TASK#1: REPORT
THEORETICAL KNOWLEDGE

CONTENT:

- **UNDERSTANDING SECURITY ASSESSMENT**
- **VAPT METHODOLOGY**
- **SECURITY STANDARDS & COMPLIANCE**
- **RISK ASSESSMENT BASICS**
- **COMMON VULNERABILITIES**
- **DOCUMENTATION FUNDAMENTALS**

UNDERSTANDING SECURITY ASSESSMENT

OBJECTIVE:

Learning how to evaluate systems without paid tools.

SECURITY ASSESMENT: identify weakness using frameworks:

It is the systematic process of identifying, analyzing, and mitigating security weaknesses in systems, networks, and applications. It helps organizations understand their security posture and reduce the risk of cyberattacks.

It follows frameworks such as:

1. NIST (National Institute of Standards and Technology)
2. ISO/IEC 27001
3. CIS (Center for Internet Security) Benchmarks

TYPES OF SECURITY TESTING:

VULNERABILITY ASSESSMENT: USE TOOLS LIKE OPENVAS (OPEN-SOURCE VULNERABILITY SCANNER):

It focuses on identifying known security flaws in systems, services, and applications. It does not exploit vulnerabilities but reports their existence and severity. It uses tool like OPENVAS.

It features are:

- It scans for known CVEs
- It Generates detailed vulnerability reports
- It Uses a continuously updated vulnerability database

COMMANDS TO START AND INITIALISE OPENVAS:

```
Sudo apt update
Sudo apt install open vas
Sudo gvm-setup
Sudo gvm-start
https:/target_ip_address/
```

PENETRATION TESTING: SIMULATE ATTACKS WITH KALI LINUX TOOLS (METASPLOIT, NMAP):

It simulates real-world cyberattacks to determine whether vulnerabilities can be exploited. It helps validate the effectiveness of security controls.

It uses platform – kali linux where the common tools used are:

- **NMAP: NETWORK SCANNING AND DISCOVERY:**

COMMANDS:

```
Nmap target_IP
```

Scanning all ports:

```
Nmap -p- target_IP
```

Scanning to get service and version detection:

```
Nmap -sV target_Ip
```

Scanning to get os detection:

```
Nmap -O target_IP
```

- **METASPLOIT COMMANDS:**

```
Msfconsole
search vsftpd
use exploit/unix/ftp/vsftpd_234_backdoor
Set RHOSTS target_IP
Exploit
```

COMPLIANCE TESTING: VALIDATE AGAINST STANDARDS USING CHECKLISTS (E.G., CIS BENCHMARKS):

It ensures that systems adhere to established security standards and policies.

STANDARDS WHICH THEY USE:

- CIS Benchmarks
- NIST SP 800-53
- ISO/IEC 27001

Compliance testing is often checklist-based and does not involve active scanning.

EXAMPLE: CIS BENCHMARK COMPLIANCE CHECK:

```
cat /etc/login.defs | grep PASS_
sudo ufw status
systemctl list-units --type=service
cut -d: -f1 /etc/passwd
```

VAPT METHODOLOGY

OBJECTIVE:

Follow a structured approach using methodologies.

It Uses a defined methodology ensures:

- Consistency in testing
- Legal and ethical compliance
- Accurate risk evaluation
- Professional reporting

PHASES:

PLANNING: DEFINE SCOPE WITH TOOLS LIKE DRADIS CE:

The planning phase defines what will be tested, how it will be tested, and which tools will be used. This phase avoids legal issues and ensures focused testing.

The Tool which is being Used are: Dradis Community Edition (CE) – Open-source collaboration and documentation tool. helps in scope definition, evidence collection, and team collaboration.

COMMANDS TO INSTALL AND EXECUTE:

```
Sudo apt install dradis
Dradis
https:/target_IP_Address/
```

DISCOVERY: USE NMAP (NETWORK SCANNING) AND OWASP ZAP (WEB APP SCANNING):

The discovery phase focuses on information gathering and vulnerability identification without exploitation.

NMAP COMMANDS:

```
Nmap target_IP
```

OWASP ZAP COMMANDS:

```
zaproxy
zap-baseline.py -t http://example.com
```

- Identify web vulnerabilities (XSS, SQL Injection, CSRF)
- Perform automated and manual testing

ATTACK: EXPLOIT VULNERABILITIES WITH METASPLOIT FRAMEWORK:

This phase validates vulnerabilities by exploiting them safely to assess real-world impact.

COMMANDS FOR METASPLOIT FRAMEWORK:

```
Msfconsole
search smb
use exploit/windows/smb/ms17_010_eternalblue
Set RHOSTS target_IP
Exploit
```

REPORTING:

- Executive summary
- Scope and methodology
- Vulnerability details
- Risk ratings
- Proof of concept
- Remediation recommendations

HOW TO LEARN: PRACTICE THE OWASP WEB SECURITY TESTING FRAMEWORK.

OWASP Web Security Testing Framework (WSTF):

The OWASP WSTF provides:

- Step-by-step web testing methodology
- Mapping of vulnerabilities to testing phases
- Hands-on learning approach

Practice Areas:

- Authentication Testing
- Session Management
- Input Validation
- Error Handling
- Business Logic Testing

SECURITY STANDARDS & COMPLIANCE

OBJECTIVE:

Security Standards and Compliance ensure that organizations protect sensitive data, maintain privacy, and follow legal and ethical security practices. Compliance reduces legal risks, improves trust, and strengthens overall security posture.

STANDARDS:

GDPR (GENERAL DATA PROTECTION REGULATION)

It Protects the personal data and privacy of individuals .

The Key Requirements are:

- Lawful data processing
- User consent
- Data minimization
- Right to access and delete data
- Breach notification within 72 hours

HIPAA (HEALTH INSURANCE PORTABILITY AND ACCOUNTABILITY ACT)

The Key Safeguards are:

- Administrative safeguards
- Physical safeguards
- Technical safeguards (access control, encryption, auditing)

ISO/IEC 27001

An international standard for establishing an Information Security Management System (ISMS).

The Core Principles are:

- Risk assessment
- Security controls (Annex A)
- Continuous improvement (PDCA cycle).

USE THE OWASP TOP 10 TO PRIORITIZE WEB VULNERABILITIES:

The OWASP Top 10 is a globally recognized list of the most critical web application security risks. It helps organizations and learners prioritize vulnerabilities based on real-world impact.

Common OWASP Top 10 Vulnerabilities:

Broken Access Control
Security Misconfiguration
Software Supply Chain Failures
Cryptographic Failures
Injection
Insecure Design
Authentication Failures
Software/Data Integrity Failures
Logging & Alerting Failures
Mishandling of Exceptional Conditions

OWASP Top 10 helps in:

- Mapping vulnerabilities to GDPR and HIPAA requirements
- Improving secure coding practices
- Enhancing ISO 27001 application security controls

RISK ASSESSMENT BASICS:

Risk Assessment is the process of identifying security vulnerabilities, analyzing their potential impact, and prioritizing them based on risk. Since resources are limited, not all vulnerabilities can be fixed at once risk assessment helps determine what to address first.

Risk is commonly defined as:
Risk = Likelihood × Impact

CVSS (COMMON VULNERABILITY SCORING SYSTEM)

The CVSS is an industry-standard scoring system used to evaluate the severity of security vulnerabilities. It provides a numerical score ranging from 0.0 to 10.0.
CVSS is widely used by:

- NVD (National Vulnerability Database)
- Vulnerability scanners
- Security teams

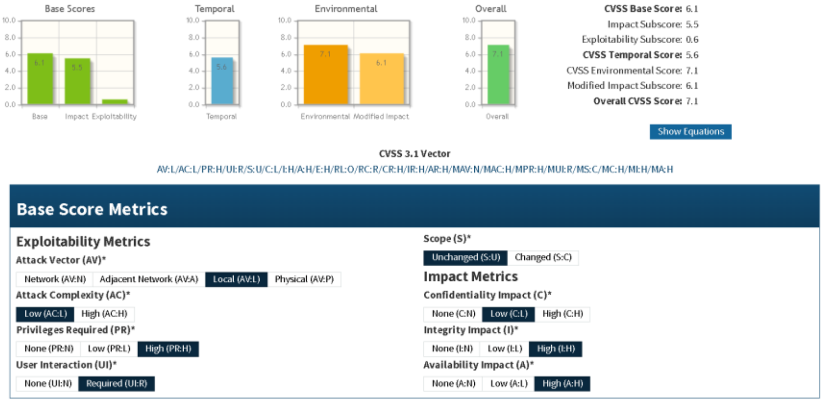
Score Range	Severity
0.0	None
0.1 – 3.9	Low
4.0 – 6.9	Medium
7.0 – 8.9	High
9.0 – 10.0	Critical

USING THE NVD CVSS CALCULATOR

The NVD CVSS Calculator allows manual calculation of vulnerability scores based on exploitability and impact metrics.

- Key Metrics are:
- Attack Vector (AV)
- Attack Complexity (AC)
- Privileges Required (PR)
- User Interaction (UI)
- Confidentiality (C)
- Integrity (I)
- Availability (A)

For example:



CVSS Base Score: 6.1
Impact Subscore: 5.5
Exploitability Subscore: 0.6
CVSS Temporal Score: 5.6
CVSS Environmental Score: 7.1
Modified Impact Subscore: 6.1
Overall CVSS Score: 7.1

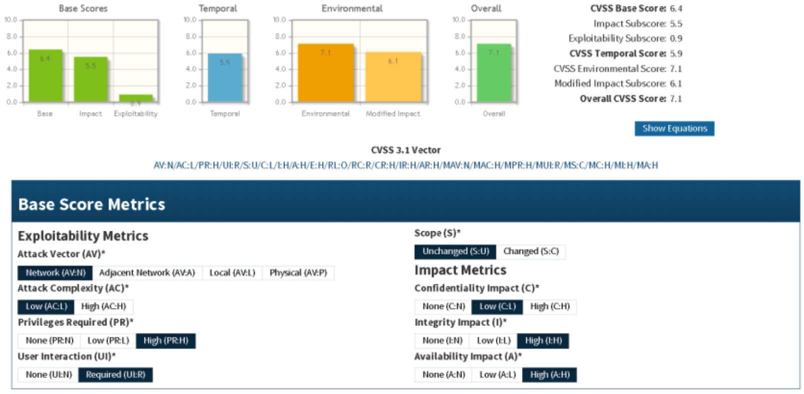
CVSS 3.1 Vector
AV:L/AC:L/PR:H/UI:R/S:U/C:L/I:H/A:H/E:H/RL:O/RC:R/CR:H/IR:H/AR:H/MAV:N/MAC:H/MPR:H/MUI:R/MS:C/MC:H/MI:H/MA:H

Base Score Metrics

Exploitability Metrics
Attack Vector (AV)*
Network (AV:N) Adjacent Network (AV:A) [✓ Local (AV:L)] Physical (AV:P)
Attack Complexity (AC)*
[✓ Low (AC:L)] High (AC:H)
Privileges Required (PR)*
None (PR:N) Low (PR:L) [✓ High (PR:H)]
User Interaction (UI)*
None (UI:N) [✓ Required (UI:R)]

Scope (S)*
[✓ Unchanged (S:U)] Changed (S:C)

Impact Metrics
Confidentiality Impact (C)*
None (C:N) [✓ Low (C:L)] High (C:H)
Integrity Impact (I)*
None (I:N) Low (I:L) [✓ High (I:H)]
Availability Impact (A)*
None (A:N) Low (A:L) [✓ High (A:H)]



CVSS Base Score: 6.4
Impact Subscore: 5.5
Exploitability Subscore: 0.9
CVSS Temporal Score: 5.9
CVSS Environmental Score: 7.1
Modified Impact Subscore: 6.1
Overall CVSS Score: 7.1

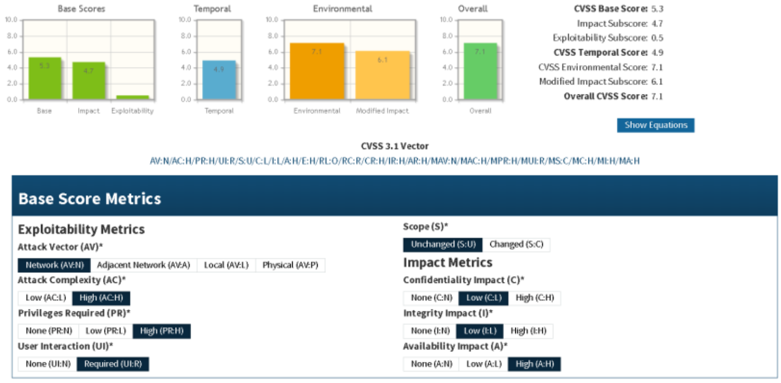
CVSS 3.1 Vector
AV:N/AC:L/PR:H/UI:R/S:U/C:L/I:H/A:H/E:H/RL:O/RC:R/CR:H/IR:H/AR:H/MAV:N/MAC:H/MPR:H/MUI:R/MS:C/MC:H/MI:H/MA:H

Base Score Metrics

Exploitability Metrics
Attack Vector (AV)*
[✓ Network (AV:N)] Adjacent Network (AV:A) Local (AV:L) Physical (AV:P)
Attack Complexity (AC)*
[✓ Low (AC:L)] High (AC:H)
Privileges Required (PR)*
None (PR:N) Low (PR:L) [✓ High (PR:H)]
User Interaction (UI)*
None (UI:N) [✓ Required (UI:R)]

Scope (S)*
[✓ Unchanged (S:U)] Changed (S:C)

Impact Metrics
Confidentiality Impact (C)*
None (C:N) [✓ Low (C:L)] High (C:H)
Integrity Impact (I)*
None (I:N) Low (I:L) [✓ High (I:H)]
Availability Impact (A)*
None (A:N) Low (A:L) [✓ High (A:H)]



CVSS Base Score: 5.3
Impact Subscore: 4.7
Exploitability Subscore: 0.5
CVSS Temporal Score: 4.9
CVSS Environmental Score: 7.1
Modified Impact Subscore: 6.1
Overall CVSS Score: 7.1

CVSS 3.1 Vector
AV:N/AC:H/PR:H/UI:R/S:U/C:L/I:L/A:H/E:H/RL:O/RC:R/CR:H/IR:H/AR:H/MAV:N/MAC:H/MPR:H/MUI:R/MS:C/MC:H/MI:H/MA:H

Base Score Metrics

Exploitability Metrics
Attack Vector (AV)*
[✓ Network (AV:N)] Adjacent Network (AV:A) Local (AV:L) Physical (AV:P)
Attack Complexity (AC)*
Low (AC:L) [✓ High (AC:H)]
Privileges Required (PR)*
None (PR:N) Low (PR:L) [✓ High (PR:H)]
User Interaction (UI)*
None (UI:N) [✓ Required (UI:R)]

Scope (S)*
[✓ Unchanged (S:U)] Changed (S:C)

Impact Metrics
Confidentiality Impact (C)*
None (C:N) [✓ Low (C:L)] High (C:H)
Integrity Impact (I)*
None (I:N) [✓ Low (I:L)] High (I:H)
Availability Impact (A)*
None (A:N) Low (A:L) [✓ High (A:H)]

COMMON VULNERABILITIES

OBJECTIVE

Common vulnerabilities are weaknesses in systems, networks, or applications that attackers can exploit to gain unauthorized access, steal data, or disrupt services. Learning these vulnerabilities through hands-on labs helps develop real-world defensive and offensive security skills.

NETWORK VULNERABILITIES: MISCONFIGURATIONS, OPEN PORTS (USE NMAP).

Common Network Issues

- Open and unused ports
- Weak or default credentials
- Outdated services
- Misconfigured firewalls
- Unnecessary running services

Identifying Network Vulnerabilities using Nmap commands are:

```
nmap target_IP
nmap -p- 1target_ip
nmap -sV target_IP
nmap -A target_IP
nmap -oN network_scan.txt target_ip
```

After that save the output.

WEB VULNERABILITIES: SQLI, XSS (PRACTICE ON OWASP JUICE SHOP)

Common Web Vulnerabilities

- SQL Injection (SQLi)
- Cross-Site Scripting (XSS)
- Broken Authentication
- Security Misconfiguration

These vulnerabilities are part of the OWASP Top 10.

PRACTICING WEB VULNERABILITIES – OWASP JUICE SHOP COMMANDS:

```
docker pull bkimminich/juice-shop
docker run -d -p 3000:3000 bkimminich/juice-shop
```

Access in browser:

```
http://localhost:3000
```

SQL Injection Example

Test login bypass:

' OR '1'='1 --

Effect:

- Bypasses authentication
- Demonstrates improper input validation

XSS Example

Inject script:

<script>alert("XSS")</script>

Effect:

- Executes JavaScript in the victim’s browser
- Can steal cookies or session tokens

USE LABS TO UNDERSTAND THE PRACTICAL APPROACH:

Metasploitable (intentionally vulnerable VM).

Metasploitable is an intentionally vulnerable Linux virtual machine designed for penetration testing practice.

Usage

- Scan using Nmap
- Exploit services using Metasploit
- Identify misconfigurations

Commands:

```
nmap -sV Metasploit_IP
```

VulnHub (vulnerable machines)

VulnHub provides downloadable vulnerable machines for practicing real-world attack scenarios.

Learning Outcomes

- Privilege escalation
- Enumeration techniques
- Exploitation chaining

Commands:

```
nmap -A <target-ip>
```

Practical Learning Workflow

- Deploy vulnerable VM (Metasploitable / VulnHub)
- Perform network scanning (Nmap)
- Identify vulnerabilities
- Exploit safely in a lab environment
- Document findings and remediation

DOCUMENTATION FUNDAMENTALS

Objective:

Create reports with tools.

IMPORTANCE OF DOCUMENTATION IN SECURITY TESTING

Documentation is a critical phase of security assessment and VAPT. Even the best technical findings lose value if they are not properly documented.

- Communicates risks to technical and non-technical stakeholders
- Serves as legal and audit evidence
- Helps track remediation and retesting
- Improves team collaboration

DRADIS COMMUNITY EDITION (CE)

Dradis CE is an open-source tool designed for collaborative security reporting. It centralizes findings from multiple tools into a single platform.

- Team collaboration
- Evidence attachment (screenshots, logs)
- Report generation (HTML, PDF)
- Integration with tools like Nmap, Metasploit

Basic Dradis Commands

```
sudo apt update
sudo apt install dradis
dradis
http://target_IP and port
```

Typical Use in VAPT

- Import Nmap and Metasploit results
- Document vulnerabilities
- Add remediation recommendations
- Generate final reports

CHERRYTREE (NOTE-TAKING FOR TECHNICAL FINDINGS).

CherryTree is a hierarchical note-taking application ideal for recording technical findings during testing.

Key Features

- Structured notes (tree format)
- Syntax highlighting
- Screenshots and file attachments
- Lightweight and offline

BASIC INSTALL AND WORK COMMANDS:

```
sudo apt install cherrytree
cherrytree
```

Typical Use

- Command outputs
- Payloads and exploits used
- Enumeration notes
- Proof-of-Concept (PoC) steps

HOW TO LEARN: USE FREE TEMPLATES FROM GITHUB.

GITHUB REPORTING TEMPLATES

Free security report templates available on GitHub help beginners understand:

- Professional formatting
- Risk presentation
- Industry terminology

COMMON TEMPLATE TYPES

- Penetration Testing Reports
- Vulnerability Assessment Reports
- Bug Bounty Reports

HOW TO PRACTICE

- Perform a lab scan (Nmap / OWASP Juice Shop)
- Take notes in CherryTree
- Import findings into Dradis
- Use a GitHub template for final report