Affected platform

Apple Devices and Software

Affected area

Sandbox

Title

macOS Recovery Safari Flaw Allows Unauthorized Read Access to Protected User Files

What is required to reproduce the issue?

- Physical Access to MacBook
- FileVault must be disabled

Detailed description

A critical information disclosure vulnerability has been identified in macOS 16.0 Recovery. The flaw allows an attacker with physical access to open and view **specific types of files** from any location on the system volume, including protected user directories. This bypasses standard file system permissions and can lead to the theft of sensitive personal and application data.

An attacker can boot a Mac into Recovery, launch Safari, and use the "Open File..." dialog (`⌘+O`) to navigate the entire mounted file system. Because Safari in Recovery runs with elevated privileges and lacks proper sandboxing, it can render any file format it natively supports. This includes common formats like `.html` , `.htm` , `.txt` , `.js` , `.css` , `.xml` , `.pdf` , and various image types (e.g., `.jpg` , `.png` , `.gif` , `.svg`). This access is granted regardless of the file's permissions or its location within a protected user folder.

CVSS Score

This vulnerability is primarily an information disclosure flaw. Its severity is high if the disk is not encrypted, as it allows a physically present attacker to read **a wide range of potentially sensitive user and system files**.

Base Score: 4.6 (Medium)

- **CVSS 3.1 Vector:** `AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N`

Metric	Value	Description
Attack Vector (AV)	Physical (P)	The attacker requires physical access to the device.
Attack Complexity (AC)	Low (L)	No specialized conditions are required; the attacker simply uses a standard menu option.
Privileges Required (PR)	None (N)	The attacker does not need any prior privileges on the system.
User Interaction (UI)	None (N)	No user interaction from the legitimate owner is required.
Scope (S)	Unchanged (U)	The exploit affects resources within the same security authority.
Confidentiality (C)	High (H)	**The attacker can access any file readable by Safari, including documents, configuration files, and source code. This leads to a total loss of confidentiality for all data stored in these formats.**
Integrity (I)	None (N)	This vulnerability only allows reading files, not modifying them.
Availability (A)	None (N)	Reading files does not impact the system's availability.

Affected Products

- macOS

Tested Devices

- 2020 Macbook Air M1

Affected Versions

- macOS 16.0 (Sequoia)
- Future versions of macOS may be affected until patched.

Non-affected Versions

- macOS 15.x (Sonoma) and earlier are not affected by this specific read-only vulnerability, though they were affected by the previously reported arbitrary-write flaw.

Limitations

- **(Untested):** This vulnerability was tested on a system with **FileVault disabled**. It is presumed that FileVault would mitigate this attack, as the system volume would be

encrypted and not accessible in Recovery Mode without prior authentication, but this has not been confirmed.

- The attacker requires physical access to the Mac.
- **The vulnerability is limited to file types that Safari can natively render.** This includes, but is not limited to, `.html`, `.txt`, `.pdf`, and various image formats. Binary files, proprietary document formats (e.g., `.pages`, `.docx`), and media files (e.g., `.mp4`, `.mov`) will not open.
- The attacker must know the path to the sensitive files or be willing to browse the file system to find them.

Resulting Capabilities

- **Unauthorized File Access:** Read sensitive user files located in protected directories such as `/Users/{username}/Documents/`, `/Users/{username}/Desktop/`, and `/Users/{username}/Downloads/`, **provided they are in a Safari-readable format.**
- **Data Theft:** View and exfiltrate the contents of personal documents (`.txt`, `.pdf`), configuration files (`.plist`, `.xml`, `.json`), source code (`.js`, `.py`, `.sh`), browser data, and any other data stored in plain text or other supported formats.
- **Bypass of File System Permissions:** The exploit completely bypasses the standard macOS Discretionary Access Control (DAC) model that should prevent one user (or the system) from reading another user's private files without authorization.

Root Cause

The vulnerability stems from insufficient sandboxing of the Safari process within the macOS Recovery environment. While the previous arbitrary-write vulnerability was partially mitigated by restricting file *saving*, the "Open File..." functionality (`⌘+O`) was not disabled or properly restricted. This dialog runs with the elevated privileges of the Recovery environment, allowing it to traverse and read from the entire mounted `Macintosh HD` volume, ignoring standard file permissions for **any file type Safari recognizes.**

Exploitation Flow / Proof of Concept

This PoC demonstrates how an attacker can read a sensitive text file from a user's desktop.

1. **Setup (as a normal user):**
   - Log into a user account on a test device.
   - Create a file on the Desktop named `Sensitive_Data.txt`.
   - Add some confidential text to the file, for example: `This is my private information and my bank password is Password123.`
2. **Exploitation (as the attacker):**

- Shut down the Mac.
- Boot the device into **Recovery Mode**.
- From the Utilities menu, open **Safari**.
- Once Safari is open, press the keyboard shortcut `⌘+O` (or go to **File → Open File...**).
- An "Open" dialog window will appear. In the sidebar, select the system volume (e.g., `Macintosh HD`).
- Navigate to the target user's directory: `Users` → `{username}` → `Desktop`.
- Select the `Sensitive_Data.txt` file and click "Open."

3. **Result:**
   - The contents of `Sensitive_Data.txt` will be displayed directly in the Safari browser window.
   - This confirms that Safari, running in Recovery, was able to read a file from a protected user directory without any authorization, demonstrating a complete confidentiality break **for supported file types.**

Relationship to Previous Arbitrary-Write Vulnerability

This information disclosure flaw is being reported alongside a separate but related arbitrary-write vulnerability affecting macOS 15 and earlier.

In macOS 16.0, it appears Apple has hardened Recovery by deploying a more restricted version of Safari that blocks file downloads, effectively mitigating the arbitrary-write flaw. However, the functionality to open local files (⌘+O) **is present and enabled** in this hardened version, likely to allow access to offline user guides. This capability, when combined with the elevated privileges of the Recovery environment, creates the information disclosure vulnerability described in this report.

Essentially, the hardening effort in macOS 16 successfully addressed the write vector but did not account for the read vector, resulting in a shift from a code execution flaw to a data exposure flaw.

Mitigation / Recommendations

- **Temporary Mitigation: Disable "Open File..." Functionality:** The most direct mitigation is to completely disable the "Open File..." menu item and the `⌘+O` shortcut for Safari when running in the Recovery environment.
- **Enforce Strict Sandboxing:** Apply a strict sandbox profile to the Recovery Safari process that prevents its file dialog from accessing any paths outside of a designated temporary or non-sensitive directory.
- **Treat User Volumes as Opaque:** The Recovery environment should not grant its applications open-read access to the user data volume by default. Access should be explicit

and brokered through authenticated processes only (e.g., the Disk Utility or Terminal after user authentication).

Credit

Yaseen Ghanem