Unknown Title

TamperedChef: Malvertising to Credential Theft

by Bert Steppe

Strategic Threat Intelligence & Research Group (STINGR)

03/10/2025

Executive Summary

TamperedChef is a sophisticated malware campaign that leveraged a convincing advertising campaign strategy and a fully functional decoy application to target European organizations. Disguised as a legitimate application such as a PDF editor, the malware operated with expected functionality for nearly two months before activating its payload to harvest browser credentials, impacting a significant number of systems.

This campaign demonstrates how even well-defined organizations can be compromised by convincing, legitimate-looking software. The consequences are severe: credential theft, potential backdoor access, and the need for full remediation. Organizations must act quickly to identify and remove this threat.

This post shares new insights uncovered by WithSecure’s Strategic Threat Intelligence & Research Group (STINGR), based on our unique visibility through telemetry into affected environments.

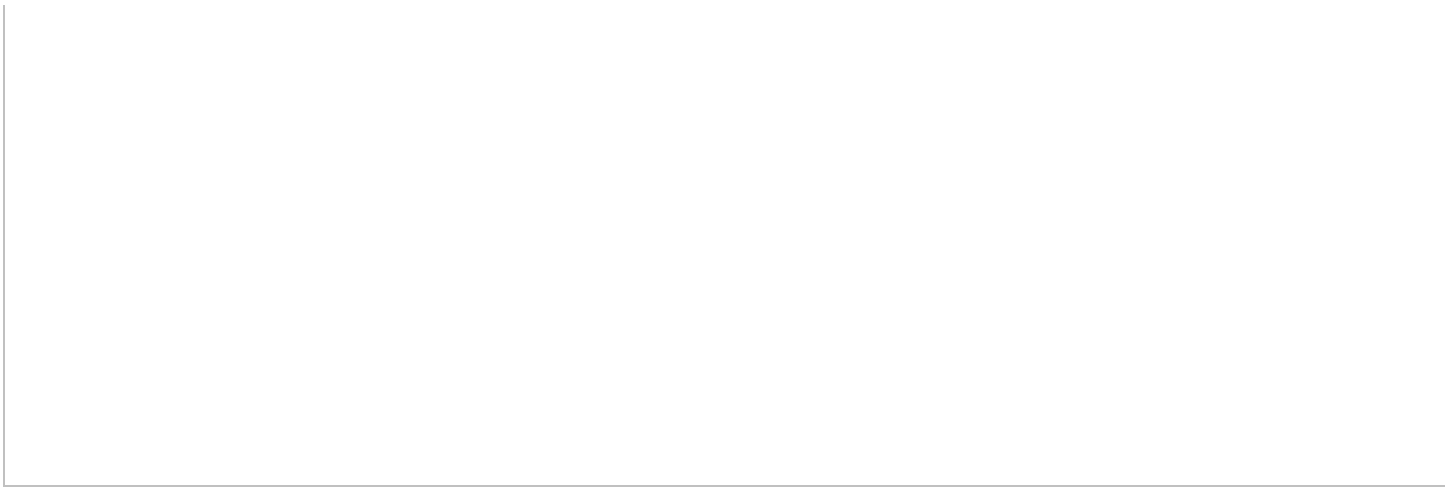

Figure 1. TamperedChef - Country Heatmap

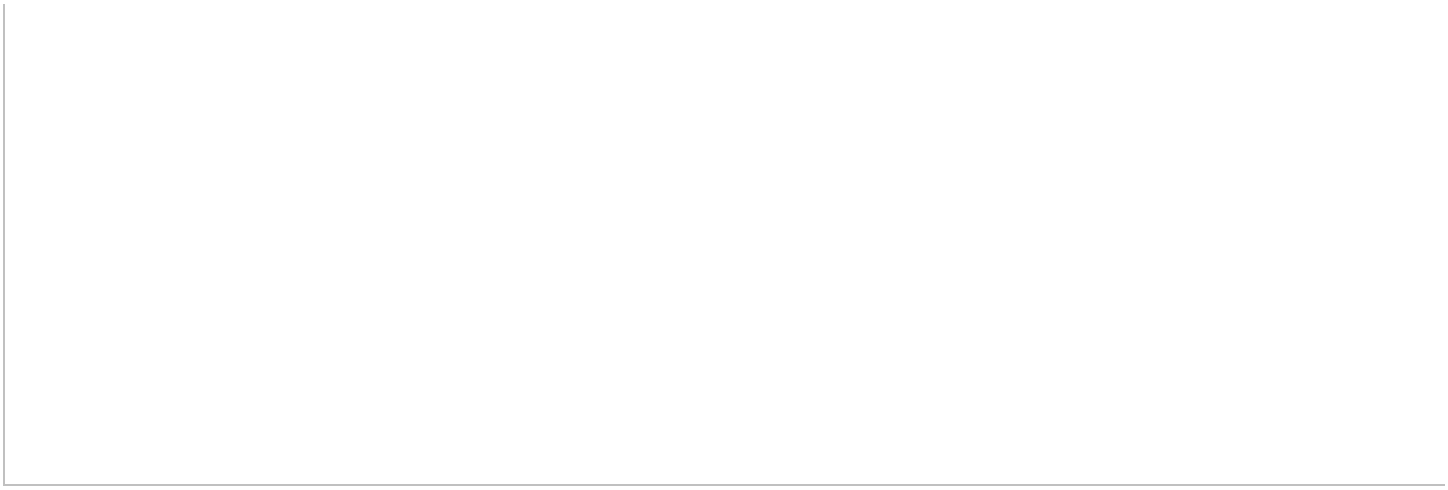

Figure 2. TamperedChef EU focus heatmap

About AppSuite PDF Editor

A typical infection flow began when a user searching for a free PDF editor encountered a malicious ad campaign. Clicking the ad redirected them to a download site controlled by the threat actors, where they downloaded and executed a Microsoft Installer (MSI) package. The installer displayed a EULA acceptance dialog, which made the app appear more legitimate and helped it evade some automated security checks, such as automatic detonation in certain sandbox environments.

Figure 3. EULA dialog displayed by the MSI installer

Figure 4. French EULA dialog, suggesting a possible localized version of ad campaigns and/or download sites.

Clicking the Accept button was the final required user interaction; from there, the MSI downloaded the application installer from vault[.]appsuites[.]ai. This installer, built using NSIS (Nullsoft Scriptable Install System), installed the app under %USERPROFILE% and created persistence by adding an autorun registry entry to launch the app at logon. Notably, the installation required no admin rights, making it effective in business environments with restricted privileges.

AppSuite PDF Editor is written in NodeJS and packaged as an Electron app. Its main executable, “PDF Editor.exe” functions as a full featured Chromium-based browser, executing bundled JavaScript. The malicious payload resides in:

- **pdfeditor.js**: The main, heavily obfuscated JavaScript file, responsible for both the limited UI and the malicious activity. Most PDF editing functionality is delivered via web content from pdf-tool[.]appsuites[.]ai, meaning the app does not work offline.
- **Utilityaddon.node**: A custom NodeJS module (native x64 DLL) used to create/delete registry entries and scheduled tasks, and more.

Figure 5. Utilityaddon.node’s exported functions

The Attacker's Response

When the malicious payload embedded in pdfeditor.js was activated on August 21, 2025, and began stealing browser credentials, the campaign’s true intent was exposed. The threat actors quickly responded by releasing new, “clean” versions of the app (1.0.40 and 1.0.41) just days later, with all malicious JavaScript code removed and the code no longer obfuscated. However, the app continued to connect to attacker-controlled infrastructure, so its use remains strongly discouraged.

Figure 6. Application directory with non-obfuscated scripts

The Abandoned: "AppSuite Print"

During our investigation, we found a similar decoy app called **AppSuite Print**. It was built and signed around the same time as the PDF Editor, on May 20, 2025. It uses an obfuscated JavaScript file named appsuite-print.js, which loads content from hxxps://pdf-tool[.]appsuites[.]ai/en/print. The executable files are signed by 'ECHO INFINI SDN. BHD.', the same certificate used in some versions of the PDF Editor. Technically, it was almost identical, but we found no evidence of its deployment among our customers. It appears the attackers abandoned this variant, likely due to lower demand for a print utility.

Figure 7. Main directory contents of the installed app

The Successor: "S3-Forge"

Shortly after the malicious payload was activated, the true intention of AppSuite PDF Editor became evident. As a result, defenders began detecting and blocking the application, rendering it useless for the threat actor. The attackers quickly pivoted and, based on shared code artifact and development patterns observed in previous campaigns, their next decoy project appears to be an early staged application named **S3-Forge.**

S3-Forge builds directly on the PDF Editor concept but remains under active development. Several artifacts in the codebase continues to reference PDF Editor, and none of the executables are signed. The main window resembles the PDF Editor interface but connects to a different domain (freeonlinetools[.]info), and the application continues to use the original PDF Editor icon.

Figure 8. S3-Forge main window connecting to freeonlinetools[.]info

Figure 9. Legacy PDF Editor configuration remnants recovered from within app.asar (stale/unused by S3-Forge at runtime)

Figure 10. Current/active S3-Forge configuration extracted from app.asar (loaded at runtime)

The S3-Forge binary is unsigned, likely because the project is still in early development. The “--cm” argument is also present in S3-Forge, as in PDF Editor, where it serves as the primary mechanism to enable malicious capabilities.

Figure 11. “--cm” command line argument present in S3-Forge

S3-Forge currently contains no malicious code and we suspect this is consistent with the application being under development and the threat actor testing the various portions of the application. It is also unclear at this point the decoy strategy of the threat actor. The "S3" name might refer to a car model, but it is more likely a reference to Amazon Web Services (AWS) cloud storage, suggesting a possible shift toward targeting software developers. Most observed S3-Forge installers are NuGet packages built by Squirrel (an open-source installation and update framework), indicating experimentation with new distribution methods. In addition to this, Utilityaddon.node and the application’s supporting scripts have been bundled into app.asar, making it less straightforward to detect.

The first known version of S3-Forge was uploaded to VirusTotal on August 26, 2025, just one day after the “de-weaponized” update of PDF Editor.

Conclusions & Recommendations

The TamperedChef campaign demonstrates a high level of planning and execution. The attackers obtained code-signing certificates, developed legitimate-looking applications, and ran a targeted ad campaign to maximize installations. The impact is significant: anyone who installed AppSuite PDF Editor should assume their browser-stored credentials were compromised.

Given the campaign’s success, it is likely the attackers will attempt similar tactics in the future, and other threat actors may follow suit. To protect against these threats:

- Avoid installing software promoted via advertisements. Always use reputable sources.
- In business environments, only install pre-approved applications.
- Invalidate sessions and rotate credentials for affected users.
- Disable browser password stores where feasible. Enforce password managers with policy.

Special thanks to Elias Koivula and Jeremy Ong for their contributions to the investigation.

Detection

WithSecure products detect and block TamperedChef via combination of generic and specific signatures, including but not limited to:

- Trojan:W32/TamperedChef.A*
- Backdoor.BDS/AVI.Agent.*
- Trojan.TR/AVI.Agent.*

Indicator of Compromise - IOC

- iocs/TamperedChef at master · WithSecureLabs/iocs

References

- https://www.truesec.com/hub/blog/tamperedchef-the-bad-pdf-editor
- https://heimdalsecurity.com/blog/heimdal-tamperedchef-investigation/
- https://expel.com/blog/you-dont-find-manualfinder-manualfinder-finds-you/
- https://www.gdatasoftware.com/blog/2025/08/38257-appsuite-pdf-editor-backdoor-analysis