Lecture 07- Network Security - Penetration Testing - Study Notes

Overview and Core Concepts

What is Penetration Testing? Penetration testing is a method for gaining assurance in IT system security by attempting to breach security using the same tools and techniques as an adversary might use. It simulates real-world attack scenarios to evaluate security posture.

The Reality Check -Penetration testing is a powerful analytical tool, but it must be:

- Properly commissioned and correctly scoped
- Integrated with routine security measures
- Used as part of a comprehensive security strategy
- NOT viewed as a primary method for identifying vulnerabilities

The Financial Audit Analogy Think of penetration testing like a financial audit:

- Your internal security team manages daily vulnerability assessments (like a finance team tracking expenditure)
- Penetration testers verify that internal processes are sufficient (like external auditors checking financial processes)
- The goal is validation, not discovery

The Ideal Scenario In an ideal situation, you should already know what penetration testers will find BEFORE they find it. This indicates:

- Good understanding of existing vulnerabilities
- Effective internal assessment processes
- Proper use of third-party verification

Purpose and Scope

Primary Purpose Penetration testing should be viewed as quality assurance for your vulnerability assessment and management processes, NOT as the primary vulnerability identification method.

What a Pen Test Should Tell You

1. Technical risk levels from software vulnerabilities
2. Hardware vulnerability exposure
3. Configuration compliance with good practices
4. Known vulnerabilities in tested components

Important: Results are only valid at the time of testing.

Scope of Assurance A well-scoped test provides confidence that:

- Products are configured according to good practice
- Security controls are properly implemented
- No common vulnerabilities exist in tested components
- No publicly known vulnerabilities are present (at test time)

Limitations and Constraints

Time-Limited Nature Critical limitation: Tests only validate security on the day of testing. Common reality includes:

- 12+ months between tests
- Vulnerabilities can exist for extended periods
- New threats emerge constantly
- Systems change and evolve

Solution: Continuous security monitoring and assessment

Appropriate Systems for Testing

SUITABLE:

- Specific operational systems
- Multi-vendor product environments
- In-house developed systems and applications
- Live production environments

NOT SUITABLE:

- Product-specific testing (use other methods)

Tester Qualifications

Essential Requirements Tests must be performed by qualified and experienced staff because:

- Tests cannot be entirely procedural
- No exhaustive test case list exists
- Quality is directly linked to tester abilities
- Expertise determines what gets discovered

CHECK Scheme Recommendation The NCSC recommends that HMG organizations use testers and companies that are part of the CHECK scheme, which provides:

- Verified qualifications
- Standardized methodologies
- Quality assurance

- Government-approved testing standards

Types of Testing

Test Basis (Information Provided)

Opaque/Black Box Testing:

- Testers have minimal prior knowledge
- Simulates external attacker perspective
- Tests what an outsider could discover
- Reveals external security posture
- Most realistic external threat simulation

Transparent/White Box Testing:

- Testers have full system knowledge
- Access to architecture, code, credentials
- Can test more thoroughly
- More efficient use of testing time
- More comprehensive coverage of potential issues

Test Types

Type 1: Bespoke Software Vulnerability Testing

- Focus: Vulnerability identification in custom or niche software
- Most commonly applied to web applications
- Must provide feedback to developers on secure coding practices and prevention strategies

Type 2: Scenario-Driven Vulnerability Testing

- Purpose: Explore specific scenarios to discover defence vulnerabilities
- Example scenarios:
  - Lost laptop with company credentials
  - Unauthorized device on internal network
  - Compromised DMZ host
  - Insider threat simulation
- Choose scenarios based on organizational risk profile and previous incidents

Type 3: Detection and Response Testing

- Enhanced scenario testing that evaluates BOTH vulnerabilities AND organizational response

- Evaluates:
  - Detection capability effectiveness
  - Response process efficiency
  - Coverage of security controls
  - Incident handling procedures

Testing Regime and Integration

Critical Principle Planned penetration tests do NOT replace normal security testing. Must continue:

- Functional testing of security controls
- Regular vulnerability assessments
- Security control validation
- Continuous monitoring

Functional Testing Requirements

Positive Tests:

- "The logon box appears when attempting to log in"
- "Authentication process initiates correctly"
- "Security controls activate as designed"

Negative Tests:

- "Cannot log in without correct password"
- "Unauthorized access attempts are blocked"
- "Invalid inputs are rejected"

Resource Allocation

NOT valuable for:

- Assessing if defined security controls are functioning
- Basic functional testing
- Routine verification tasks

Best used for:

- Complex vulnerability discovery
- Real-world attack simulation
- Validation of security posture
- Finding subtle, complex issues

Model Engagement Process

Phase 1: Initial Engagement - Team Selection

Ensure the external team has:

- Relevant qualifications (CHECK, CREST, etc.)
- Appropriate skills for your IT estate
- Experience with your system types

Highlight during bidding:

- Unusual systems (mainframes, legacy systems)
- Uncommon networking protocols
- Bespoke hardware
- Specialized requirements

Phase 2: Scoping - Critical Success Factor

Proper scoping determines:

- What systems will be tested
- What methods will be used
- What access will be provided
- What the success criteria are
- What constraints exist

Key Questions:

1. What are the test objectives?
2. Which systems are in scope?
3. What level of access is appropriate?
4. What knowledge should testers have?
5. What are the time constraints?
6. What are the business impact limits?
7. Who needs to be notified?

Remember: Poor scoping = Poor results

Phase 3: Testing - Active Assessment

During active testing:

- Testers attempt to breach system security
- Multiple tools and techniques are employed

- Various attack vectors are explored
- Findings documented in real-time
- Critical issues may be reported immediately

Your role: Monitor progress, respond to queries, maintain communication

Communication Requirements:

- Designated contact person
- Clear escalation procedures
- Rapid response to critical findings
- Regular progress updates
- Immediate notification of any issues

Phase 4: Reporting

Comprehensive report should include:

- Executive summary
- Methodology description
- Detailed findings
- Risk ratings for each issue
- Reproduction steps
- Recommended remediation
- Supporting evidence

Severity Rating Components:

- Severity level (Critical, High, Medium, Low)
- Likelihood of exploitation
- Impact if exploited
- CVSS score (where applicable)
- Business risk assessment

Rating Factors:

- Ease of exploitation
- Required attacker skill level
- Available exploits
- Potential damage
- Affected systems

- Compensating controls

Phase 5: Follow-Up and Remediation

Step 1: Internal Review Your vulnerability management group should:

- Assess the report thoroughly
- Compare findings to internal assessments
- Evaluate proposed solutions
- Consider business context
- Determine actual risk levels

Critical Principle: Risk assessment and fix decisions are YOUR responsibility

Why Testers May Rate Differently:

- Limited business context
- Incomplete system knowledge
- Standard methodology constraints
- Different risk perspectives

Previously Unknown Vulnerabilities - Special Attention When testers find something you didn't know about, ask:

1. Why didn't we find this?
2. What process gaps exist?
3. How can we spot similar issues?
4. What do we need to change?

Goal: Improve internal processes to catch these in future

Choosing Solutions

Key Principle: Proposed solutions are not always the only solutions

Beyond Simple Patching:

1. Remove functionality - eliminate unused features
2. Network segmentation - isolate vulnerable systems
3. Enhanced monitoring - detect exploitation attempts
4. Access controls - limit who can reach the vulnerability
5. WAF rules - block exploitation patterns

Example Alternatives:

- Tester suggests: "Patch this software"

- Consider: Uninstall if not required, implement additional controls, increase monitoring, accept the risk with justification, compensating controls

Consult: Your technical staff and suppliers

Mitigation as Business Process

Key Understanding Vulnerability risk assessment and mitigation is a BUSINESS PROCESS

Should NOT be:

- Wholly outsourced to test team
- Treated as purely technical
- Isolated from business context
- Decided without stakeholder input

Should include: Business owners, technical staff, risk managers, compliance team

Integration and Continuous Improvement

Penetration Testing in Context Must integrate with:

- Continuous vulnerability scanning
- Security monitoring
- Incident response
- Patch management
- Security awareness training
- Threat intelligence

Continuous Improvement Cycle Use pen test results to:

1. Validate internal processes
2. Identify process gaps
3. Improve detection capabilities
4. Enhance security controls
5. Train security staff
6. Update security procedures

Each test should make you stronger.

Common Pitfalls and Best Practices

Don't:

- Treat pen testing as one-time compliance checkbox
- Ignore findings that contradict your assumptions

- Automatically accept tester risk ratings
- Delay remediation indefinitely
- Forget about findings after initial review
- Use pen testing as substitute for regular security

Best Practices for Success:

1. Use qualified testers (CHECK scheme recommended)
2. Invest time in proper scoping
3. Maintain internal vulnerability processes
4. Communicate clearly throughout
5. Assess findings in business context
6. Use results to improve processes
7. Schedule tests appropriately
8. Plan remediation before testing

The Penetration Testing Methodology

1. Reconnaissance Gathering information about the target organization

2. Scanning & Enumeration Identifying live hosts, ports, and services

3. Vulnerability Assessment Detecting weaknesses in systems and applications

4. Exploitation Attempting to compromise identified vulnerabilities

5. Post-Exploitation Maintaining access and expanding control

6. Reporting Documenting findings and remediation recommendations

Penetration Testing Tools

Introduction to Tools

Key Points:

- Tools automate and enhance testing efficiency across different attack phases
- Categories include reconnaissance, scanning, exploitation, and post-exploitation
- Proper tool selection depends on scope, target, and testing objectives
- **Always obtain written authorization before conducting any penetration test**

Reconnaissance Tools

Passive Information Gathering

Definition: Collecting data without directly interacting with the target system

Goal: Remain undetected while learning about the target

Techniques:

- WHOIS lookups
- DNS queries
- Public website analysis
- Social media profiling
- Google dorking

Advantages:

- Low risk of detection
- Useful for early-stage reconnaissance

Limitations:

- May not reveal internal or real-time system details

Common Tools:

- **Google Dorking:** Advanced search operators to find exposed information
- **Shodan:** Search engine for internet-connected devices and services
- **theHarvester:** Collects emails, subdomains, IPs from public sources
- **Maltego:** Visual link analysis and data mining platform
- **WHOIS/DNS tools:** Domain registration and DNS record enumeration

Active Information Gathering

Definition: Directly interacting with the target system to extract information

Goal: Obtain detailed, often technical data about the system

Techniques:

- Port scanning (e.g., Nmap)
- Banner grabbing
- Vulnerability scanning
- Network sniffing

Advantages:

- Provides deeper insights into system configuration and vulnerabilities

Limitations:

- Higher risk of detection
- May trigger security alerts or countermeasures

Common Tools:

- **Nmap:** Network discovery and port scanning
- **DNSRecon:** Active DNS enumeration including zone transfers
- **Sublist3r:** Subdomain enumeration using multiple search engines
- **Fierce:** DNS reconnaissance and subdomain brute-forcing

Network Scanning with Nmap

Features:

- Industry-standard network scanner for host and service discovery
- Supports multiple scan types: SYN, TCP connect, UDP, ACK, and more
- Service version detection and OS fingerprinting capabilities
- NSE (Nmap Scripting Engine) extends functionality with custom scripts

Essential syntax: nmap -sV -sC -oA output target

Vulnerability Scanning Tools

- **Nessus:** Commercial vulnerability scanner with extensive plugin library
- **OpenVAS:** Open-source vulnerability assessment system
- **Nikto:** Web server scanner identifying common vulnerabilities
- **OWASP ZAP:** Web application security scanner and proxy

Exploitation Frameworks

Metasploit Framework

Features:

- Comprehensive exploitation and post-exploitation platform
- Contains hundreds of exploit modules for various vulnerabilities
- Payload generation, encoding, and delivery mechanisms
- Integrates with scanning tools for automated exploitation

Caution: Only use against authorized targets with proper scope

Web Application Testing Tools

- **Burp Suite:** Intercepting proxy for manual web app testing
- **SQLmap:** Automated SQL injection detection and exploitation
- **XSStrike:** Cross-site scripting vulnerability scanner
- **Gobuster/Dirb:** Directory and file brute-forcing tools
- **WPScan:** WordPress-specific vulnerability scanner

Wireless Network Testing Tools

- **Aircrack-ng suite:** Wireless packet capture and WEP/WPA cracking
- **Kismet:** Wireless network detector and packet sniffer
- **Reaver:** WPS (Wi-Fi Protected Setup) brute-force tool
- **Wifite:** Automated wireless attack tool

Requirements: Compatible wireless adapters supporting monitor mode

Password Cracking Tools

- **John the Ripper:** Fast password hash cracking with multiple attack modes
- **Hashcat:** GPU-accelerated password recovery supporting many hash types
- **Hydra:** Network login cracker for various protocols (SSH, FTP, HTTP, etc.)
- **CeWL:** Custom wordlist generator from website content

Attack Strategies: Dictionary, brute-force, and rule-based attacks

Post-Exploitation Tools

- **Mimikatz:** Windows credential extraction from memory
- **BloodHound:** Active Directory attack path mapping
- **PowerSploit:** PowerShell-based post-exploitation framework
- **Empire/Starkiller:** Post-exploitation agent and C2 framework

Social Engineering Tools

- **Social Engineering Toolkit (SET):** Framework for social engineering attacks
- **GoPhish:** Phishing campaign management and tracking
- **King Phisher:** Phishing campaign toolkit with analytics

Features: Email templates, credential harvesting, and reporting

Network Traffic Analysis Tools

- **Wireshark:** Industry-standard packet capture and analysis tool
- **tcpdump:** Command-line packet analyzer for quick captures
- **Bro/Zeek:** Network security monitoring framework
- **NetworkMiner:** Network forensics analysis tool

Automation and Scripting

- **Python:** Dominant language for security tool development
- **Bash scripting:** Automation of command-line tools and workflows
- **PowerShell:** Windows automation and Active Directory enumeration
- **Ruby:** Language behind Metasploit modules

Integrated Penetration Testing Distributions

- **Kali Linux:** Most popular distribution with 600+ pre-installed tools
- **Parrot Security OS:** Alternative with focus on privacy and development
- **BlackArch:** Arch-based distribution with 2800+ tools

Benefits:

- Pre-configured environments reduce setup time and tool conflicts
- Virtual machines and containers provide isolated testing environments

Cloud Security Testing Tools

- **ScoutSuite:** Multi-cloud security auditing tool (AWS, Azure, GCP)
- **Prowler:** AWS security assessment and compliance tool
- **Cloud Custodian:** Cloud security, compliance, and governance
- **Pacu:** AWS exploitation framework for penetration testers

Reporting and Documentation Tools

- **Dradis Framework:** Centralized reporting and collaboration platform
- **Faraday:** Collaborative penetration testing IDE
- **KeepNote/CherryTree:** Note-taking applications for pentesters
- **Markdown/LaTeX:** Professional report formatting

Legal and Ethical Considerations

Critical Requirements:

- Written authorization (scope, targets, timeframes) is mandatory before testing
- Respect rules of engagement and escalation procedures
- Bug bounty programs provide legal testing opportunities
- Data protection: handle discovered sensitive information responsibly

Tool Selection and Best Practices

Best Practices:

- Match tools to engagement objectives and scope limitations
- Understand tool capabilities and limitations—avoid blind reliance
- Verify automated findings manually to reduce false positives
- Document tool versions and commands for reproducibility
- Maintain updated toolkits addressing latest vulnerabilities
- Consider operational security and tool artifacts left on target systems

Emerging Trends and Future Directions

- AI/ML integration for intelligent vulnerability analysis
- Container and Kubernetes security testing tools
- IoT and embedded device penetration testing frameworks
- Purple teaming tools bridging offensive and defensive operations
- Continuous security validation and breach-and-attack simulation
- Emphasis on developer security training and secure coding practices