Case Study: RedLine Stealer Remediation Using DoesNotBelong

Author: Furtivex (Developer of DoesNotBelong)
Website: https://furtivex.net
Tool Version: DoesNotBelong v10.2.8
Operating System: Windows 11 Pro x64 (25H2)

Executive Summary

This case study documents the successful remediation of a **RedLine information stealer infection** using **DoesNotBelong** as the *only cleanup tool*. The incident was handled publicly on a third-party malware-removal forum, without scripting, layered scanners, or follow-up remediation utilities.

After running DoesNotBelong and reviewing its log, the assisting helper and the affected user both concluded that the system was clean and stable, and the support thread was closed without escalation.

This case demonstrates how **context-driven anomaly removal** can be effective against modern user-space malware that relies on camouflage, persistence abuse, and living-off-the-land techniques rather than traditional kernel-level exploits.

Background: RedLine Stealer

RedLine is a .NET-based information stealer commonly distributed via:

- Phishing emails and fake invoices
- Cracked software installers
- Malicious browser notifications

Rather than deeply embedding itself into the operating system, RedLine typically:

- Operates entirely in **user space**
- Drops payloads in **AppData and user directories**
- Establishes persistence via services, scheduled tasks, or policy abuse
- Abuses **DLL search-order hijacking** and fake runtime libraries
- Disables or bypasses Windows Defender via exclusions and policies

These characteristics make RedLine particularly well-suited to evasion-by-blending rather than brute force.

Incident Overview

- **Infection Type:** RedLine information stealer
- **Initial Symptoms:** User-reported malware alert and suspicious behavior
- **Environment:** Home Windows 11 system
- **Remediation Approach:** Single execution of DoesNotBelong
- **Additional Tools Used:** None

The remediation took place under public observation in a malware-removal forum context, providing independent validation of the outcome.

What DoesNotBelong Identified and Removed

1. Malicious Services

A non-standard service with a generic, authoritative-sounding name was detected and removed:

- Service names like this are frequently used by malware to appear legitimate
- The service did not match any known Windows or third-party baseline

Removing this service eliminated a potential persistence mechanism.

---

2. Confirmed RedLine Artifacts

Multiple randomly named directories and .NET project files were found within the user profile:

- Randomized folder names
- C# project artifacts (.csproj)
- Multiple staging locations

These artifacts are strongly associated with RedLine's build and deployment workflow and were safely removed.

---

3. User-Space Fake Windows Runtime Libraries

One of the most significant findings was the presence of **entire fake Windows runtime ecosystems** located in user-writable directories, including:

- AppData\Roaming\Kernel
- AppData\Roaming\Sys64
- AppData\Roaming\Microsoft\SysDriver64
- AppData\Roaming\NetworkSettings

These directories contained hundreds of files impersonating legitimate Windows components, such as:

- api-ms-win-core-*.dll
- kernel32 legacy and private API DLLs
- DirectX and input libraries

Legitimate copies of these files belong exclusively in **System32** or **WinSxS**. Their presence in AppData is a clear indicator of **DLL side-loading and execution hijacking**.

DoesNotBelong removed these directories in full, collapsing the attacker's execution environment rather than attempting selective cleanup.

---

4. Browser Persistence and Abuse

Browser push-notification abuse was detected and remediated for multiple browsers:

- Unauthorized notification permissions removed
- Reinfection and phishing vectors eliminated

This step addressed secondary persistence mechanisms often overlooked by traditional scanners.

---

5. Phishing Artifacts

Malicious PDF attachments used as likely infection vectors were identified in mail cache locations and removed. This reduced the risk of reinfection and user re-exposure.

---

6. Group Policy and Security Manipulation

System policy files were flagged alongside extensive Windows Defender exclusions, including:

- Exclusions covering entire user profile paths
- Exclusions covering ProgramData and system configuration paths

Such exclusions are highly abnormal and are commonly used by stealers to ensure long-term survival. Their removal restored Defender's visibility and protection.

---

7. AppX Package Abuse

Several consumer-facing AppX packages were removed to eliminate potential abuse chains and force clean re-provisioning by Windows. These packages are non-critical and automatically recoverable.

---

8. Anti-Forensics Indicators

Cleared event logs were detected, indicating an attempt to obscure execution history. While logs cannot be restored, surfacing this behavior provided important context confirming malicious activity.

Outcome

After the DoesNotBelong cleanup:

- No further malicious indicators were observed
- No additional scanners or scripts were requested
- The user reported normal system behavior
- The assisting helper concluded the case

The remediation was considered complete.

Why This Worked

DoesNotBelong succeeded because it did not rely solely on identifying known malware samples. Instead, it focused on **environmental legitimacy**:

- Windows runtime components were not allowed to exist outside trusted locations
- Persistence mechanisms without a legitimate baseline were removed
- Security posture manipulation was reversed

By removing what *could not reasonably belong*, the tool eliminated both the malware and the conditions required for it to function.

Conclusion

This case demonstrates that modern malware remediation does not always require layered scanners or signature-based detection. Against threats like RedLine, which depend on blending into user space, **context-aware anomaly removal** can be both safe and effective.

DoesNotBelong acted as a focused incident-response tool rather than a traditional antivirus scanner, achieving a clean outcome with a single run.

This case study is published for educational and transparency purposes. Results may vary depending on system state and threat characteristics.