Below is a **sentence-by-sentence breakdown** of the document **“Rootkits Notes.docx”**, structured as **study notes for the CompTIA A+ 220-1102 exam (Objective 2.4 – Security)**. It is formatted for **Microsoft Word** with numbered topics, minimal spacing, and professional alignment — no reformatting needed on your end.

**🛡️ Rootkits – Study Notes CompTIA A+ 220-1102 | Objective 2.4 – Security**

**1. What Is a Rootkit?**

* A **rootkit** is a specialized form of malware designed to **gain administrative-level control** (root access) over a computer **without being detected**.
* **Root** or **admin access** allows unrestricted control over the system, including installing/removing programs, and manipulating system components.
* On **Windows**, this level is called the **Administrator Account**.
* On **Linux, Unix, and macOS**, it’s called **Root Access**.
* Gaining such access is **highly advantageous for attackers**, but **very dangerous** for victims.

**2. System Permissions and Kernel Rings**

* Operating systems use a **ring architecture** to define privilege levels.
  + **Ring 3**: Where standard users operate — least privilege.
  + **Ring 0 (Kernel Mode)**: Full system control — interacts directly with hardware and drivers.
  + **Ring 1**: Sits between user and kernel — typically where **root/admin privileges** are held.
* The closer malware gets to **Ring 0**, the more dangerous and **invisible** it becomes.

**3. Rootkit Positioning**

* Rootkits aim to install at **Ring 1 or Ring 0**, enabling:
  + **Deeper system access**
  + **Evasion from security software and OS-level monitoring**
  + **Long-term persistence** on the system
* A rootkit can **hide from users, system admins**, and even from the **operating system itself**, if embedded correctly.

**4. Capabilities of Rootkits**

* Rootkits can perform **malicious operations** at any time, including:
  + Hiding their presence
  + Monitoring user activity
  + Modifying system files and processes
* Once installed, **they are extremely difficult to detect or remove**.

**5. DLL Injection**

* Rootkits use a technique called **DLL Injection** to maintain persistence.
* This involves inserting **malicious code into a running process** on a Windows system by exploiting **Dynamic Link Libraries (DLLs)**.
* Since DLLs are loaded at runtime, the **Windows OS often fails to recognize** the malicious modification.
* This allows the rootkit to **remain hidden and active**.

**6. Driver Manipulation**

* Another method rootkits use is **driver manipulation**, where they compromise **Kernel Mode device drivers**.
* These drivers operate at **high privilege levels**, making them ideal for hiding malware.
* Modifying drivers allows malware to act as if it’s part of the legitimate operating system.

**7. Shims**

* Both DLL Injection and Driver Manipulation use a **Shim**.
* A **Shim** is a piece of **intermediary software code** placed between two components to:
  + **Intercept function calls**
  + **Redirect them with malicious code**
* For example:
  + A Shim can intercept the connection between **Windows and a DLL**, redirecting it to **execute a malicious routine** instead.

**8. Why Rootkits Are Dangerous**

* Rootkits are **extremely powerful and stealthy**.
* The **operating system is often blind** to their presence.
* They can continue operating **without raising any alerts** in traditional antivirus or system logs.

**9. How to Detect Rootkits**

* Traditional system scans **may not detect rootkits**, due to their deep integration.
* The most reliable detection method is to:
  + **Boot from an external device** (e.g., USB or rescue CD).
  + **Scan the internal hard drive** from that clean external environment.
* This method bypasses the compromised OS and can **identify hidden rootkit files**.

Let me know if you’d like a **5- or 10-question CompTIA-style quiz** with answers and full grading based on this topic — formatted for Word!