DLL Hijacking Overview

Malware Persistence via `explorer.exe` DLL Injection

Attack Chain Overview

1. **Initial Compromise**:
   An attacker sends a phishing email with a malicious Office document. When the victim enables macros, a script downloads and executes a payload (e.g., a backdoor) onto the system.
2. **Privilege Escalation**:
   The backdoor exploits a local Windows vulnerability (e.g., a DLL hijacking flaw in a system utility) to gain administrative privileges.
3. **DLL Injection into `explorer.exe`**:
   The attacker deploys a malicious DLL (e.g., `malicious.dll`) and injects it into `explorer.exe`, a process that:
   - Runs under the current user's context (no admin rights needed post-injection).
   - Restarts automatically at user login, ensuring the malware survives reboots.
   - Blends into normal system activity, avoiding suspicion.

Step-by-Step Technical Execution

1. **Malicious DLL Creation**

- The attacker crafts a DLL that performs malicious actions:
  - **Persistence**: Writes a registry key (e.g., `HKCU\Software\Microsoft\Windows\CurrentVersion\Run`) to relaunch the malware if `explorer.exe` is terminated.
  - **C2 Communication**: Connects to a command-and-control (C2) server for further instructions.
  - **Payload Execution**: Drops additional malware (e.g., ransomware, spyware).
  2. **Injection into `explorer.exe
- **Method**:
  The attacker uses a tool like **Process Hacker**, **Metasploit's `post/windows/manage/dllinject`**, or custom code to:
  1. Enumerate running processes to find `explorer.exe` (PID).
  2. Allocate memory within `explorer.exe` using `VirtualAllocEx`.
  3. Write the malicious DLL path or binary into the allocated memory (`WriteProcessMemory`).
  4. Execute the DLL via `CreateRemoteThread` (or APC injection) to load it into `explorer.exe`
- **Result**:
  The malicious DLL runs under the guise of `explorer.exe`, inheriting its permissions and appearing benign to security tools.
  3. **Persistence Mechanism**
- The DLL adds a registry entry to reload itself if the process dies:

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
Value: "LegitApp" = "C:\Windows\System32\explorer.exe /loadmalicious"
```

- Because `explorer.exe` is a legitimate Windows process, security software may ignore the activity.

Real-World Example: Kovter Malware

Kovter, a fileless malware strain, abused DLL injection into `explorer.exe` for persistence:

1. **Initial Access**: Delivered via malicious ad campaigns or phishing.
2. **Injection**:
   - Stored payloads in registry keys (encoded as blobs) instead of files to evade detection.
   - Injected code into `explorer.exe` to decode and execute the payload from memory.
3. **Persistence**:
   - Modified registry keys to reload the malicious code every time `explorer.exe` started (e.g., user login).

Post-Exploitation Actions

Once the DLL is running inside `explorer.exe`:

1. **Evasion**:
   - Network traffic appears to originate from `explorer.exe`, bypassing firewalls.
   - Process hollowing or reflective DLL injection hides the malicious code.
2. **Lateral Movement**:
   - Uses `explorer.exe`'s access to network shares or clipboard data (e.g., stealing credentials).
3. **Payload Execution**:
   - Deploys ransomware (e.g., LockBit) or spyware (e.g., keyloggers).

Detection & Mitigation

How to Spot DLL Injection into `explorer.exe`

- **Tools**: Sysinternals Process Explorer, Autoruns, or EDR solutions.
- **Signatures**:
  - Unexpected child processes of `explorer.exe`.
  - Unusual registry entries under `Run` or `AppInit_DLLs`.
  - Memory anomalies in `explorer.exe` (e.g., unexpected modules).

Mitigation Strategies

1. **Restric DLL Injection**:
   Use tools like **Microsoft Attack Surface Reduction (ASR)** to block untrusted processes from injecting code.
2. **Monitor Process Behavior**:
   Alert on `explorer.exe` spawning `cmd.exe`, `powershell.exe`, or making network connections.

3. **Application Whitelisting**:
   Block unsigned DLLs from loading into critical processes.
4. **Registry Auditing**:
   Monitor changes to `Run` keys or `AppInit_DLLs`.

#dllhijacking