## Registry Persistence

## Windows Services

> The following section is sourced from this blog.

Attackers continue to abuse Windows Services for their SYSTEM-level privileges (`NT AUTHORITY\SYSTEM`), which grant unrestricted access to the system. While UAC bypasses were common on Windows 7/8, modern Windows 10/11 and Server 2016+ enforce stricter controls:

- **Protected Process Light (PPL)**: Critical services (e.g., `LSASS.exe`) are isolated from tampering.
- **Driver Signature Enforcement**: Blocks unsigned kernel-mode drivers, reducing "Own Process" service hijacking.
- **Windows Defender Application Control (WDAC)**: Restricts unauthorized DLL/EXE execution, complicating "Share Process" attacks.

| Technique of persistence | Risk Assessment | Usage |
|---|---|---|
| HKLM\System\CurrentControlSet\Services | Runs as SYSTEM, very stealthy, safe<br><br>- **High risk** if attackers bypass WDAC. | Used for long-term remote access in early stages and as backup access<br><br>- APTs deploy **phantom DLLs** (e.g., APT28’s "GreyEnergy").<br>- Mitigation: Enforce **code integrity policies** and monitor `CurrentControlSet` changes (Sysmon Event ID 13). |
| HKLM\Software run keys | Runs as Administrator, less stealthy<br><br>- Moderate risk (common in commodity malware). | Used for initial compromise<br><br>- Ransomware (e.g., BlackCat) uses Run keys for initial foothold.<br>- Mitigation: Restrict registry writes via **AppLocker** and hunt for `reg.exe` anomalies (Elastic/Splunk). |
| HKCU run keys | Early stage droppers & downloaders, noisy and dangerous<br><br>- Low risk (noisy, user-level). | Initial exploit, day zero activity<br><br>- Phishing payloads (e.g., QakBot) leverage HKCU for persistence.<br>- Mitigation: Deploy **UEBA** to detect anomalous user activity. |

## Understand Registry Persistence Types

Contemporary registry persistence techniques fall into **3 operational categories:**

| Type | Definition | Article Examples |
|---|---|---|
| **List Appenders** | Add to existing multi-item registries (e.g., multiline values, subkey lists) | LSA Authentication Packages, BootExecute |
| **Ghost Key Actors** | Create keys that Windows checks but don't exist by default | Office Test\Special\Perf key |
| **Value Overwriters** | Replace single default values in commonly monitored keys | UserInit shell replacement |

### Why these matter

- **Detection Bypass**: List appenders (e.g., adding to `BootExecute`) evade tools that only alert on *replacement* of known values.
- **Low IOC Coverage**: 83% of enterprise EDRs lack rules for "Ghost Key" creation (source: article's "Forgotten Techniques" section).
  3

## Privilege Context:

Although attackers still target Windows Services for SYSTEM-level privileges (NT AUTHORITY\SYSTEM), modern Windows 10/11 and Server 2016+ systems enforce stricter controls:

- **User Account Control (UAC)** bypass via service abuse is mitigated by *Windows Defender Credential Guard* and *Protected Process Light (PPL)* restrictions.
- **Driver Signature Enforcement** blocks unsigned kernel-mode drivers, reducing "Own Process" service hijacking.
  - **Windows Defender Application Control (WDAC)** and **AMSI** scrutinize unauthorized DLL/EXE execution, complicating "Share Process" (svchost.exe) attacks.

**Service Types in Modern Context:**

| Type | Contemporary Risks & Detection | Detection & Mitigation |
|---|---|---|
| **Own Process** | - Rare in modern attacks due to AMSI/EDR scrutiny.<br>- Seen in ransomware (e.g., LockBit's .exe droppers). | - Block unsigned executables via **WDAC**.<br>- Monitor `CreateService` API calls (Sysmon Event ID 12). |
| **Share Process** | - Still abused via **DLL sideloading** (e.g., Cobalt Strike's *execute-assembly*).<br>- Detected via anomalous svchost child processes (Sysmon Event ID 1). | - Alert on `svchost.exe` spawning unusual children (e.g., `powershell.exe`) via EDR/Sysmon (Event ID 1). |

## Registry Locations of Interest

`HKLM\System\CurrentControlSet\services`

- Loaded by the Service Controller at various times during computer operation(system startup, event trigger)
- **Driver Signature Enforcement**: Windows 10/11 blocks unsigned drivers/services unless Secure Boot is disabled.
- **Protected Services**: Critical services (e.g., LSASS) now run as **Protected Processes (PPL)**, preventing tampering
- **Detection**: Modern EDRs (CrowdStrike, Microsoft Defender) flag:
  - Unusual service binaries (e.g., `svchost.exe` spawning `powershell.exe`)
  - Services with mismatched hashes or unsigned DLLs (via **AMSI**)
  - Correlates process lineage with registry changes(ex. `svchost.exe` spawning `cmd.exe`)

`HKLM\Software\Microsoft\Windows NT\CurrentVersion\SvcHost`

- Usually heavily monitored
- Groups services that have similar privilege needs
  - Hijack deprecated services (e.g., `IISADMIN`) to load malicious DLLs
    - RBAC, Virtualization-Based Security (VBS), and Credential Guard
      - Windows 11 auto-blocks unused `netsvcs` groups via **controlled folder access**.
      - Virtualization-Based Security (VBS): Isolates critical services to prevent credential theft.
- The `netsvcs` group still exists and categorizes network-dependent services.
  - Windows 10/11 and Server 2022 have fewer unused `netsvcs` entries due to deprecated services (e.g., `Irmon`, `Ntmssvc`)
- Sysmon (Event ID 12/13) logs registry modifications to `SvcHost` keys
- Tools like **Velociraptor** baseline legitimate `netsvcs` entries and alert on anomalies

## Other Ideas

### Stealthy Registry Key Paths

User-Specific (HKCU)

1. `HKCU\Software\Adobe\Acrobat\Updater`
2. `HKCU\Software\Microsoft\Office\Common\AutoUpdate`
3. `HKCU\Software\Google\Chrome\NativeMessagingHosts`
4. `HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders`
5. `HKCU\Software\Microsoft\Windows NT\CurrentVersion\Fonts`
6. `HKCU\Software\7-Zip\FM` (if 7-Zip is installed)
   System-Wide (HKLM)
7. `HKLM\Software\Microsoft\Windows Defender\Features`
8. `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache`
9. `HKLM\Software\Microsoft\Cryptography\RNG`
10. `HKLM\Software\Microsoft\Windows\CurrentVersion\Setup\PersistenceComponents`

11. `HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate`
12. `HKLM\Software\Microsoft\Windows\CurrentVersion\Reliability`

### Legitimate-Sounding Software Names

1. Updaters:
   - `AdobeUpdater`
   - `Java Update Scheduler
   - `FlashPlayerSecurityPatch`
   - `MicrosoftEdgeAutoUpdate
2. System Tools:
   - `SecurityHealth` (mimics Windows Defender)
   - `RuntimeBroker` (real Windows process)
   - `Background Tasks Infrastructure Service
   - `Windows Error Reporting`
3. Common Software:
   - `TeamViewer_Service
   - `ZoomPresence
   - `SlackHelper
   - `OneDriveSyncEngine`
4. Generic Background Tasks:
   - `PowerManagement
   - `NetworkConfigurator
   - `DisplayCalibration`
   - `AudioEndpointBuilder
5. Hardware/Driver-Related:
   - `NvidiaDisplayContainer
   - `IntelHDAService
   - `RealtekAudioService
   - `AMDExternalEvents`

# Evasion

1. Avoid Overused Names: Don't use `svchost`, `explorer`, or `winlogon`—these are heavily monitored.
2. Use Subkeys: Hide deeper in the registry (e.g., `HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\StartupApproved\Run`).
3. Match Installed Software: Check the target's installed programs (e.g., `CCleaner` or `VLC`) and mimic their naming.
4. Environment Variables: Use paths like `%APPDATA%\Microsoft\Windows\Themes\Cursors` for payload storage.
5. Legacy Key Revival: Target deprecated-but-still-functional keys like `HKLM\...\Office Test\Special\Perf` (checked by Office apps)
6. Multi-Stage List Injection: Append to existing lists rather than replacing entries (e.g., add to LSA's `Authentication Packages` vs overwriting)

7. COM Hybrid Approach: Create both HKLM and HKCU entries for App Paths/Credential Providers to ensure persistence across privilege levels
8. Boot Sequence Mimicry:
   - Use names like `BootVerification.exe` in `%SystemRoot%\System32**
   - Match Microsoft's timestamping on registry keys
9. **Ghost Key Creation**:
   - Target registry paths checked by apps but non-existent by default
   - Example: `HKLM\Software\Microsoft\Office Test\Special\Perf` (triggers on Office launch)
   - Find via Procmon: Filter `Result=NAME NOT FOUND` on high-value processes (winword.exe, etc.)

## Emerging Threats & Defenses

1. **Cloud-Hybrid Attacks**:
   - Attackers pivot from on-prem registry persistence to **Azure Arc-enabled servers** for cross-cloud access.
   - Mitigation: Use **Azure Sentinel** to correlate registry changes with cloud identity anomalies.
2. **Zero-Trust Strategies**:
   - Enforce **conditional access policies** for service accounts.
   - Implement **just-in-time (JIT)** privilege escalation for administrators.
3. **List Appender Defense**:
   - Use **Canary Values**: Insert fake entries in multiline registries (e.g., `BootExecute`) to detect additions
   - Enable **Registry Filtering Driver** logging (Microsoft-Windows-FilterManager/0x803)

---

# Creating Persistence via Registry Key

## Scripts

### `.bat` files

```
Microsoft.Win32.RegistryKey key;
key = Microsoft.Win32.Registry.CurrentUser.CreateSubKey("My_Key"); key.SetValue("My_Key",
"Test");
key.Close();
```

1

## Terminal

Registry keys can be added from the terminal to the run keys to achieve persistence. These keys will contain a reference to the actual payload that will executed when a user logs in. The following registry locations is known to be used by threat actors and red teams that use this method of persistence.

```
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run" /v Pentestlab /t REG_SZ /d "C:\Users\pentestlab\pentestlab.exe"
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce" /v Pentestlab /t REG_SZ /d "C:\Users\pentestlab\pentestlab.exe"
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServices" /v Pentestlab /t REG_SZ /d "C:\Users\pentestlab\pentestlab.exe"
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce" /v Pentestlab /t REG_SZ /d "C:\Users\pentestlab\pentestlab.exe"
```

If elevated credentials have been obtained it is preferred to use the Local Machine registry locations instead of the Current User as the payload will executed every time that the system boots regardless of the user who is authenticating with the system.

```
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
```

Oddvar Moe discovered two more registry locations that could allow red teams to achieve persistence by executing either an arbitrary payload or a DLL. These will be executed during logon and require admin level privileges.

```
reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\0001" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
reg add
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\0001\Depend" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.dll"
```

2

## Mitigations for Registry Persistence

| Technique | Modern Detection Signature | Recommended Control |
|---|---|---|
| Boot Verification Hijack | `services.exe` launching non-MS signed binaries post-boot | WDAC + Protected Service policies |
| SMSS Configuration | Unexpected entries in `BootExecute` multiline string | Baseline monitoring of Session Manager keys |
| RDP Startup Abuse | `termsvcs.dll` anomalies in RDP sessions | Restrict `rdpwd` key modifications |

| Technique | Modern Detection Signature | Recommended Control |
|---|---|---|
| App Paths Hijacking | Non-standard EXE paths in App Paths keys | Registry access control (JEA) |

## Resources

1. https://r4bb1t.medium.com/windows-persistence-registry-run-keys-e9acb20c4a7d
2. https://pentestlab.blog/2019/10/01/persistence-registry-run-keys/
3. https://www.cyberark.com/resources/threat-research-blog/persistence-techniques-that-persist

#windows #registry #persistence