Report on Theoretical Knowledge in Vulnerability Scanning, Penetration Testing, and Exploit Development

Introduction

In modern cybersecurity practice, proactive identification and mitigation of system weaknesses are critical. Vulnerability scanning, penetration testing, and exploit development form the foundation of offensive security and defensive risk management. This report presents a theoretical overview of these three domains, outlining core concepts, objectives, tools, methodologies, and recommended learning resources. The goal is to establish a solid conceptual understanding necessary for ethical and effective security assessments.

1. Vulnerability Scanning Techniques

1.1 Overview

Vulnerability scanning is the systematic process of identifying security weaknesses in systems, networks, and applications. It serves as a foundational step in risk management by providing visibility into potential attack vectors before adversaries exploit them.

1.2 Core Concepts

1.2.1 Scan Types

- **Network Scanning**
  Focuses on identifying open ports, services, and exposed systems.
  Example tools: Nmap (TCP/UDP scans, service detection).
- **Application Scanning**
  Targets application-layer vulnerabilities, especially in web applications.
  Example tools: Nikto for web server misconfigurations and known flaws.
- **Authenticated vs. Unauthenticated Scans**
  - *Authenticated scans* provide deeper insight by scanning from a legitimate user perspective.
  - *Unauthenticated scans* simulate an external attacker's view.

1.2.2 Vulnerability Scoring

- Vulnerabilities are assessed using the **Common Vulnerability Scoring System (CVSS v4.0)**.
- Scores range from 0.0 to 10.0 and indicate severity:
  - Low, Medium, High, and Critical
- *Example:*

        - **Apache Struts CVE-2017-5638**
            - Remote Code Execution (RCE)
            - Classified as **Critical** due to high exploitability and impact.

1.2.3 False Positives

- Automated scanners may report issues that are not exploitable in reality.
- Validation methods include:
    - Manual verification of open ports
    - Configuration reviews
    - Controlled exploitation attempts

1.3 Key Objectives

- Accurately configure scans to match the target environment.
- Validate findings to reduce false positives.
- Provide actionable risk assessments based on severity and exploitability.

1.4 Learning Resources

- **OWASP Testing Guide** – Web vulnerability scanning techniques.
- **NIST SP 800-115** – Technical guidance on information security testing.
- **WannaCry Case Study** – Mapping real-world attacks to CVSS scores and vulnerabilities.

---

2. Penetration Testing Techniques

2.1 Overview

Penetration testing (pentesting) is a controlled and authorized simulation of real-world cyberattacks to evaluate an organization's security posture. Unlike vulnerability scanning, pentesting focuses on exploitation and impact.

2.2 Core Concepts

2.2.1 Phases of Penetration Testing

1. **Reconnaissance**
    - Passive and active information gathering.
    - *Example:* OSINT using Shodan.
2. **Scanning**

- Identification of vulnerabilities and services.
- *Example:* Nessus vulnerability scans.

3. **Exploitation**
   - Actively exploiting identified weaknesses.
   - *Example:* Metasploit modules.
4. **Post-Exploitation**
   - Privilege escalation, lateral movement, persistence.
5. **Reporting**
   - Documentation of findings, impact, and remediation steps.

2.2.2 Methodologies

- **PTES (Penetration Testing Execution Standard)**
  - Provides structured guidance from pre-engagement to reporting.
- **OWASP Web Security Testing Guide (WSTG)**
  - Focuses on web application testing.
- *Example:* Using PTES to define scope and rules for a web penetration test.

2.2.3 Ethics and Authorization

- Penetration testing must always be:
  - Authorized by the client
  - Conducted within a defined scope
  - Documented with legal agreements
- Unauthorized testing is illegal and unethical.

2.3 Key Objectives

- Perform structured and repeatable penetration tests.
- Identify real-world exploit paths.
- Communicate risks effectively to stakeholders.

2.4 Learning Resources

- **PTES Documentation** – Detailed explanation of pentest phases.
- **OWASP WSTG** – Web application attack vectors.
- **SANS Case Studies** – Real-world penetration testing examples.

3. Exploit Development Basics

3.1 Overview

Exploit development involves understanding vulnerabilities at a technical level and crafting code to demonstrate or leverage those weaknesses. It is essential for advanced penetration testing and vulnerability research.

3.2 Core Concepts

3.2.1 Types of Exploits

- **Buffer Overflows**
  - Overwriting memory to gain execution control.
- **SQL Injection**
  - Manipulating database queries via unsanitized input.
- **Cross-Site Scripting (XSS)**
  - Injecting malicious scripts into web applications.
  - *Example:* Reflected XSS due to unescaped user input.

3.2.2 Exploit Writing

- Exploits are often written in languages such as Python or C.
- Public proof-of-concept (PoC) code from Exploit-DB is commonly studied.
- Testing must be done in isolated lab environments.

3.2.3 Security Mitigations

- ASLR (Address Space Layout Randomization)
- Web Application Firewalls (WAFs)
- Regular Patching
- Understanding these mitigations helps explain exploit reliability and limitations.

3.3 Key Objectives

- Safely develop and test basic exploits.
- Understand how vulnerabilities translate into real attacks.
- Learn how modern defenses reduce exploit success.

3.4 Learning Resources

- **Exploit-DB** – Real-world PoCs.

- **TCM Security Exploit Development Guides**
- **TryHackMe Buffer Overflow Room** – Hands-on practice.