■■ Microsoft

Microsoft Security   Solutions ⌄   ⌨   Light ☀

🏠 [Blog home] / Threat intelligence   Products ⌄   All Microsoft ⌄

Search the blog   🔍

Research Threat intelligence Microsoft Defender Vulnerabilities and exploits

9 min read

Analyzing CVE-2025-31191: A macOS security-scoped bookmarks-based sandbox escape

By Microsoft Threat Intelligence

May 1, 2025

Ⓕ 𝕏 in

| Microsoft Defender for Endpoint | Microsoft Defender Vulnerability Management | macOS |

In April 2024, Microsoft uncovered a vulnerability in macOS that could allow specially crafted codes to escape the App Sandbox and run unrestricted on the system. An attacker could create an exploit to escape the App Sandbox without user interaction required for **any** sandboxed app using security-scoped bookmarks. With the ability to run code unrestricted on the affected device, attackers could perform further malicious actions like elevating privileges, exfiltrating data, and deploying additional payloads. Microsoft's Threat Intelligence research demonstrates that these exploits would need to be complex, and require Office macros to be enabled, in order to successfully target the Microsoft Office app.

Similar to our discovery of another sandbox escape vulnerability in 2022, we uncovered this issue while researching potential methods to run and detect malicious macros in Microsoft Office on macOS. After discovering this issue, we shared our findings with Apple through Coordinated Vulnerability Disclosure (CVD) via Microsoft Security Vulnerability Research (MSVR). Apple released a fix for this vulnerability, now identified as CVE-2025-31191, as part of security updates released on March 31, 2025. We want to thank the Apple product security team for their collaboration and responsiveness. We encourage macOS users to apply security updates as soon as possible.

This blog post details our investigation into using Office macros to escape the macOS App Sandbox and how we uncovered the CVE-2025-31191 vulnerability. We further demonstrate how the exploit could allow an attacker to delete and replace a keychain entry used to sign security-scoped bookmarks to ultimately escape the App Sandbox without user interaction. This research underscores how security solutions like Microsoft Defender for Endpoint protect devices from cross-platform threats, as well as how collaboration and responsible disclosure are essential to defend users across all platforms and devices.

The macOS App Sandbox and Office macros

The macOS App Sandbox is a security mechanism employed on macOS applications, enforcing strict fine-grained rules on what an app can or cannot do. For example, an app can specify whether it should have internet access or whether it should be able to access specific files. To get apps signed by Apple and published in the Mac App Store, developers must have sandbox rules defined for their apps.

Since 2022, Apple has made significant changes to how the App Sandbox is enforced from within Launch Services, making them aware of the XPC client being sandboxed. That means vulnerabilities that use Launch Services, such as the CVE-2022-26706 vulnerability, as well as CVE-2021-30864, CVE-2022-26696, and others, will not work anymore. Since Microsoft Office is heavily sandboxed on macOS, it seems that the impact of malicious Office macros is minimal and cannot be trivially used as an initial access vector.

Nevertheless, our team decided to perform a threat landscape analysis. With modern Microsoft Office for macOS being heavily sandboxed, two new VBA APIs have been introduced and documented:

- *AppleScriptTask*. This API allows a Microsoft Office macro to run a preassigned AppleScript. The script must be under the directory *~/Library/Application Scripts/[bundle id]/*, which is not accessible for writing from within Office itself. Therefore, script execution cannot be used for VBA-based sandbox escape purposes.
- *GrantAccessToMultipleFiles*. This API grants read and write access to files out of the sandbox from within the macro, which involves heavy user interaction to select and approve those files.

Since the *AppleScriptTask* API did not have obvious vulnerabilities, we started focusing on the *GrantAccessToMultipleFiles* API.

Interestingly, we noticed that the user's choice is persistently saved and used, even between reboots. This indicates that the user's consent is stored in a file that we can attempt to access. An attacker could aim to obtain write and read access to arbitrary files without the user's consent and then escape the macOS App Sandbox by abusing files that would later be used by other apps (such as the file *~/.zshenv* that we analyzed in the past). In such an attack, the attacker could rely on unsuspecting users approving file access to allow trivial sandbox escapes.

```vba
Sub Poc_GrantAccess()

    Dim scptPath As String
    Dim fileAccessGranted As Boolean
    Dim filePermissionCandidates
    Dim payload As String

    ' Set the payload
    payload = "echo pwn"

    ' Grant file permissions
    scptPath = "/Users/" & Environ("USER") & "/.zshenv"
    filePermissionCandidates = Array(scptPath)
    fileAccessGranted = GrantAccessToMultipleFiles(filePermissionCandidates)

    ' Create an Application Script
    Open scptPath For Output As #1
    Print #1, payload
    Close #1

End Sub
```

Figure 1. Proof of concept code for an attack that does involve user interaction
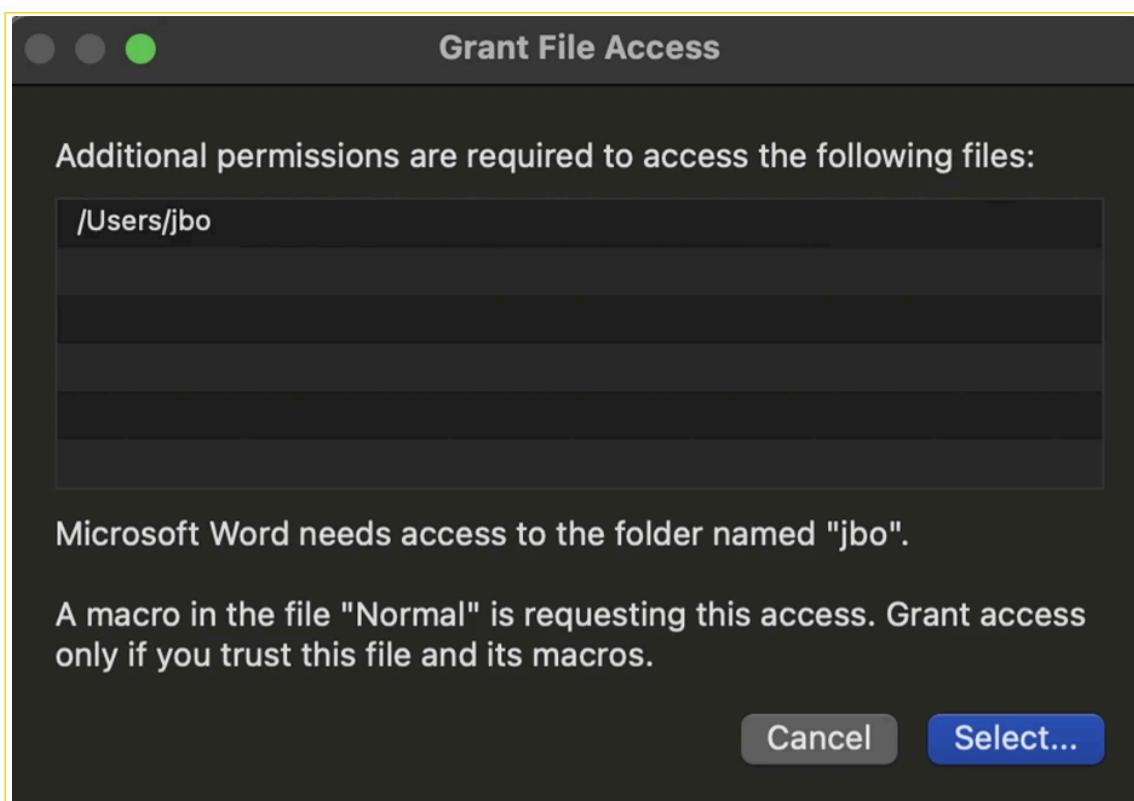
Figure 2. Typical user interaction requiring explicit selection of the folder to grant access to

File access approval using kernel tokens

We discovered that the file that persists the user's choices is a [PLIST](#) file under the *Containers* folder. The *Containers* folder is a special folder in which App Sandbox rules do not apply, which means that the sandboxed app has full access to files there. This is quite attractive for vulnerability research purposes since it means that an attacker might be able to add entries to that file and simply get access to arbitrary files mentioned in that PLIST file.

Microsoft Office uses a macOS mechanism called [security-scoped bookmarks](#), which is a mechanism designed by Apple to specifically bypass the App Sandbox rules using explicit, persistent user choices. We do note that the file seems to contain binary signatures, so frivolously adding new entries or modifying existing ones is not possible.

Figure 3. The secure bookmarks PLIST file saving the signed user choices with typical metadata

Therefore, our team decided to reverse engineer large parts of the macOS modules that support this behavior. However, to fully understand and appreciate the security design of security-scoped bookmarks, it's important to understand how sandboxed apps typically get access to files.

In general, sandboxed apps typically get access to files if a user selects them using the Open dialog. That dialog is controlled by an un-sandboxed service called *com.apple.appkit.xpc.openAndSavePanelService.xpc*. After the user selects the files, that un-sandboxed service transfers access to the selected files to the sandboxed app (using [IPC](#)) via a mechanism called sandbox extensions, which was documented well by Jonathan Levin [in the past](#). Essentially, sandbox extensions are tokens created and signed by the kernel that grant the possessing process the ability to access those files, typically using the lower-level API under *libsystem_sandbox.dylib*. In our case, the Open dialog service passes a sandbox extension token from the kernel to Microsoft Office, which then uses the token for file access purposes, bypassing App Sandbox checks. The token itself contains:

- HMAC-SHA256 authentication. The key used for that HMAC is generated in each boot by the *Sandbox.kext* kernel extension.
- Volume, node information, and other file metadata.
- Capability (such as *com.apple.app-sandbox.read-write*).
- File path.

Because the key that is used to sign the HMAC-SHA256 blob is generated in each new boot, the token cannot persist between reboots. To solve that problem, Apple came up with security-scoped bookmarks, which do something very similar. A new un-sandboxed process called *ScopedBookmarkAgent* was introduced, which can perform two important tasks:

1. Given a sandbox extension token, validate its authenticity and generate a new, serializable object called "bookmark," which will have a long-term HMAC-SHA256 authentication.
2. Given a bookmark, validate its authenticity and generate a new sandbox extension token.

Applications such as Microsoft Office could then use those capabilities to maintain long-term file access:

1. On the first call to *GrantAccessToMultipleFiles*, Office checks if there are file entries in its *securebookmarks.plist* file. Since there are no matching entries, Office consults the Open dialog service, which requires user interaction and receives a sandbox extension token. That token is sent to the *ScopredBookmarkAgent*, which validates the token and then signs it with its own unique, long-term cryptographic key. That data is then serialized by Office to the *securebookmarks.plist* file for later use.
2. On the next call to *GrantAccessToMultipleFiles*, Office finds the entry in its *securebookmarks.plist* file and sends the data to the *ScopedBookmarkAgent*, which validates the signature and generates a sandbox extension token that Office can use without user interaction involved.

The HMAC-SHA256 authentication blob generated by *ScopedBookmarkAgent* cannot be forged unless an attacker has the cryptographic key. The signing key is unique for each app and calculated as such:

cryptoKey=HMAC-256(secret, "[bundle-id]")

The bundle ID is known (for instance, *com.microsoft.Word*) and the key persists in [Keychain Access on macOS](#), saved in the keychain entry *com.apple.scopedbookmarksagent.xpc*.

Therefore, knowing the secret that is stored in the keychain is essential to retrieving the cryptoKey, and that's the only barrier against an attacker signing their own bookmark entries.

Escaping the App Sandbox via the keychain

The macOS keychain can be thought of as a built-in password manager, conceptually similar to how [Credential Manager](#) works on Windows. The keychain is a container for passwords and has Access Control Lists ([ACL](#)) that dictate which process can access each keychain item. The keychain entry we are interested in is *com.apple.scopedbookmarksagent.xpc*, and its ACL dictates only the *ScopedBookmarkAgent* has access to it, which is an excellent security decision by Apple, since injection to that process is not trivial, especially from a sandboxed context.
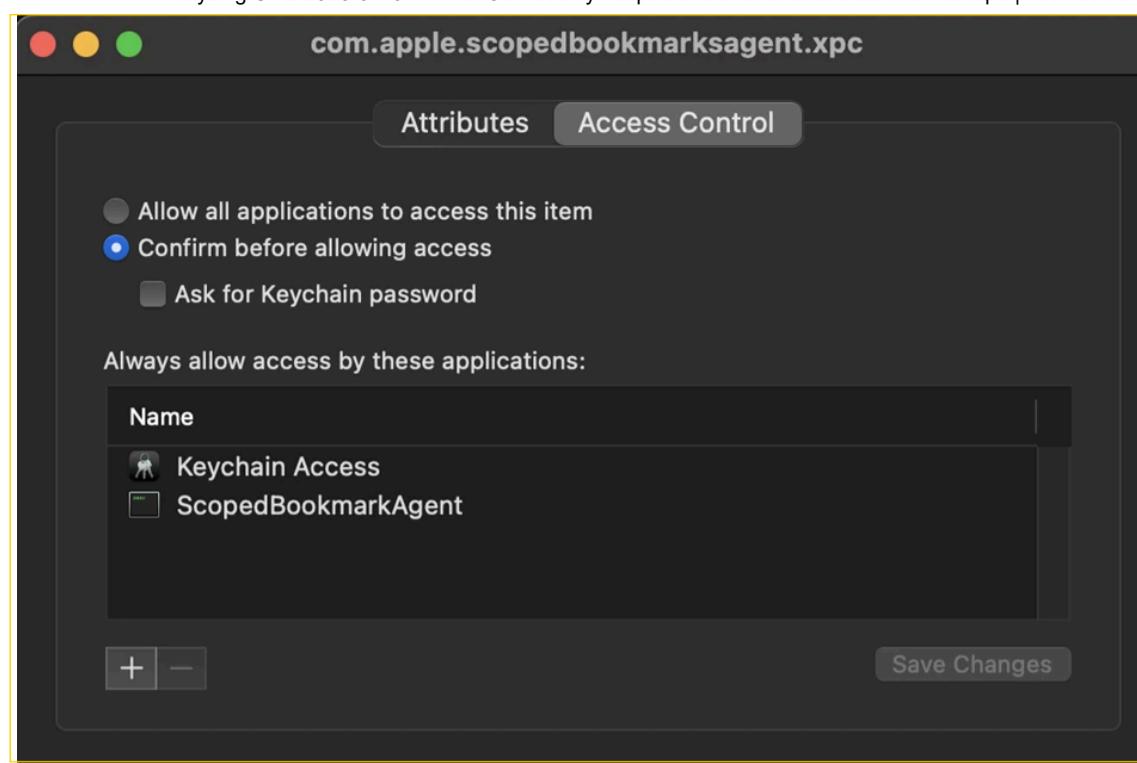
Figure 4. The Access Control List for the scoped bookmarks secret used for signing purposes

It seems as if an attacker cannot do much as they operate within the sandboxed app context and not the *ScopedBookmarkAgent* context, so attackers cannot get the key and, therefore, cannot sign arbitrary new entries in the PLIST file indirectly used by the *ScopedBookmarkAgent*. However, we discovered that the ACL only controls the ability to read the secret. An attacker could completely avoid reading the existing secret and instead can delete the existing entry and **add a new entry**, with a well-known secret. In addition, the attacker could control the new entry's ACL and allow anyone to read the contents of the secret, including *ScopedBookmarkAgent*:

Figure 5. Deletion of the old security-scoped bookmarks secret and assigning a new one from within a sandboxed session

Therefore, an attacker can create an elaborate exploit:

1. Delete the old signing secret from the keychain and decide on a new known secret that is accessible to all processes.
2. Calculate the cryptographic key for an app since its bundle ID is known ($key = HMAC\text{-}SHA256(knownSecret, [bundle\text{-}id])$).
3. Artificially sign new entries in the persistent scoped bookmarks PLIST file that is accessible since it persists in the *Containers* directory.
4. Invoke *GrantAccessToMultipleFiles*, which sends the newly self-signed bookmarks to *ScopedBookmarkAgent*. Since *ScopedBookmarkAgent* uses the new secret, the bookmarks are considered authentic, and therefore *ScopedBookmarkAgent* grants the sandboxed app the access token **without user interaction**.
5. Use the new arbitrary file access capability to escape the macOS sandbox.

As corroborated by our research, this exploit works against any sandboxed app that uses security-scoped bookmarks and is therefore a generic macOS sandbox escape.

Strengthening device security through vulnerability management and threat intelligence sharing

Security technologies such as the macOS App Sandbox are designed to protect the device from malware and other cybersecurity threats, both as a default security measure and a final safeguard. Nonetheless, attackers continue to find new ways of

breaking through these defenses for these same reasons, as they can gain full access to the device and run any files or processes they want without being detected by conventional security solutions.

Our research on the CVE-2025-31191 vulnerability highlights why organizations need a security solution like Microsoft Defender Vulnerability Management that enables them to identify and remediate vulnerabilities and misconfigurations on devices in real time and prioritize those in need of immediate attention. Additionally, Microsoft Defender for Endpoint detects and alerts on anomalous device activities using advanced behavioral analytics and machine learning. In this case, Microsoft Defender for Endpoint detects sandboxed apps controlling security keys that normally are not accessed by those apps. Moreover, in the context of our exploit, Defender for Endpoint detects such behavior as suspicious and blocks the activity, rendering the exploit unusable.
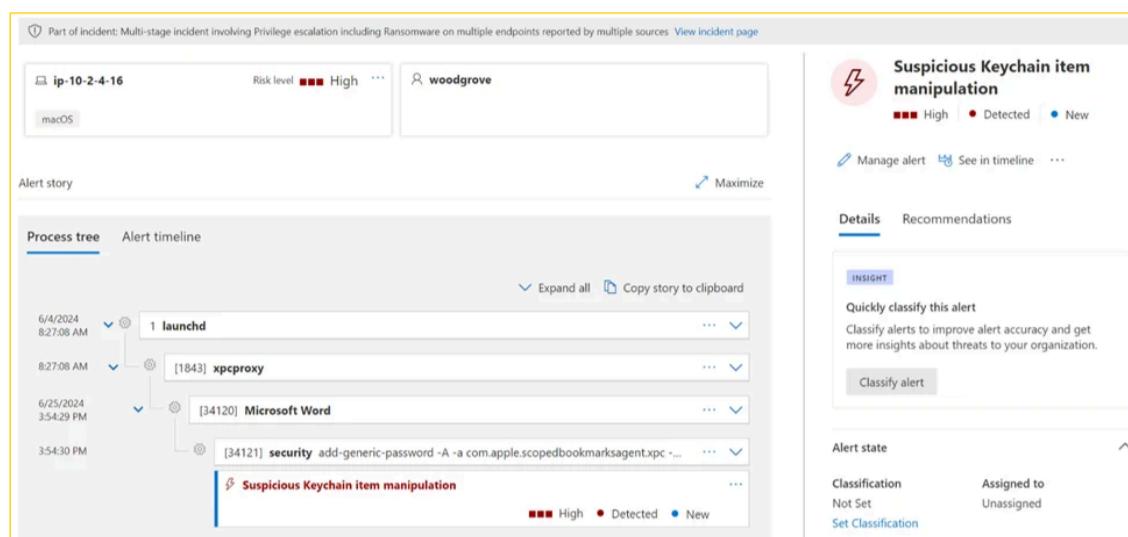

Figure 6. Detection of the exploit

Lastly, this research emphasizes the value and necessity of responsible disclosure and collaboration throughout the security community. Vulnerability discoveries, cooperation between security researchers and vendors, and coordinated response across the security community are all paramount to defend against the ever-growing and ever-changing threats across platforms. These activities, along with other forms of threat intelligence sharing, strengthen and enhance our security technologies to help safeguard users across platforms and devices.

Learn how Microsoft Defender for Endpoint delivers a complete endpoint security solution across all platforms.

Jonathan Bar Or

Microsoft Threat Intelligence

References

- https://developer.apple.com/library/archive/documentation/Security/Conceptual/AppSandboxDesignGuide/AboutAppSandbox/AboutAppSandbox.html
- https://support.apple.com/en-us/122375
- https://developer.apple.com/documentation/security/app-sandbox
- https://developer.apple.com/documentation/coreservices/launch_services
- https://developer.apple.com/documentation/xpc
- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-26706
- https://desi-jarvis.medium.com/office365-macos-sandbox-escape-fcce4fa4123c
- https://perception-point.io/a-technical-analysis-of-cve-2021-30864-bypassing-app-sandbox-restrictions/
- https://wojciechregula.blog/post/macos-sandbox-escape-via-terminal/
- https://developer.apple.com/library/archive/documentation/AppleScript/Conceptual/AppleScriptLangGuide/introduction/ASLR_intro.html
- https://attack.mitre.org/techniques/T1204/
- https://wikipedia.org/wiki/Property_list
- https://developer.apple.com/documentation/professional_video_applications/fcpxml_reference/asset/media-rep/bookmark/enabling_security-scoped_bookmark_and_url_access?language=objc
- https://wikipedia.org/wiki/Inter-process_communication

- https://newosxbook.com/files/HITSB.pdf
- https://support.apple.com/guide/keychain-access/what-is-keychain-access-kyca1083/mac
- https://developer.apple.com/documentation/security/keychain_services/access_control_lists

Learn more

For the latest security research from the Microsoft Threat Intelligence community, check out the Microsoft Threat Intelligence Blog: https://aka.ms/threatintelblog.

To get notified about new publications and to join discussions on social media, follow us on LinkedIn at https://www.linkedin.com/showcase/microsoft-threat-intelligence, and on X (formerly Twitter) at https://x.com/MsftSecIntel.

To hear stories and insights from the Microsoft Threat Intelligence community about the ever-evolving threat landscape, listen to the Microsoft Threat Intelligence podcast: https://thecyberwire.com/podcasts/microsoft-threat-intelligence.

Related Posts

Research Threat intelligence Microsoft Defender Threat Intelligence Vulnerabilities and exploits
Apr 8 · 7 min read

Exploitation of CLFS zero-day leads to ransomware activity >

Microsoft Threat Intelligence Center (MSTIC) and Microsoft Security Response Center (MSRC) have discovered post-compromise exploitation of a newly discovered zero-day vulnerability in the Windows Common Log File System (CLFS) against a small number of targets. Microsoft released security updates to address the vulnerability, tracked as CVE 2025-29824, on April 8, 2025.
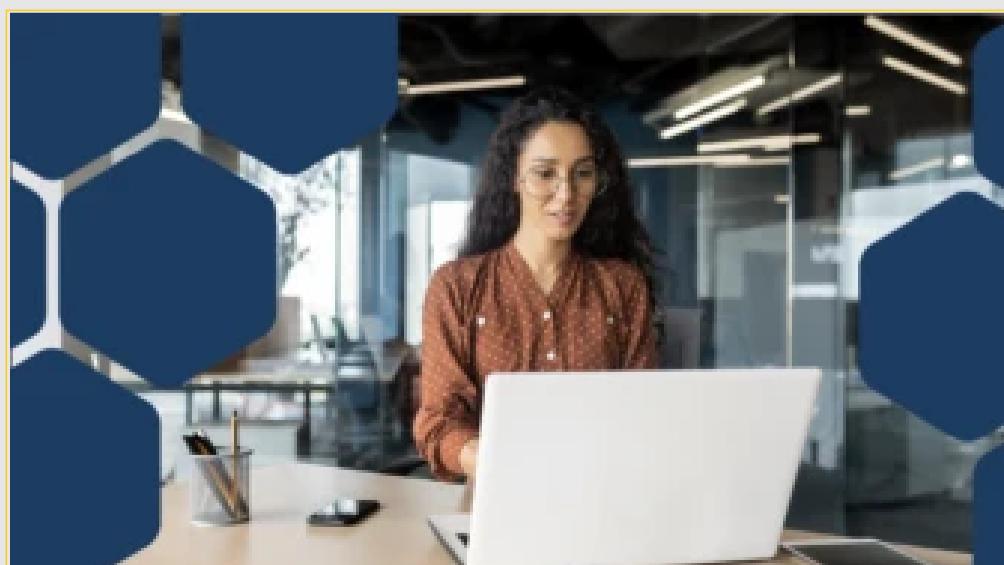
Research Threat intelligence Microsoft Security Copilot Vulnerabilities and exploits
Mar 31 · 13 min read

Analyzing open-source bootloaders: Finding vulnerabilities faster with AI >

Using Microsoft Security Copilot to expedite the discovery process, Microsoft has uncovered several vulnerabilities in multiple open-source bootloaders impacting all operating systems relying on Unified Extensible Firmware Interface (UEFI) Secure Boot. Through a series of prompts, we identified and refined security issues, ultimately uncovering an exploitable integer overflow vulnerability in the GRUB2, U-boot, and Barebox bootloaders.

Research Threat intelligence Microsoft Defender

Research Threat intelligence Microsoft Defender

Vulnerabilities and exploits
Jan 13 · 9 min read

Analyzing CVE-2024-44243, a macOS System Integrity Protection bypass through kernel extensions 〉

Microsoft discovered a macOS vulnerability allowing attackers to bypass System Integrity Protection (SIP) by loading third party kernel extensions, which could lead to serious consequences, such as allowing attackers to install rootkits, create persistent malware, bypass Transparency, Consent, and Control (TCC), and expand the attack surface to perform other unauthorized operations.

Vulnerabilities and exploits
Oct 17, 2024 · 9 min read

New macOS vulnerability, "HM Surf", could lead to unauthorized data access 〉

Microsoft Threat Intelligence uncovered a macOS vulnerability that could potentially allow an attacker to bypass the operating system's Transparency, Consent, and Control (TCC) technology and gain unauthorized access to a user's protected data. The vulnerability, which we refer to as "HM Surf", involves removing the TCC protection for the Safari browser directory and modifying a [...]

Get started with Microsoft Security

Microsoft is a leader in cybersecurity, and we embrace our responsibility to make the world a safer place.

Learn more

Protect it all
with Microsoft Security

Connect with us on social

What's new	Microsoft Store	Education	Business	Developer & IT	Company
Surface Pro	Account profile	Microsoft in education	Microsoft Cloud	Azure	Careers
Surface Laptop	Download Center	Devices for education	Microsoft Security	Microsoft Developer	About Microsoft
Surface Laptop Studio 2	Microsoft Store support	Microsoft Teams for Education	Dynamics 365	Microsoft Learn	Company news
Surface Laptop Go 3	Returns	Microsoft 365 Education	Microsoft 365	Support for AI marketplace apps	Privacy at Microsoft
Microsoft Copilot	Order tracking	How to buy for your school	Microsoft Power Platform	Microsoft Tech Community	Investors
AI in Windows	Certified Refurbished	Educator training and development	Microsoft Teams	Azure Marketplace	Diversity and inclusion
Explore Microsoft products	Microsoft Store Promise	Deals for students and parents	Microsoft 365 Copilot	AppSource	Accessibility
Windows 11 apps	Flexible Payments	AI for education	Small Business	Visual Studio	Sustainability

🌐 English (United States)   ☑️❌ Your Privacy Choices   Consumer Health Privacy

Sitemap   Contact Microsoft   Privacy   Terms of use   Trademarks   Safety & eco   Recycling   About our ads   © Microsoft 2025