Apexplanet cybersecurity internship

Task – 1

# Foundation and environment setup

Submitted by – Shourya lakhera

• Cybersecurity basics

Cyber security is the practice of **protecting computers, networks, data, and systems from digital attacks**. These attacks aim to access, change, destroy, steal, or extort information, or disrupt normal operations.

⋆ 1. CIA Triad

Confidentiality

- Ensures information is accessible only to authorized users
- Techniques: Encryption, access control, authentication
- Protects sensitive data from unauthorized disclosure

Integrity

- Ensures data is accurate, consistent, and unaltered
- Techniques: Hashing, checksums, version control
- Prevents tampering, corruption, or unauthorized modification

Availability

- Ensures systems and data are accessible when needed
- Techniques: Backups, redundancy, load-balancing, DDoS protection
- Prevents downtime and ensures business continuity

⭑ 2. Threat Types

Malware

- Includes viruses, worms, trojans, ransomware, spyware
- Damages or steals data and disrupts systems

Phishing

- Fraudulent emails/messages tricking users into giving sensitive information

Social Engineering

- Manipulating people to bypass security
- Examples: impersonation, baiting, tailgating

Ransomware

- Encrypts files and demands payment for decryption

Insider Threat

- Employees or trusted individuals misusing access intentionally or accidentally

DDoS (Distributed Denial of Service)

- Overloads servers to make services unavailable

Zero-Day Exploits

- Attacks using unknown software vulnerabilities

⭑ 3. Attack Vectors

Email

- Phishing, malicious attachments, fraudulent links

Web Applications

- SQL Injection, XSS, CSRF, insecure APIs

Network

- Man-in-the-Middle (MITM), port scanning, exploitation of open services

Endpoints

- Laptops, PCs, mobile devices targeted with malware or exploits

USB/Removable Media

- Malware spreads through infected physical devices

Social Engineering

- Psychological manipulation to gain access (calls, in-person, messages)

Cloud Services

- Misconfigurations, insecure APIs, unauthorized access

⭐ Lab Environment Setup

🔧 1. Tools Installed

Virtualization Platform

- **VirtualBox / VMware Workstation / Hyper-V**
- Used to run multiple virtual machines safely in isolation

Operating Systems

- **Kali Linux** (for penetration testing tools)
- **Windows 10/11** (target machine for testing)
- **Ubuntu Server** (optional for running services)

Security & Testing Tools

On Kali Linux

- **Nmap** – Network scanning
- **Metasploit Framework** – Exploitation platform
- **Burp Suite** – Web application testing
- **Wireshark** – Packet analysis
- **Hydra / Medusa** – Password brute forcing
- **John the Ripper / Hashcat** – Password cracking
- **Gobuster / Dirsearch** – Directory & file enumeration

On Windows (Victim Machine)

- **XVulnerable Apps** (intentionally insecure)
  - DVWA (Damn Vulnerable Web App)
  - OWASP Juice Shop
- **Sysinternals Suite** (for monitoring)
- **XAMPP** (to host vulnerable apps)

Support Tools

- **VS Code / Notepad++** – Editing scripts
- **Postman** – API testing
- **Python / PowerShell** – For automation scripts

🌐 2. Network Setup

Private / Isolated Lab Network

- Create an internal or host-only network
- Prevents internet access → safer for malware testing
- Machines communicate only inside the lab

Network Topology

- **Attacker Machine:** Kali Linux

- **Target Machine:** Windows 10/11
- **Optional Server:** Ubuntu for hosting services
- All connected through a **virtual switch**

IP Addressing

- Use static IPs for consistency:
  - Kali Linux: **192.168.56.10**
  - Windows: **192.168.56.20**
  - Ubuntu Server: **192.168.56.30**
- Subnet: **255.255.255.0**

Network Modes

- **Host-Only Adapter**
  - Best for safe, isolated testing
  - All VMs connect to each other only
- **NAT**
  - Provides internet for tools updates
  - Still isolated from the physical network

Firewall Rules

- Disable or relax firewall on target machine (Windows) for lab purposes
- Keep firewall active on host system to avoid accidental exposure

Linux Fundamentals

1. File Navigation

- `pwd` → shows current directory
- `ls` → lists files/folders
- `cd folder` → go to folder
- `cd ..` → go back
- `mkdir name` → create folder
- `rm file` → delete file
- `cp source dest` → copy
- `mv old new` → move/rename

---

2. Permissions

- Three permissions:
  - **r** = read
  - **w** = write
  - **x** = execute
- Apply to:
  - **u** = user (owner)
  - **g** = group
  - **o** = others
- Check permissions: `ls -l`
- Change permissions:
  - `chmod 755 file`
  - `chmod u+x file`
- Change owner: `chown user:group file`

---

3. Package Management

Ubuntu/Debian (APT)

- `sudo apt update`
- `sudo apt install package`
- `sudo apt remove package`

CentOS/Fedora (DNF)

- `sudo dnf install package`
- `sudo dnf remove package`

---

4. Network Commands

- `ip a` → show IP info
- `ping google.com` → test connection
- `hostname -I` → system IP
- `curl url` → fetch data
- `wget url` → download file
- `nslookup domain` → DNS info

Network Basics

1. OSI Model (7 Layers)

1. **Physical** – cables, signals
2. **Data Link** – MAC address, switches
3. **Network** – IP address, routing
4. **Transport** – TCP/UDP
5. **Session** – session control
6. **Presentation** – encryption, compression
7. **Application** – http, ftp, dns

Easy way to remember:
Please **D**o **N**ot **T**hrow **S**ausage **P**izza **A**way

2. TCP/IP Model (4 Layers)

1. **Network Access** (physical + data link)
2. **Internet** (IP, routing)
3. **Transport** (TCP/UDP)
4. **Application** (HTTP, DNS, FTP, etc.)

3. DNS, HTTP, HTTPS Working

DNS (Domain Name System)

- Converts **domain names → IP addresses**
- Example: google.com → 142.250.x.x
- Steps:
    1. User enters domain
    2. DNS resolver checks cache
    3. Queries DNS server
    4. Returns IP to the browser

HTTP (Hypertext Transfer Protocol)

- Used for web communication
- **Unsecured** (data can be read)
- Uses **port 80**

HTTPS (Secure HTTP)

- Same as HTTP but **encrypted with SSL/TLS**
- Data is secure
- Uses **port 443**

---

4. IP Addressing & Subnetting

IP Address

- Unique address of device
- Two types:
  - **IPv4** (32-bit, e.g., 192.168.1.1)
  - **IPv6** (128-bit, e.g., fe80::1)

Subnetting (simple explanation)

- Breaking a large network into **smaller parts**
- Helps in:
  - reducing network traffic
  - improving security
  - efficient IP usage
- Subnet mask example:
  - `/24 = 255.255.255.0`
  - `/16 = 255.255.0.0`

NAT (Network Address Translation)

- Converts **private IP** → **public IP** when going to internet
- Used in routers
- Types:
  - **SNAT** – source NAT (private → public)
  - **DNAT** – destination NAT (public → private)
- Helps save public IP addresses

Cryptology Basics

---

1. Encryption Types

A. Symmetric Encryption

- Same **key for encryption and decryption**
- Fast and used for large data
- Examples: **AES, DES, 3DES**
- Key problem: sharing the key safely

B. Asymmetric Encryption

- Uses **public key** (encrypt) + **private key** (decrypt)
- Secure key exchange
- Examples: **RSA, ECC**
- Slower than symmetric

C. Hybrid Encryption

- Used by HTTPS
- Symmetric key for data
- Asymmetric keys for key exchange
- Combines speed + security

---

2. Hashing

- One-way function (cannot be reversed)
- Used for passwords, file integrity
- Output is fixed length (hash value)

Common Hash Algorithms

- **MD5** – old, not secure
- **SHA-1** – broken
- **SHA-256** – widely used
- **SHA-512** – very strong

Hash Properties

- **One-way**
- **Deterministic** (same input → same output)
- **Collision resistant** (hard to find two inputs with same hash)

---

3. Certificates (Basics)

Used in **HTTPS, VPNs, secure communication**

Digital Certificate Contains

- Public key
- Owner name/domain
- Issuer (CA)
- Expiry date
- Signature of CA

Certificate Authorities (CA)

- Trusted organizations
- Examples: Let's Encrypt, DigiCert

How HTTPS Uses Certificates

1. Browser connects to website
2. Website sends **SSL/TLS certificate**
3. Browser verifies certificate with CA
4. Creates secure encrypted connection

---

4. Hands-on Commands (Very Simple)

A. Generate a Hash

```
echo "hello" | sha256sum
```

B. Generate RSA Keys

```
openssl genrsa -out private.key 2048
openssl rsa -in private.key -pubout -out public.key
```

C. Encrypt + Decrypt Using OpenSSL

Encrypt:

```
openssl rsautl -encrypt -inkey public.key -pubin -in file.txt -out file.enc
```

Decrypt:

```
openssl rsautl -decrypt -inkey private.key -in file.enc -out file.txt
```

D. View Certificate Details

```
openssl x509 -in certificate.crt -text –noout
```

Cryptology Basics

1. Encryption Types

A. Symmetric Encryption

- Same **key for encryption and decryption**
- Fast and used for large data
- Examples: **AES, DES, 3DES**
- Key problem: sharing the key safely

B. Asymmetric Encryption

- Uses **public key** (encrypt) + **private key** (decrypt)
- Secure key exchange
- Examples: **RSA, ECC**
- Slower than symmetric

C. Hybrid Encryption

- Used by HTTPS
- Symmetric key for data
- Asymmetric keys for key exchange
- Combines speed + security

2. Hashing

- One-way function (cannot be reversed)
- Used for passwords, file integrity
- Output is fixed length (hash value)

Common Hash Algorithms

- **MD5** – old, not secure
- **SHA-1** – broken
- **SHA-256** – widely used
- **SHA-512** – very strong

Hash Properties

- **One-way**
- **Deterministic** (same input → same output)
- **Collision resistant** (hard to find two inputs with same hash)

---

3. Certificates (Basics)

Used in **HTTPS, VPNs, secure communication**

Digital Certificate Contains

- Public key
- Owner name/domain
- Issuer (CA)
- Expiry date
- Signature of CA

Certificate Authorities (CA)

- Trusted organizations
- Examples: Let's Encrypt, DigiCert

How HTTPS Uses Certificates

1. Browser connects to website
2. Website sends **SSL/TLS certificate**
3. Browser verifies certificate with CA
4. Creates secure encrypted connection

---

4. Hands-on Commands (Very Simple)

A. Generate a Hash

```
echo "hello" | sha256sum
```

B. Generate RSA Keys

```
openssl genrsa -out private.key 2048
openssl rsa -in private.key -pubout -out public.key
```

C. Encrypt + Decrypt Using OpenSSL

Encrypt:

```
openssl rsautl -encrypt -inkey public.key -pubin -in file.txt -out file.enc
```

Decrypt:

```
openssl rsautl -decrypt -inkey private.key -in file.enc -out file.txt
```

D. View Certificate Details

```
openssl x509 -in certificate.crt -text -noout
```

#LINUX CHEAT SHEET

LINUX CHEAT-SHEET

FILE SYSTEM

"**ls**'
"**cd**'
"**pwd**'
"**mkdir**
"**rm** filename'

FILE PERMISSIONS

"chmod permissions
"change file permission
"chown owner
"change groupownership

PROCESS MANAGEMENT

"ps'
"**top**'

DISK USAGE

"cf -ow system disk usage

DISK USAGE

"df – show v stmd: i ɘɯusage
"ip address

SEARCH PATTERNS

"**grep pattern**'
"**find /** – name: file

PACKAGE MANAGEMENT

"apt-get update
"apt-get upgrade
"dypget installed packages

TEXT EDITORS

"**nano** filename
"**vi** filename
"**vim** filename