Unit 1 – Vulnerability Assessment and Penetration Testing (VAPT)

1. Introduction to Ethics of Ethical Hacking

What is Ethical Hacking?

Ethical hacking is the **legal practice of testing computer systems, networks, or applications** to find security weaknesses before malicious hackers exploit them.

An ethical hacker:

- Works with **permission**
- Follows **legal boundaries**
- Reports vulnerabilities responsibly

Types of Hackers

Type	Description
White Hat	Ethical hacker (security professional)
Black Hat	Malicious hacker (criminal)
Gray Hat	Between white & black (may break rules without intent to harm)

---

2. Why You Need to Understand Your Enemy's Tactics

To defend a system, you must think like an attacker.

Common Attacker Tactics

1. **Reconnaissance** – Collecting information (IP, domain, emails)
2. **Scanning** – Finding open ports & services
3. **Gaining Access** – Exploiting vulnerabilities
4. **Maintaining Access** – Installing backdoors
5. **Covering Tracks** – Deleting logs

Why it matters:

- Helps predict attack paths
- Improves system design

- Enables proactive security

---

3. Recognizing the Gray Areas in Security

Gray areas are actions that are:

- Technically illegal
- But done for learning or awareness

Examples:

- Testing a website without permission
- Scanning a network that isn't yours
- Using leaked tools for practice

⚠ Even if intention is good, **without authorization it is illegal**.

---

4. Vulnerability Assessment (VA)

Definition:

Vulnerability Assessment is the process of **identifying, analyzing, and prioritizing security weaknesses** in a system.

Goals:

- Find known vulnerabilities
- Rank them by severity
- Suggest fixes

Types of VA:

1. **Network-based VA**
2. **Host-based VA**
3. **Application VA**
4. **Wireless VA**

VA Process:

1. Asset identification
2. Vulnerability scanning
3. Risk analysis
4. Reporting
5. Remediation

Tools:

- Nessus
- OpenVAS
- Qualys
- Nexpose

---

5. Penetration Testing (PT)

Definition:

Penetration Testing is a **simulated cyber attack** to actually exploit vulnerabilities and test real-world security.

Difference: VA vs PT

Vulnerability Assessment	**Penetration Testing**
Finds vulnerabilities	Exploits them
Theoretical	Practical
Automated	Manual + automated
Less risky	Higher risk

---

6. Penetration Testing Methodology

1. **Planning & Scope**
2. **Information Gathering**
3. **Threat Modeling**

4. **Exploitation**
5. **Post-Exploitation**
6. **Reporting**

---

7. Social Engineering Attacks

Definition:

Social Engineering is the **psychological manipulation of people** to make them reveal sensitive information.

It attacks the **human**, not the system.

---

8. How a Social Engineering Attack Works

Attack Flow:

1. Research victim
2. Build trust
3. Create urgency/fear
4. Extract information
5. Exploit access

---

9. Conducting a Social Engineering Attack (Steps)

1. **Information Gathering**
    - LinkedIn, Facebook, company website
2. **Pretexting**
    - Creating fake identity (IT staff, bank officer)
3. **Interaction**
    - Email, phone, in-person
4. **Exploitation**

- Password, OTP, USB drop

---

10. Common Social Engineering Attacks

a) Phishing

Fake emails asking for credentials.

b) Spear Phishing

Targeted phishing to specific person.

c) Vishing

Voice phishing (phone calls).

d) Smishing

SMS-based phishing.

e) Baiting

Infected USB drives left in office.

f) Tailgating

Following someone into secure area.

---

11. Preparing for Face-to-Face Attacks

Face-to-face attacks involve **physical interaction**.

Examples:

- Pretending to be technician
- Asking for ID card
- Shoulder surfing

Attacker Techniques:

- Professional dress
- Fake ID cards
- Confident behavior

---

12. Defending Against Social Engineering Attacks

Technical Controls:

- Multi-factor authentication (MFA)
- Email filters
- Intrusion detection systems

Human Controls:

- Employee training
- Security awareness programs
- Verification policies

Best Practices:

- Never share OTP
- Verify identity
- Lock systems
- Report suspicious behavior

---

13. Importance of VAPT in Organizations

Benefit
Prevents data breaches
Saves financial loss
Improves compliance
Builds customer trust
Enhances cyber readiness

---

14. Ethical Hacking Legal Framework (India Example)

In India:

- Governed by **IT Act 2000**
- Sections: 43, 66
- Unauthorized access is punishable

Always get:

- Written permission
- Scope document
- NDA agreement

One-Line Exam Definitions

- **Ethical Hacking:** Authorized security testing of systems.
- **Vulnerability Assessment:** Process of identifying security weaknesses.
- **Penetration Testing:** Simulated real-world cyber attack.
- **Social Engineering:** Psychological manipulation to steal data.
- **Phishing:** Fraudulent emails for credential theft.
- **Gray Hat:** Hacker operating between legal and illegal.

Short Conclusion (For Answer Writing)

Vulnerability Assessment and Penetration Testing (VAPT) form the backbone of modern cybersecurity. While technical tools secure systems, social engineering highlights the weakest link – humans. Ethical hacking ensures organizations stay ahead of cyber threats by proactively identifying and fixing vulnerabilities before attackers exploit them.

Unit – 2: Physical Penetration Attacks & Insider Attacks

1. Physical Penetration Attacks

What is Physical Penetration?

Physical penetration refers to **gaining unauthorized physical access** to a building, server room, or secure area in order to steal data, install malware, or sabotage systems.

Unlike cyber attacks, these attacks target **physical security controls**.

---

2. Need of Physical Penetration Testing

Organizations focus mostly on cyber security, but **physical security is equally important** because:

Why it is needed:

1. To test **real-world security**
2. To identify **human weaknesses**
3. To check **entry control systems**
4. To prevent **data theft and sabotage**
5. To ensure compliance (ISO 27001, SOC2)

Example:

If an attacker can enter a server room, they can:

- Plug a USB malware
- Reset BIOS passwords
- Steal hard disks

---

3. Conducting a Physical Penetration Test

Step-by-Step Process:

1. Planning & Authorization

- Written permission from management
- Define scope (which building/areas)

2. Reconnaissance

- Study office timings
- Observe guards

- Identify camera positions

3. Entry Attempts

- Tailgating
- Fake ID
- Pretending to be vendor

4. Exploitation

- Plug malicious USB
- Photograph documents
- Access unlocked systems

5. Exit & Reporting

- Document all weaknesses
- Suggest improvements

---

4. Common Ways into a Building

a) Tailgating

Following an employee into secure area.

b) Piggybacking

Convincing someone to hold door.

c) Fake Identity

Pretending to be:

- IT staff
- Electrician
- Courier

d) Unlocked Doors

Using emergency exits.

e) Dumpster Diving

Finding sensitive info in trash.

f) Badge Cloning

Copying RFID cards.

---

5. Defending Against Physical Penetration

Physical Controls:

- Biometric access
- Turnstiles
- Mantraps
- CCTV cameras

Administrative Controls:

- Visitor policies
- Security training
- Access logs

Human Controls:

- Do not allow tailgating
- Verify identity
- Report strangers

Insider Attacks

6. What is an Insider Attack?

An insider attack is a security breach caused by **someone within the organization** (employee, contractor, intern).

These are **most dangerous** because insiders already have access.

---

7. Types of Insider Threats

Type	Description
Malicious	Intentionally causes harm
Negligent	Makes mistakes
Compromised	Account hacked

---

8. Conducting an Insider Attack

Common Insider Attack Methods:

a) Data Theft

Copying files to USB or cloud.

b) Privilege Abuse

Using admin rights for misuse.

c) Credential Sharing

Giving passwords to outsiders.

d) Planting Malware

Installing keyloggers or backdoors.

e) Sabotage

Deleting files or altering data.

---

9. Insider Attack Lifecycle

1. Access granted
2. Abuse of trust
3. Data collection
4. Exfiltration
5. Cover tracks

---

10. Defending Against Insider Attacks

Technical Controls:

- Least privilege principle
- User activity monitoring
- Data Loss Prevention (DLP)
- Log auditing
- Endpoint security

Administrative Controls:

- Background checks
- Separation of duties
- Regular audits

Human Controls:

- Employee awareness
- Strong exit procedures
- Whistleblower system

---

11. Real World Examples

Edward Snowden

Leaked NSA documents (insider threat).

Tesla Employee Case

Stole source code (2018).

---

12. Comparison: Physical vs Insider Attacks

Feature	Physical	Insider
Attacker	Outsider	Employee

Feature	Physical	Insider
Access	Gained	Already exists
Risk	High	Very high
Detection	Easier	Harder

One-Line Exam Definitions

- **Physical Penetration:** Unauthorized physical access testing.
- **Insider Attack:** Security breach by trusted person.
- **Tailgating:** Following someone into restricted area.
- **Piggybacking:** Convincing entry.
- **DLP:** Data Loss Prevention system.

Short Conclusion (Exam Use)

Physical and insider attacks highlight that cybersecurity is not just about firewalls and antivirus. Human behavior and physical access are critical components of organizational security. Effective defense requires a combination of technology, policies, and employee awareness.

Unit – 3: Metasploit & Managing a Penetration Test

Part A: Metasploit Framework

1. Metasploit – The Big Picture

What is Metasploit?

Metasploit is an **open-source penetration testing framework** used to:

- Find vulnerabilities
- Exploit systems

- Validate security flaws
- Perform real-world attack simulations

It is used by:

- Ethical hackers
- Security analysts
- Red teams
- Cybersecurity researchers

Why Metasploit is important:

- Contains **1000+ exploits**
- Supports **Windows, Linux, Android, web apps**
- Industry standard tool (used in CEH, OSCP)

---

2. Getting Metasploit

Installation Options:

- Kali Linux (pre-installed)
- Ubuntu
- Windows
- macOS

Components of Metasploit:

Component	Purpose
Exploit	Takes advantage of vulnerability
Payload	Code executed after exploit
Auxiliary	Scanning, fuzzing
Encoder	Obfuscates payload
NOP	Padding

Component	Purpose
Post	Post-exploitation modules

---

3. Using Metasploit Console (msfconsole)

msfconsole is the main command-line interface.

Basic Commands:

- search – find exploits
- use – select exploit
- set – set target info
- run / exploit – launch attack
- sessions – manage connections

Typical Flow:

1. search vulnerability
2. use exploit
3. set RHOST
4. set payload
5. exploit

---

4. Exploiting Client-Side Vulnerabilities

Client-side attacks target:

- Browsers
- PDF readers
- Email clients
- Media players

Examples:

- Malicious PDF file

- Fake website
- Trojan download

Attack Flow:

1. Create malicious file
2. Send to victim
3. Victim opens file
4. Exploit triggers
5. Attacker gains access

These attacks are very effective because they **exploit human behavior**.

---

5. Meterpreter (Heart of Metasploit)

What is Meterpreter?

Meterpreter is an **advanced in-memory payload** that gives full control over victim system.

Features:

- File system access
- Screenshot
- Keylogger
- Webcam access
- Password dump
- Privilege escalation

Why powerful:

- Runs in memory (hard to detect)
- No files written to disk
- Bypasses antivirus

---

6. Penetration Testing with Meterpreter

After exploit success:

You can:

- Browse files
- Upload malware
- Capture keystrokes
- Extract credentials
- Pivot to other machines

This phase is called **Post-Exploitation**.

---

7. Automating and Scripting Metasploit

Metasploit supports automation using:

- Ruby scripts
- Resource scripts (.rc files)

Why automation is needed:

- Large networks
- Repeated testing
- Faster assessments

Benefits:

- Saves time
- Reduces human error
- Useful for red teams

---

8. Going Further with Metasploit

Advanced uses:

- Exploit development
- Red team operations

- IDS evasion
- Payload customization
- Integration with Nmap, Nessus

Metasploit can be used with:

- Burp Suite
- Wireshark
- Nmap
- Social engineering tools

Part B: Managing a Penetration Test

This section focuses on **professional and legal execution of pentesting**.

---

9. Planning a Penetration Test

Key elements:

- Define scope
- Identify systems
- Set objectives
- Choose testing type

Types of Testing:

Type	Knowledge given
Black Box	No info
White Box	Full info
Gray Box	Partial info

---

10. Structuring a Penetration Testing Agreement

This is a **legal document**.

Must include:

- Authorization letter
- Scope of testing
- Time window
- Allowed techniques
- Liability clauses
- NDA

Without agreement → **illegal hacking**

---

11. Execution of a Penetration Test

Phases:

1. Reconnaissance
2. Scanning
3. Exploitation
4. Post-exploitation
5. Cleanup

This is where Metasploit is mainly used.

---

12. Information Sharing During a Penetration Test

Why important:

- Avoid system crashes
- Inform critical findings
- Coordinate with IT team

Rules:

- Do not hide findings

- Report high-risk issues immediately
- Maintain confidentiality

---

13. Reporting the Results

The most important deliverable.

Report includes:

- Executive summary
- Methodology
- Vulnerabilities found
- Risk levels
- Proof of concept
- Recommendations

Types of audience:

Audience	**Needs**
Management	Business impact
Technical team	Fix details

Real-World Importance

Companies like:

- Google
- Amazon
- Microsoft
- Banks
- Defense agencies

All use **Metasploit-based testing**.

Comparison Table

Feature	Metasploit
Purpose	Exploitation
Type	Open source
Users	Ethical hackers
Difficulty	Medium to advanced
Industry relevance	Very high

One-Line Exam Definitions

- **Metasploit:** A penetration testing framework.
- **Exploit:** Code that abuses vulnerability.
- **Payload:** Code executed on target.
- **Meterpreter:** Advanced control shell.
- **Black box testing:** No prior knowledge.
- **Penetration report:** Final security document.

Short Conclusion (Perfect for Exams)

Metasploit is a powerful tool that allows ethical hackers to simulate real-world cyber attacks and validate system security. However, effective penetration testing is not only technical but also procedural, requiring proper planning, legal authorization, execution, communication, and professional reporting.

Unit – 4: Linux & Windows Exploits

PART A: Basic Linux Exploits

1. Stack Operations

What is a Stack?

The **stack** is a special memory area used to store:

- Function parameters
- Local variables
- Return addresses

It follows **LIFO** (Last In, First Out).

Stack Structure:

Component
Function arguments
Return address
Base pointer
Local variables

---

2. Buffer Overflows

Definition:

A buffer overflow occurs when **more data is written into a memory buffer than it can hold**, overwriting adjacent memory.

Why dangerous?

It allows:

- Code execution
- Program crash
- Privilege escalation

---

3. How Buffer Overflow Works

Example:

char name[10];

gets(name); // unsafe

If user enters 50 characters → memory corruption.

---

4. Local Buffer Overflow Exploits

Local vs Remote:

Type	Description
Local	Attacker has system access
Remote	Exploited over network

Local attack scenario:

- Exploit vulnerable SUID program
- Gain root shell

---

5. Exploit Development Process (Linux)

Steps:

1. Find vulnerable program
2. Crash it (fuzzing)
3. Identify offset
4. Inject shellcode
5. Redirect execution
6. Get shell

---

6. Linux Memory Layout

Segment	Purpose
Stack	Local variables
Heap	Dynamic memory
BSS	Uninitialized
Data	Initialized
Text	Program code

PART B: Windows Exploits

---

7. Compiling & Debugging Windows Programs

Tools:

- MinGW / Visual Studio
- Immunity Debugger
- OllyDbg
- WinDbg

Why debugging?

To:

- Observe crashes
- Inspect registers
- Track memory

---

8. Writing Windows Exploits

Basic process:

1. Crash program
2. Find EIP offset

3. Control execution
4. Inject payload
5. Bypass protections

---

9. Structured Exception Handling (SEH)

What is SEH?

SEH is Windows' mechanism to **handle runtime errors**.

It stores:

- Exception handler address
- Pointer to next handler

SEH exploitation:

Attacker overwrites handler pointer and redirects execution.

---

10. Windows Memory Protections

Modern Windows uses **defensive mechanisms**.

a) DEP (Data Execution Prevention)

Prevents execution in data memory.

b) ASLR (Address Space Layout Randomization)

Randomizes memory addresses.

c) SEHOP

Protects SEH chain.

d) Stack Canaries

Detects stack corruption.

---

11. Windows Versions Protection

Version	Protection
XP SP3	DEP, basic ASLR
Vista	Strong ASLR
Windows 7	DEP + ASLR
Server 2008	Full protection

---

12. Bypassing Windows Memory Protections

Common techniques:

a) ROP (Return Oriented Programming)

Uses existing code snippets.

b) Heap spraying

Fills memory with shellcode.

c) SEH overwrite

Redirect handler.

d) Egghunting

Find payload in memory.

---

13. Comparison: Linux vs Windows Exploits

Feature	Linux	Windows
Complexity	Medium	High
Protections	Less	More
Tools	GDB	Immunity
Exploit style	Stack-based	ROP-based

---

14. Real-World Importance

These vulnerabilities lead to:

- Rootkits
- Ransomware
- Zero-day exploits

Famous attacks:

- WannaCry (Windows exploit)
- Dirty COW (Linux exploit)

One-Line Exam Definitions

- **Stack:** LIFO memory region
- **Buffer overflow:** Memory overwrite vulnerability
- **Shellcode:** Malicious machine code
- **SEH:** Windows error handler
- **DEP:** Blocks execution in data
- **ASLR:** Randomizes memory layout
- **ROP:** Code reuse attack

Short Conclusion (Perfect for Exams)

Linux and Windows exploits demonstrate how poor memory handling can lead to full system compromise. While Linux exploits mainly rely on stack-based vulnerabilities, Windows systems require advanced techniques to bypass modern memory protections such as DEP and ASLR. Understanding exploit development is essential for building secure software and effective defenses.

Unit – 5: Web Application Security & Vulnerability Analysis

PART A: Web Application Security Vulnerabilities

1. Overview of Top Web Application Security Vulnerabilities

Most web attacks are classified by **OWASP (Open Web Application Security Project)**.

OWASP Top 10 (Simplified List)

1. Injection
2. Broken Authentication
3. Sensitive Data Exposure
4. XML External Entities (XXE)
5. Broken Access Control
6. Security Misconfiguration
7. Cross-Site Scripting (XSS)
8. Insecure Deserialization
9. Using Components with Known Vulnerabilities
10. Insufficient Logging & Monitoring

These represent the **most critical web risks globally**.

---

2. Injection Vulnerabilities

Definition:

Injection occurs when **untrusted input is sent to an interpreter** (SQL, OS, LDAP) and executed as code.

Types:

- SQL Injection
- Command Injection
- LDAP Injection
- XML Injection

Example:

SELECT * FROM users WHERE id='1 OR 1=1';

This returns all users.

---

3. SQL Injection Vulnerabilities

What is SQL Injection?

SQL Injection allows attackers to **manipulate database queries**.

Impact:

- View private data
- Modify records
- Delete tables
- Bypass login

Types:

Type	Description
In-band	Same channel
Blind	No direct output
Out-of-band	Using DNS/HTTP

---

4. Cross-Site Scripting (XSS)

Definition:

XSS allows attackers to **inject malicious JavaScript** into web pages viewed by others.

Types:

Type	Description
Stored	Saved in database
Reflected	From URL

Type	Description
DOM-based	Client-side

Example:

<script>alert('Hacked')</script>

---

5. Impact of XSS

- Session hijacking
- Cookie theft
- Defacing website
- Phishing

---

6. The Rest of OWASP Top Ten (Important Ones)

Broken Authentication

Weak passwords, session flaws.

Sensitive Data Exposure

Unencrypted passwords.

Security Misconfiguration

Default credentials, open admin panels.

Broken Access Control

Users access admin pages.

Insecure Deserialization

Malicious objects executed.

PART B: Vulnerability Analysis

---

7. What is Vulnerability Analysis?

Vulnerability analysis is the process of **identifying weaknesses in applications before exploitation**.

---

8. Passive Analysis

Passive analysis means **no interaction with live system**.

Advantages:

- Safe
- Legal
- No system damage

---

9. Source Code Analysis

What is it?

Reviewing program source code for flaws.

Finds:

- Hardcoded passwords
- Input validation issues
- Logic errors

Tools:

- SonarQube
- Checkmarx
- Fortify

---

10. Binary Analysis

What is it?

Analyzing **compiled programs** without source code.

Techniques:

- Static analysis
- Disassembly
- Reverse engineering

Tools:

- IDA Pro
- Ghidra
- Radare2

---

11. Comparison: Source vs Binary Analysis

Feature	Source Code	Binary
Access	Full	Limited
Difficulty	Low	High
Accuracy	High	Medium
Use case	Developers	Security teams

---

12. Real-World Attacks

Attack	Vulnerability
Facebook data leak	XSS
Sony hack	SQL injection
British Airways breach	Web misconfig

One-Line Exam Definitions

- **OWASP:** Global web security organization
- **Injection:** Executing attacker input
- **SQLi:** Database manipulation

- **XSS:** JavaScript injection
- **Passive analysis:** No system interaction
- **Binary analysis:** Reverse engineering

Short Conclusion (Exam Ready)

Web applications are the primary target of cyber attacks today. Vulnerabilities like SQL injection and XSS allow attackers to compromise entire systems. Passive vulnerability analysis through source code and binary inspection helps identify flaws early and prevents costly breaches.

Client-Side Browser Exploits & Malware Analysis

PART A: Client-Side Browser Exploits

1. Why Client-Side Vulnerabilities are Interesting

Definition:

Client-side vulnerabilities exist in **user applications** like:

- Web browsers
- PDF readers
- Media players
- Email clients

Instead of attacking servers, attackers **attack users directly**.

Why attackers prefer them:

1. Easier than server hacking
2. Users are less protected
3. Bypasses firewalls
4. Targets humans, not machines
5. Works through social engineering

Example:

User clicks malicious link → browser exploited → malware installed.

---

2. Internet Explorer Security Concepts

Internet Explorer (IE) was heavily targeted due to:

Key Security Features:

Feature	Purpose
Protected Mode	Runs in low privilege
Security Zones	Different trust levels
ActiveX controls	Runs external code
DEP	Prevents execution
ASLR	Random memory

IE Security Zones:

1. Internet (untrusted)
2. Local Intranet
3. Trusted sites
4. Restricted sites

Each zone has different permissions.

---

3. History of Client-Side Exploits & Latest Trends

Early Exploits:

- Buffer overflows in IE6
- Flash player bugs
- Java vulnerabilities

Modern Exploits:

- Zero-day vulnerabilities
- Drive-by downloads

- Malvertising
- Browser sandbox escapes

Latest Trends:

- Fileless malware
- Exploit kits (Angler, RIG)
- Chrome/Edge sandbox bypass
- Supply chain attacks

---

4. Finding New Browser-Based Vulnerabilities

This is called **vulnerability research**.

Methods:

1. Fuzzing (sending random input)
2. Source code review
3. Reverse engineering
4. Bug bounty programs

Tools:

- Burp Suite
- AFL fuzzer
- Ghidra
- IDA Pro

---

5. Heap Spray to Exploit

What is Heap Spray?

Heap spraying fills memory with **malicious payload repeatedly** to increase success rate.

Why used?

Because modern systems use:

- ASLR (random memory)
  Heap spray makes payload **land somewhere predictable**.

Attack flow:

1. Load JavaScript
2. Allocate memory repeatedly
3. Insert shellcode
4. Trigger vulnerability

---

6. Protecting Yourself from Client-Side Exploits

Technical Protection:

- Keep browser updated
- Disable unnecessary plugins
- Use sandbox mode
- Enable DEP & ASLR
- Use ad blockers

Human Protection:

- Don’t click unknown links
- Avoid pirated software
- Use secure browsers
- Awareness training

PART B: Malware Analysis

---

7. What is Malware?

Malware is **malicious software designed to harm, spy, or control systems**.

Types:

Type	Purpose
Virus	Infects files
Worm	Spreads automatically
Trojan	Fake legitimate software
Ransomware	Encrypts data
Spyware	Steals data
Rootkit	Hides presence
Botnet	Controlled remotely

---

8. Collecting Malware & Initial Analysis

Sources:

- Honeypots
- Spam emails
- Infected websites
- Threat intelligence feeds

Goals:

- Understand behavior
- Identify attack method
- Develop detection rules

---

9. Honeynet Technology (Latest Trends)

What is Honeynet?

A honeynet is a **network of vulnerable systems designed to attract attackers**.

Purpose:

- Trap malware

- Study attacks
- Collect samples
- Improve defenses

Latest Trends:

- Cloud honeynets
- AI-based honeypots
- IoT honeypots
- Automated malware capture

---

10. Catching Malware – Setting the Trap

Methods:

1. Fake services
2. Open ports
3. Vulnerable websites
4. Email traps

When attacker hits → malware captured.

---

11. Initial Analysis of Malware

This is also called **triage analysis**.

Static Analysis:

Without running malware:

- File hash
- Strings analysis
- PE header inspection

Dynamic Analysis:

Running in sandbox:

- Process behavior
- Network connections
- File changes
- Registry edits

Tools:

- VirusTotal
- Cuckoo Sandbox
- Procmon
- Wireshark
- Hybrid Analysis

Comparison Table

Topic	Client Exploits	Malware
Target	Browser	System
Method	Exploit bug	Malicious code
Goal	Initial access	Persistence
Defense	Patching	Antivirus/EDR

One-Line Exam Definitions

- **Client-side exploit:** Attack on user software
- **Heap spray:** Memory flooding technique
- **Honeynet:** Trap network for attackers
- **Malware:** Harmful software
- **Static analysis:** No execution
- **Dynamic analysis:** Execute in sandbox

Short Conclusion (Exam Perfect)

Client-side browser exploits demonstrate how attackers compromise systems by exploiting software used daily by users. Malware analysis allows security professionals to study malicious code in controlled environments to understand behavior and improve defense mechanisms. Together, they form the backbone of modern cyber threat research and incident response.