Below is a comprehensive, sentence-by-sentence analysis of **“Malware Exploitation Techniques Notes”** formatted for **direct pasting into Microsoft Word** — clean, minimal spacing, numbered headings, and bullet alignment. All essential concepts are explained thoroughly and tied back to **CompTIA A+ 220-1102 Objective 2.4**.

**🛠️ Malware Exploitation Techniques – Study Notes - CompTIA A+ 220-1102 | Domain 2.0 – Security | Objective 2.4**

**1. Definition of Malware Exploitation Techniques**

* Malware exploitation techniques are **methods malware uses to infiltrate and infect systems**.
* Early malware often modified **executable files** or inserted malicious **macros** into documents.
* This ensured the malware **activated upon file execution or opening**, launching its payload.

**2. Memory-Based Infection & Worm Behavior**

* Some malware, like **worms**, target **system memory** rather than storage.
* Worms often use **Remote Procedure Calls (RPCs)** over a network to spread across systems.
* These memory-resident infections allow malware to propagate **without user interaction**.

**🔁 What Are Remote Procedure Calls (RPCs) for Worms?**

**Definition:** A **Remote Procedure Call (RPC)** is a **protocol** that allows a program on one computer to **request a service or execute a function on another computer** over a network.

**💡 Why Do Worms Use RPCs?**

* Worms are designed to **spread automatically** across networks **without user input**.
* They use RPCs to:
  + **Scan and identify vulnerabilities** on remote systems.
  + **Remotely execute malicious code** (like downloading or running a payload).
  + **Propagate themselves** from one system to another—quickly and silently.

**🧬 How It Works in an Attack**

1. The worm infects an initial system.
2. It uses **RPC protocols** to locate other systems on the same network.
3. It remotely exploits unpatched vulnerabilities (e.g., Windows RPC flaw).
4. Once successful, it **installs itself on the new host** and repeats the process.

**3. Modern Malware Evolution – Fileless Malware**

* Modern threats increasingly use **fileless techniques** to avoid detection.
* **Fileless malware**:
  + Runs scripts or **shellcode** directly in **memory**.
  + Bypasses traditional **signature-based detection** (antivirus/anti-malware).
  + Does not require presence on the **local file system**.
* These threats leave behind **minimal artifacts** or **Indicators of Compromise (IoCs)**.
* Some may write temporary data but **self-delete it** after execution, keeping detection difficult.

**4. Two-Stage Malware Deployment Model**

* Most modern malware uses a **two-stage deployment** approach:

**Stage One:** The primary function of a stage one dropper or downloader is retrieving malware code and tricking activation.

* Begins with user interaction (clicking a link or opening a file).
* Installs a **dropper** or **downloader** — a small, executable component.
  + **Dropper**: Initiates or runs other malware forms within a payload on an infected host.
  + **Downloader**: Retrieves additional tools post the initial infection facilitated by a dropper.
* The dropper runs **initial code**, initiating the infection.

**Stage Two:**

* Installs a **Remote Access Trojan (RAT)** or other persistent payload.
* Allows the attacker to perform **Command and Control (C2)** on the compromised machine.

**5. Definitions of Dropper, Downloader, and Shellcode**

* **Dropper**: Malware that carries and installs other malware payloads.
* **Downloader**: Retrieves additional components or tools **after** initial infection.
* **Shellcode**: Lightweight code designed to **exploit a vulnerability** or execute an initial action on a system.

**6. Threat Actor Objectives**

* After gaining access through Stage Two:
  + Attacker uses **remote access tools** to infect other systems.
  + Focuses on compromising **servers**, **domain controllers**, or other **high-value assets**.
  + If those targets are unreachable, attackers aim to expand **network footprint** and **permissions**.

**7. Action on Objectives Phase**

* Once inside the network, threat actors begin executing their **primary goals**:
  + **Data exfiltration**
  + **Ransomware deployment (file encryption)**
  + Other **malicious operations** (e.g., espionage, sabotage)

**8. Concealment Phase**

* Final step: **hide the intrusion** to retain access and avoid detection.
* Techniques include:
  + **Deleting logs**
  + **Erasing temporary files**
  + **Covering evidence** of malware activities

**9. Malware Delivery Techniques**

* Attackers use various methods to deliver and execute malicious code:

**9.1 Code Injection**

* Inserts malware into a legitimate process for **stealth execution**.
* For example, code injection is used to disguise the malicious code by running it with the identification of a legitimate process.

**10. Anti-Forensics and Obfuscation**

* Malware may be combined with:
  + **Encryption**
  + **Compression**
  + **Obfuscation**
* These techniques hinder detection and analysis by **security professionals**.

**11. Living off the Land (LotL) Techniques**

* Advanced attackers use **built-in system tools** to execute malware—no external files needed.
* This tactic reduces detection risk because:
  + No need to download obvious malicious files.
  + They use trusted tools like **PowerShell**.
    - This is a standard tool that is installed by default on many Windows systems and servers.
    - Power shell can be manipulated to conduct lots of different types of malicious activities.

**Example: PowerShell Abuse**

* PowerShell is a legitimate Windows tool.
* Attackers can run malicious commands within PowerShell without triggering standard security alerts.
  + This means it’s really easy for an attacker to obscure their malicious activities by making the detection of those activities more difficult by relying on internal PowerShell commands as opposed to downloading additional malicious code.

**12. Defensive Considerations**

* Cybersecurity professionals must:
  + Be aware of these **evasion techniques**.
  + Monitor **legitimate tools** like PowerShell for **unusual behavior**.
  + Use behavior-based and memory scanning tools—not just signature-based antivirus.