THREATLOCKER® RINGFENCING™ LOCK DOWN POWERSHELL VULNERABILITIES

BY JHEROME DEGUZMAN, THREATLOCKER SOLUTIONS ENGINEER MANAGER

A deep dive into real-world threats and practical mitigations

PowerShell plays a critical role in the Windows ecosystem. Its versatility makes it an invaluable tool in most enterprise environments. Much of modern Windows system administration is performed through PowerShell. It boasts a trusted reputation through its omnipresence and the industry expectation that all sysadmins use it, or at least know of it. But that trust can be easily exploited. Cybercriminals often abuse PowerShell's deep access to carry out fileless attacks that bypass traditional detection solutions. It can be used to secretly download malicious payloads, move laterally within a network, escalate privileges, and exfiltrate sensitive data—all without ever creating a detectable file on disk.

In short, PowerShell can represent a significant security risk if left unchecked, but ThreatLocker Ringfencing helps mitigate that risk. It restricts lateral movement, limits access to sensitive files and executables, blocks unauthorized changes to the Windows Registry, and precisely governs which applications—including PowerShell—can connect to the internet.

Securing PowerShell requires understanding exactly what it can do once it's running. ThreatLocker Ringfencing shifts the conversation from whether it's running or not to discovering exactly how, when, and where PowerShell is acting. In this guide, we'll walk through real-world attack vectors and look at the exact ThreatLocker Ringfencing policies that can be implemented to keep PowerShell activity in check. ThreatLocker provides default PowerShell Ringfencing policies, but they can be modified to fit your environment.

LIVING-OFF-THE-LAND ATTACKS

Suppose an attacker gains access to a non-admin, low-privilege endpoint. Once inside, they use legitimate PowerShell commands like Invoke-WebRequest, Invoke-Expression, or Start-Process to download and execute malicious payloads directly in memory. These commands will leave minimal observable activity against a hard drive, allowing them to bypass many traditional antivirus and endpoint detection tools.

THREATLOCKER RINGFENCING SOLUTION: Block all unnecessary internet connections from PowerShell

Ringfencing rule:

- Application: PowerShell (Built-in)
- Restriction: Restrict this application from accessing the internet
- Exceptions: None

This policy blocks PowerShell from establishing outbound connections to command-and-control infrastructure, cloud storage services, or file repositories used to deliver payloads.

READING SENSITIVE DATA FROM THE FILE SYSTEM

Attackers often use PowerShell to scrape data from local drives, network shares, or mapped drives. This harvested data can then be exfiltrated or leveraged to facilitate further actions like lateral movement and escalating privileges.

THREATLOCKER RINGFENCING SOLUTION: Restrict PowerShell file access

Ringfencing rule:

- Application: PowerShell (Built-in)
- Restriction: Restrict this application from accessing files
- Exceptions: C:\Scripts*

This principle of least privilege at the filesystem level ensures that PowerShell can't be used to harvest data from the endpoint and the drive shares that the endpoint can access. Ringfencing applies Zero Trust by restricting PowerShell to designated folders and files.

LAUNCHING OTHER PROCESSES

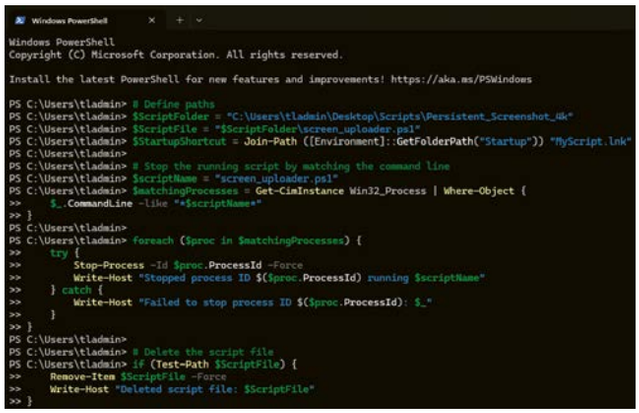
Attackers frequently use PowerShell as part of a chained attack strategy. It's an application that, by its nature, casts a wide net. So it's wise to ensure that other applications cannot launch PowerShell in order to stop attackers from being able to execute additional payloads, establish persistence mechanisms, or obscure malicious activity by blending it with legitimate system processes.

THREATLOCKER RINGFENCING SOLUTION: Block other applications from launching or interacting with PowerShell

Ringfencing rule:

- Application: Microsoft Office
- Restriction: Restrict this application from interacting with other applications
- Exceptions: None

Even if malware manages to execute on an endpoint, it's effectively contained—unable to leverage PowerShell to interact with other applications, access sensitive files, or communicate with the broader system.



ThreatLocker engineers test Ringfencing against a vast catalog of malicious PowerShell scripts

LAUNCHING BITLOCKER

Attackers often use PowerShell to execute malicious commands and encrypt entire drives using BitLocker. By using Ringfencing rules in the opposite direction, we can specifically restrict PowerShell's ability to interact with BitLocker.

THREATLOCKER RINGFENCING SOLUTION: Block PowerShell from launching or interacting with BitLocker

Ringfencing rule:

- Application: PowerShell (Built-in)
- Restriction: Allow this application to interact with other applications
- Exceptions: BitLocker

This stops PowerShell from being able to use BitLocker to encrypt the drive, preventing ransomware from using BitLocker as a malicious tool.

BUILDING A LEAST PRIVILEGE RINGFENCING POLICY FOR POWERSHELL

1. Audit behavior first
Use the Unified Audit to review what PowerShell is doing in your environment. Identify legitimate paths, network calls, and scripts.

2. Define use cases
Determine whether PowerShell is required on all endpoints, who should be able to use it, and what it should be able to access.

3. Create controlled Ringfencing policy
Restrict file and location access, deny external network access, and disable lateral movement and child process execution.

4. Create exceptions via approvals
If a PowerShell script needs elevated access, create a policy with a time-bound elevation exception or approval workflow.

5. Monitor continuously
Set alerts on policy violations for real-time response and refine the policy iteratively and regularly.