The effectiveness of Microsoft Application Allowlisting 2025 Edition

Yearly Report by AppControl.AI

Authored by Kim Oppalfens, AppControl.AI Security Architect

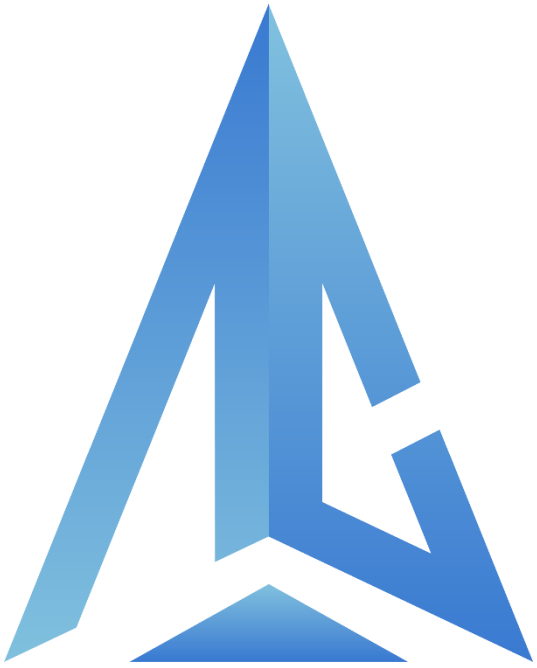


Table of Contents

Executive Summary

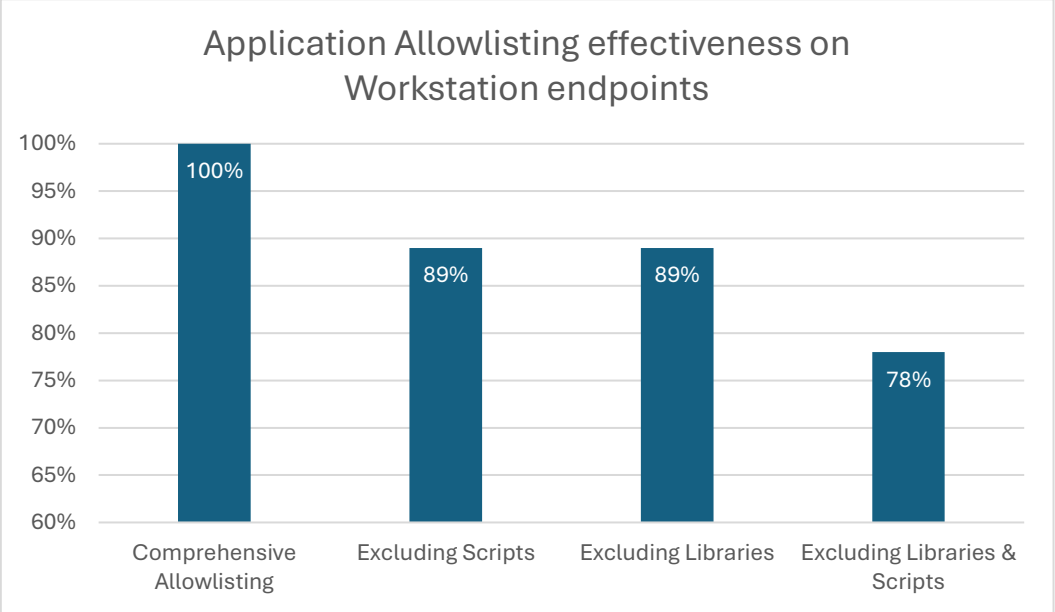
Appcontrol.AI's Microsoft application allowlisting report uses Microsoft Defender for Endpoint Threat Intel reports to objectively quantify the effectiveness of implementing Windows application allowlisting against a comprehensive dataset.

Key Findings

1. Implementing application allowlisting that targets executables, libraries and scripts **prevented every Threat report produced by the Microsoft Defender for Endpoint team in 2025** that depended on code execution on workstation devices that did not use SimpleHelp as their Remote Management & Monitoring tool.
2. Implementing application allowlisting that targets executables, libraries and scripts **prevented all but 6 Threat reports produced by the Microsoft Defender for Endpoint team in 2025** that depended on code execution on devices running a server workload. It is recommended to have a look at the additional measures that are proposed for Server workloads that run under a privileged account.
3. Attack Campaign reports describe coordinated, time-bound attack operations observed in the wild. **No Attack Campaigns took place in 2025 that would have been effective in an environment that implemented application allowlisting targeting executables, libraries and scripts.**
4. **Excluding Script allowlisting** from your allowlisting implementation opens the environment up to **11 Threat reports** produced by the Microsoft Defender for Endpoint team. These 11 threat reports are made up of no less than **9 actual Attack campaigns** and 2 vulnerability profiles.
5. **Excluding Library allowlisting** (dll's, etc…) opens the environment up to **11 Threat reports** produced by the Microsoft Defender for Endpoint team. These 11 threat reports are made up of **4 actual attack campaigns**, 5 Tool and Technique reports and 2 vulnerability profiles.

Report Conclusions

1. The analysis indicates that allowlisting provides preventive protection at the execution layer, reducing the dependency on detection and response in many attack scenarios.
2. Allowlisting effectiveness is proportional to policy scope; gaps in scripts or libraries create exploitable pathways that adversaries actively use.
3. Server environments present distinct risk factors—primarily privileged, network-facing applications—that are not mitigated by allowlisting alone and require complementary controls as advised in the appendix of this report.
4. Script execution control is the most significant differentiator in effectiveness; leaving scripts open materially increases the likelihood of successful real-world campaigns.
5. Library-based allowlisting is a significant element in disrupting the operations of ransomware gangs.



Methodology

This report documents the approach used to evaluate and demonstrate the effectiveness of Microsoft application allowlisting solutions against contemporary security threats observed in 2025. The objective was to ground the assessment in real-world threat activity while ensuring relevance to Windows environments and to allowlisting as a preventive control.

Source of Threat Intelligence

To author this report we needed a good sample of real-life threats to the security of Windows endpoints. The choice was made to use Microsoft Defender for Endpoint Threat Analytics as the primary base of threats as a common well-known source of Threat intel. All analyst reports published during the 2025 calendar year were reviewed and catalogued. Defender Threat Analytics was selected as the baseline due to its direct alignment with Microsoft security controls, its focus on active and emerging threats, and its detailed technical analysis of adversary behavior affecting enterprise environments and detailed analyst reports that go with it.

Threat Selection and Filtering Criteria

Each 2025 analyst report was evaluated against a defined relevance framework to determine its suitability for assessing application allowlisting effectiveness. Reports were excluded from further analysis if they met one or more of the following criteria:

- **No observable code execution**: Threats that relied solely on social engineering, credential misuse, misconfiguration, or abuse of legitimate cloud services without introducing or executing code on an endpoint were marked as irrelevant, as application allowlisting would have limited or no opportunity to intervene.
- **Insufficient technical detail**: Reports that lacked clarity on execution chains, payload delivery, process creation, or binary/script characteristics were excluded, as they did not provide enough information to reasonably assess allowlisting impact.
- **Non-Windows targeting**: Threats clearly designed for non-Windows platforms (e.g., mobile OSs, Linux-only workloads, MacOS, Network equipment, or cloud-native services without endpoint execution) were excluded to maintain platform relevance.

Only reports that demonstrated credible execution of binaries, scripts, installers, or living-off-the-land tooling on Windows endpoints were retained for analysis.

Analysis Approach

For each relevant threat, the execution chain described in the analyst report was mapped against Microsoft application allowlisting capabilities, including Windows Defender Application Control (WDAC) and related policy-based execution controls. The analysis focused on identifying where allowlisting could have:

- Prevented initial execution of untrusted or unsigned binaries.
- Blocked execution of malicious scripts, installers, or payloads delivered through common attack vectors.
- Restricted abuse of legitimate tools when not explicitly authorized by policy.

- Reduced attacker flexibility by enforcing strict trust boundaries on executable content.

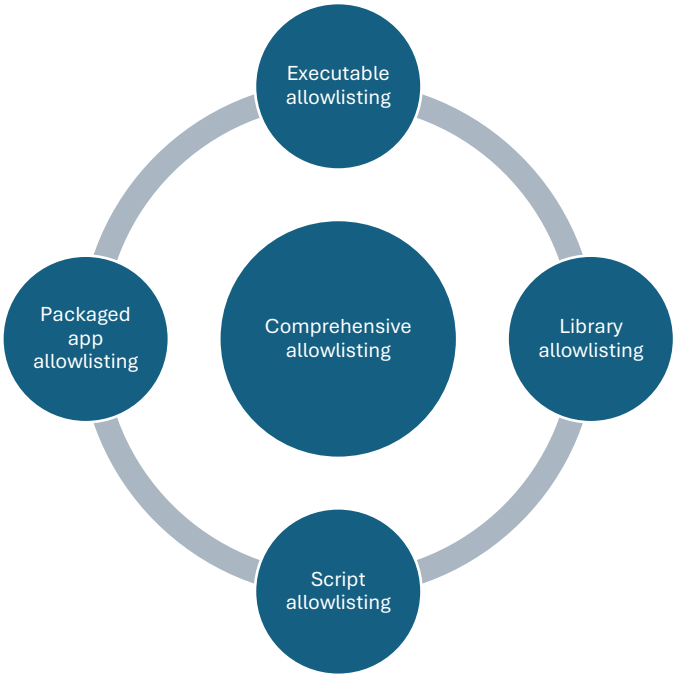
This assessment was performed conceptually, based on documented threat behavior, rather than through live detonation or emulation, ensuring consistency across all analyzed threats.

Allowlisting Configuration Model and Measurement Approach

The effectiveness of Microsoft application allowlisting was assessed using three distinct high-level configuration models. Each model represents a deliberate policy posture designed to isolate the impact of controlling different execution primitives on Windows endpoints.

Configuration Dimensions

The three configuration models evaluated were:



1. **Executable Allowlisting**
   This configuration enforces allowlisting controls on executable content, including portable executable (PE) files such as .exe and .com. Only explicitly trusted and policy-authorized binaries are permitted to run. This model reflects the most common baseline allowlisting deployment and is intended to prevent the execution of untrusted or unsigned applications delivered through common attack vectors.

2. **Library Allowlisting**
   In addition to executable control, this configuration enforces restrictions on dynamically loaded libraries, such as .dll files. Blocking untrusted libraries addresses a class of threats that rely on DLL search order hijacking, side-loading, or malicious library injection into otherwise trusted processes. This model demonstrates the additional risk exposure that remains when executable control is present but library enforcement is absent.

3. **Script Allowlisting**
   This configuration applies allowlisting controls to script-based execution, PowerShell, Windows Script Host, JavaScript. Script blocking targets threats that avoid dropping traditional binaries and instead rely on living-off-the-land techniques, fileless execution, or inline script payloads.

Outcome and Purpose

The resulting dataset represents a curated sample of real-world threats from 2025 where application allowlisting is a meaningful and measurable control. By excluding irrelevant or insufficiently detailed threats, the analysis avoids overstating allowlisting coverage while providing a realistic view of its defensive value.

This methodology ensures that conclusions drawn in the report are evidence-based, threat-informed, and directly applicable to Windows enterprise environments considering or operating Microsoft application allowlisting as part of their endpoint security strategy.

Analyst report types explained

Vulnerability

Vulnerability reports focus on a specific software or firmware weakness, typically identified by a CVE. These reports describe:

- The nature of the vulnerability and affected products
- Observed or potential exploitation in the wild
- Threat actors or campaigns leveraging the vulnerability
- Recommended mitigations and security controls

From an allowlisting perspective, these reports are often only relevant when exploitation results in post-exploitation code execution (e.g., dropped payloads, loaders, or scripts).

ActivityGroup

ActivityGroup reports profile a specific threat actor or adversary group (e.g., nation-state, cybercriminal, ransomware operator). They typically include:

- Known aliases and attribution confidence
- Targeting patterns and motivations
- Common tools, malware families, and techniques
- Historical and recent activity observed by Microsoft

These reports are useful for allowlisting analysis when they clearly document endpoint execution behaviors, such as custom malware, loaders, or abuse of scripting engines.

ToolOrTechnique

ToolOrTechnique reports focus on a single tool, malware family, or adversary technique. Examples include remote access tools, loaders, script-based techniques, or LOLBins. These reports detail:

- How the tool or technique operates
- Common execution methods and artifacts
- Detection and mitigation coverage

This category is often the most directly relevant to application allowlisting, as it frequently documents executable content, scripts, or interpreters that can be explicitly allowed or denied via policy.

AttackCampaign

AttackCampaign reports describe coordinated, time-bound attack operations observed in the wild. They typically include:

- Initial access vectors
- End-to-end attack chains
- Payload delivery and execution
- Impact and objectives (e.g., ransomware, espionage)

Campaign reports are useful for demonstrating allowlisting effectiveness across entire execution chains, showing where policy enforcement could disrupt or stop an attack at multiple stages.

Key findings

The Microsoft Defender for Endpoint base that provides the Threat intel for this report classifies threats in several different report types.

- AttackCampaign
- Vulnerability
- ActivityGroup
- ToolOrTechnique
- AttackSurface

The key findings will be given for the overall effectiveness across all report types and specific highlights per report type where AppControl.AI as the authors of this report consider the highlight relevant.

Additionally the report will highlight differences across the different allowlisting configuration models:

- Executable Allowlisting
- Library allowlisting
- Script allowlisting

KF1 - Zero threats affect Workstation Endpoints*

The analysis demonstrates that, under a fully enforced application allowlisting configuration—where executables, libraries, and scripts are all subject to allowlisting controls—zero analyst reports across all report types resulted in a successful impact on Windows workstation operating system–based endpoints.

Across the full 2025 Defender for Endpoint Threat Analytics dataset reviewed, no threat that met the report's relevance criteria was able to complete its documented execution chain when all three execution primitives were constrained. In each case, the attack depended on at least one of the following being permitted by policy:

- Execution of an untrusted or attacker-delivered executable,
- Loading of a malicious or abused dynamic library, or
- Execution of script-based or interpreter-driven content.

When these three control classes were simultaneously enforced, the documented attack paths were fully disrupted, eliminating the attacker's ability to establish code execution or progress beyond initial access.

This result underscores two key conclusions. First, modern threats affecting Windows endpoints in 2025 continue to rely on one or more traditional execution surfaces, even when employing living-off-the-land or multi-stage techniques. Second, application allowlisting, when implemented comprehensively rather than partially, functions as a highly effective preventative control against a broad spectrum of real-world threats.

Importantly, this outcome does not rely on detection or post-execution response. Instead, it reflects preventative denial of execution, reinforcing the value of allowlisting as a foundational

endpoint hardening control when deployed with complete coverage across executables, libraries, and scripts.

(*) There is one Caveat for organizations running the SimpleHelp remote management & monitoring tool. There was one vulnerability profile that could have been used to bypass an allowlist implementation that required the deployment of the security patch issued by SimpleHelp.

Relevant Microsoft Defender for Endpoint Threat Intel Reports

Vulnerability Profile: CVE-2024-57726 - Multiple vulnerabilities found in SimpleHelp Remote Support Software

KF2 – 6 threats Impact Windows Server Under Full Allowlisting

While full application allowlisting eliminated observed impact on Windows workstation operating systems, the analysis identified a residual exposure affecting approximately 6% of reviewed threat intelligence reports when applied to Windows server environments.

It is important to contextualize this figure. The identified successes are not indicative of a systemic failure of allowlisting, nor do they apply broadly across all servers. Instead, these scenarios are tightly coupled to the presence of specific vulnerable server-side applications. In practical terms, the affected threat reports would only succeed on a subset of servers running the targeted software; environments that do not deploy these applications are not impacted.

Root Cause Characteristics

The dominant recurring factor across these server-impacting scenarios is the architectural pattern common to many server workloads:

- Long-running services that listen for inbound network traffic,
- Execution under highly privileged service accounts, most notably LocalSystem,
- Exposure of application-level vulnerabilities that can be triggered remotely.

In these cases, the attacker does not rely on introducing new executable content onto the system. Instead, exploitation occurs within the context of an already trusted, allowlisted service, effectively bypassing execution-based controls. Once code execution is achieved inside the privileged process, application allowlisting no longer has an opportunity to intervene as the attacker has gained privileges that allow him to alter the policy or disable application allowlisting altogether.

Implications for Control Effectiveness

These findings reinforce a key distinction between workstation and server threat models. Application allowlisting is highly effective at preventing the introduction and execution of untrusted code, but it cannot mitigate flaws in trusted, privileged, network-facing applications.

As a result, the observed 6% residual risk should be understood as:

- Configuration- and workload-specific, not universal to all servers,
- Dependent on software inventory and patch posture, and
- Outside the primary enforcement scope of allowlisting controls.

This outcome highlights the need for complementary controls in server environments, such as rigorous patch management, service isolation, privilege reduction, and attack surface minimization, alongside allowlisting to address risks inherent to trusted server applications.

The appendix contains specific Application Control for Business – Formerly Windows Defender Application Control configurations that can alleviate the challenge of privileged workloads listening on the network.

Relevant Microsoft Defender for Endpoint Threat Intel Reports

- Technique Profile: VM extension abuse
- Threat Overview Profile: SharePoint Server and Exchange Server threats
- Vulnerability Profile: CVE-2025-31324 - SAP NetWeaver
- Vulnerability Profile: CVE-2025-30406 - Gladinet CentreStack and Triofox
- Vulnerability Profile: CVE-2024-57726 - Multiple vulnerabilities found in SimpleHelp Remote Support Software
- Vulnerability Profile: CVE-2024-40711 - Veeam Backup
- Technique Profile: Code injection attacks using disclosed ASP.NET machine keys

KF3 – No AttackCampaign reports that are successful on Server or Workstation OS's

The analysis determined that no Attack Campaign reports from 2025 would have resulted in a successful compromise of either Windows workstation or Windows server operating systems when evaluated against the allowlisting configurations defined in this report.

Attack Campaign reports represent a particularly significant category within Defender for Endpoint Threat Analytics. Unlike theoretical techniques or isolated tooling, these reports document real-world attack operations observed by Microsoft in active environments, including coordinated campaigns, end-to-end attack chains, and confirmed adversary behavior in the wild.

Observed Impact

Across all reviewed Attack Campaigns, successful progression depended on one or more execution paths that are directly constrained by application allowlisting controls, including:

- Execution of attacker-delivered binaries,
- Abuse of script-based execution engines,
- Loading of malicious or attacker-controlled libraries.

When these execution paths were restricted, the documented campaigns were unable to progress beyond initial access or early-stage execution, regardless of whether the target environment was a workstation or a server.

Analytical Significance

The absence of successful Attack Campaigns under the evaluated configurations is a strong indicator of practical defensive value. It demonstrates that application allowlisting is not only effective against isolated techniques or tools, but also against complete, operational attack chains actively used by adversaries.

This result is particularly important because Attack Campaign reports reflect threats that have already bypassed other organizations' controls. Preventing these campaigns at the execution stage underscores the role of allowlisting as a preventative control that can stop real-world attacks before detection, response, or remediation are required.

Conclusion

The findings suggest that, for the 2025 threat landscape as observed by Microsoft, comprehensively enforced application allowlisting would have disrupted all documented attack campaigns affecting Windows endpoints, across both workstation and server classes. This reinforces allowlisting's value as a foundational control for mitigating modern, in-the-wild attack activity when deployed with full coverage and appropriate policy design.

Relevant Microsoft Defender for Endpoint Threat Intel Reports

Not applicable

KF4 – Not implementing allowlisting for scripts reduces your effectiveness with 11% and allows 9 attack campaigns to succeed

The analysis shows that 11 threat intelligence reports would have resulted in successful compromise on systems where application allowlisting was deployed but script-based execution was not enforced. This outcome highlights a material reduction in control effectiveness when scripting engines are left outside the allowlisting boundary.

This finding aligns with a well-established security industry trend: PowerShell- and script-based attacks remain one of the most prevalent and effective execution mechanisms used by adversaries on Windows platforms. By operating through trusted interpreters rather than dropping traditional binaries, attackers are able to bypass allowlisting policies that focus exclusively on executables and libraries.

Real-World Significance

The practical impact of this gap is underscored by the composition of the affected reports. Of the 11 threats that would have succeeded under this configuration:

- 9 were classified as Attack Campaigns, representing confirmed, real-world attack operations observed by Microsoft.
- These campaigns documented active exploitation, lateral movement, and post-compromise activity, indicating actual attacker success and tangible organizational impact, not theoretical risk.

In these scenarios, attackers relied on script-based payloads, inline commands, or abuse of PowerShell and related scripting hosts to execute malicious logic within the context of trusted, allowlisted components.

Implications for Allowlisting Strategy

These results demonstrate that allowlisting without script enforcement leaves a critical execution surface exposed. While executable and library restrictions significantly reduce attack options, they do not adequately address adversaries that deliberately avoid dropping binaries.

The findings reinforce that script enforcement is not an optional enhancement but a core requirement for effective application allowlisting on Windows. Omitting script controls materially increases exposure to in-the-wild attack campaigns and undermines the preventative value of the control as a whole.

Conclusion

The prevalence of successful outcomes under this configuration provides concrete, threat-driven evidence that modern attackers actively exploit script execution paths. Comprehensive allowlisting must therefore include explicit control over script-based execution to meaningfully reduce real-world attack risk.

Relevant Microsoft Defender for Endpoint Threat Intel Reports

- Vulnerability Profile: CVE-2025-59287 - Windows Server Update Service
- Activity Profile: Opal Sleet impersonating diplomats to spearphish embassies worldwide
- Activity Profile: Forest Blizzard using new TroikaTwist malware in campaign targeting NATO member states
- Activity Profile: Seashell Blizzard impersonates AV firm in cyberespionage campaign against targets in Ukraine
- Vulnerability Profile: CVE-2025-34028 - Commvault Command Center Innovation Release
- Activity Profile: Marbled Dust leverages zero-day in Output Messenger for regional espionage
- Activity Profile: Threat actors using fake Chrome updates to deliver Lumma Stealer
- Activity Profile: Secret Blizzard and Aqua Blizzard collaborate to target Ukrainian military devices
- Activity Profile: ClickFix and spoofed IT apps deliver RATs via Node.js over Cloudflare Quick Tunnels
- Activity Profile: Phishing campaign impersonates Booking.com delivers multiple commodity malware types
- Activity Profile: Aqua Blizzard implementing Cloudflare Tunnel service to conceal C2

KF5 – Not implementing allowlisting for libraries reduces your effectiveness with 11% and allows 4 attack campaigns to succeed

The analysis indicates that, while less prevalent than in previous years, failing to enforce library-level allowlisting still results in 11 threat reports that would have been successful on systems where executable and script controls were in place but dynamic library loading was left unrestricted.

This finding demonstrates that a subset of adversaries continues to deliberately exploit gaps in basic allowlisting implementations, specifically by relying on malicious or abused libraries rather than standalone executables to achieve code execution.

Threat Composition and Adversary Behavior

Of the 11 threat reports affected under this configuration:

- 4 were Attack Campaigns, reflecting real-world attacks observed by Microsoft in active environments.
- 4 were ToolOrTechnique reports, centered on tools analyzed by Microsoft that are commonly offered to ransomware operators.
- The remaining reports further reinforced patterns of library-based execution and evasion.

Notably, the tools documented in the ToolOrTechnique reports are frequently associated with Ransomware-as-a-Service (RaaS) ecosystems, where affiliates are provided with purpose-built tooling to maximize reliability and evade common defensive controls.

Library-Centric Evasion Techniques

Several of the analyzed tools are explicitly designed to avoid reliance on traditional .exe files. Instead, they employ techniques such as:

- Malicious DLL side-loading alongside trusted binaries,
- Abuse of legitimate application loading behavior,
- Injection of attacker-controlled libraries into trusted, allowlisted processes.

By operating within the execution context of trusted applications, these techniques allow attackers to bypass allowlisting policies that focus solely on executable control.

Implications for Allowlisting Design

Although the frequency of library-centric attacks has decreased relative to prior years, their continued presence—particularly within ransomware tooling—demonstrates that library enforcement remains a critical component of a resilient allowlisting strategy.

The presence of confirmed attack campaigns and commercially distributed ransomware tools among the affected reports highlights that this is not a theoretical concern. Adversaries that invest in operational tooling continue to account for environments with incomplete allowlisting coverage.

Conclusion

These findings confirm that omitting library enforcement introduces a measurable and exploitable gap, even in otherwise mature allowlisting deployments. Comprehensive application allowlisting must therefore extend beyond executables and scripts to include explicit control over dynamic library loading, particularly in environments seeking to defend against ransomware and other financially motivated threat actors.

Relevant Microsoft Defender for Endpoint Threat Intel Reports

- Technique Profile: Malicious browser extensions
- Vulnerability Profile: CVE-2025-0678
- Activity Profile: Manatee Tempest shifts tooling in recent activity
- Tool Profile: RadiantMaze
- Tool Profile: CandleStone
- Tool Profile: SesameOp backdoor
- Activity Profile: Forest Blizzard weaponizes Microsoft Office files to distribute ZooFlip malware in cyberespionag
- Activity Profile: Storm-0249 uses ClickFix to deliver malware
- Vulnerability Profile: CVE-2025-30397 - Microsoft Scripting Engine
- Tool Profile: Reedbed
- Activity Profile: Diamond Sleet shifting tactics in recent hijacked software activity

Appendix

Increasing the protection of privileged server workloads

Policy Code Signing as a Mitigation for Privileged Code Execution

Microsoft App Control for Business provides the capability to digitally sign application control policies, enabling the operating system to enforce the integrity and authenticity of the policy itself. This capability plays a critical role in defending against advanced attack scenarios where adversaries achieve privileged code execution, such as those described in the analysis of server-side vulnerabilities.

Threat Context

As previously identified, a subset of server-focused threats succeeds not by introducing new executables, but by exploiting vulnerabilities in trusted, network-facing applications that run under highly privileged accounts (e.g., LocalSystem). In these scenarios, an attacker may gain the ability to execute code within a trusted process and, by extension, attempt to modify or disable security controls.

Without additional safeguards, such privileged execution could allow an attacker to tamper with or replace application control policies, effectively weakening or bypassing allowlisting protections after initial compromise.

Role of Policy Code Signing

By code signing the App Control for Business policy, organizations ensure that:

- Only policies signed by a trusted signing certificate are accepted by the operating system.
- Unauthorized modifications to the policy are rejected, even when attempted by highly privileged processes.
- An attacker with local or system-level execution cannot trivially weaken allowlisting enforcement to facilitate follow-on payload execution.

The operating system validates the policy signature before applying it, creating a trust boundary that remains intact even in the presence of privileged code execution within a compromised service.

Security Impact

Policy code signing directly addresses the residual risk observed in server environments by preventing attackers from converting a single exploit into sustained or expanded execution capability. While it does not remediate the underlying application vulnerability, it contains the blast radius by preserving allowlisting enforcement after exploitation.

In effect, this control shifts an attacker's options from post-exploitation persistence and tool execution to a far more constrained environment, where additional malicious payloads remain blocked despite elevated privileges.

Conclusion

The ability to code sign App Control for Business policies significantly strengthens allowlisting as a defensive control in server environments. By protecting the integrity of the allowlisting policy itself, it mitigates a key weakness exposed by remote code execution vulnerabilities in

privileged services and reinforces application allowlisting as a resilient, preventative control—even under partial compromise conditions.

Plugin and Child Process Control for Privileged Applications

App Control for Business also provides granular control over plugin and child process execution, enabling allowlisting policies that restrict *how* and *by whom* specific components may be executed. This capability allows administrators to permit a plugin or executable only when launched by an explicitly authorized parent process, rather than universally across the system.

Control Description

Using this functionality, an application allowlisting policy can be constructed to:

- Allow a plugin or auxiliary component to load only within its intended host application.
- Prevent the same component from being invoked directly or by an unauthorized process.
- Restrict which child processes a given executable is permitted to spawn.

This model moves beyond simple file-based trust and introduces context-aware execution control, where trust is dependent on execution lineage.

Security Value for Privileged Processes

This capability is particularly valuable for highly privileged, long-running processes, such as server applications and services running under LocalSystem or other elevated service accounts. These processes are frequently targeted for exploitation due to their privilege level and network exposure.

By explicitly restricting their ability to launch secondary executables—especially general-purpose shells such as cmd.exe and powershell.exe—organizations can significantly reduce post-exploitation options available to an attacker.

In practice, this means that even if a privileged application is compromised:

- The attacker is prevented from spawning interactive shells or script hosts.
- Common post-exploitation techniques that rely on shell access are blocked.
- The attack is constrained to the original process context, limiting lateral movement and persistence.

Alignment With Observed Threats

This control directly mitigates several execution patterns observed in the analyzed threat reports, particularly those involving:

- Abuse of trusted services to launch command interpreters,
- Script-based follow-on payloads after initial exploitation,
- Living-off-the-land techniques executed from privileged contexts.

By enforcing parent–child execution relationships, App Control for Business reduces the effectiveness of these techniques even when initial exploitation succeeds.

Conclusion

Plugin and child process control extends application allowlisting from static execution prevention to behavioral execution governance. When applied to privileged applications, it provides a powerful mitigation against shell spawning and post-exploitation activity, reinforcing allowlisting as an effective containment control in both workstation and server environments.

Defender for Endpoint Threat Intel highlevel numbers

231 Analyst reports in 2025

92 Marked as irrelevant for Analysis

6 left marked as not involving Code Execution on the device

33 Attack Campaigns after these filters

Leaves an even 100 Analyst reports.

Windows Defender Application Control Training with ViaMonstra

Interested in learning how to implement Microsoft Application allowlisting. The author of this report teamed up with the Appcontrol.AI founder to build a comprehensive training around Windows Defender Application control offered by ViaMonstra.

As you've reached the end of this document, and as such showed determination, you're offered with a 12% discount voucher for the training here:

- Discount code: **12off4defender-526**

Training Details

Masterclass Schedule

Date and Time for Live Webinars

Dates and start times for the next 5-Day Masterclass:

- Tuesday, May 12, 2026, 9:00 AM-12:30 PM Central Time (US and Canada)
- Thursday, May 14, 2026, 9:00 AM-12:30 PM Central Time (US and Canada)
- Tuesday, May 19, 2026, 9:00 AM-12:30 PM Central Time (US and Canada)
- Thursday, May 21, 2026, 9:00 AM-12:30 PM Central Time (US and Canada)
- Tuesday, May 26, 2026, 9:00 AM-12:30 PM Central time (US and Canada)

Training- Windows Defender Application Control

Registration

Mastering Windows Defender Application Control Using ConfigMgr and Intune - May 2026 Edition