data or further refine the time range for more precise insights.

Relevance to Digital Forensics

This dashboard provides an at-a-glance summary of SSH traffic trends and anomalies. It allows forensic investigators to identify patterns (e.g., frequent connections from a single source IP) that may indicate malicious activity such as brute force attacks or unauthorized access attempts.

20 Elastic - Detailed Network Connections View



Description

This screenshot shows the detailed log view in Kibana, listing individual network connection events related to SSH traffic.

Details

- **Tool**: Elastic Kibana.
- **Search Query**: `tags:ssh`.
- **Key Observations**:
    - The logs reveal timestamps, source and destination IPs, ports, and unique session IDs for each connection.
    - Source IP (`192.168.1.222`) and destination IP (`192.168.1.248`) are consistently involved across all events.
    - Destination port `22` (SSH) confirms the focus on secure shell traffic.
    - Unique identifiers (`CYRGvj3Ya0i03qWJHe`) provide session-level granularity for tracking and correlating events.

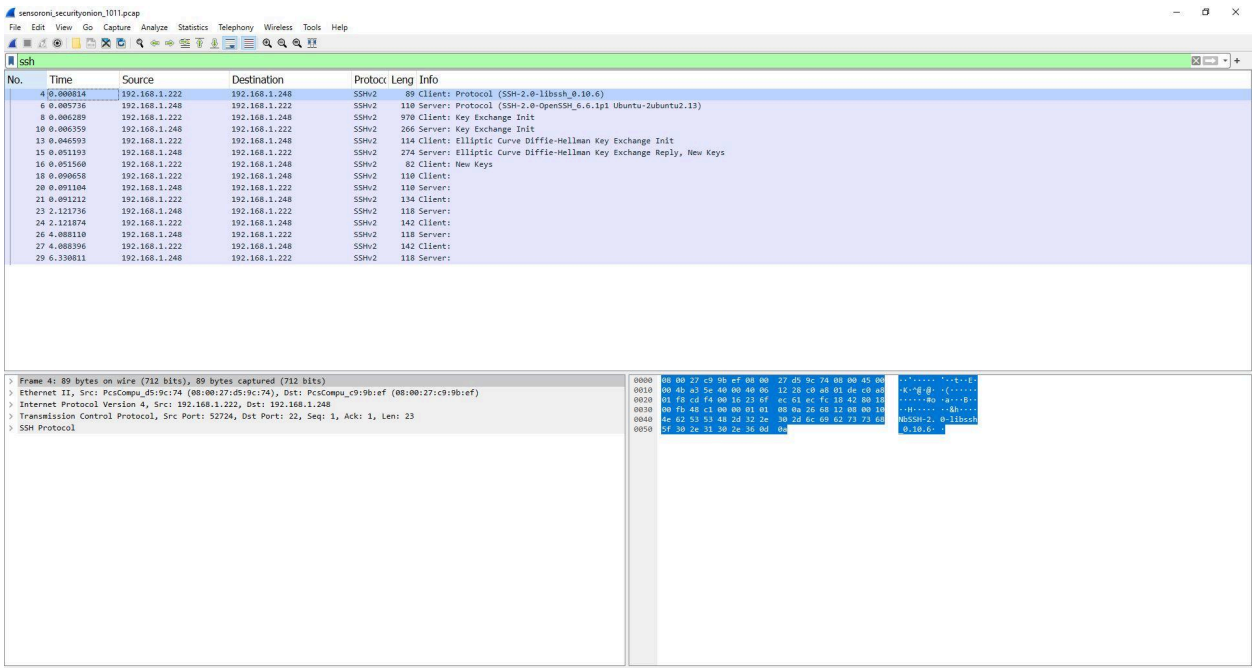
Steps Performed

1. **Event Filtering**: The analyst filtered the logs to focus on SSH events using specific tags.
2. **Inspection**: Details such as timestamps and session IDs were reviewed to establish the timeline and pattern of connections.
3. **Correlation**: Logs can be cross-referenced with other dashboards (e.g., SSH client/server details) to understand the nature and source of activity.

Relevance to Digital Forensics

Detailed network logs are essential for reconstructing the timeline of an incident. By analyzing connection details, investigators can identify the scale of an attack, track the source, and gather evidence for remediation and potential legal actions.

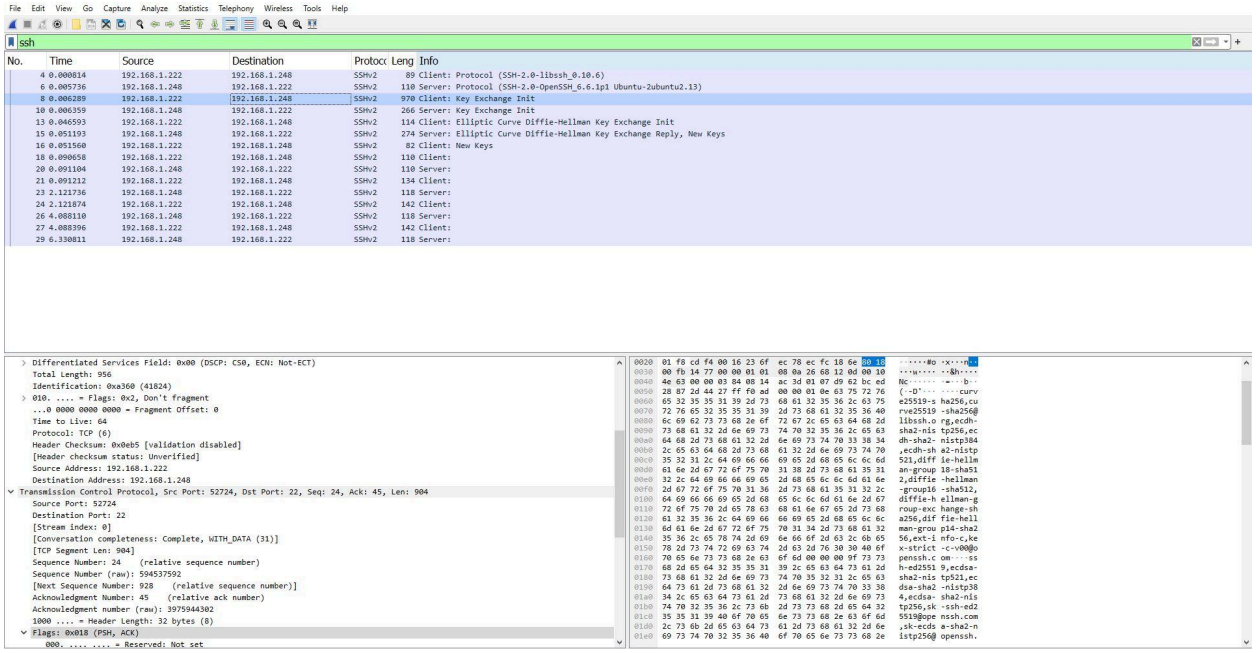
21. wireshark



This screenshot represents a packet capture session analyzed in Wireshark, showcasing SSH protocol communications. Here are the details:

1. **Context of Analysis:**
   - The capture file indicates SSHv2 (Secure Shell version 2) traffic between two IP addresses: `192.168.1.222` (client) and `192.168.1.248` (server).
   - This is part of the process of securing remote communication channels.
2. **Captured Packets:**
   - The initial packet establishes the protocol version used by the client (`SSH-2.0-libssh_0.10.6`) and the server (`SSH-2.0-OpenSSH_6.6.1p1`).
   - Subsequent packets involve the key exchange, specifically the Diffie-Hellman elliptic curve variant. This step ensures secure encryption between the client and server.
3. **Significance in Digital Forensics:**
   - Observing SSH traffic can help investigate unauthorized access to systems, as SSH is often used for secure logins.
   - Analyzing specific key exchange methods and identifying encryption versions are critical for assessing vulnerabilities or outdated configurations.
4. **Technical Notes:**
   - **Source and Destination Ports:** The client uses a dynamic source port (`52724`), and the server communicates on the default SSH port (`22`).
   - **Key Exchange Process:** Includes Diffie-Hellman elliptic curve and "new keys" notifications, signaling the handshake's completion.

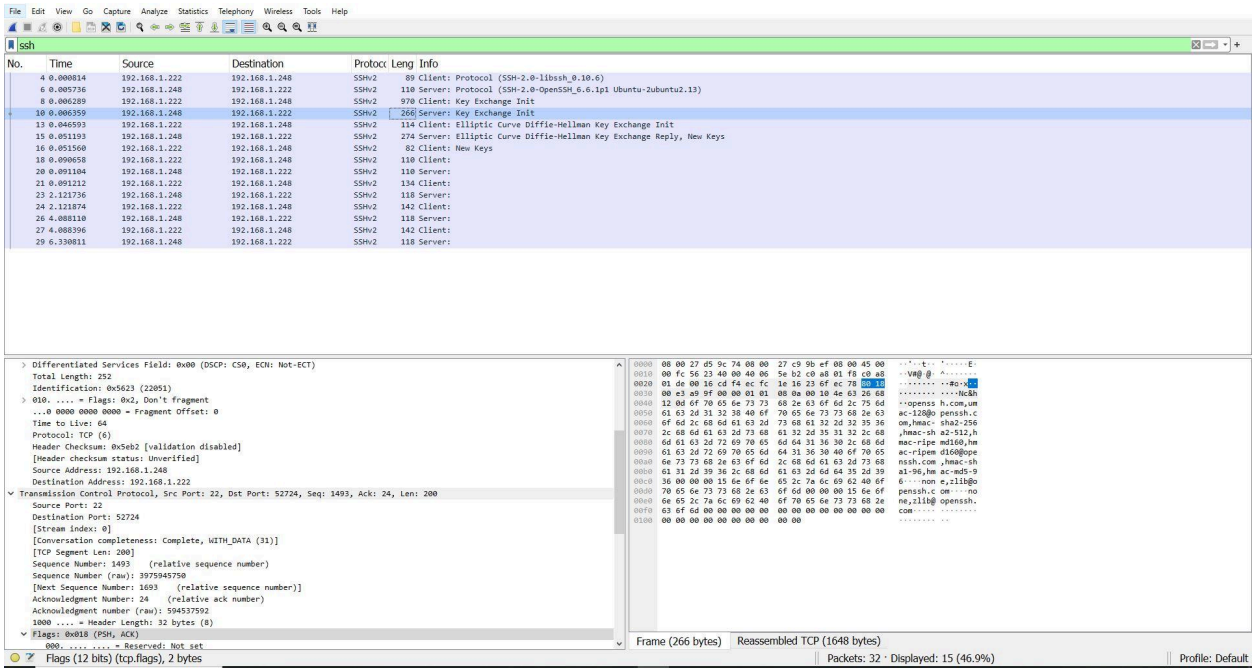
22 wiresharkflags_client



This screenshot focuses on a specific packet exchange during the SSH handshake.

1. **Context of Analysis:**
    - The view highlights the details of a packet originating from the client (`192.168.1.222`) during the key exchange process.
    - Detailed fields such as source/destination IP addresses, TCP headers, and payload information are displayed.
2. **Packet Analysis:**
    - **TCP Header:** The flags section shows `PSH, ACK`, indicating that the client is sending data and expects an acknowledgment from the server.
    - **Sequence and Acknowledgment Numbers:** Sequence (`904` relative) and acknowledgment (`45`) numbers confirm reliable data transfer.
3. **Forensic Relevance:**
    - The `PSH` (Push) flag signals the urgency of delivering this data, making it an essential packet for establishing the session's encryption context.
    - By reconstructing such packets, forensic analysts can investigate SSH-related attacks or analyze secure communication anomalies.
4. **Additional Observations:**
    - The client initiates specific encryption configurations (`ecdh-sha2-nistp256`, `aes128-ctr`) found in the packet payload. This demonstrates the selection of cryptographic protocols during negotiation.

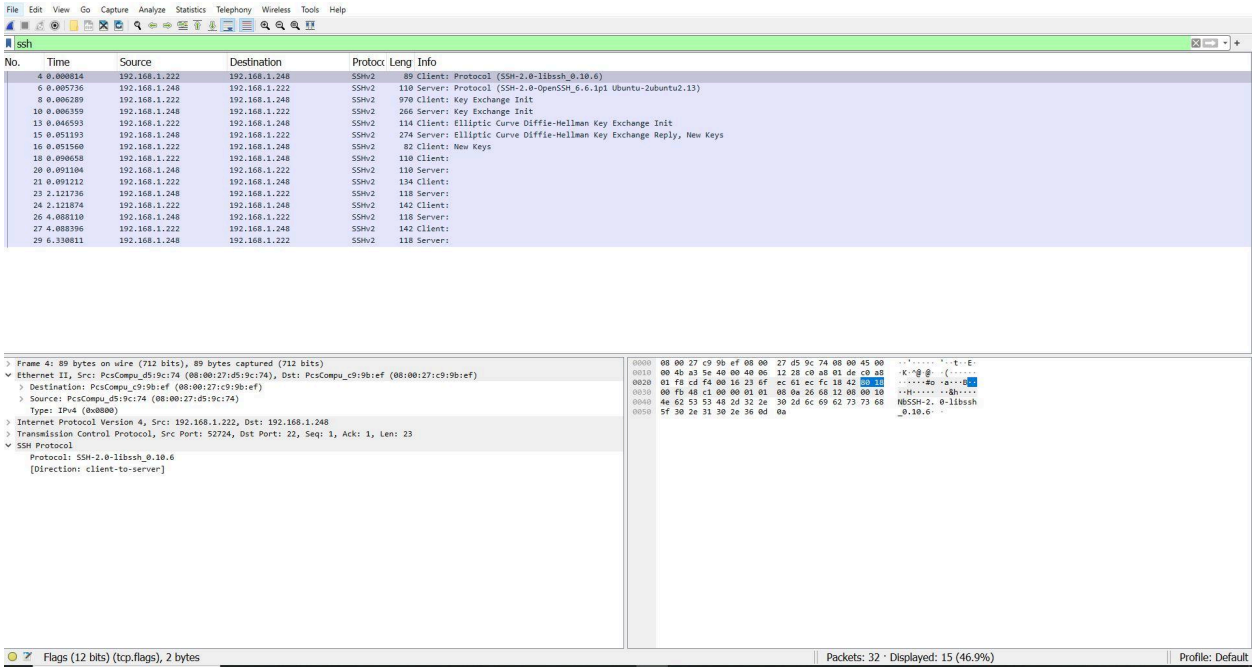
23 wiresharkflags_server



his screenshot examines a response packet from the server (`192.168.1.248`) during the SSH handshake.

1. **Context of Analysis:**
    - The server's response includes cryptographic protocol negotiations, ensuring compatibility with the client.
    - This packet confirms the server's readiness to proceed with encrypted communication.
2. **Packet Details:**
    - **TCP Flags:** The `PSH, ACK` flags again indicate the server is pushing data while acknowledging receipt of previous data from the client.
    - **Key Negotiation Data:** In the payload, the server lists supported encryption algorithms (`hmac-sha256`, `diffie-hellman-group14-sha1`, `ecdh-sha2-nistp256`), further facilitating secure communication.
3. **Relevance to Forensics:**
    - Understanding the cryptographic negotiation helps analysts determine if vulnerabilities (e.g., weak ciphers) could be exploited.
    - Identifying mismatched or anomalous configurations might indicate a man-in-the-middle (MITM) attack during the handshake.
4. **Technical Insights:**
    - Payload analysis reveals adherence to modern cryptographic standards. However, usage of deprecated algorithms (e.g., `diffie-hellman-group1-sha1`) should be flagged in a forensic audit.

24 Wireshark Packet Capture (SSH Traffic)



The third screenshot shows a **Wireshark** session capturing SSH traffic between two systems.

Tools and Commands Used:

- **Tool**: Wireshark.
- **Filter Applied**: `ssh` (displays SSH-related packets).

Analysis:

1. **Key Details**:
    - **Traffic Overview**:
        - Source: `192.168.1.222`.
        - Destination: `192.168.1.248`.
        - Protocol: SSHv2.
        - Length: Each packet is 89 bytes.
    - **Session Details**:
        - Packets are part of an SSH handshake process, including:
            - **Key Exchange Init**: Initiating Diffie-Hellman key exchange.
            - **New Keys**: Establishing encryption for further communication.
2. **Hexadecimal View**:
    - Displays raw packet data. For example:

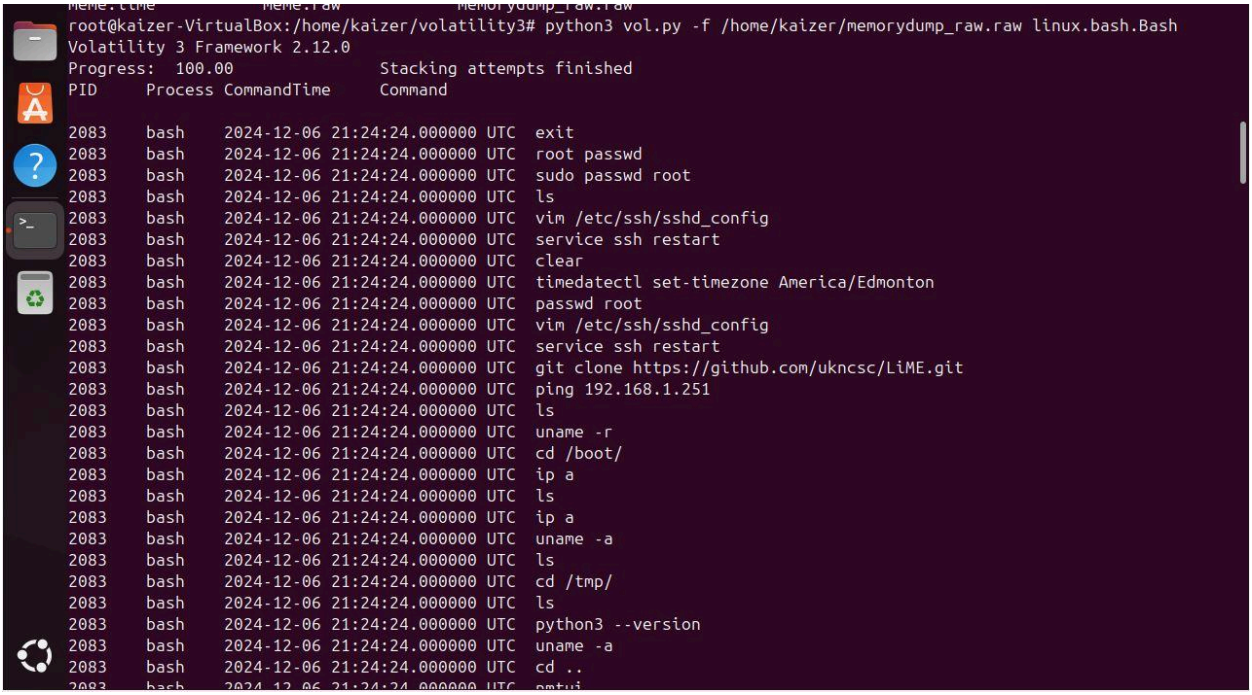
- **Ethernet II**: MAC addresses are captured for both source (`08:00:27:d9:c6:74`) and destination (`08:00:27:c9:9b:ef`).
- **SSH Header**: Protocol identifier (`SSH-2.0-libssh_0.10.6`), indicating the library and version used for encryption.

3. **Forensic Relevance**:
   - **Secure Communication**:
     - SSH traffic typically involves encrypted sessions; however, metadata such as IPs, ports, and protocol versions is visible.
     - Analysis of SSH activity helps in identifying potential brute force attempts or unauthorized access.
   - **MAC Addresses**:
     - Useful for tracing physical devices in the network.
   - **Library Version**:
     - The `libssh` version (`0.10.6`) can be checked for known vulnerabilities that might have been exploited.

—--

Memory Analysis

Bashcommand 1 **Analysis of Memory Dump Using Volatility 3**

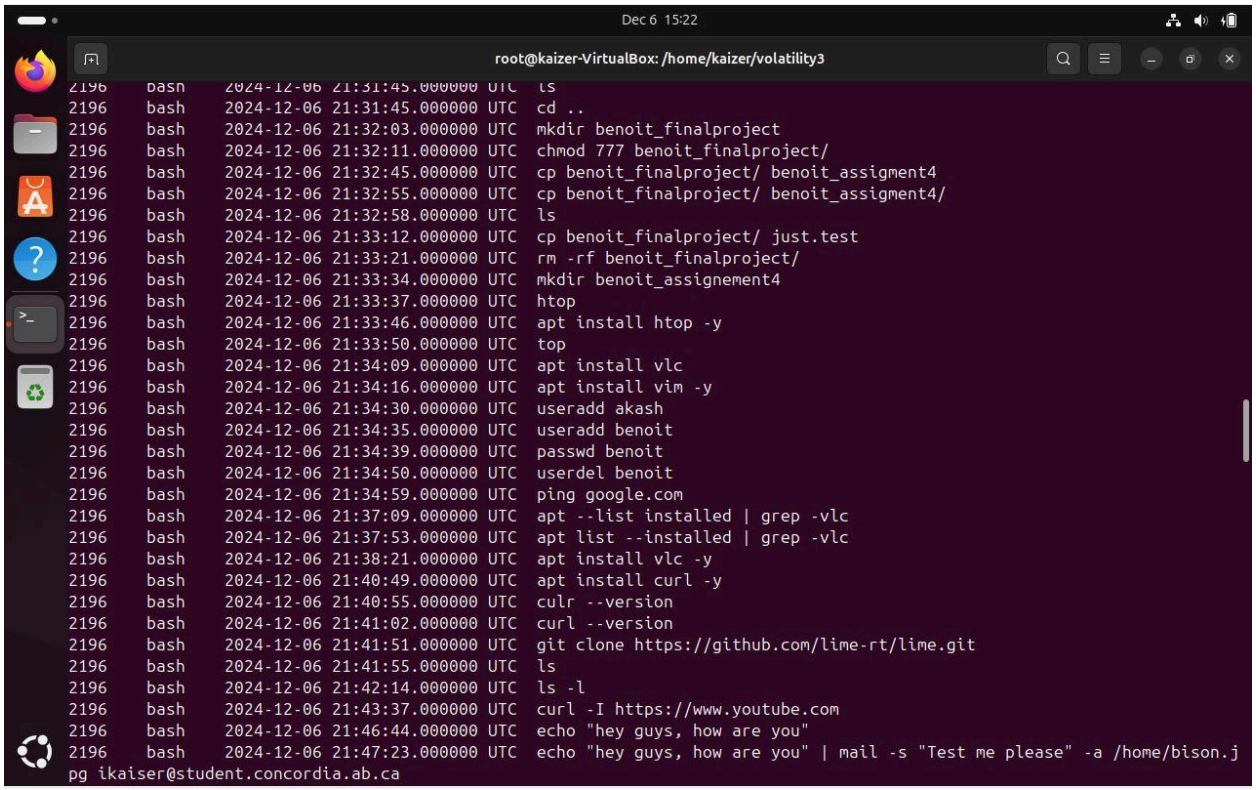


1. **Overview:**
    - The first screenshot showcases the analysis of a memory dump using Volatility 3, a powerful open-source memory forensics tool.
    - The command executed is `python3 vol.py -f /home/kaizer/memorydump_raw.raw linux.bash.Bash`, indicating an attempt to extract Bash command history from a raw memory image file.
2. **Technical Details:**
    - **Tool Used:** Volatility 3 Framework (v2.12.0).
    - **Memory Image File:** `/home/kaizer/memorydump_raw.raw`.
    - **Plugin Executed:** `linux.bash.Bash`, which retrieves Bash command history by inspecting process memory associated with `bash`.
    - **Output Fields:** The output includes the Process ID (PID), process name (`bash`), timestamps, and the commands executed.
3. **Findings and Relevance to Forensics:**
    - The extracted commands include administrative activities such as modifying user credentials (`passwd` and `sudo passwd root`), editing SSH configurations (`vim /etc/ssh/sshd_config`), restarting services, and cloning a repository (`git clone https://github.com/ukncsc/LiME.git`).
    - The timestamp field suggests the exact timing of each command execution, a critical factor for establishing a timeline of events in a forensic investigation.
    - Key activities like setting the system timezone (`timedatectl set-timezone America/Edmonton`) and network-related

commands (`ping 192.168.1.251`) are identified, which may indicate attempts to configure or test remote connections.

4. **Significance:**
    - Analyzing Bash history provides insights into user behavior and potential misuse or malicious activity on the system.
    - Commands such as editing `sshd_config` and restarting the SSH service are of particular interest, as they may indicate unauthorized access or persistence setup.

Bash command 2 **Extended Bash History and User Management**

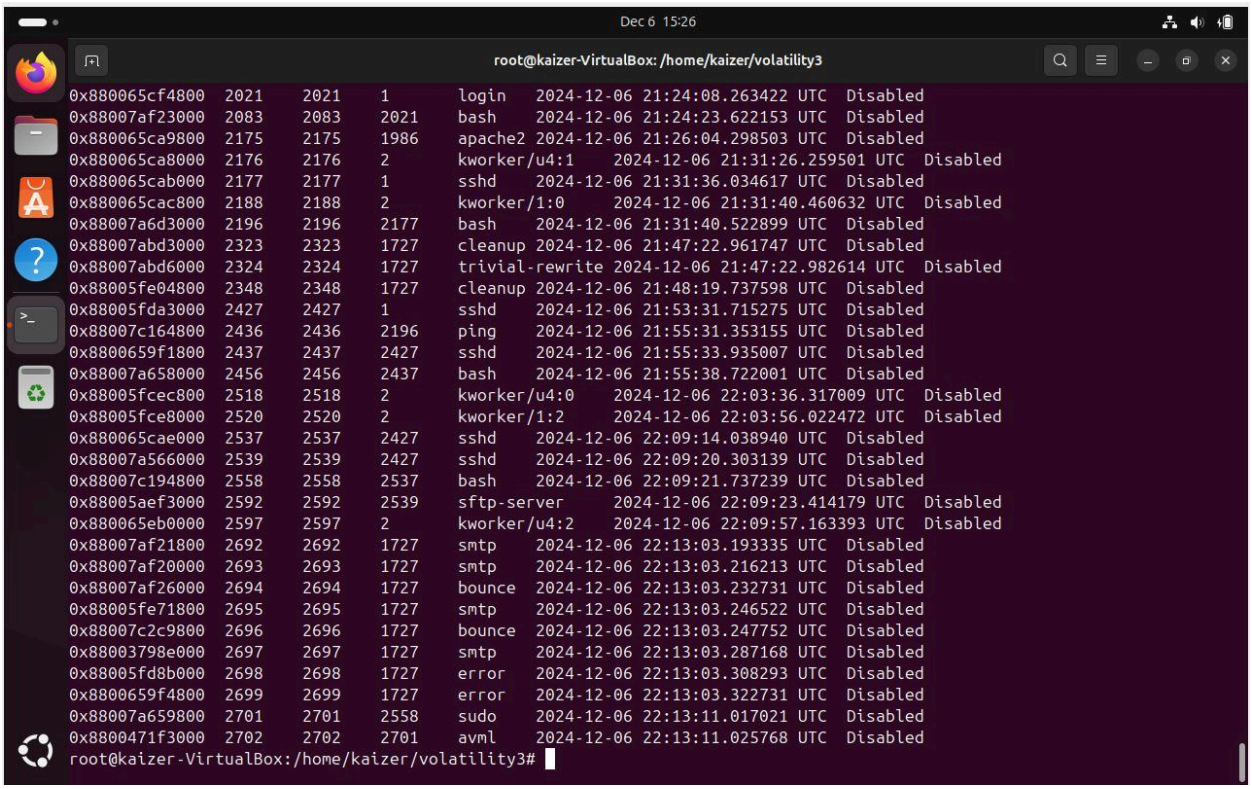


1. **Overview:**
    - The second screenshot extends the analysis, displaying additional commands executed on the system.

    - The PID shown is `2196`, indicating a different session from the first screenshot, with new activities being traced.
2. **Technical Details:**
    - **Tool Used:** Volatility 3 Framework, similar to the previous screenshot.
    - **Plugin:** `linux.bash.Bash`.
    - **Commands Observed:**
        - User directory and file manipulations (`mkdir`, `chmod`, `cp`, `rm`).
        - Software installation commands (`apt install htop -y`, `apt install vim -y`).
        - Network and system monitoring commands (`htop`, `top`, `ping google.com`).
        - User account management commands (`useradd`, `passwd`, `userdel`).
        - An attempt to fetch data from an external URL using `curl`.
3. **Findings and Relevance to Forensics:**
    - File and directory manipulations such as creating project directories and copying files (`cp benoit_finalproject/`) may indicate the movement or exfiltration of data.
    - Commands involving user management (`useradd akash`, `passwd benoit`, `userdel benoit`) point to the addition or removal of user accounts, which could be used for privilege escalation or erasing traces of activity.
    - The `curl` command suggests an attempt to connect to external resources or perform data exfiltration, while the `mail` command hints at sending data to external email addresses.
4. **Significance:**
    - Observing administrative commands such as creating user accounts and installing packages may help identify the intent behind these actions—whether they are legitimate or malicious.

- The interaction with external domains (`ping google.com`, `curl`) indicates potential evidence of network activity worth investigating.
- Email-related commands using `mail` point to potential data leakage or an attempt to communicate sensitive information externally.

1 plist **`pslist` Command Output (Processes List)**



The first screenshot showcases the output of the **Volatility 3 framework's `linux.pslist` plugin**, which is part of a memory forensic analysis. The `pslist` command is used to extract active processes from a memory dump and is a crucial step in analyzing the runtime state of an operating system at the time of the memory capture.

Technical Details:

- **Tool**: Volatility 3, a framework for advanced memory forensics.

Command Used:

```
python3 vol.py -f /home/kaizer/memorydump_raw.raw
linux.pslist.PsList
```

- This command specifies:
  - `-f`: Path to the memory dump file (`memorydump_raw.raw`).
  - `linux.pslist`: The plugin used to retrieve the process list from the memory dump.

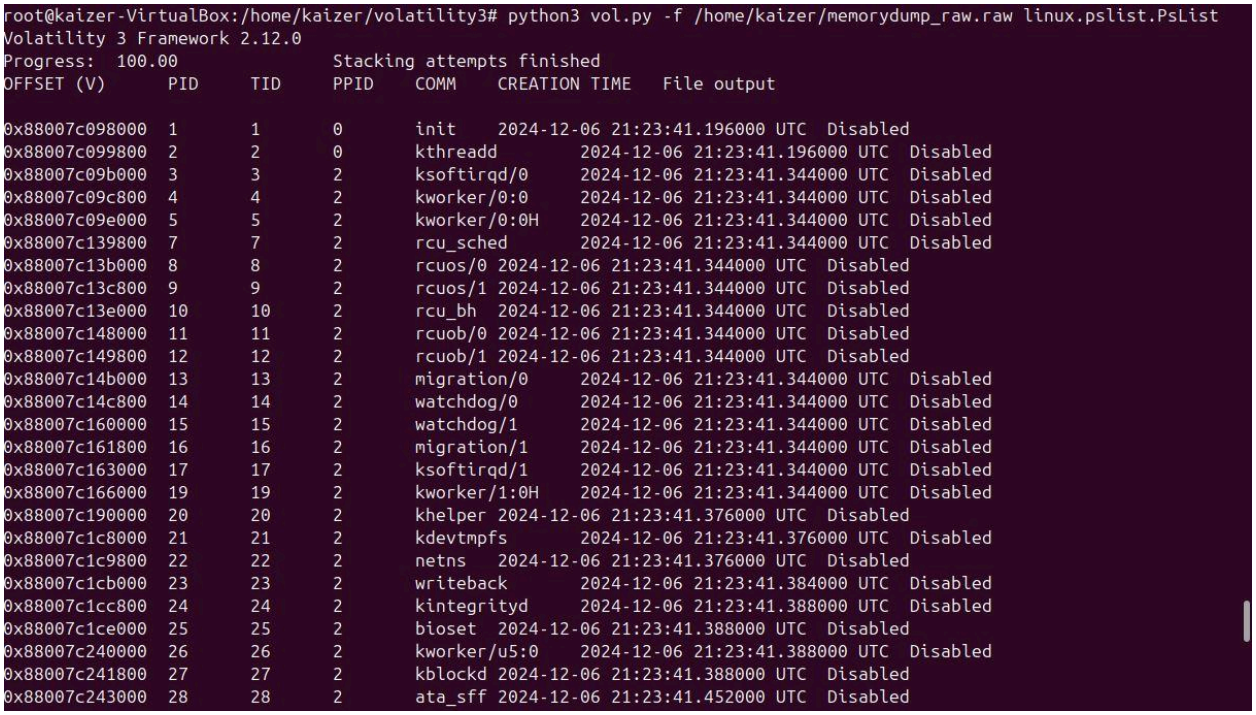
Key Observations:

1. **Process Hierarchy**:
   - The table includes columns such as `PID` (Process ID), `PPID` (Parent Process ID), and `COMM` (Command name).
   - The `PPID` column establishes the parent-child relationships between processes, which is essential for identifying malicious or unusual process trees.
2. **Processes and Timestamps**:
   - Processes such as `init`, `kthreadd`, `kworker`, and `rcu_sched` are visible. These are kernel/system processes typically seen in a Linux operating system.
   - The `CREATION TIME` column shows the UTC timestamps when each process was created.
3. **Relevance to Forensics**:
   - The output is valuable for identifying anomalies, such as rogue processes or mismatched parent-child relationships.
   - Processes like `bash` or `ssh` could hint at user activity or external connections that may require further investigation.

Forensic Relevance:

- Examining the process list allows analysts to establish a timeline of events, understand the system state, and pinpoint malicious activities or indicators of compromise (IoCs).
- This output serves as a foundational analysis step for deeper investigations, such as examining process memory, file descriptors, or network activity associated with suspicious processes.

2 plist **Additional Processes Analysis**



```
root@kaizer-VirtualBox:/home/kaizer/volatility3# python3 vol.py -f /home/kaizer/memorydump_raw.raw linux.pslist.PsList
Volatility 3 Framework 2.12.0
Progress:  100.00               Stacking attempts finished
OFFSET (V)      PID     TID     PPID    COMM    CREATION TIME   File output

0x88007c098000  1       1       0       init    2024-12-06 21:23:41.196000 UTC  Disabled
0x88007c099800  2       2       0       kthreadd        2024-12-06 21:23:41.196000 UTC  Disabled
0x88007c09b000  3       3       2       ksoftirqd/0     2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c09c800  4       4       2       kworker/0:0     2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c09e000  5       5       2       kworker/0:0H    2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c139800  7       7       2       rcu_sched       2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c13b000  8       8       2       rcuos/0 2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c13c800  9       9       2       rcuos/1 2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c13e000  10      10      2       rcu_bh  2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c148000  11      11      2       rcuob/0 2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c149800  12      12      2       rcuob/1 2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c14b000  13      13      2       migration/0     2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c14c800  14      14      2       watchdog/0      2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c160000  15      15      2       watchdog/1      2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c161800  16      16      2       migration/1     2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c163000  17      17      2       ksoftirqd/1     2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c166000  19      19      2       kworker/1:0H    2024-12-06 21:23:41.344000 UTC  Disabled
0x88007c190000  20      20      2       khelper 2024-12-06 21:23:41.376000 UTC  Disabled
0x88007c1c8000  21      21      2       kdevtmpfs       2024-12-06 21:23:41.376000 UTC  Disabled
0x88007c1c9800  22      22      2       netns   2024-12-06 21:23:41.376000 UTC  Disabled
0x88007c1cb000  23      23      2       writeback       2024-12-06 21:23:41.384000 UTC  Disabled
0x88007c1cc800  24      24      2       kintegrityd     2024-12-06 21:23:41.388000 UTC  Disabled
0x88007c1ce000  25      25      2       bioset  2024-12-06 21:23:41.388000 UTC  Disabled
0x88007c240000  26      26      2       kworker/u5:0    2024-12-06 21:23:41.388000 UTC  Disabled
0x88007c241800  27      27      2       kblockd 2024-12-06 21:23:41.388000 UTC  Disabled
0x88007c243000  28      28      2       ata_sff 2024-12-06 21:23:41.452000 UTC  Disabled
```

The second screenshot provides an expanded list of processes from the same `pslist` command, further detailing potentially user-initiated or server processes.

Key Details:

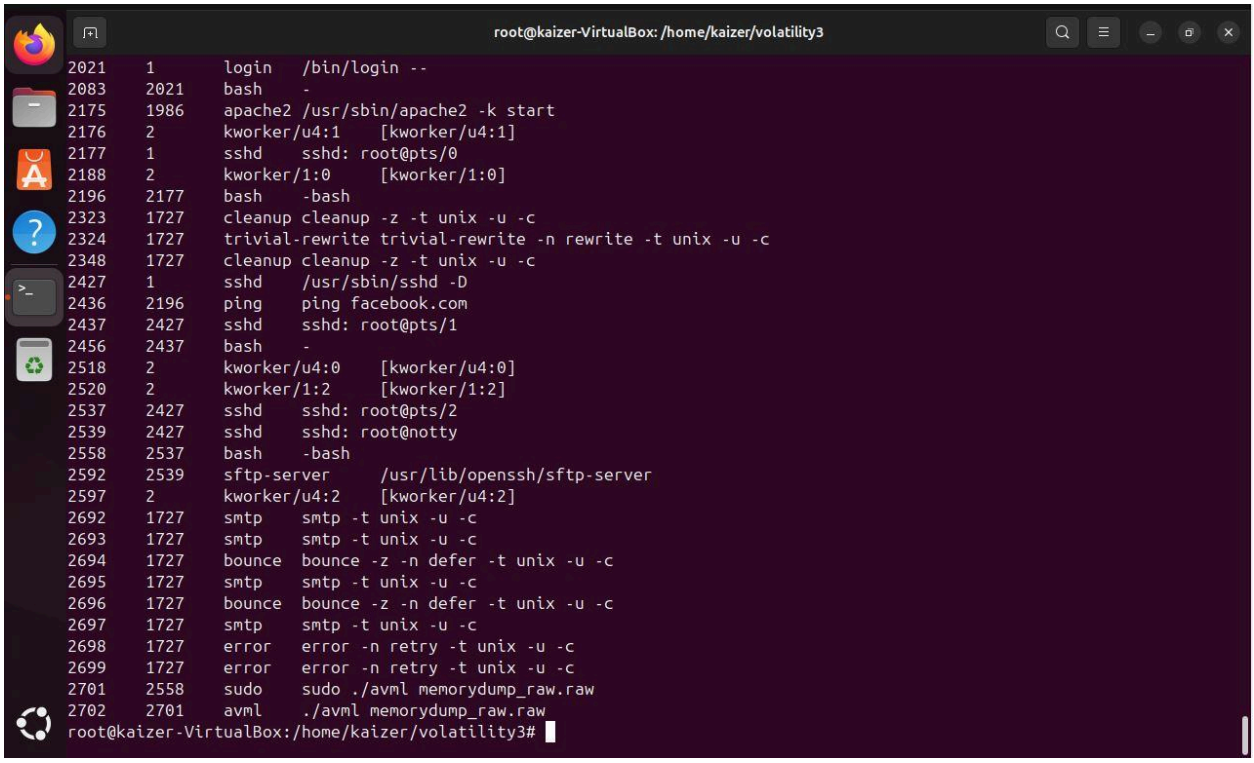
1. **Notable Processes**:

    - `sshd`: Indicates an active or recently active SSH connection.
    - `apache2`: A web server process, suggesting that this system is hosting a web application or service.
    - `bash`: Several instances of `bash` shells are visible, which may imply interactive sessions.
2. **User and System Processes**:
    - A mix of user-level and kernel-level processes is present, providing insight into both user activity and system background operations.
    - The presence of `ping` and `facebook.com` indicates outgoing network connections, which can be cross-referenced with network forensic artifacts for verification.

Forensic Significance:

- By correlating timestamps and parent-child relationships, it becomes possible to establish whether certain user actions (e.g., SSH sessions or web server activity) align with suspicious behaviors or external intrusion attempts.
- The `bash` and `sshd` processes are particularly noteworthy in forensic investigations, as they might indicate remote logins or command execution that could be unauthorized.

1 psaux **Command-Specific Arguments**



This screenshot appears to provide further details, potentially extracted with a command like `psaux`, showing detailed arguments used when processes were executed.

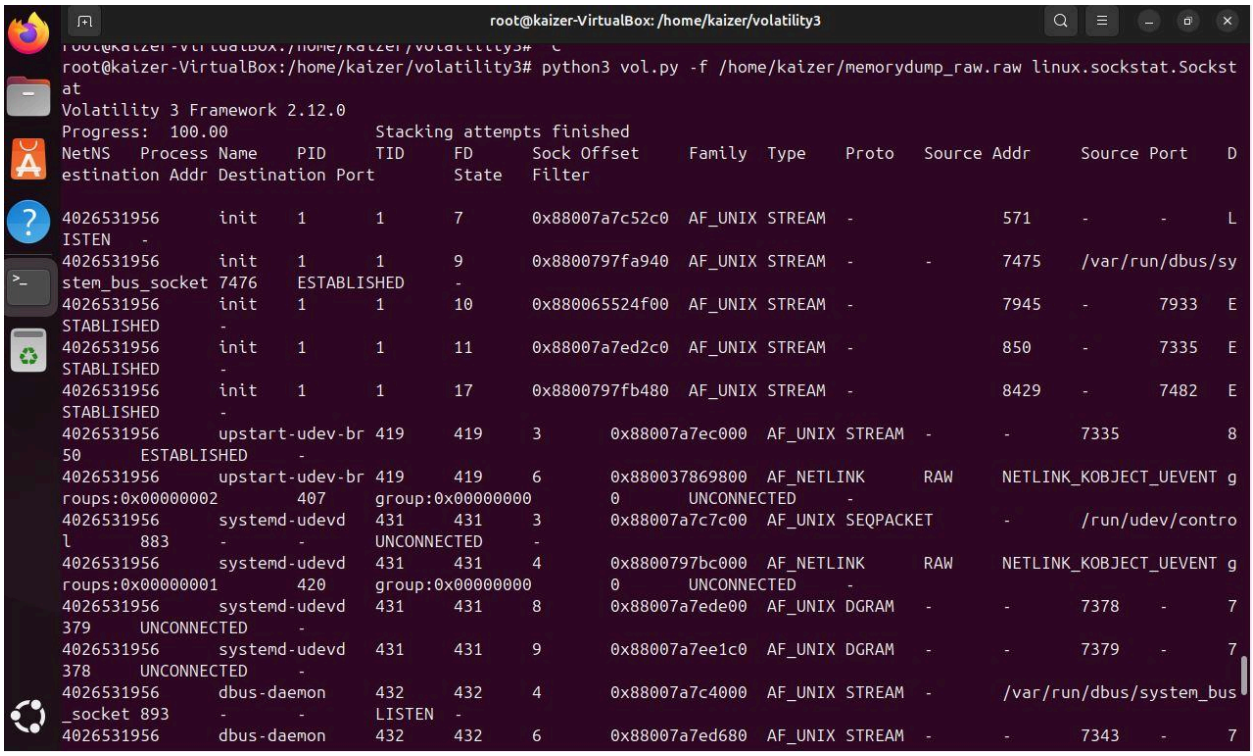
Key Observations:

1. **Command-Line Arguments**:
    - Commands like `/usr/sbin/apache2 -k start` show how the `apache2` process was started, including specific flags.
    - The `sshd` processes include arguments indicating specific SSH sessions, such as `root@pts/0` and `root@notty`.
2. **Forensic Implications**:
    - This level of detail is essential for understanding how processes were initiated. For instance, arguments for SSH sessions (`root@pts/2`) can reveal remote logins or shell access.

Relevance to Forensics:

- Investigators can use command-line arguments to determine the intent of processes. For example, unusual arguments for critical system services like sshd or apache2 may indicate exploitation or misuse.
- This screenshot complements the earlier ones by providing granular detail on process execution, which can be used to trace malicious activity.

1 socketused **Volatility Framework Output (Socket Analysis)**



The first screenshot shows the output of the **Volatility 3 Framework**, specifically analyzing socket information from a memory dump. Below is the detailed write-up:

Tools and Commands Used:

- **Tool**: Volatility 3 Framework (v2.12.0).

Command:

```
python3 vol.py -f /home/kaizer/memorydump_raw.raw
linux.sockstat.Socketstat
```

- This command extracts socket statistics from a Linux memory dump file (`memorydump_raw.raw`).

Analysis:

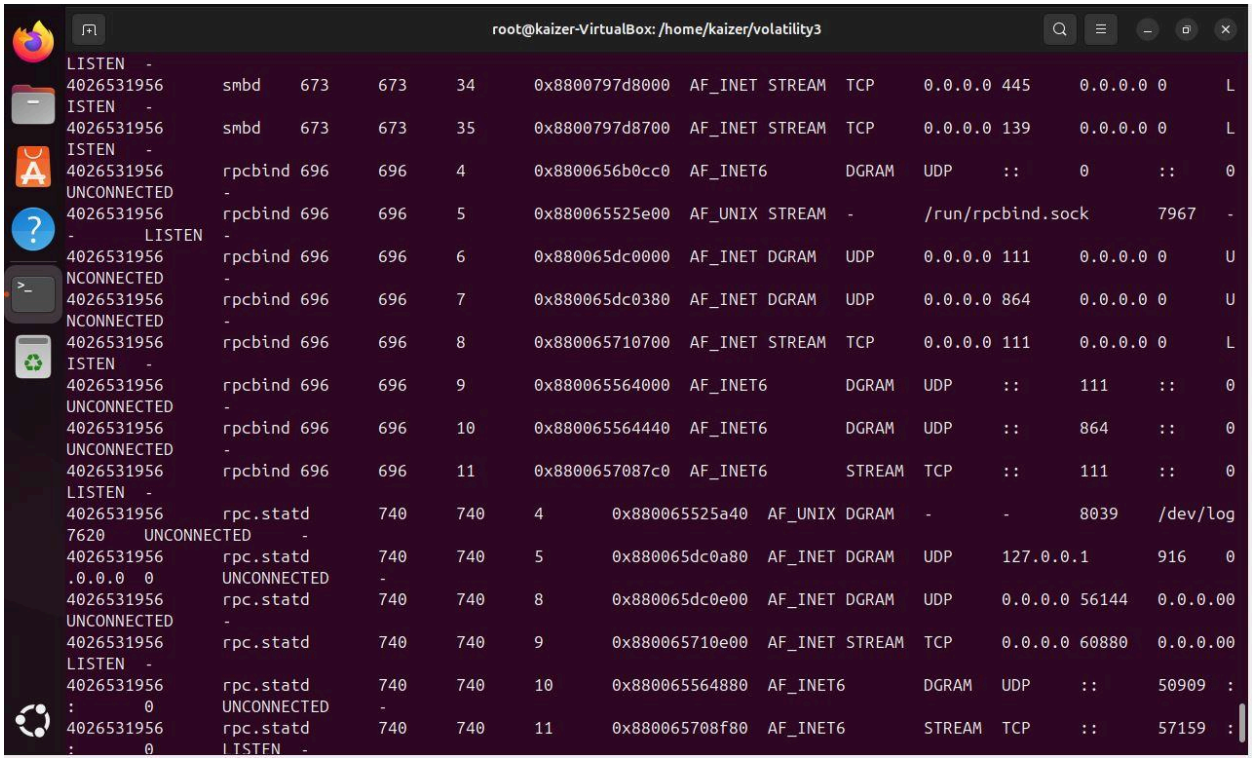
1. **Purpose**: The `linux.sockstat.Socketstat` plugin is used to examine active sockets on a system by retrieving their details from the memory dump. This is critical in digital forensics for detecting unauthorized network activities or malware communication.
2. **Key Observations**:
    - **Columns Explained**:
        - **NetNS**: Network Namespace Identifier.
        - **Process Name**: The name of the process associated with the socket.
        - **PID (Process ID)**: Unique identifier of the process.
        - **TID (Thread ID)**: Unique identifier of the thread associated with the process.
        - **FD (File Descriptor)**: The file descriptor linked to the socket.
        - **State**: The current state of the socket (e.g., `LISTEN`, `ESTABLISHED`, `UNCONNECTED`).
        - **Family**: Protocol family (e.g., `AF_UNIX`, `AF_INET`).
        - **Type**: Type of socket (e.g., `STREAM`, `DGRAM`).
        - **Proto**: Protocol used (e.g., `TCP`, `UDP`).
        - **Source Addr/Port**: Local address and port.
        - **Destination Addr/Port**: Remote address and port.
    - **Important Findings**:
        - Several `AF_UNIX STREAM` sockets in an `ESTABLISHED` state, indicating inter-process communication (IPC) through UNIX sockets.

    - Connections associated with processes like `init`, `upstart-udev-br`, and `systemd-udevd`, which are system-critical services.
    - Listening sockets (`LISTEN` state) associated with `dbus-daemon`, likely for system-wide messaging services.

Forensic Relevance:

- **Socket Analysis**: This provides insight into live network and IPC activities captured at the time of memory acquisition. Suspicious sockets can indicate unauthorized access or malware operations.
- **Artifacts**: Information about active and listening sockets can help correlate potential malicious activity, like rogue network connections or exfiltration attempts.

2 socketused **Volatility Framework (IPv4 and IPv6 Sockets)**



The second screenshot provides additional socket-related data from the memory dump, including both IPv4 and IPv6 connections.

Tools and Commands Used:

- **Tool**: Volatility 3 Framework.

Command:

```
python3 vol.py -f /home/kaizer/memorydump_raw.raw
linux.sockstat.Socketstat
```

-

Analysis:

1. **Key Observations**:
    - **Network Protocols**:

        - Several `AF_INET` (IPv4) and `AF_INET6` (IPv6) sockets are identified.
        - Protocols include both `TCP` and `UDP`.
    - **Notable Connections**:
        - `smbd` process (PID 673) has two listening TCP sockets (ports `445` and `139`), which are standard for **SMB (Server Message Block)** protocol. SMB is used for file sharing and could indicate a file server or networked resource.
        - `rpcbind` process (PID 696) is listening on multiple ports (e.g., `111` for RPC) with both TCP and UDP. RPC services can be exploited if not properly secured.
        - `rpc.statd` (PID 740) has both listening and unconnected sockets, which is typical for network status monitoring in NFS environments.
2. **Relevance**:
    - **Potential Vulnerabilities**: Open SMB ports (`445` and `139`) and RPC services could be exploited by attackers for lateral movement or unauthorized file access.
    - **IPv6 Activity**: Indicates systems configured for dual-stack networking, which expands the attack surface and requires special consideration during forensic investigations.
    - **Correlation**: This data can be cross-referenced with logs or network captures to identify malicious activity targeting specific ports or services.

Tab 2

1) Uploading a PCAP (Packet Capture) file is a critical step in network forensic investigations, marking the initiation phase of analyzing captured network traffic for malicious activities, unauthorized access, or data breaches. This process involves selecting and uploading a PCAP file, typically obtained from a network monitoring system or a network tap, into a forensic analysis platform like Security Onion or ELK (Elasticsearch, Logstash, Kibana). These platforms provide a robust framework for parsing, analyzing, and correlating traffic patterns to identify potential security threats.

The selected PCAP file undergoes a validation process before it is analyzed. This ensures the file's integrity and compatibility with analysis tools such as Wireshark, Zeek, Suricata, or Snort. Validation is a vital step, as corrupted or improperly formatted files can lead to inaccurate findings. The upload interface is designed to be user-friendly, allowing seamless integration with automated analysis workflows. Once uploaded, the PCAP file is available for inspection, with systems leveraging preconfigured rules to identify anomalies.

Analyzing PCAP files provides critical insights into network activity, offering forensic investigators the ability to detect indicators of compromise (IoCs) such as suspicious IP addresses, abnormal port usage, or unusual traffic volumes. For instance, examining PCAP data may reveal recurring communication patterns indicative of reconnaissance activities, lateral movements, or data exfiltration attempts. Such observations bridge the gap between raw data collection and in-depth forensic analysis, making this step foundational for the entire investigative process.

From a technical perspective, network security platforms inspect PCAPs by applying rule-based engines like Suricata or Snort, which automate the detection of known threats. These tools parse network packets to generate alerts for anomalies and provide event correlation to trace attack vectors. For example, Suricata may highlight SSH brute-force attempts based on repetitive failed login attempts captured within the PCAP.

The significance of this step extends beyond technical analysis. It ensures that all network activity is preserved and documented for further examination, maintaining the integrity of the evidence chain. Additionally, uploading PCAPs allows investigators to leverage visual dashboards and analytical tools to identify trends, anomalies, and high-risk activities effectively.

In professional practice, proper handling of PCAP files underpins the reliability of forensic findings. This step sets the stage for more refined analyses, such as traffic pattern reconstruction, alert generation, and behavioral analysis of network entities. By methodically processing PCAP data, forensic investigators lay a solid foundation for uncovering attacker methodologies and mitigating future risks, emphasizing the importance of meticulous data management in network forensics

2) The alert logs generated by Suricata provide critical insights into network security, marking a pivotal step in identifying malicious activities and potential threats. Suricata, an open-source

intrusion detection system (IDS) and intrusion prevention system (IPS), inspects network traffic by applying predefined rules to detect anomalies. The snapshot highlights how these alerts, triggered by analyzing traffic patterns, serve as potential indicators of compromise (IoCs) and guide forensic investigators in prioritizing incidents.

Each alert contains essential details that help analysts assess the nature of suspicious activity. Key elements include timestamps, which specify when the alert was generated, and event types that offer a descriptive categorization of the detected threat (e.g., "Trojan detected"). Source and destination IPs provide the communication endpoints involved in the flagged activity, while protocol details, such as TCP or UDP, define the nature of the transmission. These alerts are ranked by severity, enabling investigators to focus on high-priority threats and streamline their analysis process.

In this case, the PCAP file likely uploaded earlier was processed by Suricata, or real-time network monitoring was configured. Suricata matched the captured traffic against its signature database, generating logs that were visualized on a platform such as Security Onion. These logs served as the foundation for identifying traffic anomalies, unauthorized access attempts, or data exfiltration activities. For example, repeated failed SSH login attempts might trigger an alert labeled as a brute-force attack, prompting further examination of related logs and traffic patterns.

Suricata's functionality extends beyond signature-based detection by incorporating anomaly-based methods to identify previously unknown threats. This versatility allows it to detect advanced persistent threats (APTs) and zero-day exploits that evade traditional rule-based detection systems. When integrated with platforms like Kibana or Splunk, these alerts can be visualized and correlated with other data points for comprehensive analysis.

The relevance of Suricata alerts in digital forensics cannot be overstated. They act as a starting point for investigations, highlighting critical events amidst large datasets. For instance, an alert flagging unusual traffic to a high-risk destination IP could indicate malware communication with a command-and-control (C2) server. By correlating such alerts with timestamps, network logs, and other forensic data, investigators can reconstruct attack timelines and identify the methods and tools used by attackers.

From a professional standpoint, Suricata alerts are integral to the incident response lifecycle. They enable analysts to isolate threats, understand attacker behavior, and implement mitigative actions such as blocking malicious IPs or tightening firewall rules. By leveraging these alerts, forensic investigators ensure a systematic approach to threat detection and response, reinforcing the overall security posture of the network.

3) The process of creating a case within a forensic analysis platform is a foundational step in digital investigations, providing structure, clarity, and organization to the workflow. This snapshot highlights the importance of documenting findings, consolidating evidence, and facilitating collaboration among investigators. By leveraging integrated case management features,

forensic analysts ensure that all aspects of an investigation are systematically recorded, maintaining the integrity of the process and the evidence.

At the outset, the investigator assigns a descriptive name to the case, which serves as an anchor for organizing related activities. A unique identifier, or case ID, is often generated to facilitate easy tracking and retrieval of the case in the future. The interface showcased in the snapshot appears user-friendly and professional, suggesting the use of platforms such as TheHive, Autopsy, or EnCase. These platforms allow seamless navigation and integration with other forensic tools, further streamlining the investigation process.

The comments section within the case plays a pivotal role in maintaining transparency and clarity. This area is used to record observations, interpretations, hypotheses, and next steps, making it easier for investigators to revisit their thought processes. It may also include links to critical evidence files, summaries of alerts, or detailed analyses of previously captured data. Such annotations are particularly helpful in collaborative settings where multiple investigators are involved, ensuring that all team members have a unified understanding of the case.

The case creation process often begins with the investigator providing a name and optional metadata. Metadata can include timestamps, associated incidents, or related artifacts like network alerts, PCAP files, or host-specific logs. Once created, the case acts as a central repository, enabling investigators to consolidate findings, manage alerts, and link additional artifacts as the investigation progresses. For instance, if a Suricata alert flags a brute-force SSH attempt, the associated traffic logs and PCAP files can be added to the case, creating a comprehensive record.

From a digital forensics perspective, creating and managing cases is essential for maintaining the chain of custody and ensuring evidence admissibility. A well-documented case preserves a clear timeline of events and provides a defensible record of the investigation. This structure is invaluable in legal or compliance contexts, where transparency and accountability are paramount.

The integration of case management tools with tagging, search, and reporting features further enhances workflow efficiency. Investigators can filter and prioritize cases, search for related artifacts, and generate detailed reports for stakeholders. Additionally, collaboration is facilitated through shared access and task assignments, enabling teams to divide responsibilities effectively.

In professional practice, a thoughtfully documented case not only supports the investigative process but also serves as a reference for future incidents. By adding detailed comments and linking relevant evidence, investigators create a transparent and cohesive narrative that strengthens the overall integrity and effectiveness of the forensic investigation.

4) The escalation of alerts to a case analysis within Security Onion is a critical step in the forensic workflow, bridging the gap between detection and structured investigation. This snapshot illustrates the "Cases" section of Security Onion, where alerts generated by intrusion

detection systems like Suricata are tracked and managed systematically as cases. In this instance, the escalated alert pertains to a potential SSH brute force attack categorized under the title "ET SCAN LibSSH Based Frequent SSH Connections Likely BruteForce Attack." This categorization reflects the system's detection of suspicious and repetitive SSH connection attempts, warranting further investigation.

The case interface is tailored to streamline an investigator's focus on actionable incidents. Filtering options, such as excluding closed cases (`NOT so_case.status:closed`) and templates (`NOT so_case.category:template`), ensure that only open and relevant cases are displayed. This enables forensic analysts to prioritize ongoing threats without being encumbered by resolved or irrelevant data. Each case includes timestamps for creation and last modification, providing a clear timeline essential for reconstructing events and ensuring chronological accuracy in the investigation process.

Key metadata associated with the case highlights its urgency. For example, the "new" status signifies that the case requires immediate attention, while its "high" priority level underscores its criticality to the organization's cybersecurity posture. By integrating these elements, Security Onion not only aids in managing alerts but also enforces a structured workflow that facilitates prompt action.

From a technical standpoint, transitioning alerts into cases ensures that potential threats are documented comprehensively for legal, compliance, and reporting purposes. The case contains details such as the nature of the alert, affected systems, and timestamps that support forensic time-lining. These elements allow analysts to correlate alerts with broader attack patterns or other incidents, building a cohesive picture of the threat landscape.

The escalation process is particularly relevant in scenarios involving persistent threats like brute force attacks. SSH brute force attempts, characterized by repetitive login attempts targeting port 22, pose significant risks of unauthorized access and privilege escalation. By elevating such alerts to cases, analysts can consolidate evidence, monitor subsequent activity, and implement countermeasures such as IP blocking, rate limiting, or multi-factor authentication.

Additionally, the ability to categorize and prioritize cases ensures efficient resource allocation. Investigators can focus their efforts on high-priority cases while using metadata and timestamps to trace attack vectors or identify vulnerabilities in affected systems. This structured approach not only enhances investigative efficiency but also ensures that findings are properly documented for future audits, incident reviews, or legal proceedings.

In professional practice, the escalation of alerts into cases exemplifies the importance of integrating detection and response workflows. By maintaining a clear and organized repository of incidents, Security Onion empowers investigators to address threats with precision, ensuring robust network security and effective incident response. This process serves as a cornerstone of digital forensic investigations, reinforcing the importance of meticulous case management in combating cyber threats.

5) Case metadata plays a crucial role in the forensic investigation process, ensuring that each case is organized, attributed, and classified for effective tracking and analysis. The screenshot in question delves into the metadata and descriptive attributes of an escalated case, offering insights into how forensic professionals manage and document the details of incidents. This metadata provides essential context for understanding the case, including its classification, tags, and timestamps, which are fundamental for maintaining the integrity and transparency of the investigation.

One of the key elements in the metadata is the **Traffic Light Protocol (TLP)**, which is marked as "unknown" in this instance. The TLP is a classification system used to indicate the sensitivity level of the information in the case, guiding how it should be handled and shared. In this case, the "unknown" designation suggests that the sensitivity level has yet to be determined, highlighting the need for further analysis before deciding on the appropriate level of confidentiality. Similarly, the **Permissible Actions Protocol (PAP)** is also marked as "unknown," implying that the guidelines for responding to or disclosing the case's findings are still to be established. These protocols are essential for ensuring that the investigation remains compliant with legal and organizational standards.

The **category** field classifies the case as "SSH_BRUTE," indicating that the incident involves a brute-force attack targeting SSH services. This classification is important as it provides a clear context for the nature of the attack, helping forensic analysts quickly identify trends and align the investigation with known attack patterns. Brute force attacks are common attack vectors, and classifying them as "SSH_BRUTE" allows analysts to efficiently track similar incidents and develop threat intelligence for more proactive defense strategies.

Adding further specificity, the **tag** "SSH_FORCE" refines the categorization and aids in correlating this case with other incidents involving SSH-based brute force attacks. Tags are essential for linking related events, incidents, or patterns, enhancing the investigative process by creating a searchable and filterable index of threats across multiple cases. For instance, the "SSH_FORCE" tag could help analysts identify multiple brute-force attempts across different systems or networks, contributing to a broader understanding of an attacker's tactics.

User attribution is another critical component of case metadata. In this instance, the case was created and updated by **ikaiser@student.concordia**, establishing a clear chain of custody for accountability. Proper attribution ensures that all actions taken during the investigation are traceable to the responsible individuals, adding a layer of transparency and accountability to the process. This also assists in collaboration, as other team members can see who was involved and who made updates to the case, ensuring that the investigation remains cohesive.

The **timestamps** associated with the case—both for creation and last modification—reflect a time span of just two minutes, indicating prompt escalation by the analyst. This rapid response is essential in digital forensics, as timely actions can significantly impact the ability to preserve evidence and understand the scope of an incident. The timestamp not only marks the start of the investigation but also helps establish a timeline of events, which is critical when correlating activities across systems or when building a narrative for legal or compliance purposes.

From a **technical perspective**, case metadata forms the backbone of forensic case management. It enables forensic professionals to manage and track incidents systematically, ensuring that critical information is accessible and organized for further analysis. The proper categorization, use of tags, and user attribution facilitate efficient collaboration and investigative continuity. Moreover, case metadata serves as a valuable record for audits, future investigations, or litigation, providing a detailed and transparent account of actions taken during the investigation process. By leveraging this metadata, analysts can not only streamline their current investigations but also enhance their ability to identify and mitigate future risks effectively.

6) . Dashboard Metrics for Analysis

The **Dashboards section** of Security Onion plays a vital role in providing a macro-level view of the security landscape by aggregating metrics and trends from monitored events over a specified period. This feature is indispensable for forensic analysts, as it offers visual insights that help identify patterns, understand activity spikes, and assess the overall network security posture. By visually displaying network activity, the dashboard allows investigators to spot anomalies, prioritize critical threats, and focus their efforts where they are most needed.

The **Onion Query Language** (OQL) is used to aggregate data by several key dimensions, such as event categories, modules, datasets, source and destination IP addresses, and ports. This query syntax allows the analyst to organize and filter vast amounts of network data efficiently. By grouping data based on these fields, investigators can easily pinpoint emerging patterns, such as a sudden surge in brute-force login attempts targeting specific services or IPs. This can be particularly valuable in identifying potential ongoing attacks, such as brute-force attempts on SSH, or detecting misconfigurations that may have exposed vulnerabilities.

One of the essential features of the dashboard is its **basic metrics visualization**, which includes bar graphs that illustrate the volume of events across various categories. These visual representations allow analysts to quickly identify the most prominent threats within the system, helping them prioritize their investigation. For example, if a particular event category (such as failed login attempts or port scanning) shows an unusually high volume, it may warrant further examination. The **timeline trends** graph, on the other hand, tracks the event count over time, offering a temporal perspective on network activity. By observing the trends, analysts can correlate spikes in activity with potential attack windows, aiding in the identification of suspicious periods that might align with cyberattacks or unauthorized activity.

The **search scope** in the dashboard is set to the "Last 10 days," ensuring that recent activity is captured in the analysis. With a total of 173,217 events found during this period, the sheer volume of data reinforces the need to narrow down the search to specific event types or incidents. This high level of activity necessitates a focused approach, where analysts can filter the events based on severity, type, or relevance to ongoing investigations. For example, the dashboard can filter out benign activity, such as routine traffic, and isolate events like potential brute-force attacks or unauthorized access attempts that are more critical for investigation.

From a **technical perspective**, the dashboard provides essential insights for **security incident triage**. The ability to aggregate, visualize, and analyze data across various parameters allows forensic analysts to see beyond isolated events and understand broader network behavior. Trends identified on the dashboard can be cross-referenced with specific cases, like an ongoing SSH brute-force attack, helping to determine whether the events are part of a larger pattern or isolated incidents. This helps in quickly identifying systemic vulnerabilities and prioritizing response efforts. Additionally, the timeline and metrics allow investigators to assess the scale of the attack, potentially revealing connections to other incidents or threats across the network.

In the context of **digital forensics**, the dashboard is an indispensable tool for quickly spotting emerging threats and efficiently managing large-scale security incidents. By using visual aids like bar graphs and timeline trends, investigators can easily track the progression of potential attacks, correlate incidents, and focus their analysis on high-priority threats. This enhanced situational awareness is crucial for effective incident response, ensuring that resources are allocated appropriately to mitigate risks and protect the network from further compromise.

7) The **Actions menu** in the Security Onion Dashboard is an invaluable tool for forensic investigators, allowing them to interact with data results and refine their analysis. This interface is designed to support the detailed workflow of digital forensics, facilitating data management, event correlation, and case-building. By providing several key options for filtering and exporting data, the Actions menu helps investigators streamline their process, making it easier to focus on the most relevant findings and add them to an ongoing case.

One of the standout features of the **Actions menu** is its filtering capabilities, including options such as **Include**, **Exclude**, and **Only**. These filters allow analysts to isolate specific

8) The **"Add to Case"** feature within Security Onion is an essential component of its case management workflow, designed to enhance the organization and documentation of suspicious events during a forensic investigation. This functionality allows investigators to systematically add individual events or sets of data to a specific case, ensuring that all findings are accurately recorded, easily accessible, and linked to the broader investigative effort. It streamlines the process of correlating disparate events and organizing the investigation's progress into structured, actionable steps.

The **Add to Case workflow** offers investigators the flexibility to either create a new case or add events to cases that have already been initiated. This ensures that new incidents are captured in real-time, while ongoing investigations can be continuously updated with new findings. The cases are clearly listed in the interface, with descriptions that provide context for each, such as "ET SCAN LibSSH Based Frequent SSH Connections Likely BruteForce Attack," which highlights malicious SSH activity. Other case types, like "ET USER_AGENTS Suspicious User-Agent" and "OpenSSH_Vulnerable," help track reconnaissance activities or the exploitation of known vulnerabilities. These case labels and classifications play a key role in categorizing events based on their nature and severity, guiding forensic experts to assess the potential impact of each event in the broader context of the security incident.

In the **event context**, the filtered query (event.dataset) is essential for narrowing down which logs should be analyzed and added to a case. Specific datasets like `system.syslog` and `elastic_agent.fleet_server` are commonly used to detect anomalies, ensure system integrity, and flag suspicious activity. These datasets contain critical information, such as system-level logs and agent data, which can provide detailed insights into the nature of the threat, the methods used by attackers, and the potential vulnerabilities that may have been exploited. By filtering data based on these sources, investigators can focus on the most relevant logs, reducing the volume of data and making the investigation process more manageable and efficient.

The relevance of this functionality to **digital forensics** is immense. It provides a structured repository for all investigative actions, ensuring that evidence is preserved and documented in a way that adheres to forensic principles, such as the chain of custody. Maintaining this chain of custody is crucial for the integrity of the investigation, particularly if the findings need to be presented in legal or compliance contexts. By organizing events under specific cases, the process of gathering, analyzing, and reporting findings becomes more efficient, as investigators can trace the origins of events, correlate different pieces of evidence, and ensure consistency in their analysis.

The process itself is straightforward. After filtering the events by dataset types on the **Security Onion Dashboard**, the user can select the "Add to Case" action. This action opens options to either create a new case or update an existing one. Upon selecting the appropriate case, the investigator can log their findings, ensuring that they are linked to the relevant case. This not only helps in keeping the investigation organized but also ensures that all findings are centralized in one place, accessible for further review, analysis, and reporting.

Ultimately, the **Add to Case** feature is a vital tool in the management of forensic investigations, offering clear organization, transparency, and consistency throughout the investigative process. It ensures that forensic principles are adhered to, making it easier for investigators to track progress, share findings with team members, and document their actions accurately. This functionality supports the overall goal of maintaining forensic integrity while enabling efficient and systematic analysis of suspicious events and potential threats.

9) Filtering and aggregating network traffic based on source and destination details is an essential step in identifying, analyzing, and tracking suspicious activities during a digital forensic investigation. This screenshot highlights how **Security Onion** allows forensic analysts to refine their investigation by isolating data based on critical communication endpoints, such as source IPs, destination IPs, and ports. By filtering network traffic through these dimensions, investigators can pinpoint anomalous or malicious activities that might otherwise be overlooked in large datasets.

In the **filter context**, the data is grouped by key fields, with specific attention given to the **Source IPs**, **Destination IP**, and **Port**. For instance, the source IPs, such as `192.168.1.222` and `192.168.1.246`, likely represent the devices initiating the connections. These IP

addresses can be critical in identifying compromised systems or devices within the network. The **Destination IP**, `192.168.1.248`, serves as the target for these connections, which further investigation suggests might be involved in lateral movement or data exfiltration attempts. By isolating the destination IP, investigators can narrow their focus on the specific system or service being targeted by malicious actors.

The **Port** is another vital piece of information; in this case, only **port 22** (SSH) is visible, indicating that the traffic pertains to secure shell connections. This detail is crucial, as SSH is often used for remote administrative access, and any unusual traffic directed at this port could point to brute-force login attempts or unauthorized administrative access. By focusing on SSH traffic, analysts can efficiently filter out irrelevant data and target the specific threat vectors associated with this protocol.

The **counts and patterns** generated by the analysis are another crucial feature in forensic investigations. For example, the source IP `192.168.1.222` is noted for having the highest interaction count at 78, suggesting a concentrated effort to initiate connections to the destination IP. A high frequency of interactions from a particular source IP could be indicative of a brute-force attack, where an attacker is attempting to gain unauthorized access by repeatedly trying different credentials. These counts and patterns allow analysts to prioritize their investigation by identifying high-risk sources and focusing their efforts on critical data points.

From a **forensic perspective**, identifying the source and destination IPs, along with the corresponding ports, helps trace the origin of suspicious traffic and provides essential evidence of malicious or unauthorized access attempts. Logs that record this communication activity serve as concrete evidence in digital forensics, supporting the reconstruction of events and aiding in the identification of attack vectors. Furthermore, these details play a crucial role in the incident response process, as they can inform decisions on mitigating ongoing threats, such as blocking specific IP addresses or isolating compromised systems.

The steps performed in this analysis begin with accessing the **Dashboards menu** in Security Onion and selecting relevant data filters for **source.ip**, **destination.ip**, and **destination.port**. Once the filters are applied, visualizations are used to aggregate traffic counts and identify sources with the most interactions. This allows investigators to isolate high-priority sources and investigate further. Additionally, the filtered findings can be prepared for **further analysis or addition to a case**, ensuring that the most relevant and critical data is organized for subsequent investigation steps.

In conclusion, the ability to filter results based on source and destination details is invaluable in digital forensics. By honing in on specific communication endpoints and ports, forensic analysts can quickly identify potential intrusion attempts, correlate activity across multiple systems, and efficiently allocate resources for further investigation. This functionality within Security Onion significantly enhances the overall forensic process, ensuring that analysts can focus on high-priority threats and respond to incidents in a timely and effective manner.

10) The **Hunt view** in Security Onion is an essential tool in digital forensics, specifically designed to investigate suspicious network activity triggered by intrusion detection systems like Suricata. The snapshot provided highlights a Suricata alert, offering forensic investigators a powerful platform to conduct detailed investigations of potential security threats. By using the Hunt view, investigators can correlate events, analyze attack patterns, and trace potential intrusion attempts across the network.

One of the most critical elements of the **Hunt view** is the **timestamp**, which in this case is **2024-11-27T11:58:19.588Z**. Precise timestamps are indispensable in digital forensics, as they allow investigators to synchronize events across multiple systems, helping to build an accurate timeline of activities. This becomes especially crucial when analyzing network traffic over extended periods, as small gaps in time can often be linked to key phases in an attack, such as initial exploitation, lateral movement, or data exfiltration.

The **source and destination details** in this case are pivotal in understanding the nature of the activity. The **source IP**, `192.168.1.222`, originates from port `52724`, while the **destination IP**, `192.168.1.248`, is targeted on port `22` (SSH). The presence of traffic directed to port 22 is a strong indicator of potential malicious activity, as SSH is commonly used for remote administrative access. This interaction suggests the possibility of a brute-force attack, where an attacker repeatedly attempts to gain unauthorized access to the system using different passwords or login credentials. Such activities are typical of attempts to exploit weak SSH configurations or guess credentials.

The **Suricata alert details** further validate the potential threat, as the dataset name `suricata.alert` confirms that this log was generated by Suricata, a robust network intrusion detection and prevention system (IDS/IPS). Suricata is specifically designed to inspect network traffic and identify patterns that match known attack signatures. By categorizing this data under the **so** namespace (Security Onion), the platform helps in organizing and identifying the source of the alert, ensuring that all findings are attributed to the correct tool and system for easy reference.

Additionally, the event is processed and categorized by the **Elastic Agent**, version 8.14.3, under the **network category**. This categorization is crucial for streamlining the investigative process, as it provides clarity on the type of activity being observed. By processing the event into this category, investigators are provided with structured and categorized data that simplifies the analysis of network activity and helps identify attack vectors more quickly.

In terms of **digital forensics relevance**, this log provides **initial evidence** of a potential intrusion attempt. The key forensic tasks here involve cross-referencing this Suricata event with other system logs to identify any related activities or trends that might indicate further malicious activity. For example, **SSH configurations** on `192.168.1.248` should be inspected to verify if there are any weak or default credentials in use, which would make the system more susceptible to brute-force attacks. Additionally, forensic investigators would examine

authentication attempts to identify patterns that suggest unauthorized access attempts, such as multiple failed logins or unusual login times.

This investigation would also require identifying whether the traffic fits the typical patterns of a **brute-force attack**, such as numerous failed login attempts originating from a single source or an unusually high volume of SSH connection attempts. Such patterns are key indicators of an ongoing brute-force attack that may require immediate mitigation, such as blocking the source IP, enforcing rate-limiting measures, or implementing stronger authentication protocols like multi-factor authentication (MFA).

In summary, the **Hunt view** in Security Onion provides forensic investigators with a powerful interface for analyzing network traffic and identifying potential security incidents. By focusing on specific details such as timestamps, source and destination IPs, and targeted ports, investigators can efficiently track suspicious activity and assess the nature of potential attacks. The ability to categorize and correlate events further enhances the forensic process, allowing for a more organized and effective investigation.

11) The **Network and Rule Details** section of the Security Onion platform provides an in-depth look at the network traffic and the rules that were triggered by Suricata, offering valuable context for forensic investigations. This screenshot dives into the details of a specific network connection, providing analysts with key data points to better understand the nature of the activity and the potential threats posed by the connection. By analyzing these details, forensic experts can assess attack vectors, identify vulnerabilities, and determine whether the connection is part of a broader exploitation effort.

One of the key observations in this analysis is the **network context**, which includes the use of a unique identifier called the **network.community_id**. This identifier serves to link related traffic across different systems, helping investigators correlate seemingly disparate events that might otherwise go unnoticed. In forensic investigations, the ability to correlate traffic from various sources can be critical for understanding the scope of an attack and determining whether it is part of a larger campaign or a targeted incident. By tracking the traffic across multiple systems, forensic analysts can build a more comprehensive picture of the attacker's methods and objectives.

The **decoded data** is particularly significant in understanding the nature of the communication. In this case, the traffic includes a captured **SSH handshake**, identified as **SSH-2.0-libssh_0.10.6**. This handshake is the initial step in establishing a secure SSH connection, and the fact that it has been captured provides insight into the specific cryptographic algorithms being used. For example, the use of **curve25519-sha256** and **aes128-ctr** algorithms provides information on the encryption standards employed, which can help analysts identify whether the communication is secure or if there are any weaknesses in the cryptographic protocols being used. In this context, outdated or weak encryption standards could indicate a vulnerability that attackers may exploit.

The **transport protocol** used in the captured traffic is **TCP**, which is the standard protocol for most network communications, including SSH. Understanding the protocol helps investigators narrow down the types of activities they are dealing with, ensuring that only relevant traffic is analyzed. By confirming that the attack is happening over TCP, analysts can focus on network behaviors associated with this protocol, such as port scans, connection attempts, or repeated login failures, all of which are common in brute-force SSH attacks.

The **observer** responsible for flagging the suspicious traffic is identified as **securityonion**, which indicates that the event was monitored and detected by the Security Onion platform. This provides clarity on the source of the alert and ensures that it can be traced back to the correct system for further analysis. Additionally, the **rule action** associated with this traffic is marked as "allowed," which indicates that the traffic was permitted to pass through the system despite being flagged as potentially malicious. This categorization helps forensic experts understand the event in the context of the network's security policies, which could reveal misconfigurations or gaps in security controls.

The **rule metadata** provides further insight into the nature of the threat. In this case, the category is labeled as **"Attempted Administrator Privilege Gain,"** which points to an attempt to exploit administrative access—an alarming sign of an attack targeting high-level system privileges. The rule itself was first created in **2010** and updated in **2019**, which underscores its relevance in detecting both older and evolving threats. This demonstrates the continued importance of maintaining and updating security rules to stay ahead of attackers, particularly those attempting to exploit known vulnerabilities in widely used protocols like SSH.

From a **digital forensics** perspective, this detailed analysis of the captured network handshake and associated rule metadata provides critical information that can guide further investigation. By examining the specific cryptographic algorithms and the type of SSH communication being used, investigators can identify potential weaknesses in the system's configuration or encryption methods. If the **source IP**, in this case, `192.168.1.222`, is linked to known malicious activities, this could provide a strong indication that the traffic is part of an ongoing attack. Furthermore, cross-referencing this data with other logs, such as authentication attempts or system configuration files, can help forensic analysts identify unauthorized access attempts and trace the source of the attack.

Overall, this level of detailed analysis is essential for uncovering specific **threat vectors** targeting SSH services, particularly when dealing with common attack methods like brute-force attacks. By focusing on network packet sources, cryptographic weaknesses, and rule metadata, digital forensics teams can enhance their ability to detect, investigate, and mitigate threats before they escalate into more significant security breaches.

12)The **Alert Metadata and Threat Intelligence** section of this screenshot provides essential context and categorization for understanding the severity and potential impact of the detected alert. This information is crucial for digital forensics investigations, as it allows forensic analysts to assess the nature of the detected threat, prioritize investigation efforts, and correlate the event with external threat intelligence sources. The metadata captured by Security Onion and

related tools provides an organized summary of key data, making it easier for investigators to quickly identify and respond to potential threats.

A significant **rule name** present in the metadata is **ET SCAN LibSSH Based Frequent SSH Connections Likely BruteForce Attack**. This rule strongly indicates a brute-force attempt targeting SSH services, which is a common attack method used to gain unauthorized access to systems by guessing login credentials. Brute force attacks typically involve repeatedly attempting different username and password combinations until the correct one is found, and they are often used as the precursor to more serious activities, such as privilege escalation or lateral movement across the network. In this case, the **Emerging Threats** reference ties the detection to a trusted threat intelligence source, providing further credibility to the alert. Emerging Threats is widely recognized for providing comprehensive and up-to-date signatures for various types of attacks, and referencing this source strengthens the reliability of the detection.

The **severity level** of the rule is marked as **Level 1 (low)**, suggesting that this alert represents the early stages of a potential attack or a scan activity. While the severity is relatively low, it still requires attention as brute-force scans are often the initial phase of more sophisticated attacks. They can serve as the starting point for further exploitation, such as gaining unauthorized administrative access, escalating privileges, or moving laterally within the network. By recognizing this early stage, investigators can take proactive measures to block the attack and mitigate the risk of escalation before it becomes a more serious threat.

The **source IP** of the malicious activity is **192.168.1.222**, confirming its role as the origin of the brute-force attack. Repeated tagging of the event with the label "alert" highlights its significance in the broader dataset, ensuring it draws attention from analysts who are reviewing large volumes of network data. This repeated tagging is an important feature of the system, as it helps prioritize alerts based on their frequency and potential impact. The identification of the source IP is a critical step in the forensic process, as it allows investigators to trace the origins of the attack and, if necessary, block or investigate the malicious actor responsible for initiating the brute-force attempt.

From a **digital forensics** perspective, the alert metadata provides strong contextual support for identifying the event as a likely brute-force attack. To further investigate, forensic analysts would take several steps. One of the first steps would be to **identify patterns** of repeated access attempts originating from `192.168.1.222`. This can help determine whether the attack is ongoing, if it is targeting specific user accounts, or if it is being executed at specific times to avoid detection. **Threat intelligence databases** can then be consulted to check whether the observed IP address has been flagged for malicious activity in other incidents. This external correlation can provide valuable insights into the attacker's history and techniques.

Another essential step in the forensic investigation would involve analyzing the **SSH server logs** to identify **failed authentication attempts**, **unusual login times**, or other **indicators of compromise**. Failed login attempts that occur in rapid succession or from unusual locations are often a telltale sign of a brute-force attack. Investigators should also look for patterns such as

multiple login attempts across different user accounts or the use of common default credentials, which could further support the hypothesis that a brute-force attack is underway.

In summary, the **alert metadata** and **threat intelligence** contained in this section provide essential information for understanding the nature of the attack and guiding forensic investigation. The identification of a potential brute-force SSH attack, combined with the severity level and source IP details, offers forensic analysts the context they need to prioritize their response. By correlating this data with external threat intelligence, reviewing server logs, and identifying patterns in access attempts, investigators can effectively trace the source of the attack, mitigate the risk of further exploitation, and strengthen the system's defenses against similar attacks in the future.

13) The **Hunt Interface** in Security Onion is an advanced tool designed for forensic investigators to search, filter, and analyze network and log data, providing a comprehensive view of captured packets and related logs. This screenshot highlights how the Hunt feature is used to conduct detailed analysis of a **Suricata alert** related to a potential **SSH brute-force attack**. The interface is part of the broader network security and digital forensics toolkit provided by Security Onion, which helps analysts track, investigate, and respond to network security threats in real-time.

One of the most critical aspects of the **Hunt tool** is its ability to **search** and **filter** data. In this case, the alert, flagged by the Suricata rule "ET SCAN LibSSH Based Frequent SSH Connections Likely Brute Force Attack," indicates suspicious SSH activity. The **source IP** `192.168.1.222` is identified as the origin of the suspicious connections, and it is flagged for initiating the potential brute-force attack. This is an essential finding because it allows investigators to focus on a specific endpoint, trace its activity, and determine whether it is part of a broader attack campaign or an isolated incident.

The **decoded payload** of the captured traffic provides valuable insights into the nature of the attack. It includes key **SSH handshake details**, including the cryptographic algorithms being used, such as **aes256-ctr** for encryption and **hmac-sha2-512** for message authentication. These cryptographic details allow investigators to understand the security of the session and identify potential weaknesses or misconfigurations. For example, weak or outdated encryption methods may be a sign of an attack targeting known vulnerabilities in SSH configurations. Analyzing the handshake also provides insight into the session's establishment, revealing patterns that might indicate whether the connection is part of an ongoing brute-force attack.

The **rule action**, marked as "Allowed," reveals that while the network traffic was flagged as suspicious, it was not blocked and was permitted to pass through the system. This action may seem counterintuitive at first, but it provides critical information for forensic analysis. The fact that the traffic was allowed indicates that the system may not have had sufficient defenses in place, such as rate limiting, account lockout, or multi-factor authentication, to prevent repeated login attempts. Understanding why the traffic was permitted despite being flagged as suspicious helps identify potential gaps in the network's security posture and can guide future improvements to prevent similar attacks.

The **steps performed** by the analyst involve several key actions within the Hunt interface. The investigator used filters like **"Include," "Exclude,"** or **"Group By"** to refine the dataset, making it easier to isolate the relevant events from a large volume of data. These filters allow the analyst to focus on specific aspects of the traffic, such as repeated SSH connection attempts or interactions with critical systems. The **payload inspection** feature is particularly useful in this case, as it allows the analyst to directly examine the SSH session details and the captured data stream. By inspecting the decoded payload, investigators can identify further anomalies, such as unusual session parameters or the use of common usernames in the attack.

Another valuable feature of the Hunt interface is the ability to **correlate the alert with other events**, which helps analysts build a more comprehensive understanding of the attack. For instance, correlating this SSH brute-force activity with other network logs, such as firewall logs or host-based logs, can reveal whether the source IP is part of a broader attack or if it is targeting multiple systems in the network. Additionally, the option to **add the event to a case** or **download the associated PCAP** (Packet Capture) file allows the investigator to document their findings and carry out further in-depth analysis. The PCAP can be used to examine the network traffic in more detail, or to share the raw data with other forensic tools for advanced inspection.

From a **digital forensics** perspective, the **Hunt Interface** is indispensable in the identification and response to network intrusions. By analyzing **SSH session details** and correlating them with **threat intelligence**, analysts can assess whether the activity is part of an ongoing attack campaign, such as a larger brute-force or credential stuffing effort. Furthermore, correlating this activity with other logs and alerts provides a broader picture of the attacker's methods and objectives, allowing for a more thorough investigation. For example, the presence of repeated login attempts from a single IP, combined with information about the session's cryptographic parameters, may indicate an attacker's attempts to exploit weak SSH configurations, potentially targeting high-value systems or administrative accounts.

In summary, the **Hunt Interface** in Security Onion offers a comprehensive, powerful tool for digital forensic investigations, allowing analysts to dissect network traffic, correlate events, and gain actionable insights into suspicious activities. By utilizing advanced filtering, payload inspection, and event correlation capabilities, investigators can identify potential intrusions, assess the severity of threats, and take appropriate action to mitigate further risks. This interface is an essential component of any forensic investigation, helping teams respond swiftly and effectively to emerging security incidents.

14) The attachment of a **Packet Capture (PCAP)** file to a case in Security Onion is a crucial step in documenting network-related forensic evidence for further analysis. This screenshot illustrates the integration of a specific PCAP file, **sensoroni_securityonion_1011.pcap**, into the Security Onion dashboard. The PCAP file, which is vital for analyzing raw network traffic, provides forensic investigators with detailed data related to network interactions, such as potential brute-force authentication attempts. By attaching this file to the case, investigators can ensure that all relevant traffic data is linked to the specific incident, providing a cohesive and organized record for further examination and analysis.

The **file metadata** for the PCAP includes key information such as the **filename**, **sensoroni_securityonion_1011.pcap**, which clearly identifies the file, and the **file size** of **6,019 bytes**. This size provides an indication of the volume of data captured, helping investigators assess the scope of the network traffic recorded in the capture. More critically, the **hash values**—SHA256, SHA1, and MD5—serve an essential role in validating the integrity and authenticity of the PCAP file. These hash values ensure that the file has not been altered or corrupted during the collection, storage, or transmission process, making them indispensable for maintaining the **chain of custody** in a digital forensic investigation. Forensic tools such as **shasum** or **md5sum** can be used to verify the file's integrity by comparing the hash values calculated for the file against the ones provided. This is an essential practice for ensuring the reliability of evidence that may be used in legal or compliance proceedings.

In the **case details**, the **description** of the attached PCAP file is noted as **"PcapFor_bruteForce_authentication"**, which indicates that the file likely contains network traffic related to brute-force authentication attempts. These attempts, typically characterized by repeated failed login attempts to services like SSH, are often precursors to more serious attacks, such as privilege escalation or unauthorized access. The **severity** of the case is marked as **Critical**, underlining the potential significance of the brute-force attempts and highlighting the need for immediate attention and action. The **priority** of **0** (highest priority) reinforces this urgency, indicating that this case is one of the most pressing issues within the investigation and should be addressed as quickly as possible.

From a **digital forensics** perspective, the attachment of this PCAP file is crucial for analyzing network traffic that could provide insights into the nature of the attack. Brute-force authentication attempts are a common attack vector, and analyzing the captured traffic can help forensic investigators identify the source of the attack, the targeted system, and the methodology used. By examining the raw network traffic in the PCAP file, investigators can also verify if there were any successful attempts, detect unusual login patterns, and confirm whether the attackers gained unauthorized access to critical systems.

The use of **Security Onion** as the monitoring and intrusion detection system (IDS/IPS) provides a centralized platform for forensic investigations. It allows investigators to integrate network traffic data, such as the PCAP file, with alerts and logs from tools like **Suricata** and **Zeek**. This centralized case-management workflow makes it easier to track and correlate findings across different systems, ensuring a streamlined investigative process. The ability to attach PCAP files to cases and maintain associated metadata not only enhances the organization of the investigation but also ensures that all artifacts are documented in a manner that supports the overall integrity of the analysis.

In summary, the attachment of the **PCAP file** to a case within Security Onion is an important step in ensuring the integrity, organization, and thorough analysis of network traffic data during digital forensic investigations. By utilizing hash verification, categorizing the case with appropriate priority and severity, and linking the PCAP file to the broader investigation, forensic analysts can maintain an effective workflow and ensure the reliability of evidence for reporting and legal proceedings. This process helps in identifying and mitigating threats such as

brute-force attacks, reinforcing the importance of thorough network traffic analysis in safeguarding systems and networks.

15) The **Packet Analysis via Hunt Interface** in Security Onion is a powerful forensic tool used to inspect, filter, and analyze network traffic captured during suspicious activities. This specific screenshot visualizes the packet flow between two hosts—**192.168.1.222** (source IP) and **192.168.1.248** (destination IP)—highlighting a potential brute-force attempt over **SSH (port 22)**. By analyzing this traffic, forensic investigators can identify key patterns and behaviors associated with brute-force attacks, which often involve high volumes of connection attempts targeting login services like SSH.

Key observations in this analysis include the **source IP** (`192.168.1.222`) and **destination IP** (`192.168.1.248`), indicating that the attack is directed at the destination system, likely attempting to gain unauthorized access via SSH. The **source port** (`52724`) and **destination port** (`22`) provide additional context, confirming that the traffic is targeting SSH, which is commonly used for remote administration of servers and systems. These details are crucial in understanding the nature of the attack and identifying potential vulnerabilities in the target system's configuration.

The **TCP flags** captured in the packet headers—**SYN**, **ACK**, and **PSH**—are significant indicators of the session's establishment and data transfer. The SYN flag indicates the initiation of a TCP session, the ACK flag confirms the acknowledgment of the handshake process, and the PSH flag suggests that data transfer is occurring. The combination of these flags is typical for establishing a session, and their presence in repeated connection attempts is a common behavior in brute-force attacks, where an attacker systematically tries multiple login combinations in a short time frame.

The **timestamp details** from the packet capture provide further insights into the behavior of the attack. Multiple requests are logged within a span of just **milliseconds** (from **04:58:19.497** to **04:58:19.503**), suggesting a high volume of connection attempts—a hallmark of brute-force activity. The rapid, repetitive nature of these attempts indicates the use of automated tools, which are designed to repeatedly initiate SSH sessions in an attempt to guess the correct login credentials.

From a **digital forensics** perspective, identifying **brute-force behavior** is crucial for detecting unauthorized access attempts that could lead to the compromise of critical systems. The presence of repeated **SYN flags** in the packet headers further supports the hypothesis that an attacker is attempting to establish multiple SSH connections, likely as part of an automated process to crack passwords. These patterns are significant because they point to a **credential-guessing** technique commonly employed by attackers to bypass weak or default authentication configurations.

The **timeline of events** plays a critical role in reconstructing the attack sequence. By analyzing the timestamps and packet flow, investigators can build a chronological picture of the attack, detailing the sequence of connection attempts and any associated patterns. This timeline is

essential for understanding the scale of the brute-force attack, identifying potential windows of exposure, and gathering evidence for further investigation or legal purposes. The ability to reconstruct the attack timeline helps analysts to pinpoint the moment the attack started, how long it persisted, and whether any successful logins occurred.

Tools like **Security Onion's Hunt interface** are invaluable for performing this detailed traffic inspection. The interface enables investigators to apply filters, such as "Include" or "Exclude," to narrow down the dataset and focus on the most relevant traffic. Analysts can also utilize the interface to **export flow data** or **drill down** into the packet payloads for more granular analysis. This capability is crucial for deep-diving into the raw packet data to uncover additional details, such as specific usernames or failed login attempts, which may provide further insights into the attacker's tactics.

In summary, the **Packet Analysis via Hunt Interface** provides forensic investigators with the tools needed to identify, inspect, and analyze brute-force attack traffic. By capturing detailed packet flow, TCP flags, timestamps, and filtering network traffic, the Hunt interface offers essential visibility into the attack's structure and behavior. This detailed analysis is crucial for identifying unauthorized access attempts, reconstructing attack sequences, and strengthening the security posture of the affected systems by mitigating similar attacks in the future.

16) The **Alert Dashboard in Kibana** is an essential feature of the Security Onion platform, enabling forensic investigators to visualize and interpret alerts generated by Suricata. Kibana’s powerful data visualization capabilities offer a comprehensive summary of potentially malicious activity, helping analysts quickly identify trends, assess the severity of incidents, and prioritize their investigative efforts. In this case, the dashboard aggregates data from Suricata, an intrusion detection and prevention system, to provide an overview of network security events that require further attention.

The **alert statistics** displayed in the dashboard indicate that **16 alerts** were triggered, each corresponding to different types of suspicious activity within the network. These alerts are generated by Suricata, a tool that analyzes network traffic for patterns that align with known attack signatures. The **modules involved** include **Suricata**, with 16 alerts detailing specific types of suspicious events. Examples include **ET SCAN Potential SSH Scan** and **ET SCAN LibSSH Based Frequency**, both of which highlight activity related to SSH scanning or brute-force attempts. The alert statistics offer a quick overview of the volume and type of potential threats, helping investigators prioritize critical events.

The **severity breakdown** of these alerts further refines the analysis by categorizing events based on their level of threat. **Medium-severity alerts** may indicate potentially suspicious activity, but they require further investigation to determine their significance. These events might include traffic patterns that are not immediately indicative of an attack but are worth monitoring. In contrast, **high-severity alerts** are critical and point to active threats that need immediate attention. For example, an alert flagged as a **Potential SSH Scan** indicates that the system detected reconnaissance or active attempts to exploit SSH services, which could lead to

unauthorized access. This categorization is vital for investigators, as it helps them focus on the most urgent threats and allocate resources accordingly.

The technical details within the dashboard highlight the specific rule that triggered the alert. In this case, the **ET SCAN Potential SSH Scan** rule is activated when Suricata detects traffic patterns associated with SSH scanning or brute-force attempts. These types of attacks are typically the first step in gaining unauthorized access to systems, as attackers try to guess login credentials using automated tools. By triggering this rule, Suricata alerts investigators to the presence of suspicious activity that may indicate a larger attack campaign targeting SSH services. The **timeline visualization** within Kibana is also a powerful feature, enabling analysts to correlate alerts with known events or anomalies within the network. This allows for a more comprehensive understanding of the attack's progression and context, helping to pinpoint the exact time of the attack and its impact on other systems.

From a **digital forensics** perspective, the alerts generated in this dashboard provide early **indicators of compromise (IOC)**, which are critical for identifying unauthorized access or malicious activity within the network. The visual format of the Kibana dashboard enhances situational awareness, enabling analysts to quickly spot patterns or clusters of suspicious activity. By correlating these alerts with historical data or related incidents, investigators can identify emerging threats and assess whether the activity is part of a broader attack or an isolated event. This is especially useful in large networks where multiple incidents may occur simultaneously, making it difficult to detect individual threats without effective visualization tools.

The integration of Kibana with Security Onion makes it easier for investigators to handle large volumes of security alerts. By using Kibana dashboards, forensic teams can triage alerts, identify recurring patterns, and determine the next steps in the investigation. For instance, an alert indicating a potential SSH brute-force attack might prompt an analyst to cross-reference it with other security logs, such as system authentication records or firewall logs, to determine if the attack was successful or if it was thwarted by existing defenses. Additionally, the dashboard's interactive features allow analysts to filter and drill down into specific data, exporting relevant information for further analysis or adding it to an ongoing case for more detailed investigation.

In summary, the **Alert Dashboard in Kibana** is a critical tool in the digital forensic process, enabling investigators to efficiently manage and analyze network security alerts. By leveraging Kibana's powerful data visualization capabilities, forensic teams can quickly identify and respond to emerging threats, correlate incidents across the network, and gain deeper insights into potential security breaches. The integration of Suricata alerts with Kibana ensures that suspicious traffic patterns are flagged early, enabling prompt investigation and mitigation.

17) The **Kibana Dashboard - Security Onion Alerts Overview** provides a comprehensive view of network security alerts generated by Security Onion, aggregating and visualizing data over the past 10 days. This dashboard is crucial for digital forensic investigators as it helps track potential security breaches, correlate suspicious activities, and prioritize investigations based on alert categories. By organizing the alerts according to activity, source IPs, and destination IPs,

the dashboard simplifies the analysis of complex network traffic patterns, enabling forensic teams to focus on the most significant threats.

The **alert data** is organized into two primary categories: **Attempted Information Leak** and **Attempted Administrator Privilege Escalation**. The first category, **Attempted Information Leak**, appears 12 times, suggesting that there were multiple instances where there was a potential unauthorized attempt to extract sensitive or confidential information from the monitored environment. These leaks might involve sensitive data, intellectual property, or personally identifiable information (PII), making them a high-priority threat for investigators. The second category, **Attempted Administrator Privilege Escalation**, was recorded 4 times, indicating that there were attempts to gain unauthorized administrative access to the network or systems. Privilege escalation is a common tactic used by attackers to increase their control over a compromised system, often as a precursor to deeper exploitation, lateral movement, or data exfiltration.

By categorizing the alerts, the **Kibana dashboard** allows forensic investigators to understand the nature of the potential security breaches more clearly and prioritize their investigation efforts accordingly. For instance, while both categories are significant, **Attempted Administrator Privilege Escalation** events are typically of higher priority due to the potential severity of unauthorized administrative access. This categorization helps investigators decide where to allocate resources and which alerts require immediate attention.

The **source IP analysis** reveals that **192.168.1.222** was flagged in **16 alerts**, indicating that this internal IP address is involved in a considerable amount of suspicious activity. This repeated occurrence of alerts linked to a specific source IP could signify that the device or system with this IP address is compromised or is being used as a pivot point for attacks. It is a clear indicator of potential malicious activity within the network, making it a point of focus for further investigation. Investigators would likely attempt to trace the activity back to this source, analyzing logs, system behavior, and potential vulnerabilities to determine the root cause.

The **destination IP analysis** shows that all 16 alerts were targeted at **192.168.1.248**, a specific asset or service within the network. The concentration of attacks on a single destination IP raises suspicions that this system could be a high-value target or that attackers are focusing their efforts on exploiting a known vulnerability in that system. By tracking the frequent targeting of this IP, forensic investigators can determine whether it is a critical system, such as a server or network device, and whether its security has been compromised. This targeted attack pattern could also suggest that the attackers are attempting to gain unauthorized access or exfiltrate sensitive data from this specific asset.

The **tools used** in this process include **Kibana**, a powerful data visualization and exploration tool integrated with Security Onion. Kibana allows analysts to filter, sort, and visualize large volumes of data, making it easier to identify patterns and track suspicious activities. **Security Onion**, an open-source platform for network monitoring and intrusion detection, provides the underlying infrastructure for capturing and analyzing alerts generated by various tools such as Suricata. By leveraging these tools, forensic investigators can gain deeper insights into network

activity, identify the source of potential breaches, and determine the extent of the damage or compromise.

In terms of **digital forensics**, the Kibana dashboard provides invaluable insights into potential malicious activities within the network. By categorizing alerts based on their severity and relevance, investigators can more easily pinpoint compromised systems, track the origin of attacks, and assess the potential damage. The ability to visualize and analyze these alerts in real-time gives investigators a comprehensive view of the security landscape, making it easier to detect threats early and take appropriate action to mitigate risks. Furthermore, the dashboard's capabilities in correlating alerts across systems allow for the identification of broader attack trends, providing a clear understanding of the attacker's methods and helping to anticipate their next move.

In summary, the **Kibana Dashboard** in Security Onion is a crucial tool for digital forensic investigations, offering powerful data visualization capabilities to track and analyze network security alerts. By categorizing alerts and focusing on key IP addresses, forensic teams can prioritize their efforts, streamline incident response, and better understand the scope of potential security breaches. This dashboard plays a central role in helping investigators detect unauthorized access, identify compromised systems, and respond to emerging threats effectively.

18) The **Kibana Dashboard - SSH Authentication Analysis** is an essential tool for monitoring and analyzing SSH-related events in the **Security Onion** platform, focusing specifically on authentication success rates and associated hash values. This feature is critical for forensic investigators, as it provides a detailed overview of SSH traffic over a specific period (in this case, the last 10 days). The dashboard highlights crucial data points, such as the success and failure of SSH logins, as well as hash values that can help correlate SSH sessions with specific user credentials or system activities.

The **authentication success rates** shown in the pie chart provide immediate insights into the SSH authentication process. The chart distinguishes between **successful** and **unsuccessful** SSH authentications, with **"False" (failed authentications)** accounting for the majority of the recorded attempts. This is a significant finding, as a high number of failed login attempts is often indicative of a **brute-force attack**, where an attacker repeatedly tries different combinations of usernames and passwords in an attempt to gain unauthorized access. Brute-force attacks targeting SSH are a common method used by attackers to exploit weak or default credentials, and the prevalence of failed authentication attempts serves as an early warning of such activity. On the other hand, the smaller fraction of **"True" (successful authentications)** implies that, while there are fewer legitimate or successful login attempts, some authorized access may have been granted, which can be important for identifying whether attackers succeeded in compromising accounts.

The **hash analysis** section provides a deeper layer of insight by identifying two distinct hash values associated with SSH sessions. **Hash 1** (`742b4fd5532ca4f243aae081017fe8c5`) appears **29 times**, suggesting it is linked to a recurring user credential or **SSH fingerprint**

observed in multiple authentication events. This pattern could indicate that the same user or system is involved in several sessions, providing valuable context for identifying compromised accounts or repeat access attempts. **Hash 2** (`aae6b9604f6f3356543709a376d7f657`), which was recorded only **1 time**, may represent a unique or isolated event, possibly a one-off authentication attempt or a session involving a different user or system. These hash values could correspond to **cryptographic fingerprints** associated with SSH sessions, such as **session keys**, **usernames**, or **authentication tokens**. Analyzing these hashes helps investigators correlate specific sessions to particular users or systems, making it easier to identify compromised credentials or ongoing malicious activity.

In the **direction data** section, the "Direction" panel reveals that **no data was found**, which could be due to either a lack of classification for inbound or outbound traffic or a filtering issue in the query. This absence of directional data might limit the ability to fully analyze the flow of traffic between systems, but it can also prompt forensic analysts to review their query settings or investigate further to determine if any directional data was missed or misclassified.

Tools used for this analysis include **Kibana**, which is employed for visualizing and querying SSH traffic events, and **Security Onion**, the platform responsible for capturing and processing SSH session logs. Kibana allows forensic investigators to filter, sort, and drill down into SSH-related data, providing a clear view of authentication attempts and related activities. **Security Onion** plays a central role in collecting network data, integrating alerts, and generating logs that can be used for further analysis.

From a **digital forensics** perspective, **SSH logs** are invaluable in investigations of **unauthorized access**. A high number of failed authentications strongly suggests an ongoing **brute-force attack**, making it critical to focus on these events to identify the scope of the attack and prevent further access. The **hash analysis** section is particularly important for correlating SSH sessions with **specific user credentials or systems**, allowing forensic investigators to pinpoint compromised accounts or systems. By analyzing this data, forensic teams can better understand the nature of the attack, determine whether attackers have successfully gained access, and assess the potential impact on the network or infrastructure.

In summary, the **Kibana Dashboard - SSH Authentication Analysis** is an indispensable tool for forensic investigations related to SSH activity. By providing visual insights into authentication success rates, analyzing hash values, and presenting key event data, this dashboard helps forensic analysts detect and respond to suspicious activities, such as brute-force attacks. It also aids in identifying compromised credentials, correlating SSH sessions with user accounts, and assessing the full scope of an intrusion. With the ability to filter, visualize, and analyze SSH data, this tool plays a crucial role in maintaining network security and ensuring timely response to emerging threats.

19) The **Elastic Kibana interface**, integrated with **Security Onion**, provides a powerful visualization tool for analyzing network and log data, with a specific focus on **SSH activity** over the last 10 days. The **SSH Dashboard Summary** presents a clear and concise overview of key SSH-related events, offering insights into potential security incidents and enabling forensic

investigators to identify suspicious activities quickly. By providing detailed metrics and filtering capabilities, this dashboard is instrumental in pinpointing malicious actions such as brute-force attacks or unauthorized access attempts targeting SSH services.

One of the **key metrics** displayed on the dashboard is the **total count** of **30 SSH-related events** detected within the monitoring period. This statistic gives a high-level overview of the frequency of SSH traffic that is potentially suspicious and warrants further investigation. The dashboard highlights critical data points, such as **source IPs** and **destination IPs**. The **source IP** `192.168.1.222` initiated all 30 SSH events, suggesting that this IP address is the point of origin for the suspicious activity. This is a significant observation, as repeated attempts from a single IP address can indicate a **brute-force attack** or attempts to exploit weak SSH credentials. The **destination IP**, `192.168.1.248`, is the target of all 30 SSH events, pointing to a specific system or service within the network that may be under threat. This concentration of activity on a single IP raises concerns about the system's security and could indicate a vulnerability that attackers are trying to exploit.

The dashboard also displays the **SSH clients** used in the sessions. **libssh_0.1** was recorded in **29 events**, with a single instance of **OpenSSH** being used. The variation in the SSH client indicates that different methods or tools may have been used in an attempt to connect to the target system, suggesting that the attacker may be employing different techniques to bypass security defenses. The **SSH server** analysis reveals that all sessions were connected to an **OpenSSH_6.6** server, which could be a potential vulnerability if this version is outdated or lacks critical security patches. This detail is important for forensic investigators to examine, as it may help identify whether the system's SSH configuration or software version contributed to the attack.

The **steps performed** in this investigation begin with a **tag-based search** using the keyword `tags:ssh` to focus the dashboard on SSH-specific events. This type of search allows forensic analysts to filter out unrelated traffic and zoom in on the most relevant data, streamlining the investigation process. The dashboard's **visualization and analysis** features, including pre-built widgets, display key metrics, such as the count of events over time, the source and destination IPs, and SSH protocol details. This visualization makes it easy to spot patterns, such as unusual spikes in SSH connections or activity concentrated on specific IP addresses, which could indicate malicious activity. The **drill-down options** in Kibana allow analysts to refine the time range of the analysis, enabling them to zoom in on specific periods of interest where suspicious activity might have occurred. Additionally, analysts can export the data for further analysis or to share it with other teams for additional investigation.

From a **digital forensics** perspective, this dashboard is an essential tool for identifying **patterns and anomalies** in SSH traffic. The ability to focus on SSH-specific events, combined with the detailed breakdown of source and destination IPs, SSH clients, and server information, allows forensic investigators to detect potential threats early. For example, frequent connection attempts from a single source IP (`192.168.1.222`) could signal a **brute-force attack**, while the targeting of a specific destination IP (`192.168.1.248`) suggests that the attacker is

focusing on a critical asset within the network. Additionally, analyzing variations in the SSH clients used and the SSH server version can provide valuable context for understanding the attack method and identifying potential vulnerabilities in the system.

This dashboard also plays a key role in incident response, as it allows forensic investigators to **prioritize their efforts** based on the severity of the detected activity. By identifying potential threats early and correlating them with known attack patterns, investigators can take swift action to contain the threat, mitigate risks, and prevent further compromise. The data captured in the dashboard can also be used to **correlate SSH events** with other network activities, enabling a more comprehensive investigation of potential intrusions.

In conclusion, the **Elastic - SSH Dashboard** is an indispensable tool for digital forensic investigations, offering detailed insights into SSH-related traffic patterns and providing a comprehensive view of potential security incidents. By enabling forensic teams to quickly identify and analyze suspicious activity, this dashboard helps streamline the investigation process, prioritize response efforts, and enhance overall network security. The integration of Kibana with Security Onion makes it a powerful platform for real-time analysis of network threats, ensuring that investigators can stay ahead of attackers and mitigate risks effectively.

20)The **Elastic - Detailed Network Connections View** in Kibana is an advanced feature used to inspect and analyze specific network connection events, focusing on SSH traffic. This screenshot offers a granular look at individual SSH-related events by revealing important connection details such as timestamps, source and destination IP addresses, ports, and unique session identifiers. This level of detail is crucial for forensic investigators, as it provides the necessary data to trace the sequence of events, identify potential security incidents, and reconstruct the timeline of an attack.

One of the key elements observed in the logs is the **source IP** (`192.168.1.222`) and **destination IP** (`192.168.1.248`), both of which consistently appear across all events. This repetition highlights a significant pattern, indicating that the suspicious activity is originating from a specific internal IP and targeting a particular system or service. The **destination port 22** confirms that the traffic is associated with **SSH**—a protocol commonly used for secure remote administration of systems. This consistent targeting of port 22 suggests that the traffic is related to SSH connection attempts, which are often vulnerable to brute-force attacks or unauthorized access attempts.

The logs also reveal **unique session identifiers**, such as `CYRGvj3Ya0i03qWJHe`, which provide session-level granularity, allowing forensic analysts to track and correlate events across multiple network interactions. These identifiers are crucial for investigating the progression of a potential attack, as they link specific actions or activities to individual sessions. By examining these session identifiers, analysts can trace the timeline of the connection, analyze the authentication process, and identify whether multiple sessions are part of the same attack or whether different sessions involve different attack methods.

In terms of **steps performed**, the **event filtering** process is fundamental in narrowing down the focus to SSH-related events. The analyst filters the logs using the `tags:ssh` query to isolate SSH traffic, removing irrelevant data and allowing the investigator to zero in on the most relevant connection events. Once the logs are filtered, the **inspection** process begins, where details like timestamps and session IDs are reviewed to establish the **timeline** and **patterns** of the connections. By analyzing the sequence of connection attempts, investigators can understand whether the attacker was successful in gaining access, identify possible brute-force attempts, and assess the scale of the attack.

The **correlation** process involves cross-referencing these logs with other relevant dashboards, such as those detailing SSH client/server information. This cross-referencing helps provide context for the attack, including the type of SSH client and server involved, which can indicate whether attackers are exploiting specific vulnerabilities or using tools to bypass security defenses. Correlating this data across multiple logs and dashboards allows investigators to piece together a more complete picture of the attack, enhancing their understanding of the threat landscape and improving the effectiveness of their investigation.

From a **digital forensics** perspective, the **detailed network logs** captured in this view are invaluable for reconstructing the **timeline** of an incident. By analyzing the connection details, forensic investigators can determine the sequence of events, assess the volume of attempts, and identify which systems or services were targeted. This granular level of detail allows investigators to track the **source** of the attack, determine whether the attack was successful, and correlate the attack with other activities occurring on the network. Furthermore, these logs provide essential evidence that can be used for **remediation** efforts, such as blocking malicious IP addresses or strengthening security measures, as well as for potential **legal actions** where a clear and detailed timeline of events is required.

In summary, the **Elastic - Detailed Network Connections View** is a critical tool in the digital forensic toolkit. By providing granular details about SSH traffic and connection events, it allows investigators to track suspicious activity, correlate sessions, and understand the scope of a potential attack. The ability to filter, inspect, and correlate network logs in this way enables forensic teams to identify compromised systems, respond to emerging threats, and ensure that appropriate measures are taken to mitigate risks and prevent further security breaches.